100%
ONE HUNDRED PERCENT
COMPREHENSIVE
AUTHORITATIVE
WHAT YOU NEED
ONE HUNDRED PERCENT

Explore all the key Web site technologies, including HTML, CSS, RSS, JavaScript™, Flash™, and XML

Build Yahoo!® and Amazon.com® online stores and make them stand out in Google™ searches

Create dynamic sites, great eBay® auction pages, and blogs, using the latest tools



Creating Web Sites Bible

2nd Edition

David Crowder and Andrew Bailey

BONUS
COMPANION
WEB SITE
Download examples at
www.wiley.com/compbooks/crowder

www.it-ebooks.info

Creating Web Sites Bible

Second Edition

www.it-ebooks.info

www.it-ebooks.info

Creating Web Sites Bible

Second Edition

David A. Crowder and Andrew Bailey



WILEY

Wiley Publishing, Inc.

www.it-ebooks.info

Creating Web Sites Bible

Published by
Wiley Publishing, Inc.
10475 Crosspoint Boulevard
Indianapolis, IN 46256
www.wiley.com

Copyright © 2004 by Wiley Publishing, Inc., Indianapolis, Indiana

Published by Wiley Publishing, Inc., Indianapolis, Indiana

Published simultaneously in Canada

eISBN: 0-7645-7944-4

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

XX/XX/XX/XX/XX

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 646-8700. Requests to the Publisher for permission should be addressed to the Legal Department, Wiley Publishing, Inc., 10475 Crosspoint Blvd., Indianapolis, IN 46256, (317) 572-3447, fax (317) 572-4355, E-Mail: brandreview@wiley.com

LIMIT OF LIABILITY/DISCLAIMER OF WARRANTY: THE PUBLISHER AND THE AUTHOR MAKE NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE ACCURACY OR COMPLETENESS OF THE CONTENTS OF THIS WORK AND SPECIFICALLY DISCLAIM ALL WARRANTIES, INCLUDING WITHOUT LIMITATION WARRANTIES OF FITNESS FOR A PARTICULAR PURPOSE. NO WARRANTY MAY BE CREATED OR EXTENDED BY SALES OR PROMOTIONAL MATERIALS. THE ADVICE AND STRATEGIES CONTAINED HEREIN MAY NOT BE SUITABLE FOR EVERY SITUATION. THIS WORK IS SOLD WITH THE UNDERSTANDING THAT THE PUBLISHER IS NOT ENGAGED IN RENDERING LEGAL, ACCOUNTING, OR OTHER PROFESSIONAL SERVICES. IF PROFESSIONAL ASSISTANCE IS REQUIRED, THE SERVICES OF A COMPETENT PROFESSIONAL PERSON SHOULD BE SOUGHT. NEITHER THE PUBLISHER NOR THE AUTHOR SHALL BE LIABLE FOR DAMAGES ARISING HEREFROM. THE FACT THAT AN ORGANIZATION OR WEBSITE IS REFERRED TO IN THIS WORK AS A CITATION AND/OR A POTENTIAL SOURCE OF FURTHER INFORMATION DOES NOT MEAN THAT THE AUTHOR OR THE PUBLISHER ENDORSES THE INFORMATION THE ORGANIZATION OR WEBSITE MAY PROVIDE OR RECOMMENDATIONS IT MAY MAKE. FURTHER, READERS SHOULD BE AWARE THAT INTERNET WEBSITES LISTED IN THIS WORK MAY HAVE CHANGED OR DISAPPEARED BETWEEN WHEN THIS WORK WAS WRITTEN AND WHEN IT IS READ.

For general information on our other products and services or to obtain technical support, please contact our Customer Care Department within the U.S. at (800) 762-2974, outside the U.S. at (317) 572-3993 or fax (317) 572-4002.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic books.

Library of Congress Cataloging-in-Publication Data

Crowder, David.
Creating Web Sites Bible/David A. Crowder and Andrew Bailey.
p. cm.
Includes bibliographical references and index.
ISBN 0-7645-7498-1 (pbk.)
1. Web site development. 2. Web sites–Design. I. Bailey, Andrew 1968– II. Title.
TK5105.8888.C76 2004
006.7–dc22 2004017335

Trademarks: Wiley, the Wiley logo, and related trade dress are trademarks or registered trademarks of John Wiley & Sons, Inc. and/or its affiliates, in the United States and other countries, and may not be used without written permission. All other trademarks are the property of their respective owners. Wiley Publishing, Inc., is not associated with any product or vendor mentioned in this book.

www.it-ebooks.info

About the Authors

David A. Crowder has authored or coauthored more than 20 books, including popular bestsellers such as *Building a Web Site For Dummies* and *Cliffs Notes Getting on the Internet*. He was selling hypertext systems back in the days when you had to explain to people what the word meant. He has been involved in the online community since its inception and is the recipient of several awards for his work, including the *NetGuide Magazine* Gold Site Award. When he is not writing, he spends his time with his wife Angela, wandering through villages in the Andes or frolicking in the Caribbean surf.

Andrew Bailey was born in England where he studied Natural Sciences at Cambridge University. He worked for Cray Research as a systems analyst from 1989 to 1992. Then he moved to Colombia, South America. At the present moment he is the Head of Investigation in Computer Sciences at the San Buenaventura University in Medellín where he also teaches Distributed Computing and the Design of Web Applications.

He owns HazloRealidad.com, a company dedicated to the design and hosting of Web sites and the development of custom-built software based on open standards and freely licensed or open source software. HazloRealidad is Spanish for "make it real," which, according to Andrew, "refers to making your dreams come true, well at least the ones related to software". To contact him send an email to `andybailey@hazlorealidad.com`.

About the Contributors

Bud Smith is a Web consultant, teacher, and writer who learned his trade in Silicon Valley and is now living in London, England. He has written more than a dozen books on Web site design and other computing topics, with one best seller placing among the Top 10 computer books in annual sales recently. Among his other work, Bud teaches online classes on Web design, reviewing his students' Web sites as they progress.

Doug Sahlin is an author, graphic designer, and Web site designer living in Lakeland, Florida. He is the author of 14 books on image editing, Web design, and office applications. His best-selling book ranked as high as 22nd on Amazon's Top 50 Computer Books list. The author uses multiple applications to design Web sites that feature animation, video, and images.

www.it-ebooks.info

Credits

Executive Editor
Chris Webb

Production Editor
Felicia Robinson

Development Editor
Sharon Nash

Copy Editor
Michael Koch

Editorial Manager
Kathryn Malm

Vice President &
Executive Group Publisher
Richard Swadley

Vice President and Publisher
Joseph B. Wikert

Executive Editorial Director
Mary Bednarek

Proofreading and Indexing
TECHBOOKS Production Services

www.it-ebooks.info

For my parents, with love and gratitude.

www.it-ebooks.info

www.it-ebooks.info

Contents at a Glance

Preface ix
Acknowledgments xiii

Part I: The Basics of Building Web Pages and Sites 1
Chapter 1: Planning Your Web Pages 3
Chapter 2: Promoting Your Site 17
Chapter 3: Basic HTML Building Blocks 37
Chapter 4: Organizing Your Site 65
Chapter 5: Putting It on the Web 79

Part II: Enhancing Web Pages with Fonts, Images, and Color 109
Chapter 6: Getting Fancy with Text 111
Chapter 7: Finding Images 137
Chapter 8: Blending Images into Your Pages 145
Chapter 9: Creating and Enhancing Images 173
Chapter 10: Adding Color Throughout Your Site 195

Part III: Designing Web Layouts 209
Chapter 11: Harnessing the Power of Tables 211
Chapter 12: Organizing Your Site with Frames 241
Chapter 13: Styling Web Pages with Cascading Style Sheets 269
Chapter 14: Positioning Elements with DIVs 315

Part IV: Adding Interactivity With Flash, Forms, and Other Tools 329
Chapter 15: Getting Input with Forms 331
Chapter 16: Making Dynamic Pages with JavaScript 365
Chapter 17: Navigating Your Web Site 407
Chapter 18: Adding Dynamic Page Elements with DHTML 429
Chapter 19: Animating with Macromedia Flash 449
Chapter 20: Adding Multimedia and Other Objects 487
Chapter 21: Blogging 507

Part V: Cashing In on E-Commerce 527
Chapter 22: Setting Up Your Store 529
Chapter 23: Selling on eBay 549

www.it-ebooks.info

Chapter 24: Getting Paid 571
Chapter 25: Using Advertising 579
Chapter 26: Covering All the Bases 595

Part VI: Maintaining and Updating Your Site 611

Chapter 27: Maintaining Your Site 613
Chapter 28: Updating Your Site 625
Chapter 29: Designing with XML 641
Chapter 30: Making Your Site Mobile with WAP/WML 659

Part VII: Appendixes 679

Appendix A: HTML 4.01 Specification 681
Appendix B: XHTML 1.0, Second Edition Specification 701
Appendix C: XML 1.0, Third Edition Specification 727
Appendix D: WML 2.0 Specification 787
Appendix E: JavaScript Reference 865
Glossary 869
Index 883

www.it-ebooks.info

Contents

Preface ix

Acknowledgments xiii

Part I: The Basics of Building Web Pages and Sites 1

Chapter 1: Planning Your Web Pages 3

Analyzing Web Site Types 3
- Personal home pages 4
- Informational sites 4
- Organizational sites 6
- Political sites 8
- Commercial sites 8

Understanding Internet Demographics 10
- Finding good information 10
- Surveying site visitors 12

Determining Your Approach 13
- Focusing on goals 13
- Deciding on complexity levels 14
- Establishing a budget 15

Summary 15

Chapter 2: Promoting Your Site 17

Exploiting Search Engines 18
- Understanding robots and spiders 20
- Submitting your site 20
- Keeping your site searchable 21
- Pulling it all together 23
- Ranking 24
- Portals 25
- Blocking search engines 26

Working the Web Directories 27
- Ranking in a directory 27
- Registering with the major sites 28
- Listing on Free-For-All link pages 29

Issuing Press Releases 29
- Saying the right thing 31
- Finding e-mail addresses 31
- Formatting the e-mail—to use HTML or just plain text? 32

Setting Up Reciprocal Links 32

Evaluating Banner Exchanges 32
- Building the banner 33
- Design do's and don'ts 33
- Multimedia ads 34

www.it-ebooks.info

Banner Exchanges 35
Traditional advertising 35
Summary 35

Chapter 3: Basic HTML Building Blocks 37
Grasping the Classic HTML Structure 37
Understanding the Basic Elements 38
Defining elements with start and end tags 38
Nesting elements via container relationships 38
Structuring pages with block-level elements 39
Fine-tuning with inline elements 40
Adding Flexibility with Attributes 40
Working with common attributes 40
Using specialized attributes 42
Accepting defaults and setting values 42
Adding Text 42
Heading elements 43
Adding line breaks 43
Dividing with horizontal lines 44
Understanding white space 47
Understanding the Types of Authoring Tools 48
Text editors 49
HTML editors 51
WYSIWYG programs 51
Word processors 54
Exceptions to the rule 55
Choosing Your Tools 55
Trying demos 56
Experimenting with shareware 57
Using freeware 57
Going Beyond the Basics 57
Link checkers 58
Code validators 58
Compatibility testers 58
Server monitors 59
Working with Specific Tools 60
Notepad 60
HomeSite+ 60
Dreamweaver 61
Summary 63

Chapter 4: Organizing Your Site 65
Comparing Page Design to Site Design 65
Color 66
Text style 67
Navigation 67
Understanding the importance of content 68
Making your site functional 68
Planning Your Site's Layout 69
Creating topic-specific Web pages 70
Finding natural breakpoints 71
Addressing Resources with URLs 72
Adding Links 73
Getting hypertext references right 75

www.it-ebooks.info

Creating internal links with local anchors 76
Sending e-mail with mailto links 77
Other types of links 77
Locking relative URLs with the BASE element 77
Summary 78

Chapter 5: Putting It on the Web 79
Getting Your Domain Name 79
Choosing a domain name 79
Picking a registrar 82
Coping with registration pitfalls 83
Finding a Web Space Provider 84
Picking the right Web server 84
Determining your space needs 87
Choosing a non-virtual, virtual, dedicated, or co-located server 87
Balancing price-service ratios 89
Investigating Web space providers 90
Testing Responsiveness to Your Needs 91
Obtaining customer service 92
Assessing technical support 92
Avoiding Common Provider Scams 92
Thinking like a crook 93
Suffering domain name theft 93
Getting "unlimited" traffic or space 94
Rushing deadlines for special deals 94
Exploring Bells and Whistles 95
Getting extra e-mail accounts 95
Redirecting messages 95
Using autoresponders to provide information 97
Getting visitor statistics 97
Using control panels to set site options 98
Becoming a reseller 101
Uploading Your Pages 101
Transferring with FTP 102
Uploading with Web Browsers via HTTP 107
Summary 107

Part II: Enhancing Web Pages with Fonts, Images, and Color 109

Chapter 6: Getting Fancy with Text 111
Customizing Text with Character Styles 111
Italicizing text 112
Bolding text 112
Preformatting text 113
Adding superscripts and subscripts 114
Using the FONT and BASEFONT Elements 116
Specifying a size 116
Using relative size 117
Comparing font size with heading size 117
Overriding default font faces 117
Coloring text 119

www.it-ebooks.info

Aligning and Indenting Text 119
Dealing with deprecation 120
Indenting with BLOCKQUOTE 122
Choosing Character Sets 123
Using Special Characters and Entities 127
Summary 136

Chapter 7: Finding Images 137

Exploring Graphics File Types 137
GIF 137
JPEG 138
PNG 138
Getting Free Images off the Web 138
Finding public domain photographs 138
Downloading free artwork 139
Finding suitable artists 139
Using stock photographs 140
Avoiding Legal Problems 141
Using copyrighted material 142
Respecting trademarks 143
Summary 143

Chapter 8: Blending Images into Your Pages 145

Adding Images 145
Using the height and width attributes 146
Setting horizontal and vertical spacing 147
Setting border size 148
Covering All the Bases with Placeholders 150
Adding alternative text for accessibility 150
Using low-resolution images to speed page download time 152
Aligning Text and Images 153
Floating images out of the flow 153
Demystifying word wrap around images 155
Aligning multiple images 157
Using Images to Link Files 157
Using borders 158
Making multiple links with image maps 160
Solving Image Problems 161
Using thumbnail images to speed up page downloads 161
Embedding special fonts 162
Adding Background Images 163
Calculating sizes for tiling 164
Dealing with “sidebar” backgrounds 165
Checking seamlessness 167
Avoiding busy backgrounds 169
Choosing color and contrast 169
Summary 171

Chapter 9: Creating and Enhancing Images 173

Choosing an Image Editing Tool 173
Photoshop 173
Fireworks 174
Painter 174
Paint Shop Pro 174

www.it-ebooks.info

Modifying Images 176
Cropping 177
Resizing and resampling 178
Rotating and flipping 180
Sharpening and blurring 184
Applying artistic effects filters 184
Using 3D Graphics Programs 187
Poser 188
Bryce 189
trueSpace 189
iSpace 191
3D models and more 192
Summary 192

Chapter 10: Adding Color Throughout Your Site 195
Setting Color Attributes 195
Setting background colors 196
Specifying foreground colors 197
Changing link colors 197
Using Color Names and Hex Numbers 198
Decoding RGB triplets 198
Facing limitations of named colors 199
Using color pickers 200
Using the Three C's: Complement, Contrast, and Coordination 204
Choosing complementary colors 204
Ensuring contrast and visibility 206
Coordinating color schemes 206
Understanding Color Warmth 206
Summary 207

Part III: Designing Web Layouts 209

Chapter 11: Harnessing the Power of Tables 211
Adding Tables and Setting Table Size 211
Managing Borders 217
Using Padding and Spacing 218
Aligning Tables and Cell Contents 220
Setting horizontal alignment 220
Setting vertical alignment 228
Preventing word wrap 228
Spanning Rows and Columns 229
Working with Images and Color 232
Specifying background images 233
Specifying background colors 234
Summary 240

Chapter 12: Organizing Your Site with Frames 241
Designing Frame Layouts 241
Analyzing functional needs 242
Creating navigation layouts 243
Setting up action or result layouts 244

www.it-ebooks.info

Creating Framesets 244
Setting columns and rows 246
Mixing pixel, percentage, and relative sizing 250
Preventing resizing 253
Specifying frame names and contents 253
Nesting framesets 255
Adding NOFRAMES content 257
Setting Targets for Your Links 258
Targeting custom-named frames 258
Using reserved frame names 259
Customizing Frame Borders and Margins 260
Setting border width 260
Handling border colors 262
Setting margin width 262
Setting Scroll Bar Options 263
Alternatives to framesets and frames 263
Avoiding Frame Problems 265
Making sure you have enough frames 265
Adding foreign elements 265
What size is the user's screen? 266
Using too many frames 266
Providing backup navigation 266
Summary 267

Chapter 13: Styling Web Pages with Cascading Style Sheets 269

Testing Your Browser 270
Coping with Browser Support Issues 270
Cascading Priorities 270
Going inline with the style attribute 271
Embedding with the <STYLE> tag 272
Linking external style sheets 273
Redefining HTML Elements 274
Assigning Classes 275
Using IDs 275
Defining Nested Elements with Contextual Selectors 276
Cool effects with CSS 277
Consulting the CSS Reference 279
CSS1 properties 280
CSS2 properties 288
Using Style Sheet–Creation Programs 312
Interesting CSS sites 313
Summary 313

Chapter 14: Positioning Elements with DIVs 315

Adding Layers 315
Using CSS positioning properties 315
Going with the flow 316
Choosing absolute or relative positioning 317
Nested elements 320
Overlapping Layers 321
Stacking layers with z-order 321
Using transparency and background color 321
Clipping Layer Content 322
Visibility of layers 324

www.it-ebooks.info

Handling Large Elements with Overflow Attributes 324
Visible overflow 324
Hidden overflow 325
Scroll bars 326
Summary 327

Part IV: Adding Interactivity With Flash, Forms, and Other Tools 329

Chapter 15: Getting Input with Forms 331

Adding Forms 331
The INPUT element 332
Naming elements 333
Getting Short Data with Text Boxes 334
Setting box size 335
Defining the maximum length of input 336
Prefilling a text box 337
Locking content with readonly and disabled 338
Gathering Information with Text Areas 339
Sizing text areas 339
Setting word wrap 339
Setting default contents 342
Making Choices with Check Boxes and Option Buttons 342
Adding check boxes 343
Grouping option buttons 344
Setting a default choice 346
Saving Space with the SELECT and OPTION Elements 346
Specifying values 349
Setting menu types 349
Allowing multiple selections 350
Setting a default choice 351
Using INPUT Buttons 352
Adding the Submit button 352
Adding a Reset button 352
Creating custom buttons 353
Going pictorial with graphical buttons 354
Using the BUTTON Element 356
Adding Hidden Fields 357
Adding Labels 358
Setting Tab Order 359
Specifying Access Keys 360
Submitting the Form 361
Summary 363

Chapter 16: Making Dynamic Pages with JavaScript 365

Trying Out JavaScript 365
Understanding JavaScript 366
Working with variables and literals 366
Affecting values with operators 369
Adding statements 371
Combining statements into functions 379

www.it-ebooks.info

Making choices with If and If . . . Else 381
Going in loops 382
Using Events to Trigger Scripts 386
Triggering on page loads and unloads 387
Reacting to mouse movements 388
Clicking and double-clicking 391
Pressing and releasing keys 392
Seeing Elements as Objects 393
Understanding properties 394
Working with methods 395
Form Validation 396
Examining Browser Compatibility 400
Summary 404

Chapter 17: Navigating Your Web Site 407

Setting Navigation Rules 407
Avoiding home page blunders 407
Avoiding orphan pages 410
Adding Navigation Bars 410
Using graphics to enhance bars 412
Including pointers 413
Using JavaScript rollovers 415
Choosing bar orientation 417
Displaying Links with Lists and Menus 418
Using plain text link listings 418
Creating ordered and unordered link lists 420
Making menus with the SELECT element 425
Summary 427

Chapter 18: Adding Dynamic Page Elements with DHTML 429

Understanding the Hazards of Animation 429
Getting Dynamic with Rollovers 430
Swapping images 430
Triggering changes in other elements 432
Putting Elements in Motion 435
Calculating absolute position 435
Support for old browsers 437
Determining screen size 441
Color Fades 445
Summary 448

Chapter 19: Animating with Macromedia Flash 449

Understanding the Flash Layout 449
The Stage 450
Scenes 450
Layers 451
The Timeline 451
The toolbox 452
Tool panels 454
Creating Objects 454
Drawing lines with the Line tools 456
Setting stroke characteristics 456
Selecting and deleting objects 457
Making shapes with tools 458

www.it-ebooks.info

Saving your work 459
Importing art 459
Using the Library panel 460
Modifying Objects 460
Stretching and distorting objects 460
Grouping objects 461
Scaling 462
Rotating and skewing 463
Straightening and smoothing 463
Using gradient fills 463
Working with bitmap fills 465
Working with Text 466
Adding text 466
Choosing fonts 466
Setting font characteristics 467
Creating text effects 468
Working with Animation Timelines 469
Using frames 469
Assigning keyframes 470
Adding layers 470
Adding objects to the layers 470
Converting objects to symbols 470
Creating animation 472
Tweening action 472
Creating more layer effects 473
Using sound 476
Adding MP3 sound 477
Synchronizing sound and action 477
Using Advanced Techniques 477
Programming with ActionScript 477
Adding actions 478
Exporting and Publishing Movies 479
Exporting movies 479
Publishing movies 481
Tips for optimizing your movies 483
Summary 485

Chapter 20: Adding Multimedia and Other Objects 487

Adding Audio 487
Choosing a file type 488
Embedding audio 488
Setting volume 490
Using NOEMBED 490
Finding digital audio sources 490
Embedding Video 494
Running a movie 496
Finding digital video sources 496
Adding Java Applets 498
Setting PARAM values 498
Adding alternative content 499
Making it work 499
Applets are objects too 502
Finding applet sources 505
Summary 506

www.it-ebooks.info

Chapter 21: Blogging 507
The World of Blogging 507
Using Blogger 509
A brief history of Blogger 509
Setting up your blog on Blogger 510
Adding to your blog 514
Alternative Blogging Tools 518
AOL's Hometown 518
blogs.com and TypePad 519
LiveJournal 520
Radio UserLand 521
Blogging add-ons 523
Integrating Blogging 524
Blogging as a writing style 524
Integration by reference 524
Full integration 524
Summary 525

Part V: Cashing In on E-Commerce 527

Chapter 22: Setting Up Your Store 529
Choosing Your E-Commerce Setup 529
Single product sites and specialty stores 529
General stores 530
Online superstores 531
Online computer stores 532
Service sites 532
Shopping malls 534
Setting Up Shopping Carts 535
Addressing concerns about security 536
Joining mall-provided programs 536
Getting commercial programs 538
Using open source programs 538
Taking Payment 540
Credit and debit cards 541
Checks 541
E-cash 542
Setting up an affiliate store 544
Riding Internet Buying Patterns 544
Summary 547

Chapter 23: Selling on eBay 549
Working with eBay 549
Setting up an eBay account 551
Fees for selling on eBay 552
Setting Up an eBay Auction 554
Using the Buy Now option 560
Setting up a Dutch (multiple item) auction 560
Payment and Shipping Options 561
Setting up a PayPal account 561
Getting paid with checks 562
Shipping Merchandise 563
Getting the Most from eBay 564
Establishing yourself as a reputable dealer 564

www.it-ebooks.info

Setting a start and finish time 565
Monitoring your auction 565
After the auction 565
Enhancing your chances for success 566
Using the eBay template 566
Modifying the eBay template 567
Setting Up an eBay Store 568
Summary 568

Chapter 24: Getting Paid 571
Credit and debit cards 571
Establishing Return and Refund Policies 573
Watching out for chargebacks 575
Checks and Paper Drafts 576
E-cash 577
Summary 578

Chapter 25: Using Advertising 579
Advertising with Google and Other Search Engines 579
Advertising with Google 579
Advertising with other search engines 585
Advertising with banners 587
Working with Advertisement Agencies 588
Finding an Internet advertisement agency 588
Working with affiliates 589
Investigating Alternative Approaches 590
E-mail advertising 590
Amassing a mailing list 591
Sending newsletters 592
Summary 592

Chapter 26: Covering All the Bases 595
Using Domain Names and Trademarks 595
Domain name disputes 596
Anti-Cybersquatting Consumer Protection Act 598
Researching domain names online 598
Maintaining your domain name 598
Researching trademarks online 599
Soothing Common Fears 600
Publicizing your security 601
Creating a privacy policy 603
Establishing Return and Refund Policies 603
Investigating Alternative Approaches 607
Accepting advertising 607
Optimize Your Site for Search Engines 607
Summary 610

Part VI: Maintaining and Updating Your Site 611

Chapter 27: Maintaining Your Site 613
Testing Your Site 613
Visit your site as a stranger 614
Finding outside testers 615
Evaluating input from testers and visitors 617

www.it-ebooks.info

Coping with Visitors' Changing Your Site 618
Browser preference settings 618
Maintaining Consistency on Your Site 621
Visitor comfort 621
Common maintenance tasks 621
Site improvement tools 622
Server monitors 622
Summary 623

Chapter 28: Updating Your Site 625

Keeping Your Site Fresh 625
Using incremental uploads 626
Establishing a schedule 626
Finding new material 627
Having your visitors provide material 627
Adding Extra Value 629
Producing newsletters and e-zines 629
Writing product reviews 630
Creating tutorials 630
Facing the XML Future 631
Classifying content versus defining presentation 633
Customizing tags 633
Specifying changes with a document type definition 634
Getting different XML vocabularies to work together 636
Transitioning to XHTML 1.0 637
Making HTML XML-compliant 637
Contrasting XHTML with HTML 4.0 639
Facing browser compatibility problems 639
Summary 640

Chapter 29: Designing with XML 641

Designing with XML 641
XML Rules 642
Getting different XML vocabularies to work together—Namespaces 642
Classifying content versus defining presentation 643
Specifying a CSS 644
Specifying an XSL style sheet 645
Defining the structure of an XML document 649
Customizing tags 651
More on DTD 652
Transitioning to XHTML 1.0 653
Making HTML XML compliant 654
Contrasting XHTML with HTML 4.0 655
Facing browser compatibility problems 656
Summary 657

Chapter 30: Making Your Site Mobile with WAP/WML 659

What Is WML? 659
Dealing with WML 660
WML tags 662
Shuffling the cards 664
Interacting with Users 666
Forms in WML 667
Specifying the format 667

www.it-ebooks.info

Specifying required values 669
Menus a la card 669
Multiple-choice menus 671
Differences between HTML lists and WML lists 672
Transforming XHTML into WML 673
Summary 677

Part VII: Appendixes 679

Appendix A: HTML 4.01 Specification 681

Appendix B: XHTML 1.0, Second Edition Specification 701

Appendix C: XML 1.0, Third Edition Specification 727

Appendix D: WML 2.0 Specification 787

Appendix E: JavaScript Reference 865

Glossary 869

Index 883

www.it-ebooks.info

www.it-ebooks.info

Preface

Welcome to *Creating Web Sites Bible*. This book includes everything you need to know to put up a cutting-edge site on the World Wide Web. It covers everything from the basic HTML you have to use all the way on up to the tricks of the trade used by JavaScript programmers and Flash wizards.

How This Book Is Organized

Creating Web Sites Bible consists of 30 chapters divided into six parts and a set of six appendixes, including a glossary.

Part I: The Basics of Building Web Pages and Sites

Part I covers planning your site and shows how to use HTML—the Hypertext Markup Language—to create Web pages. It then covers how to organize those pages into a coherent Web site and explores your various options in putting that site on the World Wide Web itself.

Part II: Enhancing Web Pages with Fonts, Images, and Color

Part II explores the use of text, images, and color in the design of a good-looking Web site. It demonstrates how all these elements must be carefully integrated to create the best look and feel you can achieve.

Part III: Designing Web Layouts

Part III delves into the mysteries of using frames and framesets to put multiple Web pages onscreen simultaneously. It also covers the use of tables and shows how they can be used for so much more than merely displaying information. The chapters in this part go on to cover the latest advances in cascading style sheets and the use of layers to achieve total control over the layout of your Web pages.

Part IV: Adding Interactivity with Flash, Forms, and Other Tools

Part IV gets down to brass tacks about using forms to gather information from your site's visitors. It then moves on to show you how to use JavaScript for both

www.it-ebooks.info

animation and site navigation and covers Macromedia Flash, the premier Web animation tool, in depth. For good measure, it also covers how to add multimedia, such as sound and video, to your Web site.

Part V: Cashing in on E-Commerce

Part V goes where the money is—e-commerce. It shows you how to determine what kind of setup you need and how to take credit cards online in a secure environment, and it explores a variety of options, including affiliate programs.

Part VI: Maintaining and Updating Your Site

Part VI shows you how to handle the critical issues of maintaining your site and keeping your material fresh and interesting. It wraps up with coverage of where Web technology is heading and what you'll need to know in the coming years, including XML and XHTML and the latest advances in wireless Web, WAP and WML.

Part VII: Appendixes

The book contains several useful appendixes: "HTML 4.01 Specification," "XHTML 1.0, Second Edition Specification," "XML 1.0, Third Edition Specification," "WML 2.0 Specification," and "JavaScript Reference." The final appendix is a glossary of useful terms.

About the Web Site

This book provides a companion Web site from which you can download the code from various chapters. All the code listings reside in zip files at `www.wiley.com/compbooks/crowder`. If you don't currently have WinZip, you can download an evaluation version from `www.winzip.com`.

Conventions

This book includes the following icons to draw your attention to specialized information:



This icon points out some potential "gotchas" of which you need to be aware.



This icon points you to additional information on a topic that appears elsewhere in the book.

www.it-ebooks.info



Notes add a little bit of extra knowledge to your armament.



Tips toss in other ways of accomplishing a goal or point out some optional approach that may suit you better.

www.it-ebooks.info

www.it-ebooks.info

Acknowledgments

Thanks are due to Chris Webb and Sharon Nash, my fine editors, who were there for me every step of the way. Also Carol Sheehan, Laura Brown, Valerie Perry, Carmen Krikorian, Eric Butow, Matthew David, Patricia Hartman, Stephanie Cottrell Bryant, Wendy Willard, Chris Stone, Doug Sahlin, Bud Smith, and Andy Bailey, without whom this book would not be in your hands right now. All helped to make this the best book we could all put together for you. And they're just the tip of the iceberg: about a zillion people work their tails off anonymously and behind the scenes at Wiley to bring you the finest books they can possibly produce. My hat is off to all of them, from the top editors to the humblest laborer on the loading dock. Last, but by no means least, I'd like to say how much I appreciate all the hard work done by my literary agent, Robert G. Diforio, without whose help I would be lost in the intricacies of the publishing world.

www.it-ebooks.info

www.it-ebooks.info

P A R T



The Basics of Building Web Pages and Sites



In This Part

Chapter 1
Planning Your Web Pages

Chapter 2
Promoting Your Site

Chapter 3
Basic HTML Building Blocks

Chapter 4
Organizing Your Site

Chapter 5
Putting It on the Web



www.it-ebooks.info

www.it-ebooks.info

CHAPTER 1

Planning Your Web Pages

In This Chapter

Analyzing Web site types

Understanding Internet demographics

Determining your approach

If you have a completely solid, totally clear idea of exactly what you want to do on the Web, you can safely skip this chapter. Maybe. Are you absolutely certain that you have considered all the details? How does your site compare to other similar ones? Where does it fit in? What are the differences between them? Are those differences enough to distinguish yours, to attract visitors away from competing sites?

Have you covered everything from setting a budget to grasping current Internet demographics? Has it occurred to you to avoid some useful technologies? If your answer to any of these questions isn't a resounding yes, you should start right here, before you do anything else.

Analyzing Web Site Types

You could argue endlessly about how to categorize the overwhelming variety of sites on the World Wide Web. For all practical purposes, however, there are only five basic types of sites—personal, informational, organizational, political, and commercial.

As with any attempt at taxonomy—the art of lumping disparate entities into categories based on similarities—there are gray areas, fuzzy situations, and downright overlaps. What do you make of a personal home page that's largely political? In our system, it's still a personal home page. What about a genealogy site? Is that an extension of a personal home page or is it an informational site? We vote for it being informational, and labeling a site personal only if its focus is limited to an individual and his or her immediate family and friends.

To categorize sites, the basic method we use is to try to determine the intent of the Web site. If it's to make money, then it's a commercial site, no matter what other angles there

www.it-ebooks.info

are to it. If it's to promote a particular candidate in an election, then it's a political site, even if it's made by one individual.

Personal home pages

Personal home pages exist for the purpose of introducing individuals to the world. Despite the phrase "personal home page," which has become generically accepted as a description of this type of Web site, some personal sites involve many different Web pages and can be as elaborate as a corporate Web site. Generally speaking, the information in them is somewhat limited in scope, appealing mainly to the close friends and immediate family of the person involved. This isn't to say that some of them aren't very interesting, indeed. As with any other type of site, a carefully designed and well-written production is a real joy to explore and experience.

Just as you'd find when talking to strangers at a party, some people are fascinating and have great stories to tell, whereas others send you running from the room, yawning from the boredom. Personal Web sites can be a similar experience. Some home pages grab you, and others make you wish you had never clicked that link.

When putting together a personal site, it's a good idea to open up yourself as much as possible within your comfort level. The best personal sites enable you to get to know the people who created them; by the time you're done browsing these sites, you feel as if you could sit down to dinner with their authors and talk as if old friends. This isn't to say that you've got to spill your guts about every detail of your personal life, but the site *is* about you—and if *you* aren't really in it, what's the point of creating it?

Informational sites

Sites that are dedicated to providing information on a particular topic are common on the Web. If they weren't, it would be a pretty boring place. In fact, most people use the Web primarily to find the answer to some sort of question. Many of the informational Web sites are run by public-minded organizations whose goal is simply to make the public aware of particular issues. The Monroe County Community School Corporation, for instance, provides much-needed information on the confusing situation of Indiana's handling of daylight savings time at `www.mccsc.edu/time.html` (see Figure 1-1).

Search engines, on the other hand, are a perfect example of the kind of gray area we discussed earlier, where it's difficult to slot many of them into one category. The vast majority of them are commercial operations designed to generate advertising revenue—some of them even rank the results by how much they're paid instead of by true relevance—but the only interest that Web surfers have in them is to find information. The same is true of news sites.

Another type of informational site is one that offers a limited amount of information at no charge, but which provides a greater amount for a fee. Spanish Learning Resources (`www.studyspanish.com`) is a prime example of this. The site, shown in Figure 1-2, provides about as much training as you would get in a typical first-year Spanish language course for free. If you simply need to understand such niceties as why *ser* and *estar* both mean "to be," or need a basic grasp of Spanish vocabulary

www.it-ebooks.info

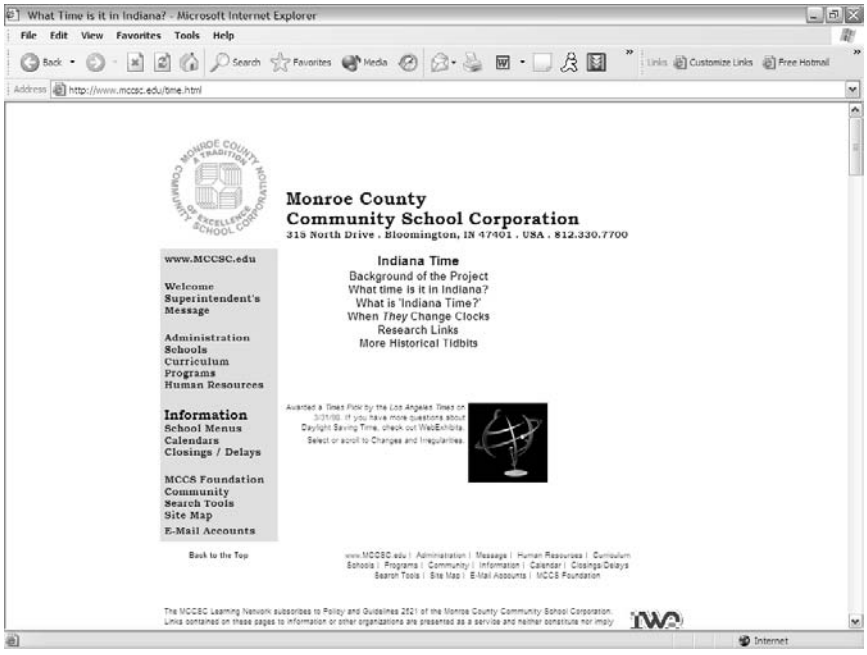


Figure 1-1: The Monroe County Community School Corporation provides free instructions on understanding Indiana time.

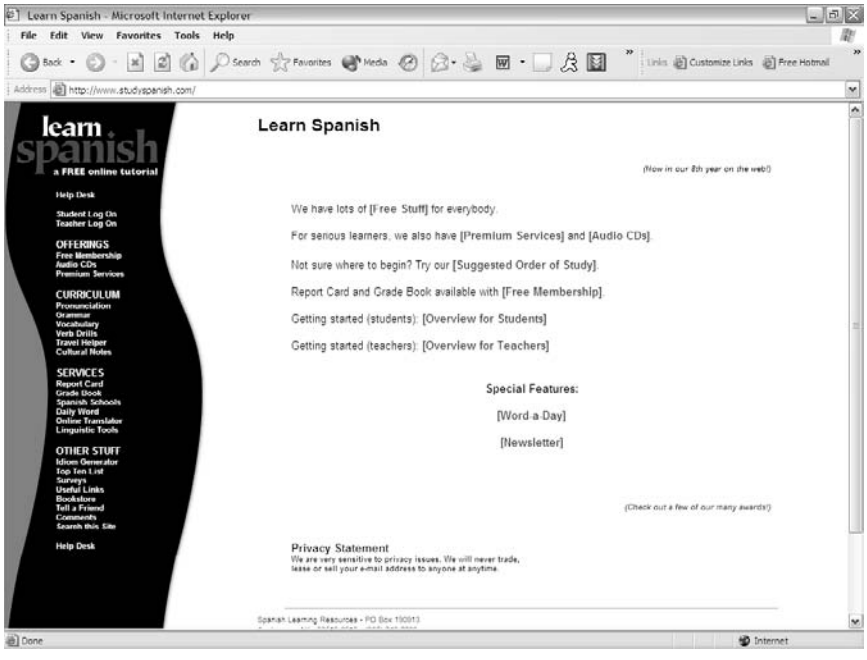


Figure 1-2: Spanish Learning Resources provides both free and premium information services.

www.it-ebooks.info

and grammar, visit this site. (You should still pick up a copy of *Spanish For Dummies*, though—you never know when your Internet connection might go down.)

Table 1-1 lists the URLs (Uniform Resource Locators) of several informational sites that you can browse and study.

Table 1-1
Informational Web Sites

Site	*URL*
AcqWeb's Directory of Book Reviews on the Web	acqweb.library.vanderbilt.edu/acqweb/bookrev.html
AnyWho	www.tollfree.att.net
Catalog of Solar Eclipses	sunearth.gsfc.nasa.gov/eclipse/SEcat/SEcatalog.html
CIA World Factbook	odci.gov/cia/publications/factbook
CIAC	http://ciac.llnl.gov/ciac/index.html
Dead People Server	www.dpsinfo.com/dps/
Indiana Department of Agriculture	ai.org/oca
Internet Traffic Report	InternetTrafficReport.com
King Arthur and the Matter of Britain	legends.dm.net/kingarthur
Letters and Dispatches of Horatio Nelson	wtj.com/archives/nelson
Sir Francis Drake	www.mcn.org/2/oseeler/drake.htm
Veterinary Information Network	vin.com

Organizational sites

Organizational Web sites are concerned with presenting information about—you guessed it—a particular organization. The reason we distinguish these from pure informational sites is their narrow focus. Their sole function is to describe the work of a particular organization, its structure, its personnel, and history. These sites don't diverge from their function at any point—no sidetracks to outside issues or topics exist except for occasional links to other sites that partner with or support the goals and purposes of the organization.

The Internet Engineering Task Force (IETF) site, located at ietf.cnri.reston.va.us, is a classic example of an organizational site. It covers everything from the structure of the organization to the process for joining it (see Figure 1-3).

Table 1-2 gives the addresses of some organizational sites on the World Wide Web.

www.it-ebooks.info

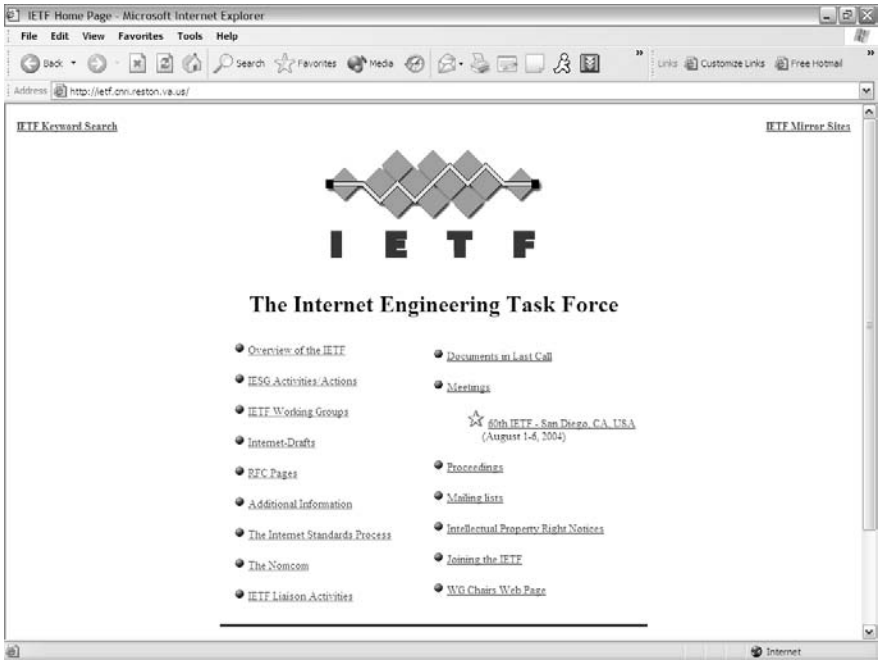


Figure 1-3: The Internet Engineering Task Force (IETF) Web site has a full explanation of the structure, goals, and functions of the organization.

Table 1-2
Organizational Web Sites

Site	*URL*
International Center for Reiki Training	reiki.org
JustLinux	justlinux.com
National Weather Service	www.nws.noaa.gov
Nautical Research Guild	naut-res-guild.org
Nicholas Roerich Museum	roerich.org
Small Business Administration	www.sba.gov
State Library of New South Wales	www.sl.nsw.gov.au
The Naval Dockyards Society	www.hants.gov.uk/navaldockyard
The Royal Society	www.royalsoc.ac.uk
The Society for Creative Anachronism	sca.org
World Wide Web Consortium	w3.org

www.it-ebooks.info

Political sites

Sites that have a political agenda to push aren't limited to dealing with particular candidates for certain offices in election years. Politics isn't all about elections. In our definition, such sites also include those with any social agenda that would be furthered by the passing of appropriate legislation. Thus, Web sites covering issues such as gun control, the medical usage of marijuana, development of wetland areas, or the control of personal information on the Internet—regardless of whether the site's editorial thrust is for or against—would qualify as political sites.

Figure 1-4 shows the Web site for the Democratic National Committee (DNC) at `democrats.org`. It could be argued that it is, in fact, an organizational Web site, representing as it does the DNC, its views, and activities. Because those activities are strictly political, however, it belongs here.

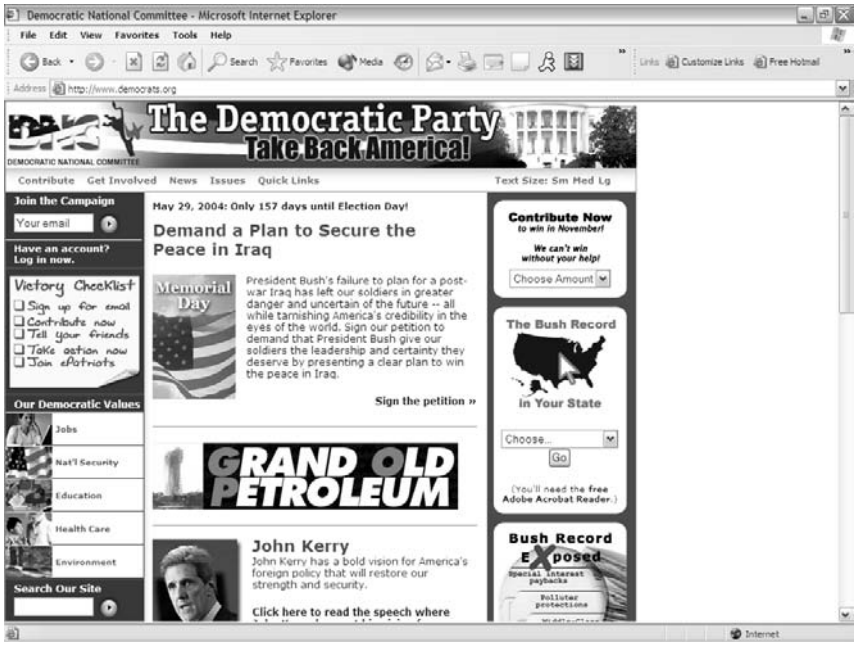


Figure 1-4: The Democratic National Committee site is a prime example of a political Web site.

Table 1-3 provides the URLs of several political Web sites.

Commercial sites

This is the one simple, unambiguous area in the taxonomy—if the site is created strictly out of a profit motive, then it's a commercial site. Period. While other sites—particularly political sites—often engage in some kind of fund-raising activity that is secondary to their main purpose. With purely commercial sites, there is never

www.it-ebooks.info

Table 1-3
Political Web Sites

Site	*URL*
Australian Politics Online	www.ozpolitics.com.au
Chinese Politics Online	www.politics.unimelb.edu.au/cpo
Conservative Party Website	www.conservatives.com/home.cfm
Labour Party Website	www.labour.org.uk
Republican National Committee	rnc.org
Southeast European Politics Online	www.seep.ceu.hu
Washington Center for Politics & Journalism	wcpj.org

any doubt—the main thrust is simply "Buy our product or service" or "Support our sponsors."

The Barnes & Noble Web site (www.bn.com), shown in Figure 1-5, goes straight to the point. It's there for one thing and one thing only—to enable you to find and buy books and music.



Figure 1-5: The Barnes & Noble Web site is a prime example of a well-designed and successful commercial Web site.

www.it-ebooks.info

Simply accepting advertising or tossing in a few links to a bookstore doesn't magically transform another kind of site into a commercial Web site. There are few sites in any category that do not attempt to raise some money (often unsuccessfully) by a half-hearted foray into banner ads.

Sometimes, especially for sponsored sites or those which are participants in an affiliates program, the "buy" message is so subtle that the commercial nature of the site isn't obvious to the casual visitor. Many times, there is no mention at all of any sales intent, but the site's links lead to sponsored pages.



See Chapter 25 for more information on affiliates programs and advertising.

Table 1-4 guides you to a number of commercial Web sites.

Table 1-4
Commercial Web Sites

Site	*URL*
Australian Bush Flower Essences	www.ausflowers.com.au
Gibson Research Corporation	grc.com
Jagex Software	jagex.com
Janes Information Group	www.janes.com
Magix Entertainment Software	www.magix.net
Ptaah	www.ptaah.com/home.html
Staples Office Supplies	staples.com
The History Channel	www.historychannel.com/ontv/index.html
TV Guide	tvguide.com

Understanding Internet Demographics

There are tons of "facts" available about the Internet. Unfortunately, most of these statistics can be classified by one of Mark Twain's most famous quip, "There are lies, damned lies, and statistics." As with so many of the greatest American writer's sayings, the truth of this rides into our present and, doubtless, our future as well.

Finding good information

It's easy to find anything you want to know about the Internet—just look on the Internet. But how do you know if you can trust what you find? One of the simplest

www.it-ebooks.info

ways is also one of the best—know the reputation of the source. If someone's Web site says that there are eight billion people with Internet access, you might want to know who they are and how many planets they think are hooked up to the Internet.

The same is true, however, of even less obviously suspect figures. News stories are, unfortunately, one of the least reliable of all statistical sources, because they are generally written in a hurry by people who do not really understand statistics. They also often fall prey to the fault of accepting statistics from yet other news stories. This can easily lead to the kind of situation where one reporter gets something wrong, but it is published in a major publication. Because another reporter believes that anything in *The Wall Street Journal* or *The New York Times* must be true, the figure is incorporated into yet another article, which is taken as gospel by someone else, creating a chain that perpetuates a number that is far removed from its original source.

We'd be inclined to trust, without more than a reflexive professional doubt, a report from companies such as Jupiter Communications (see Figure 1-6), Nielsen, or WebSideStory, all of which are professional outfits that have proved reliable in the past.

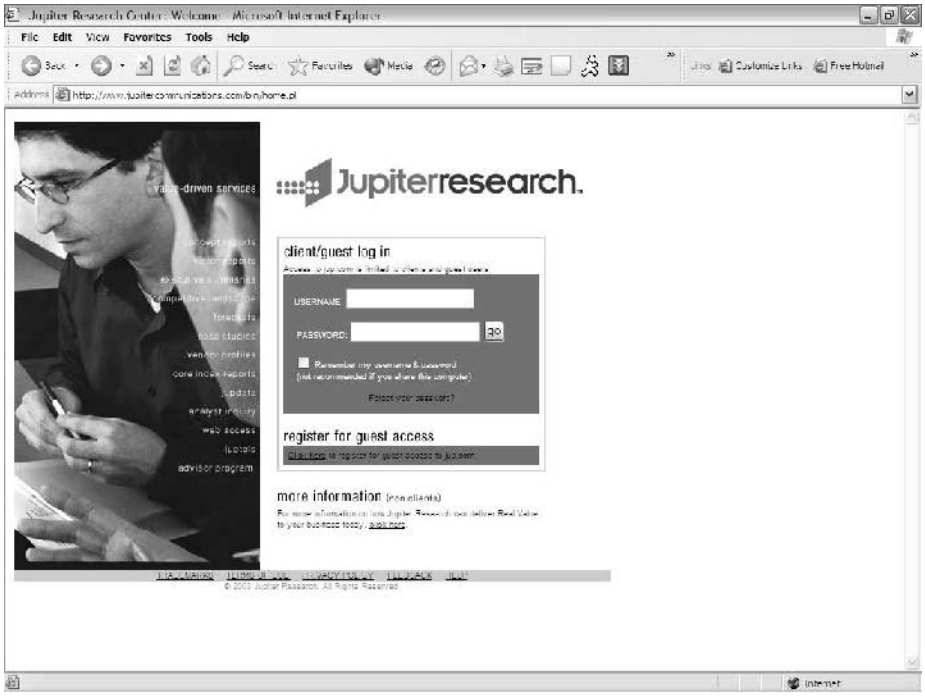


Figure 1-6: Jupiter Communications is one of the best sources of Internet facts.

Table 1-5 lists the URLs of some of the top Internet research firms.

www.it-ebooks.info

Table 1-5
Internet Research Firms

Company	*Web Address*
comScore Networks	www.comscore.com
Forrester Research	www.forrester.com
Harris Interactive	www.harrisinteractive.com
Jupiter Communications	jupitercommunications.com
Nielsen//NetRatings	nielsennetratings.com
Nua Internet Surveys	www.nua.ie/surveys
Statistical Research, Inc.	statisticalresearch.com
WebSideStory	websidestory.com
Zandl Group	zandlgroup.com

You can never really trust any statistics, no matter what the source, unless you know how those figures were arrived at—not if you're going to bet your future on them. Any reputable research company is glad to provide you with the details of the methodology they used in arriving at certain figures.



Neither the numbers nor the methodology are necessarily free. Most of these companies do issue some public reports but make their livings conducting custom research projects or selling the results of their own initiated studies. Often, the publicly released figures are only the tip of the iceberg. You can sometimes cut down on the cost of conducting a study by participating in an *omnibus study*, in which a group of companies each pay part of the costs while all share in the results.

Surveying site visitors

When it comes to learning about your particular site's visitors—or what they like—there's not much of a substitute for simply asking them. There are drawbacks, however, to the survey concept. The biggest problem is that people don't always respond truthfully for a variety of reasons, sometimes with the best of intentions, sometimes out of plain vanity. If you ask people whether they read tabloids or intellectual magazines, for instance, the vast majority of them say that they don't read tabloids. While this has proved to be true in many different surveys, tabloids outsell intellectual magazines by millions of copies, so somebody's not telling the truth. Remember, when devising your survey questions, that any time you ask people to admit something that's less than flattering to their self-image, you are asking them to respond less than truthfully.

www.it-ebooks.info

This phenomenon is widely recognized in the consumer research field, and a common and simple way around it was developed long ago. The trick is to ask other questions that help qualify the responses. The confirmatory questions should not be close to the sensitive one, but later, and scattered about within the survey. For example, you might ask visitors to choose their favorite columnists from a list. If they fail to select a single one from the intellectual magazines, you've got a red flag on their previous answer—why would they read those magazines if they didn't enjoy any of their writers? In that case, you should probably exclude that respondent's answers from your tally.

When designing a form for your Web site, however, you should bear in mind that you don't want to make it too long. People—especially Web surfers—do not have endless patience for answering zillions of questions, which limits your ability to mix in confirmatory questions.



If at all possible, resist the tremendous temptation to force your site visitors to fill out endless forms in order to access your site. You will drive away more people than you'll attract, if you don't avoid this temptation. You may think that you have the upper hand because you can withhold access, but visitors have the upper hand because they can refuse to participate.

Determining Your Approach

Before you take the first step toward actually putting your Web site together, you have to determine how you are going to approach your intended audience. You must first set your goals, decide how complex your site's technology will be, and, armed with that knowledge, set a realistic budget.

Focusing on goals

Every Web site, in order to be successful, must fulfill some sort of need for its visitors. The word "need" doesn't have to be interpreted too strictly. After all, how many people really *need* a $400 leather jacket? Or a handcrafted miniature car? Yet, there are sites that are successfully selling such items.

Your goals must be set by one key rule: The average person visiting your site should walk away happy. The key to making your visitors happy—and making them want to come back again and again—is to satisfy people's desires. Most Web site designers make the mistake of assuming that people only come for what they really have to have—or, more accurately, what the designer *thinks* people really need. That's a fatal error to make. On the Internet, people are still people and, as in the outside world, people are driven more by emotions than by logic.

Put yourself in the shoes of your potential site visitors and take a look at what you're offering. Does your site solve their problems (or, more accurately, their *perceived* problems)? Does it provide them with the information or products they're looking for? Does it do it all easily, without making your visitors work too hard to get what they want?

www.it-ebooks.info

Deciding on complexity levels

How deeply are you going to get involved in Internet technology? When we're talking "technology" here, we mean both the hardware and the software—not just on your end, but on your site visitors' end, too.

Living and dying on the edge

Internet technology is constantly changing, and this means that you need to make one specific choice—whether or not to keep your Web site on the leading edge of that change. The leading edge is often sardonically referred to as "the bleeding edge," because lots of people and companies have suffered tremendously by trying to stay up with it.

The reasons for this are twofold. First, leading-edge technology has much higher expenses. Second, nobody can guarantee which of several competing approaches will end up becoming the standard. What looks as if it's the most promising approach to accomplishing something can turn out to be unpopular in the end, regardless of its technical merits.

Our best recommendation is to hang back a little bit and not join the latest rush to the New Hot Internet Thing. But just for a little while. Just long enough to be sure it's the real thing and not just some fad that won't pan out in the end. As soon as you're sure the change is permanent and the technology is here to stay, move ahead with it. Let some other company do the bleeding, then move in fast enough to reap the rewards—if there are any to be had.

Staying deliberately obsolete

That said, there are good and solid reasons to deliberately ignore even a well-established Internet technology in favor of still older approaches. And those reasons are? Stability and demographic reach. As new standards for Web design and applications emerge, anyone on the leading edge of implementing them will quickly find that the latest Web browsers, despite anything their makers say, only partially support the standards when they are first introduced.

To further confuse the situation, the average Web surfer doesn't keep up-to-date with the latest Web browser. Thus, even if the newest versions of browsers finally get around to supporting the latest advance in its entirety, you'll still lose the audience share that uses old browsers if your site uses the latest ideas. If you have committed to following hot on the heels of the standards committees, your site had better be an experimental one and not meant to serve any practical purpose.



See Chapter 13 for more information on cascading style sheets.

The basic question you need to consider is whether or not you want to reach the vast majority of Web surfers—those who are using Web browsers that are, at least to some degree, obsolete. If so, you should consider using the lowest level of Internet technology that still achieves your basic goals.

www.it-ebooks.info

On the other hand, if your target audience is users of higher-level technology—you run a Web site that has nothing but tutorials on Macromedia Flash, for instance—you can confidently expect that the only low-tech visitors to your site will have wandered in by accident.

Establishing a budget

How much are you willing to spend on your site? This question, more than any other one, sets the practical limitations on what you're able to do.

If you're whipping up a personal site, the odds are pretty good that all you need to invest is nothing more than the amount that you have already put into getting online in the first place. All your needs are probably thoroughly covered by the basic amount of Web space that your Internet Service Provider (ISP) supplies as a basic part of their service. If the amount of Web space your ISP provides is inadequate, you can doubtless find plenty more available for little or no further investment.

You need to sit down and figure out exactly how much money you're willing to commit to your project if you're going for anything beyond that rock bottom minimum. If the money's going to come out of your own pocket, you have to take a hard look at how much you're willing to lose because every Internet startup is a gamble. Unless you're very rich or planning on a really small operation, you may have to put a substantial percentage of your wealth into the first year of your site's operation. Are you willing to risk giving up all your disposable income to try to make your dream come true? Half of it? Ten percent?

Whether your site is designed to generate an income or not, it still costs money to run. And, while there are exceptions to the rule, the best sites tend to cost the most money. Are you passionate enough about what you want to do on the Web to divert that money from all the other things you could do with it? If the answer is yes, then you stand a better chance of success, but if you're hesitant about committing, you may want to think twice—maybe even thrice.

Summary

No matter what you want to do on the World Wide Web, you're going to have to do some thinking and planning first. This chapter introduced you to the following concepts and topics:

- ✦ The kind of material you will present will determine who your audience will be.
- ✦ Understanding who your visitors are is critical to your success.
- ✦ It's often a good idea to let others test out the "bleeding edge" first, then follow a little behind the leader.

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 2

Promoting Your Site

In This Chapter

- Submitting to search engines
- Robots and spiders
- Customizing Web pages for search engines
- Blocking spiders
- Web directories
- Registering with major sites
- Issuing press releases
- Marketing with e-mail
- Linking with other sites
- Banner exchanges
- Using online banners
- Leveraging traditional advertising

So you have developed a Web site that is rich in content, interactivity, and chock-full of information and ways to link like-minded people together. But no one is coming to your site. The Web has some 350 million sites. And all these sites are pushing their content hard. But what do you have to do to attract visitors? Do you have to heavily promote your site?

Savvy sites employ a multitude of popular services to drive users to their sites. From search engines to ad banners to e-mails, cleverly used services can drive customers to any site no matter how large or small.

The most popular service used is the search engine, such as Google, where a customer can search a term and bring up a list of sites that meet the criteria. Web directories, such as Yahoo!, are similar in concept to search engines.

E-mail is often a more focused method of distributing information. Though often lambasted in the press as *spam*, the door-to-door salesperson of the Internet, the clever use of e-mail can often yield higher traffic *click-thrus*, the term for driving a user to a site, than any other online marketing tool. Along with e-mail, an equally rich method for driving visitors to a site is the use of banners. Thoughtfully designed banners that work in conjunction with banner exchanges—large groups of users who share ad space with each other—can provide a cost-effective and statistically accurate vehicle for driving sales. In the last couple of years online ads have matured. This has enabled the technology to increase its effectiveness.

As an old fortune cookie predicts, “The man who shouts from the top of the tree gets heard.”

www.it-ebooks.info

Exploiting Search Engines

Search engines have enjoyed a long history on the Internet. Search engines such as Lycos, WebCrawler, and AltaVista have been part of the Internet since the mid-1990s, although relative newcomer Google has certainly come out on top.

A search engine is a site that enables a user to search for information by entering a term, clicking a button, and enabling a database containing information on millions of Web sites to return a page of results in response to the query. Search engines immediately attract new Internet users for one simple reason: it is easy to find any Web site within a click or two.

Although Google is number one, some of the early search engines are still very popular: Lycos, Excite, and AltaVista are among just a few (see Figure 2-1).

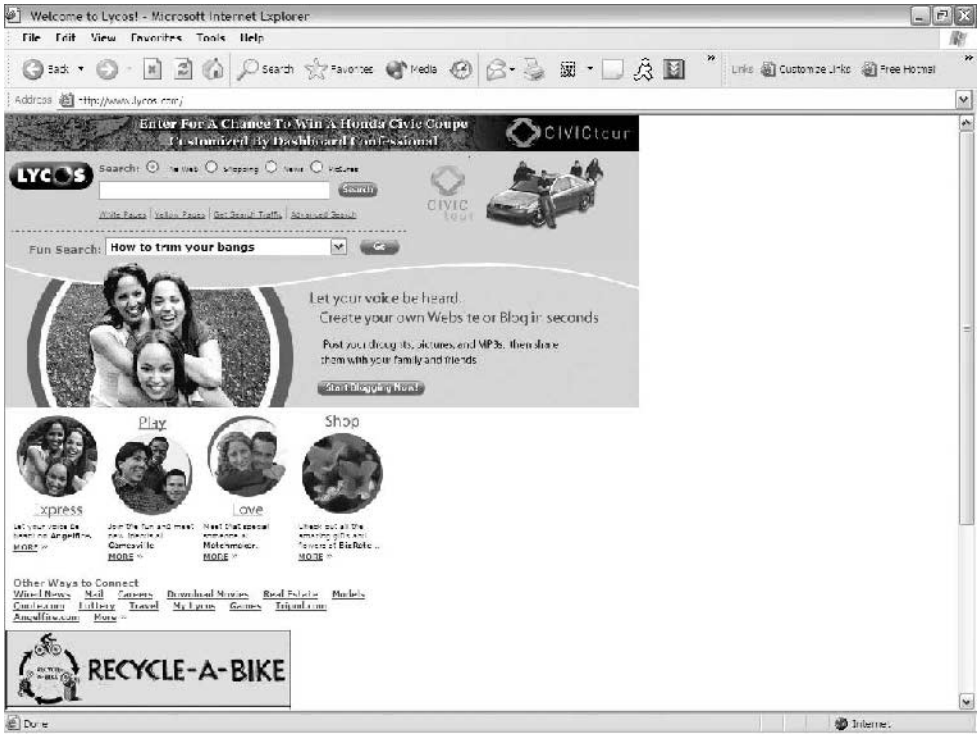


Figure 2-1: Lycos search site.



Lycos was one of the first search engines. The site now serves as a springboard to the Web.

Search engines are designed to be central aggregations of content. Most users, looking for information, start with search engines. Users search topics ranging from "brain surgery" to "VCR repair" to "Ritz crackers" (yes, thousands of Web sites are dedicated to the noble cracker).

www.it-ebooks.info

A search engine works by taking pieces of information on Web sites and storing it in a massive database. When a search is performed, the submitted search words are sent to a database that returns a set of records, matching the user's request, and generating a results page.

The information in the search engine database is collected by software called Robots and Spiders. The robots and spiders move throughout the Web from hypertext links on Web pages and return formatted information from these sites to the main search engine. The end result of this sort of action is information like that shown in Figure 2-2.

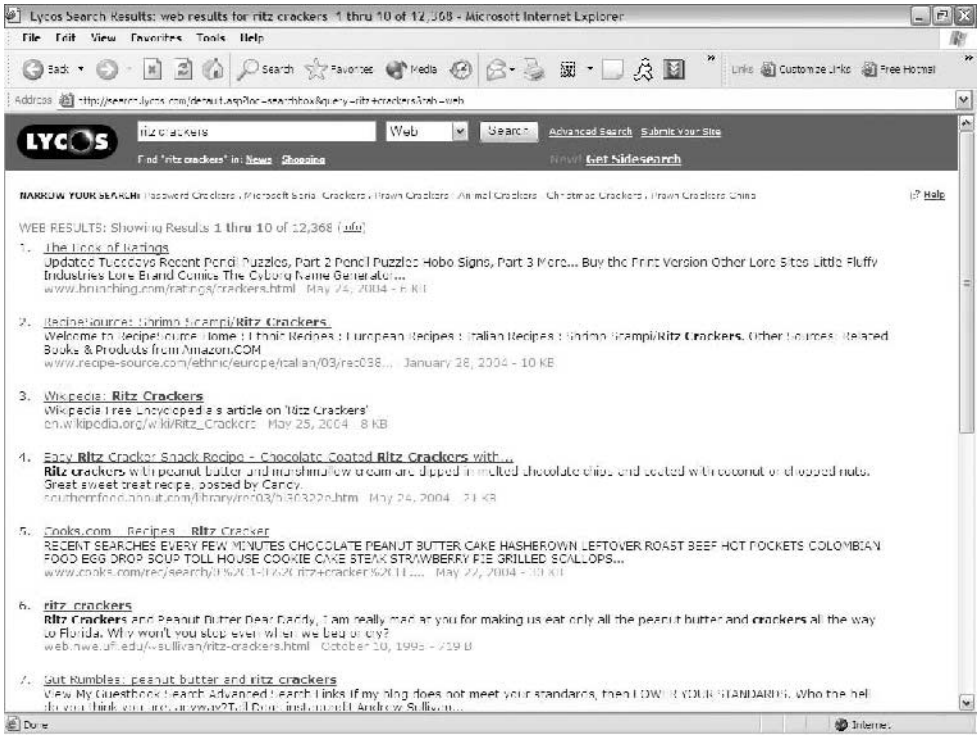


Figure 2-2: Lycos results page.



Typically, only 10 to 20 Web pages are listed on the initial results page for a search engine. The trick is to ensure that your site makes the results list!

Search engines share these commonalities:

- ✦ A link to a Web site that contains the searched word
- ✦ A description of the site
- ✦ The full URL to the Web site that has been returned

In a sweeping glance, a user can look down the list of results and click the link that makes sense. Brief information is key. If your site is listed but does not accurately define what your site is about, a customer may potentially glance over your listing.

www.it-ebooks.info

Each piece of information that is delivered in the results page is critical. The link must be meaningful, the description must be detailed, and the name of the Web site must be easy to remember.

Understanding robots and spiders

A search engine works by cataloging information contained on Web sites throughout the entire World Wide Web. Search engines cannot catalogue in detail all of the information on each and every site due to the massive amounts of data this would create. Search engines rely on software programs called *robots* or *spiders* that are launched onto the Web. Robots follow URLs from one Web page to another cataloging key information about each page. This information is sent back to the main search database.

If a robot or spider has not investigated your site, you will not be in the search database. A robot can be forced to visit your site by completing a registration form with a search engine.

To determine whether or not your site has been visited by a search engine's robot, you will need to do some digging around. Each and every Web server maintains a set of records called *logs*, which provide rich data about the type of people visiting your site. You can ascertain what type of Web browser a user has, how long they stayed at your site, and where the user came from. Logs also show if a robot or spider has been to your site. Software such as WebTrends can analyze Web server logs to see if a robot has visited a Web site.

Robots are always searching Web sites. If a robot visits your site, many methods are available to ensure that the most amount of information about your site is exchanged with the robot.

Submitting your site

All search engines require that a form be completed before a search is done. Each search engine has its own form. The following is a list of links to seven search engines:

- **All the Web:** `www.alltheweb.com/help/webmaster/submit_site`
- **AltaVista:** `addurl.altavista.com/addurl/new`
- **Ask Jeeves:** `ask.ineedhits.com/sitesubmit.asp`
- **Excite:** `https://secure.ah-ha.com/guaranteed_inclusion/teaser.aspx`
- **Google:** `www.google.com/addurl.html`
- **Lycos:** `insite.lycos.com`
- **Webcrawler:** `https://secure.ah-ha.com/guaranteed_inclusion/teaser.aspx?network=WebCrawler`

www.it-ebooks.info

Going to each of these pages ensures that a comprehensive registration form for each site is completed. This can be labor intensive. To make it easier, online services such as `www.submitit.com` will for a fee submit a site to over 400 search engines.

Keeping your site searchable

Robots search for key information about a Web site. The search engine uses this information to categorize your site; to determine how to display the site in the results page; and to seduce the customer into clicking through to the site.

Within the HTML on any Web site, you can add code to help make the site more search-friendly. The primary areas investigated by crawling spiders and robots are as follows:

- `TITLE` tag
- `META` tags
- `ALT` tag

TITLE

In many ways the most important tag on any page is the `<TITLE>` tag that appears at the top of each page. This tag is located within the `HEAD` section of the HTML page. It appears as this:

```
<html>
<head>
<title>This is where the Title for a Web Page is placed</title>
</head>
```

The `<TITLE>` tag is very important because when a search engine displays a results page the text in the `<TITLE>` tag is used as the link to your Web site. The link on the results page is underlined, which draws attention to it. Generally, a user glances down a list of highlighted links and chooses the most meaningful link.

When giving a page a title consider the content on the page. If the page is the front page to a site, do not call it `<TITLE>Home</TITLE>`. Add more meaning to the title such as the following:

```
<TITLE>Company XYZ Home: Where you will find the best
plumbing</TITLE>
```

This title now tells the user about the company name, where the page is, and a little about what the company sells.

The title for a page does have some limitations. Only 255 characters can be used, and HTML cannot be used to format the text. Wrapping a company name with the `<B>` tag to bold the text does not work. On the flip side of the coin, special characters, such as ? (question mark), * (asterisk), # (pound), and ! (exclamation

www.it-ebooks.info

mark) can be used. Some sites go to great lengths to use these characters to differentiate their titles from other sites. For instance, the following will work:

```
<title>^^^ ----!STOP! CLICK THIS LINK FOR SUPER SALES! OR CALL MORTY @
555 1212---^^^ </title>
```

Small amounts of extra information can make the difference between a user clicking through and ignoring the link.

META

<META> tags are a great tool for fine-tuning how the results page will display a site. As with the TITLE tag, Meta tags are designed for multiple purposes, but their most common purpose is to add content to a site.

META tags are located at the top of a Web page within the HEAD tag. There are two META tags that are frequently used by search engines: keywords and description. The following code shows two META tags in conjunction with the TITLE tag and their placement on the Web page:

```
<html>
<head>
<meta name="keywords" content="book, bible, publishing, Wiley, Web,
HTML, promotion">
<meta name="description" content="The Web Publishing Bible provides a user
with the expertise to build, execute, and promote a Web site">
<title>This is the title</title></head>
```

In the above code, the Keywords and Description META tags display slightly different information that is used by the search engine.

"Keywords" are used to categorize a site. Determining where to place a site depends on what words are used in the Keywords META tag. In the above example, the keywords book, bible, publishing, Wiley, Web, HTML, promotion are used. When a customer enters a search on any of these terms, this site will be presented. The more keywords you use to describe the content and theme of the site the higher the probability of a site being retrieved by a search engine.

One caveat is that the keywords META tag is limited to 255 separate words. Most robots and spiders ignore any more words.

The second META tag "description" is an area enabling a description of the site to be filled out. In the previous example, the "description" explains that the site is about Web Publishing.

The description META tag is not limited to 250 words. As such, it should be used to provide a clear description of the site.

ALT

The final tag that is often used by search engines is the ALT tag. The ALT tag is most commonly associated with the IMG tag. The following HTML code shows the ALT tag being used in the IMG tag:

www.it-ebooks.info

```
<img src="foo.jpg" width="417" height="163" alt="The ALT tag is
similar to the Description Meta tag and Title tags">
```

The ALT tag does a lot of double duty. For search engines, the ALT tag is another method of extracting information from a page. This is important because graphic files cannot be searched by robots or spiders, but adding the ALT tag enables a description to be associated with a graphic. The description can explain what the graphic file is and where it links to, as well as provide supplemental information about the content of the page. The search engine uses the ALT tag to catalog the site.

Use of the ALT tag is especially important for sites designed for aurally impaired Web surfers. Today, over 10 million people surf the Web with computers that are enhanced to "speak" Web pages. Clever use of the ALT tag can provide rich information about a Web site.

The ALT tag is not exclusively used by the IMG tag, but Image MAPS placed on top of graphics to create hot spots also leverage the ALT tag. The following code shows a graphic with an image map:

```
<img src="foo.jpg" width="417" height="163" alt="The ALT tag is similar to
the TITLE tag" usemap="#Map" border="0">
<map name="Map">
  <area shape="rect" coords="47,45,97,92" href="home.htm" alt="This links goes
to the home page" title="This links goes to the home page">
  <area shape="circle" coords="187,69,55" href="contact.htm" alt="Click here
for further information on how to contact us " title="Click here for further
information on how to contact us ">
  <area shape="rect" coords="286,38,332,103" href="resources.htm" alt="Click
here for further information on the site " title="Click here for further
information on the site ">
</map>
```

Each hot spot has its own link and ALT tag. A unique description is now associated with each and every hot spot through the use of the ALT tag. This information, in turn, is used to further categorize the site.

Extensive use of the ALT tag adds rich content on any given site.

Pulling it all together

The three sets of tags demonstrate how to add rich information to a site, but how does it translate to the search engine? Google clearly demonstrates the importance of meaningful information. Figure 2-3 demonstrates how all of this pulls together.



Use of the TITLE, META, and ALT tags combine to provide a meaningful results page.

The top of the page identifies the word "publishing" as the word being searched. The search has resulted in 12,300,000 entries, of which the top 1 to 10 are listed on this page.

The TITLE tag is used to create the link to each page.

www.it-ebooks.info

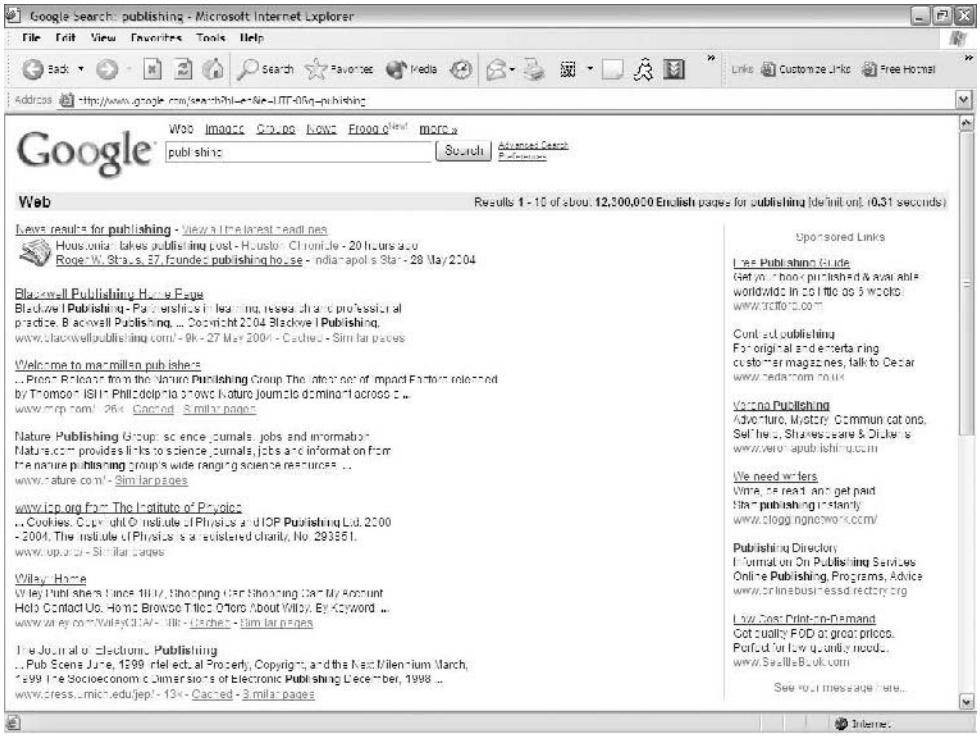


Figure 2-3: The Google results page provides meaningful information.

The second "description" is extracted from a Description Meta tag.

Textual content on the cataloged Web page and the `ALT` tags is used to create the description of the site.

Finally, the URL to the Web site is given along with the file size of the Web site.

Combined together, the user is presented with a rich set of information that will drive the decision on whether or not to click and view the site.

Ranking

Being listed or *ranked* among the top 10 results for any site is extremely important, especially when 8 million other sites are fighting for the same business. To a certain degree, this can be achieved through the use of `META`, `TITLE`, and `ALT` tags.

A number of services, like `www.webposition.com`, are available to let you know where your site is ranked.

While search engines can change your site's ranking, most charge to increase the ranking. Paying for higher ranking may be worthwhile, if a search engine drives a huge number of users to sites.

www.it-ebooks.info

Portals

Search engine sites are in the business of driving users to their service. In the past couple of years, many search engines, influenced by the massive numbers of users coming to their sites each day, have evolved to add additional tools and services such as e-mail, calendars, shopping, auctions, and online communities. The list of new services is ever increasing. These new sites are evolving to become central aggregates of content or *portals*. The depth of the tools and services available at search or portal sites is demonstrated in Figure 2-4 by the Excite site. You will notice that the Excite home page surrounds its search engine with rich content.

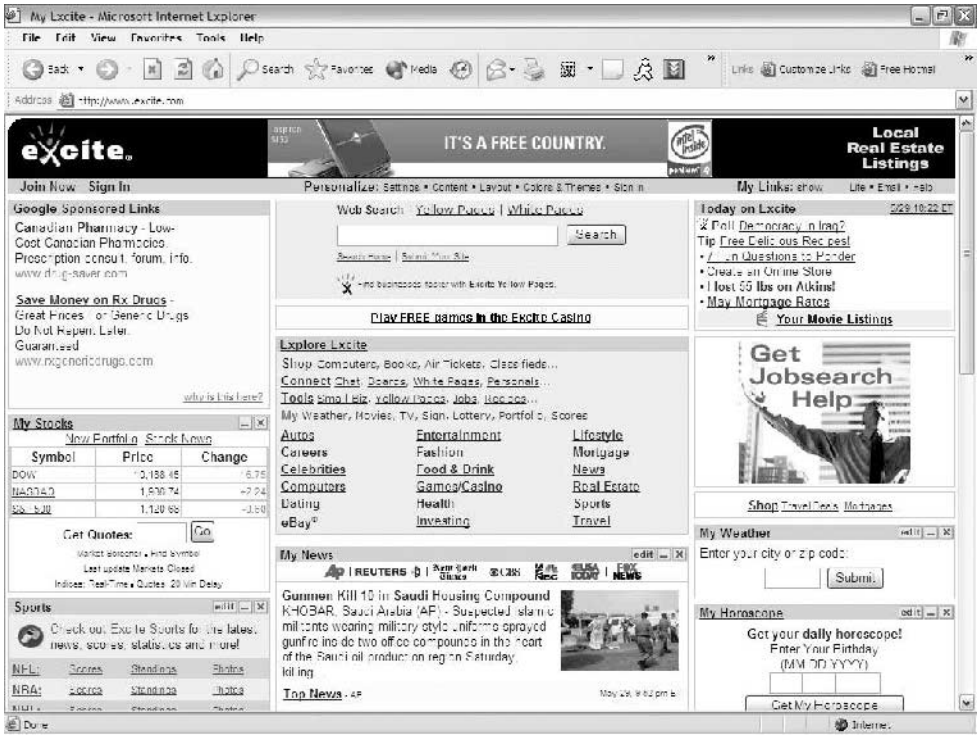


Figure 2-4: The Excite portal page surrounds its search engine field with content.



Portals provide engaging content, driving users to the site. Once at the site, the primary service is the search engine.

In providing a rich set of tools and services, search engine portals are more likely to encourage users to keep coming back. For this reason, it's important that search engine sites are highly promoted because when the results page for a search is presented, you want to ensure your site is there.

www.it-ebooks.info

Blocking search engines

A robot or spider shouldn't be able to search for every Web page ever created. For one of a hundred reasons such as the content of the page, or based on a company decision, you may want to block a robot or spider.

While many ways are available to block spiders and robots, the easiest method to block is through the use of the `ROBOTS` Meta tag. The tag is as follows:

```
<HEAD>
<TITLE>This page will not be indexed by a Robot or Spider</TITLE>
<META NAME="ROBOTS" CONTENT="NOINDEX">
</HEAD>
```

Placing this line at the top of a Web page stops most search engines.

But not all search engines employ this method of blocking spiders and robots. Another method is to use the `robots.txt` file.

The robots.txt file is placed in the top-most directory for a site. When a search engine arrives at a site it first looks to see if the `robots.txt` file is being used. In a Web browser, the link to the robots.txt file appears as follows:

```
http://www.companyname.com/robots.txt
```

The content within the `robots.txt` file determines how restrictive the search on the site will be. The following code prevents all robots from searching any part of the site:

```
User-agent: *
Disallow: /
```

In contrast, the `robots.txt` file can also enable any robot to search the entire site. This is done with the following code:

```
User-agent: *
Disallow:
```

Individual directories can be identified and blocked from robots:

```
User-agent: *
Disallow: /cgi-bin/
Disallow: /tmp/
Disallow: /private/
Disallow: /finance/results/
```

To fine-tune the `robots.txt` file, individual search engines can be excluded. In the following example, the Lycos spiders will not be able to search the site:

```
User-agent: Lycos
Disallow: /
```

www.it-ebooks.info

The opposite would be to only enable the Lycos spiders to index the site with no other search engines being allowed to do so:

```
User-agent: Lycos
Disallow:

User-agent: *
Disallow: /
```



The use of `robots.txt` is case sensitive. It must be in lowercase. If you use `ROBOTS.txt` it is simply ignored by any search engine.

Working the Web Directories

Often confused as search engines, Web directories are a must-have for any Web site promotion. A Web directory differs from a search engine in that it presents a customer with an ordered series of directories through which a Web site can be found. For instance, a Web directory may present a customer with a series of high-level categories, such as hobbies, shopping, and computers. Each category, when selected, exposes another set of categories that further defines the main directory. If hobbies is selected, a user could be presented with fishing, hunting, sports, reading, or music. Selecting the category music produces a results page similar to a search engine results page. The significant difference is the method in which the results page is found.

The most popular Web directory is Yahoo! The most important feature on the page is the Web Site Directory. Directories, however, force users to take a number of steps to filter their result. In the case of Yahoo!, you might click the category "News & Media." The linked page shows a set of categories within the broader scope of the Media.



The categorization style enables a user to fine-tune a Web site search.

It is not until you choose a subcategory that a set of links to actual Web sites is presented. By filtering through the categorized levels of the sites, a customer finds a Web site with arguably more accurate information. As Figure 2-5 demonstrates, the final results page may list only a handful of results. The benefit in the extra work is that each result holds information pertinent to the search.



A directory search yields more accurate links.

Ranking in a directory

Unlike search engines that rely on robots to find Web sites to populate a gargantuan database, directories rely on a human's touch. Each site that is submitted to a

www.it-ebooks.info

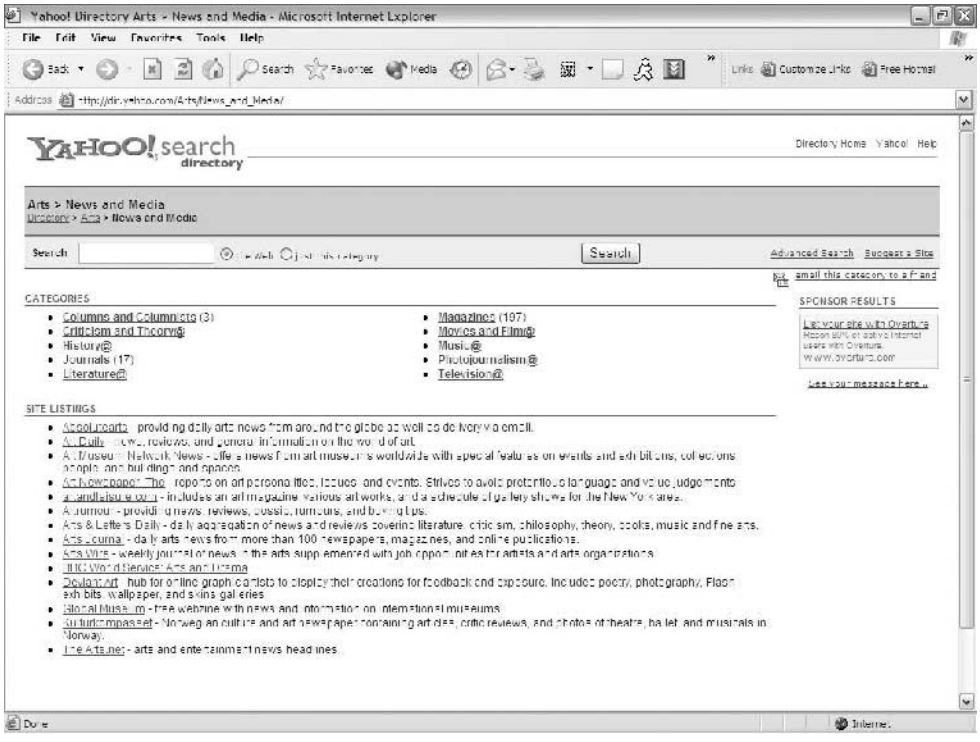


Figure 2-5: The final Yahoo! results page only yields few examples in this case.

directory is initially categorized by the person submitting the site, and then, in the case of Yahoo!, it is inspected by a Yahoo! employee. When this process is completed, the submitted site is placed within the directory.

Where and when a site is placed and ranked within the directory is important in driving traffic to your site. Many directories charge a fee for raising the ranking of a site. With many surfers unwilling to dig too deeply for information from a search engine, it may be worthwhile to pay to increase ranking.

Registering with the major sites

The registration process for a directory is slightly different than the process of registering with a search engine. The primary difference is that a site must be placed in a category. Often, two or three sets of categories are presented before a site can be fully categorized.

Yahoo! provides an excellent example on how to submit a site for registration to a directory. On each results page is a link to Suggest a Site.



If a directory is too broad, Yahoo! requests that it be further refined in order to place the site.

www.it-ebooks.info

Selecting the Suggest a Site button takes a user to Step One of the submission process.

Step one identifies the cost of partnering with Yahoo! if you run a business. This once-time free service now costs $299, though listings for private Web sites are still free.



Yahoo! is the most popular location on the Web. Placement within their directory is crucial for promotion.

With the largest Internet audience attracted to Yahoo!, a presence within their directory is critical for attracting business. In return, Yahoo! wants to guarantee that the level of content submitted to their site is rich and unique. By having a human analyze each site for content and originality, Yahoo! ensures that only elite sites are submitted.

The next form a user must agree to lists terms and conditions specified by Yahoo! to all directory members.



The terms and conditions enable Yahoo! to maintain a high level of quality for the content in its directory.

When all terms and conditions are met, the submission form requests title information, URL, and a description. On submission, Yahoo! checks that the site is not currently in the Yahoo! directory and that it is a unique site. Yahoo! now asks for contact information and finalizes the contract and payment.

Other directory services are taking their cue from Yahoo! and are now charging for their services. With an ever-increasing number of users accessing the Internet, these types of fees are likely to appear on more and more sites.

Listing on Free-For-All link pages

A slightly different version of a directory is a Free-For-All (FFA) site, which collects links to specialty sites. Unlike commercial sites, such as Yahoo!, FFA directories are often run by hobbyists. Guests are encouraged to leave a link to other sites on the same topic as the hobbyist's site. This, in turn, begins building a database of links.

FFA directories are very cost effective. They are free! The significant downfall comes with the return on investment. The traffic to your site from a link on a FFA directory may not be as heavy as a mainstream directory or a search engine.

Issuing Press Releases

Placement in search engines and directories is the first part of an online marketing campaign. As in the real world, some old standbys such as a press release can have Internet twists.

www.it-ebooks.info

Reuters and other press companies make their business from broadcasting press releases. Reuters has built a global reputation for high-quality, immediate information. Reuters charges for its collection of press releases, which are submitted to them from companies and news agencies. Reuters allows people to subscribe to the service. Through charging a fee, Reuters places a higher value on the content over free news services. More customers are likely to notice a press release and read it than wait to see it reported on CNN.

The Wall Street Journal is another site that has built a successful business model around charging for press releases that appear on its Web sites before reaching its printed paper or other sites (see Figure 2-6).

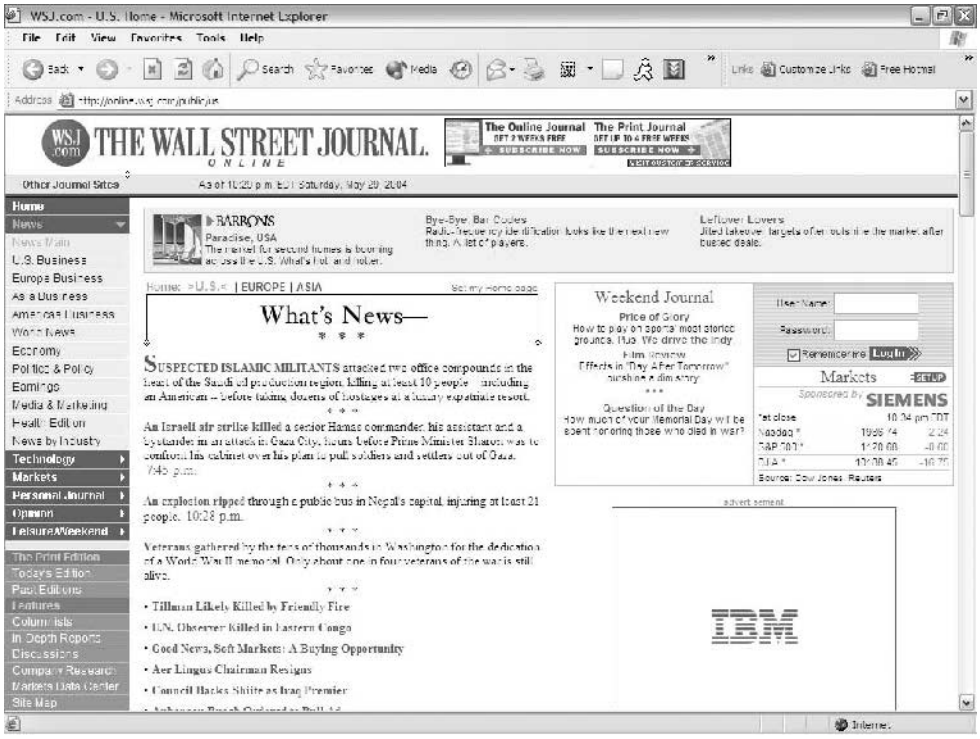


Figure 2-6: The Wall Street Journal is another example of a successful information site.



The Wall Street Journal places value in its news content by charging for it.

A company launching a new Web site or product should accompany its launch with press releases. Not only does this make for good news, but it also provides rich information for investors.

www.it-ebooks.info

Saying the right thing

The site is built. It has been posted to all of the search engines and directories. Now, customers are starting to come to your site. As any salesperson will tell you, the hardest *sell* is to a customer who has already purchased from you. If you can persuade them to buy twice, then the third, fourth, and fifth sales are much easier.

In the computer-centric world of the Internet, making the repeat sale is no easy task. Selling often requires two people discussing a product—one selling the product, the other buying it. Because Internet customers do not have this human interaction, they can experience a sense of anxiety. Did I buy that book? Will the company charge my credit card correctly? Did my mom receive the flowers I sent?

By anticipating the anxiety and proactively working with customers, you can develop loyal relationships.

The most effective method of actively reducing customer anxiety is through the use of e-mail. Even though it's an old Internet tool, it is still the most effective and widely used Internet service. Every customer coming to purchase a product from your online store has an e-mail address. These addresses must be collected during the purchasing process. When a customer presses the "submit" button to purchase products from a site, an e-mail must be immediately issued to the customer, telling the customer that the transaction has been successful, how much has been spent, and what has been purchased. If the product is to be shipped, an additional e-mail should be submitted to let the customer know the day the product shipped. UPS, FedEx, and many package services have Web sites that can track packages as they move around the world. Generally, these services are free. Linking to these sites in any confirmation e-mail enables a customer to track their own packages.

The key is to communicate with customers immediately. This reduces anxiety and increases consumer confidence. The customer is likely to return to make additional purchases in the future.

Finding e-mail addresses

It can't be denied that the strength and success of e-mail is its capability of penetrating directly into any user's in-box. It is extremely easy to write an e-mail that can be instantly sent to millions of people. But how do you know whom to target with an e-mail?

Many companies supply e-mail distribution lists to millions of users for less than $1,000. Sending unsolicited e-mail to a large number of users is referred to as spamming. Spam, the official term for e-mail sent blindly to massive e-mail lists in the hopes of procuring sales, is not a good method for communicating with potential customers. Spam is the junk mail of the Internet, and most users rarely read it and seldom act on it.

Meanwhile, e-mail can be effectively used when customers request it. For example, during registration, asks users to mark a check box next to a question that asks them

www.it-ebooks.info

whether or not they would like to receive additional information. Users who select this option expect to receive information. In this way, you can generate a list of loyal readers.

Formatting the e-mail—to use HTML or just plain text?

Today, e-mail offers much greater opportunities than ever before. Many companies are no longer just sending plain text e-mail messages but designing rich and engaging e-mail messages written in HTML. HTML formatted e-mail is supported by most of the popular e-mail clients such as Microsoft Outlook and Eudora. Certainly, free e-mail services such as Hotmail, support HTML formatted messages. When sending e-mail, ensure that you include instructions at the bottom of the e-mail on how to unsubscribe from the mailing list. This provides a user with the control over what e-mail is sent to them. If they do not want to receive e-mailers, they can unsubscribe.

Setting Up Reciprocal Links

Partnering is how businesses grow. The same happens on the Web. Web sites that offer similar or supporting services and products that complement your offering can help drive business to your site. The value of the two sites can be immediately increased by partnering, offering a link to the other site and, in return, having the site link to yours. For example, a customer buying software from a site that links to another site for training on the software is receiving a fuller and more gratifying shopping experience.

As with any partnership, understanding what you are getting into is the key to success. Before beginning any partnership, you must investigate the potential partnering site. Use a checklist, as the following one:

- ✦ Do the products or services enhance and support your current products or services?
- ✦ Are the same standards of design used on the site, and is the site similar in style to your own site? Using similar styles tells the customer that both sites have discussed and proactively agreed to work with each other.
- ✦ Is the customer service to the same standards as your own site?
- ✦ Does the potential partner site currently link with any sites that are considered competitors?

If all of the criteria match your needs, then begin by approaching someone at the company you want to partner with.

Evaluating Banner Exchanges

Web banners and ads are a mainstay of the Internet. Large entertainment sites are littered with banner ads. Why? Because they drive users to sites.

www.it-ebooks.info

Today, banner ads appear on almost every site. The success of a banner is measured by a user clicking the ad, going to the advertised site, and making a successful purchase. This is known as the *click-thru rate*. Most banners have a very low click-thru rate of 0.04 percent. Depending on how many people hit a site determines how much to charge.

The following list demonstrates some of the typical monthly charges for online ads if the host guarantees 500,000 placements of the ad per month:

- Search Engines—$20,000 to $50,000
- Keyword Advertising—$40,000 to $70,000
- City Guides — $20,000 to $80,000
- Top 100 Web sites—$25,000 to $100,000

With so much money riding on online advertising, the ad itself must sell.

Building the banner

What the banner looks like, how it is worded, and how it engages a user contribute to the final success of any banner.

To bring some standardization, the Internet Advertising Bureau (`www.iab.net/`) has defined widths and heights for different styles of banners. The most common banner, the Full Banner, stretches across the top of the Web page and is 468 pixels wide by 60 pixels high.

Other sizes of banners that can be used are

- 392 × 72 pixels (full banner with vertical navigation bar)
- 234 × 60 pixels (half banner)
- 120 × 240 pixels (vertical banner)
- 120 × 90 pixels (button 1)
- 120 × 60 pixels (button 2)
- 125 × 125 pixels (square button)
- 88 × 31 pixels (micro button)

Almost all Web sites that place banners on their sites conform to these standards.

Design do's and don'ts

Creating the perfect banner size is one thing, but now the banner must be designed to be effective on the site.

The default graphic format for most banners is GIF. The format is limiting in color depth, only 255 colors, but it makes up for this limitation with the capability to be animated. Though not elegant, several GIF images can be placed on top of each other

www.it-ebooks.info

to form a simple animation. Nielsen//Netratings (`www.nielsennetratings.com/`) track the success of different sizes of online ads as shown in Figure 2-7.

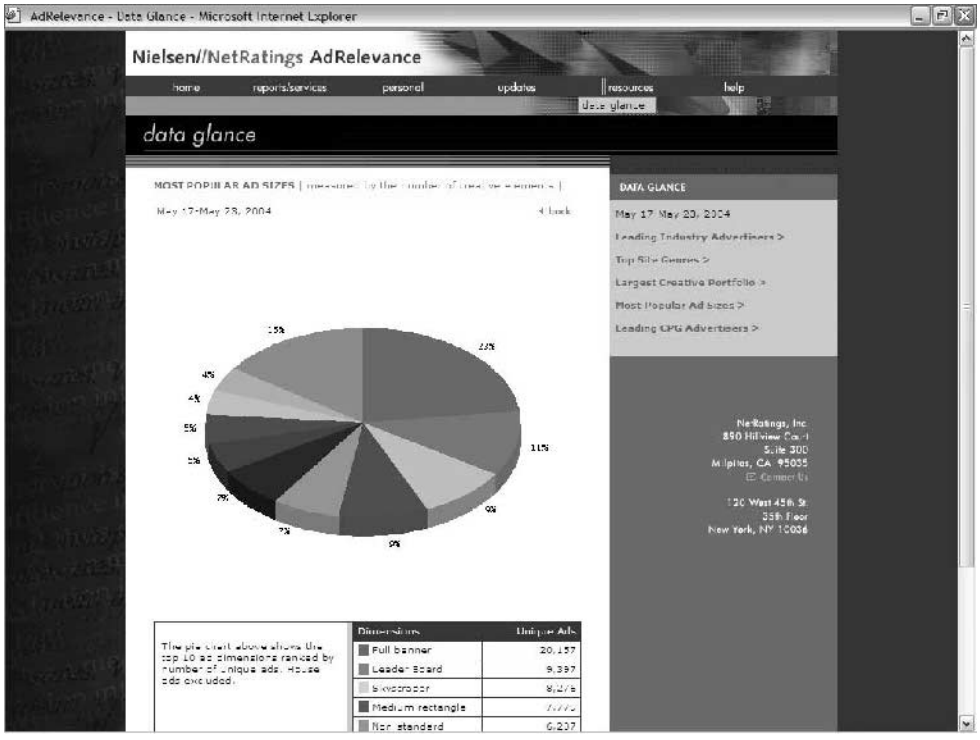


Figure 2-7: Nielsen//Netratings track the success of online ads.

Another common feature on all banner ads is the inclusion of the words "click here." These words provide the consumer with an action to perform.

Multimedia ads

The days of the animated GIF image are numbered. More efficient and elegant media formats have matured over the last few years to increase the tools available to agencies for use.

The most interesting format to emerge is the Macromedia Flash format. Flash is an animation format that can be scripted to interact with the user. Flash is engaging for several significant reasons. First, it is capable of bringing broadcast quality animation over low-speed Internet modems. Second, its file sizes are smaller than animated GIFs (this enables the ad to start playing faster). And third, Flash boasts a user base of some 94 percent of the entire Web surfing population.



Macromedia Shockwave uses Flash as the main media format to produce slick, elegant ads.

www.it-ebooks.info

Other media formats, such as video and audio, are being developed. Within a few short years, animated GIFs will become part of Internet history.

Banner Exchanges

Where and how to place an ad on a page can be almost an art form. To help reduce this potential headache, a number of Banner Exchanges manage the placement of banners on millions of sites. An Exchange places banners on sites that relate to the content of the banner; for instance, a gardening site will receive banners on lawn equipment.

Each Exchange functions similarly. A customer can buy advertising space by category and placement. Each Exchange supplies statistics outlining the success of a banner. Future placement of the ad and the design of the ad should be influenced by this information.

Traditional advertising

With the increasing attention garnered by Internet and electronic advertising, it can be too easy to forget about more traditional means of advertising. Cable, magazine, and billboards can drive stunning results.

Unlike the Internet, more traditional forms of advertising do not have the exact empirical data online banners offer. There are no charts or graphs to measure click-thru rates. This does not diminish the effectiveness of advertising. Knowing who your audience is determines what type of ads to run. Running billboards, radio, and local cable ads may be a cost-effective method of reinforcing your brand name to offline users.

Yes, the Web store is, by nature, located online—a banner makes any Web site only one click away. However, other methods of advertising can help build brand equity and drive traffic to any site. A subtle mixture of both provides the greatest impact for the widest range of audience.

Summary

It doesn't much matter how good your Web site is if nobody even knows that it exists. This chapter introduced you to the following concepts and topics:

- ✦ Register with the major search engines and Internet directories as a matter of course.
- ✦ Avoid at all costs using spam—unsolicited commercial e-mail messages.
- ✦ Reciprocal links are one of the key methods of driving visitors to your site. Deliberately seek out similar sites and ask their Webmasters for permission to set up links between your own site and theirs.
- ✦ Banner exchanges, properly used, can bring a lot of new traffic to your site.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 3

Basic HTML Building Blocks



In This Chapter

Grasping the classic HTML structure

Understanding the basic elements

Adding flexibility with attributes

Adding text

Understanding the types of authoring tools

Choosing your tools

Going beyond the basics

Working with specific tools



HTML is the heart of the World Wide Web. But, these days, with so many visual tools available for building Web pages, why should anyone even bother to learn anything about HTML? The basic answer is that there's no way around it. Despite the claims of some software publishers, there is just no way in the world to build a significant Web page—let alone an entire Web site—without understanding something about what's going on under the hood.

You will find that, whether you're typing your HTML code by hand in a simple text editor or filling in glitzy dialog boxes in an expensive page creation program, you're dealing with the same things. If the dialog box has a nice, neat interface that enables you to enter the "source of the image," you're still doing the exact same thing as a hand-coder who is typing in a value for the `SRC` attribute of the `IMG` element. The fancier programs do make things quite a bit easier—but only if you know what you're doing to begin with. If you don't know at least the basics of how a Web page is made, then all the bells and whistles in the world won't help.

Grasping the Classic HTML Structure

The basic structure of nearly every Web page can be summarized by this HTML code skeleton:

```
<HTML>
<HEAD>
<TITLE>
</TITLE>
</HEAD>
<BODY>
</BODY>
</HTML>
```

www.it-ebooks.info

This structure is defined by *elements*. The HTML element is the container of containers. Beginning with the <HTML> start tag and running until it reaches the </HTML> end tag, it contains all the other elements on a Web page, but nothing contains it. It has no attributes.

The HEAD element, which runs from the <HEAD> start tag to the </HEAD> end tag, contains basic information about the Web page, none of which end up in the visible display in a Web browser. The TITLE element goes into the HEAD element and contains the title of the Web page.

The BODY element is where all the visible content of a Web page goes. It contains such things as text, images, and tables.

Understanding the Basic Elements

When you visit a Web page, your Web browser reads the code, translates it into a displayable image, and puts that image on your screen. The browser is translating a set of instructions that tell it what the content and layout of the page are. At its most basic, that set of instructions is a group of *elements* and the content of those elements. An element is an item, such as an image or a paragraph, that is used to specify what a particular part of the page contains or looks like.

Generally speaking, element names are easily recognizable and easy to remember. A paragraph, for example, is represented by the P element; an image is represented by the IMG element; a table is represented by the TABLE element.



Element names are traditionally written in uppercase. However, they don't have to be because HTML is not case sensitive. As far as a Web browser is concerned, you can write them in uppercase, lowercase, or a combination. Many Web designers have developed the new habit of using lowercase for HTML elements, since that is how XML does it.

Defining elements with start and end tags

Elements, of course, have a beginning and an ending. In HTML, these are shown by the *start tag* and the *end tag*. The beginning of a paragraph is identified by the <P> start tag and its end is identified by the </P> end tag. The start tag is always the element name surrounded by angle braces and the end tag is the same, but contains a slash.

Just to make things a bit challenging, not all elements have end tags—the <IMG> tag is one such. Some other elements, such as the BR (line break) element, don't always require them. It's good practice, however, to include an end tag if one exists. This not only helps you to clarify exactly where things begin and end, but it also helps to build forward compatibility with the newer forms of markup languages such as XML.

Nesting elements via container relationships

The most critical thing to understand in HTML is that nearly all elements are *nested,* or contained by other elements. The following HTML code example illustrates how this works:

www.it-ebooks.info

```
<P>An element that is inside another is said to be <I>contained</I> within
it.</P>
```

This HTML code simply creates a paragraph in which the word *contained* is italicized. The `I` element, which does the italicizing, is completely inside the `P` element, so the paragraph element contains the italicization element. The word *contained* is, for that matter, the content of the `I` element.

Everything between the start tag of an element and its end tag is the content of that element.

Structuring pages with block-level elements

Block-level elements are those that automatically create a line break after them. For this reason, none of them can possibly be on the same line as any other block-level element. Some of the elements discussed in this chapter that are block level include heading elements, tables, paragraph elements, and horizontal rules.



This doesn't mean that you can't put the code for block-level elements on the same line of text in a text editor. However, the results, when displayed in a Web browser, won't be on the same line.

Figure 3-1 shows several block-level elements.

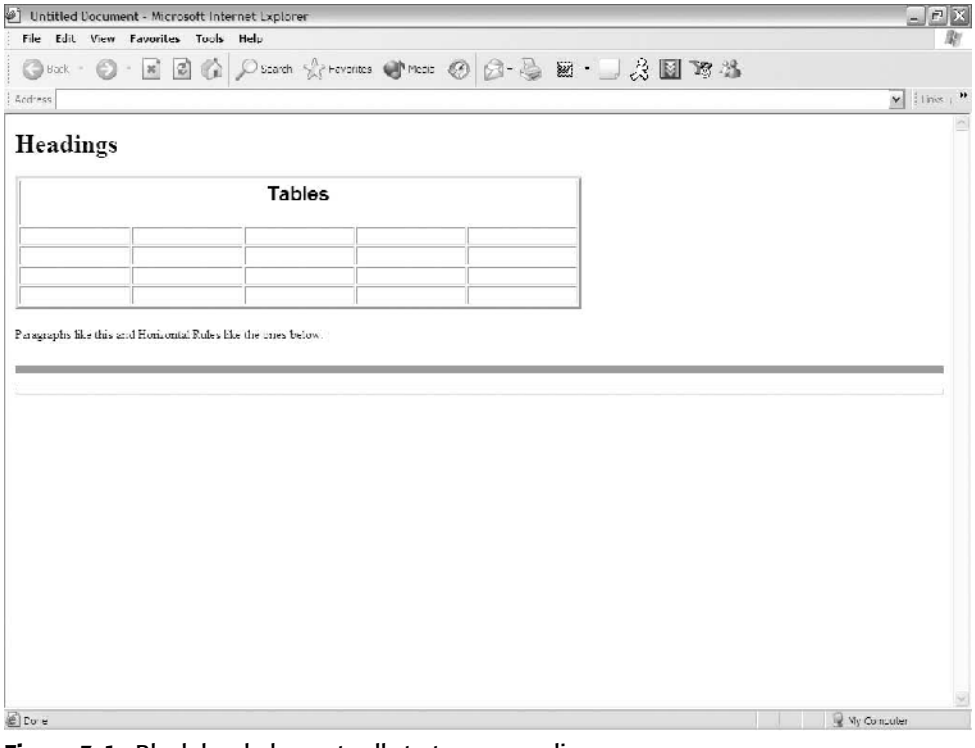


Figure 3-1: Block-level elements all start on a new line.

www.it-ebooks.info

Fine-tuning with inline elements

Inline elements are those that don't create a line break after them. They're sometimes called *text-level elements* because many of them deal with the appearance of textual characters. Among the most commonly used are the `B` (bold) and `I` (italic) elements. In the following HTML code, the `B` element is an example of an inline element.

```
<P>This text is <B>bold</B>.</P>
```



Inline elements are always contained within block-level elements, never the other way around.

One element that many people are surprised to learn is an inline element is the `IMG` element. It is actually possible to put it right in a line of text just as though it were just another word. The reason for this is that images are often used together with words in a variety of ways that would be impossible if `IMG` were a block-level element.



See Chapter 8 for more information on images.

Adding Flexibility with Attributes

Although the basic elements that HTML supplies provide some pretty powerful tools, they are capable of modification via *attributes*. Sometimes, an attribute isn't optional but required to make the element work. The `src` attribute of the `IMG` element, for example, is an absolute necessity because it specifies the source of the image file to be displayed.

Working with common attributes

Some attributes are common to all elements—the core attributes, i18n (international) attributes, and events. The following tables show these attributes and their official definitions from the HTML standard.

The core attributes in Table 3-1 are mostly used for working with style sheets, although the `id` attribute is also often used with JavaScript.

Table 3-1
Core Attributes

Attribute	*Description*
id	Document-wide unique id
class	Space-separated list of classes
style	Associated style info
title	Advisory title

www.it-ebooks.info



See Chapter 13 for information on style sheets. See Chapter 16 for information on JavaScript.

The i18n attributes in Table 3-2 deal with internationalization issues, specifying the (human) language of the Web page and which direction the words are written in.

Table 3-2
i18n Attributes

Attribute	*Description*
lang	Language code
dir (ltr|rtl)	Direction for weak/neutral text

The event attributes in Table 3-3 recognize when various things happen to an element, such as a mouse pointer moving over it. They are strictly for use with scripting languages such as JavaScript and have no other function.

Table 3-3
Event Attributes

Attribute	*Description*
onclick	A pointer button was clicked
ondblclick	A pointer button was double-clicked
onmousedown	A pointer button was pressed down
onmouseup	A pointer button was released
onmouseover	A pointer was moved onto
onmousemove	A pointer was moved within
onmouseout	A pointer was moved away
onkeypress	A key was pressed and released
onkeydown	A key was pressed down
onkeyup	A key was released



See Chapter 16 for information on using events.

www.it-ebooks.info

Using specialized attributes

In addition to the attributes that are common to all elements, most elements have their own specialized attributes as well. For example, a heading element (see the section "Heading elements" later in this chapter) has an `align` attribute that can be used to set whether it is aligned against the left or right margin, or in the center of the page. More than a dozen other elements also have an `align` attribute, but not all elements do because the `align` attribute can't be used with all elements.

Accepting defaults and setting values

In most cases, if you do not specify a value for a particular attribute, a default value will be used instead. The default alignment for heading or paragraph elements, for example, is left alignment unless you say otherwise.

In some cases, a default value doesn't exist—the framers of the HTML standard could not possibly guess what image file you wanted to use for the source of an `IMG` element, for instance.

With some attributes, the HTML standard doesn't specify a default value even where it might be useful to do so. A horizontal rule, for example, does not have a default thickness, and each Web browser renders them a little bit differently as a result.

Values are specified for attributes in this manner:

```
<ELEMENT attribute="value">
```

For example, to specify that the background color of the `BODY` element should be green, you would write this HTML code:

```
<BODY bgcolor="green">
```

Adding Text

The `P` (paragraph) element, already mentioned in a few examples, is the most popular way to put text on a Web page. It's a block-level element that separates different blocks of text from one another. The following HTML code shows how three different paragraphs would be coded:

```
<P>This is the first paragraph.</P>
<P>This is the second paragraph.</P>
<P>This is the third paragraph.</P>
```

Figure 3-2 shows how these paragraphs are displayed in a Web browser.

In addition to the `P` element, however, there are other ways to enter text, chief among them the heading elements.

www.it-ebooks.info

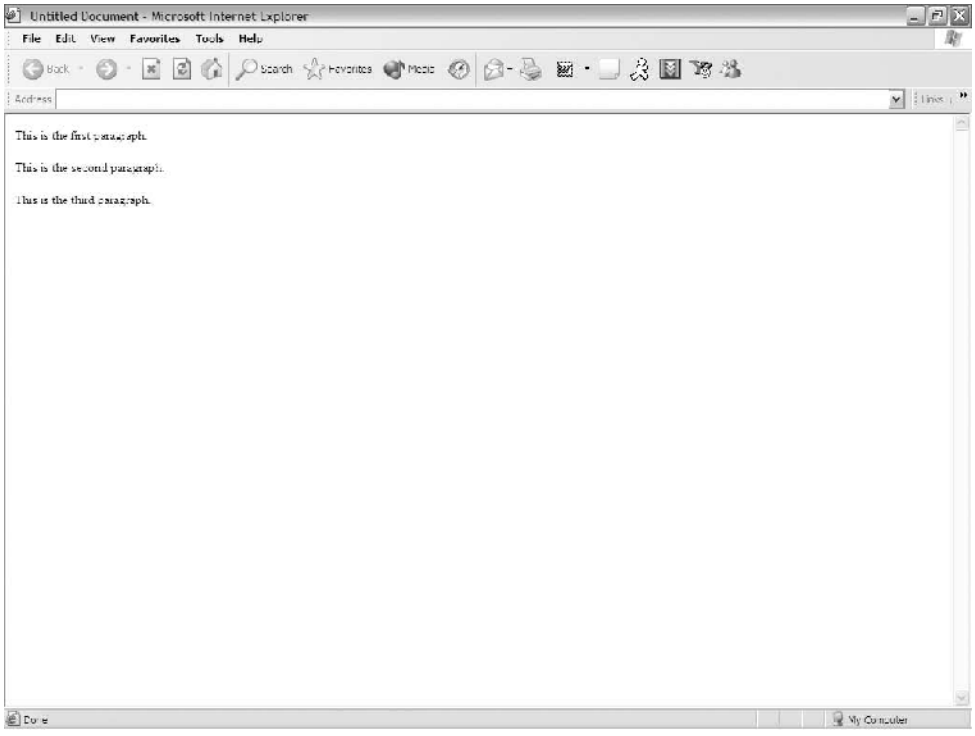


Figure 3-2: P (paragraph) elements contain text.

Heading elements

Heading elements come in six sizes, ranging from H1 (the largest) to H6 (the smallest). Their purpose is to introduce a new section and differentiate it from the text that preceded it. Figure 3-3 shows the six heading sizes.

Although heading sizes smaller than H4 are rarely used because they are small enough to be confused with normal text, they do have an important function. For blind Web surfers, the size of the lettering is of no consequence, but the structure provided by the heading type is critical. Although they cannot see the visual layout, the programs they use to surf the Web do specify which heading levels are present.

Adding line breaks

Text on a Web page automatically wraps down to the next line when it encounters the right margin. You can, however, make it do so at any point by using the BR (line break) element. In the following HTML code, the second line will start right after the word “way”:

```
<P>This line will not reach all the way <BR> to the edge of the
screen.</P>
```

www.it-ebooks.info

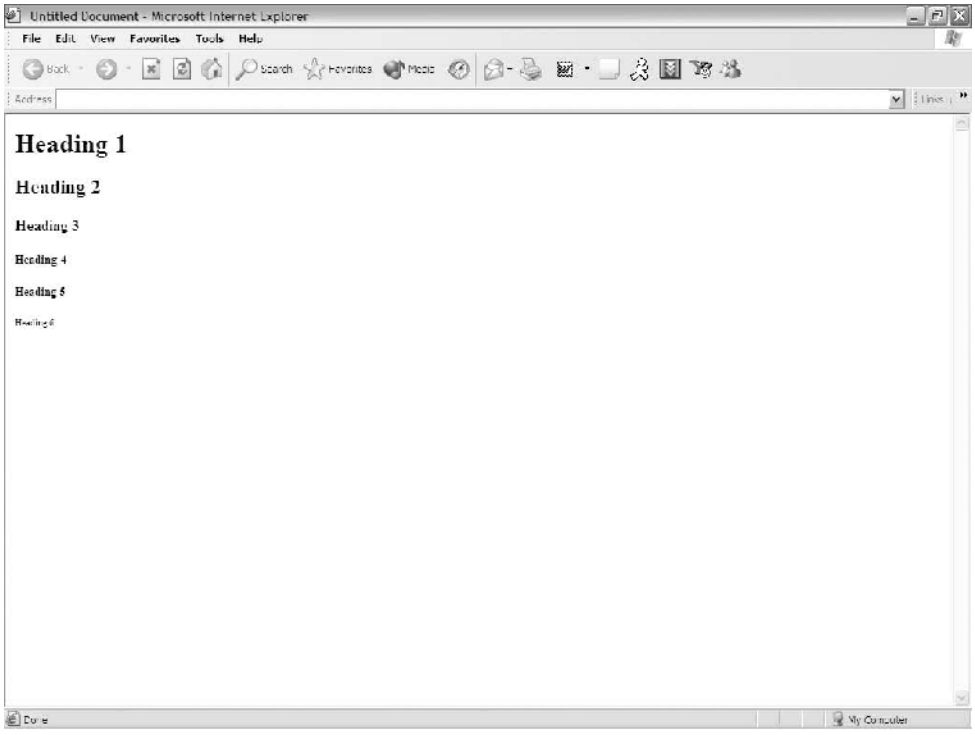


Figure 3-3: There are six different heading sizes.



The `BR` element has no end tag.

You could achieve a similar effect by using two `P` elements, but the effect is not identical. If you were to use the following HTML code, you would have two lines that were farther apart than with the `BR` element.

```
<P>This line will not reach all the way</P>
<P>to the edge of the screen.</P>
```

Figure 3-4 illustrates the two approaches and how their spacing differs.

Dividing with horizontal lines

You can add horizontal lines to your Web pages with the `HR` (horizontal rule) element. It has four specialized attributes: `align`, `noshade`, `size`, and `width`. The first, of course, sets how it is aligned, the choices being `left`, `right`, or `center`. The `size` attribute sets how thick the line is, and the `width` attribute sets how wide it is. I'll deal with `noshade` momentarily.

www.it-ebooks.info

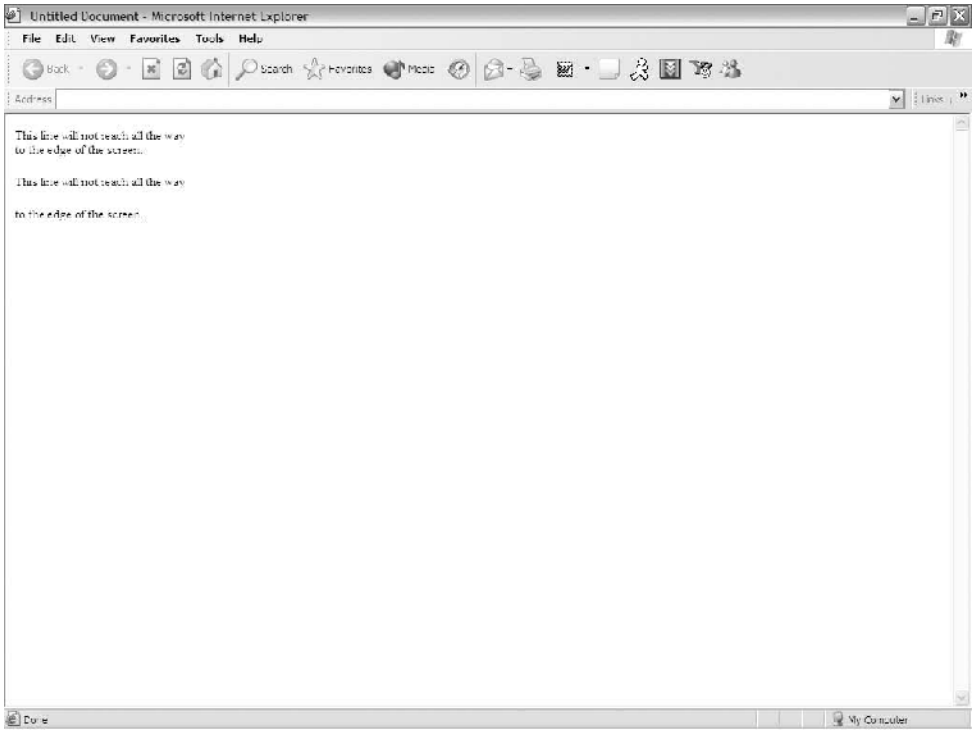


Figure 3-4: The `BR` element produces a closer pair of lines than two `P` elements do.

To add a horizontal rule that takes up half the width of the Web page and is aligned to the right side of it, you would use the following HTML code:

```
<HR align="right" width="50%">
```

Figure 3-5 shows the results.

The default horizontal rule is 100 percent of the screen width, and its default alignment is centered, which is kind of silly because if the line stretches all the way across the page, then it doesn't matter what kind of alignment it has. It's only when the horizontal rule is less than the screen width that you can tell any difference from the alignment setting.

The thickness of the line, specified with the `size` setting, doesn't have a default value, so it's up to whatever Web browser is used to view the page—unless you specify a size. For most purposes, a size of somewhere between 3 and 6 pixels is good. To add a thickness specification to the preceding HTML code, you'd do this:

```
<HR align="right" width="50%" size="6">
```

Figure 3-6 shows several different sizes of horizontal rules.

www.it-ebooks.info

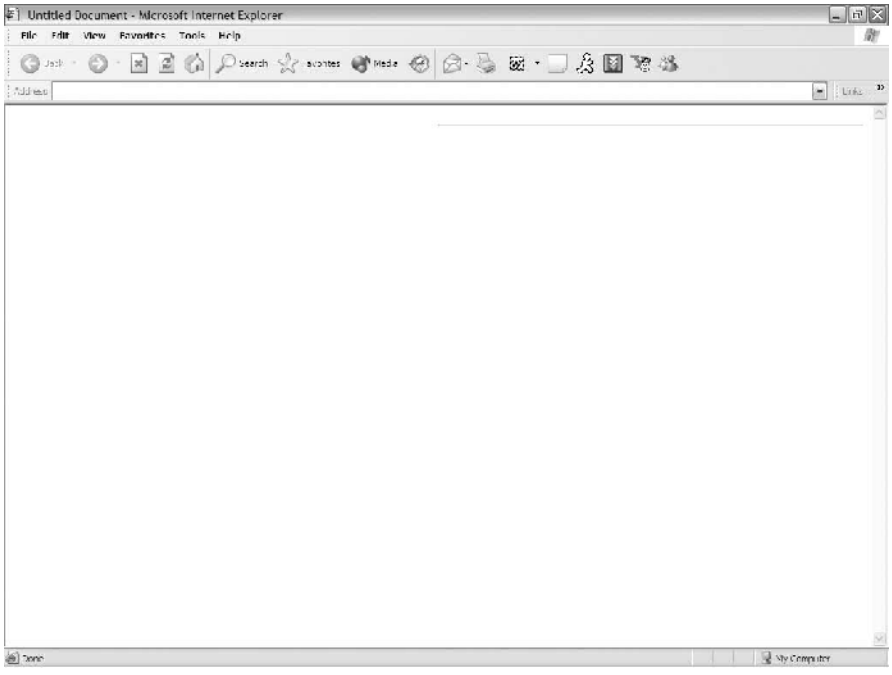


Figure 3-5: A basic horizontal rule, half the screen size and right-aligned

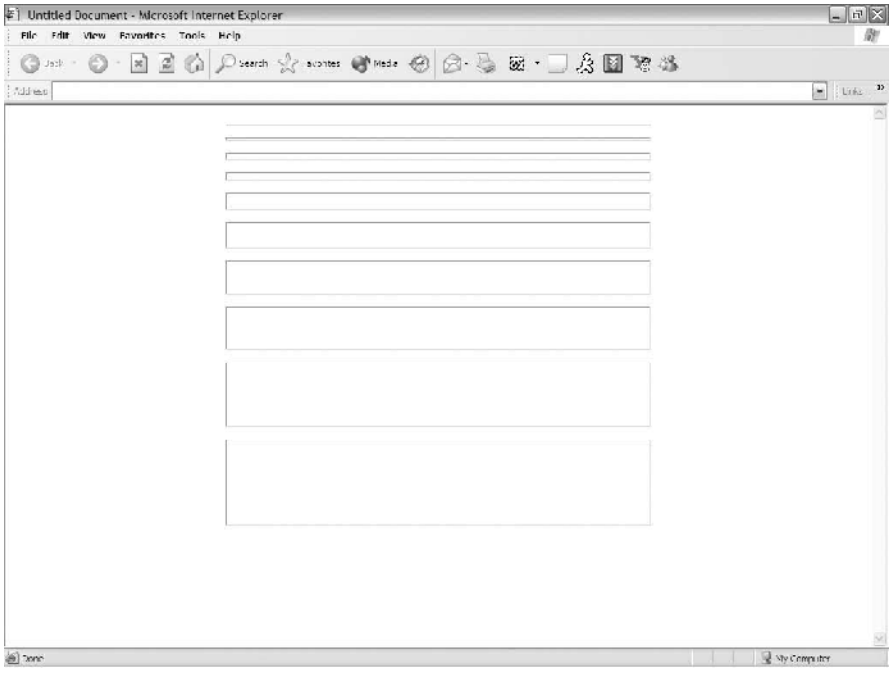


Figure 3-6: The `size` attribute enables you to control the thickness of horizontal rules.

www.it-ebooks.info

The noshade attribute has a bit of a confusing name, considering that it sets the horizontal rule to be solid instead of hollow. It does actually make sense, however, because in the old days when all Web pages were medium gray in color, a hollow horizontal rule had the appearance of being embedded in the page, casting a 3D shadow. Setting it to noshade removed the shadow and just left the line itself. This attribute takes no value, but simply stands on its own:

```
<HR noshade>
```



The size attribute also affects how apparent the hollowness of a default horizontal rule is. If the line is too thin, then you cannot tell if it is hollow or solid.

Understanding white space

White space is composed of all the characters that can be used on a Web page that don't produce any visible output: the space between words, a tab, or a carriage return.

According to the HTML standard, "user agents should collapse input white space sequences when producing output inter-word space." What this means in plain language is that any extra spaces you have in your Web page will be eliminated when the page is displayed in a Web browser.

As far as a Web browser is concerned, the following two lines of code are absolutely identical and will be displayed the same way:

```
<P>                    This demonstrates how white space is
handled.</P>
<P>This demonstrates how white space is handled.</P>
```

Figure 3-7 shows the results.

A tab will be interpreted as a single space. This means, by the way, that there is no way in plain HTML to indent the first line of a paragraph. You can, however, do so by using cascading style sheets (CSS) in concert with HTML.



See Chapter 13 for more information on cascading style sheets.

The one exception to the white space rules is preformatted text, represented by the PRE element. Any text placed within a PRE element will be displayed exactly as it is written, including all white space. Thus, in the following HTML code, the leading spaces would be preserved:

```
<PRE>                  This will be shown in a Web browser exactly as
it is here, including the white space.</PRE>
```

www.it-ebooks.info

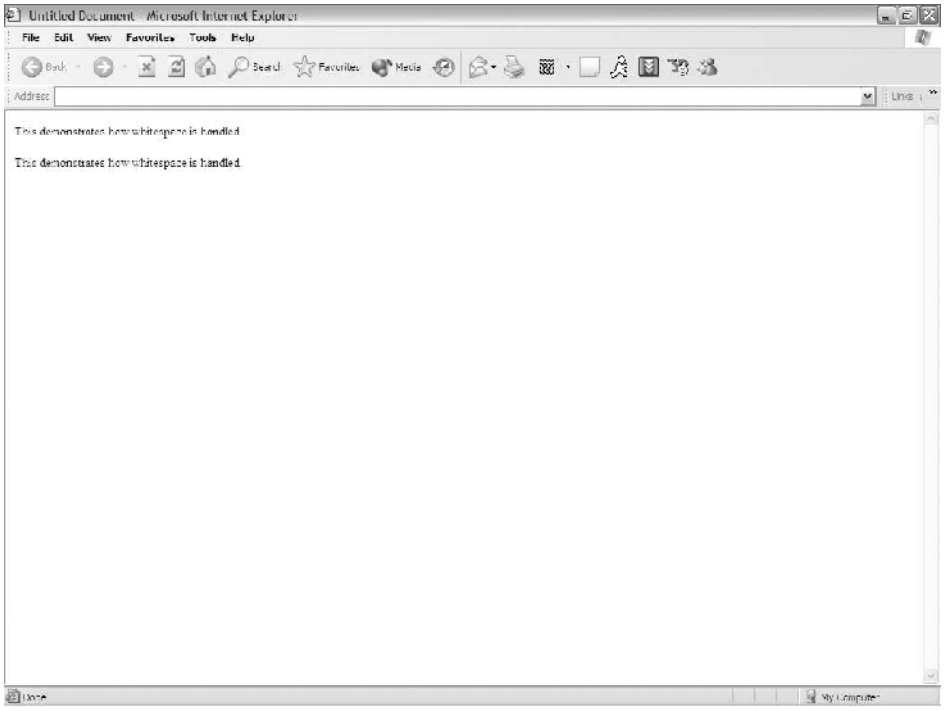


Figure 3-7: Extra white space is ignored by Web browsers.

Understanding the Types of Authoring Tools

All craftspeople are only as good as the tools they use. It doesn't matter if they're carpenters or artists—or Web site designers. A carpenter with an inaccurate measuring tape, a blunted saw, and a plane whose blade keeps falling off will always produce lousy work. An artist whose brush is shedding and who has to work with old paint on a badly made canvas will never be able to create any masterpieces. And Web designers whose authoring programs are not suited to their individual needs won't win any awards, either.

On the flip side of the coin, no tools, no matter how fine and wonderful, can make the person who uses them into a winner. It is only the skilled craftsperson using the best tools who has a chance to produce the best work. A poor musician will play a Stradivarius as badly as a dime store violin. A poet without heart will write bad poetry on a computer or with a quill pen. And Web designers who don't bring skill and artistry to the process won't benefit from the most sophisticated of programs.

The various chapters of this book cover both the skill and artistry parts. But there's one other thing that's needed, and you have to bring it—the earnest desire to make the best Web pages you can. If you have that, I'll supply the rest and you can add a few more glittering strands to this wonderful place called the World Wide Web.

www.it-ebooks.info

There are four basic types of Web page authoring tools:

- ✦ Text editors
- ✦ HTML editors
- ✦ WYSIWYG programs
- ✦ Word processors

Text editors

Text editors are simple, no-frills programs that allow you to type from the standard keyboard and save a plain text file. These are the most difficult programs for beginners to work with because they offer no help. They require you to know everything there is to know about the languages you use and how their quirks can be applied to your situation. Many intermediate- to expert-level Web people, however, prefer to work this way. Different text editors offer various features, including spell checking and syntax highlighting (where different kinds of items in the code show up in different colors so that they can be easily identified at a glance). Figure 3-8 shows the GWD Text Editor, a tool designed for use by programmers. It can, however, be very useful to a Web page designer.

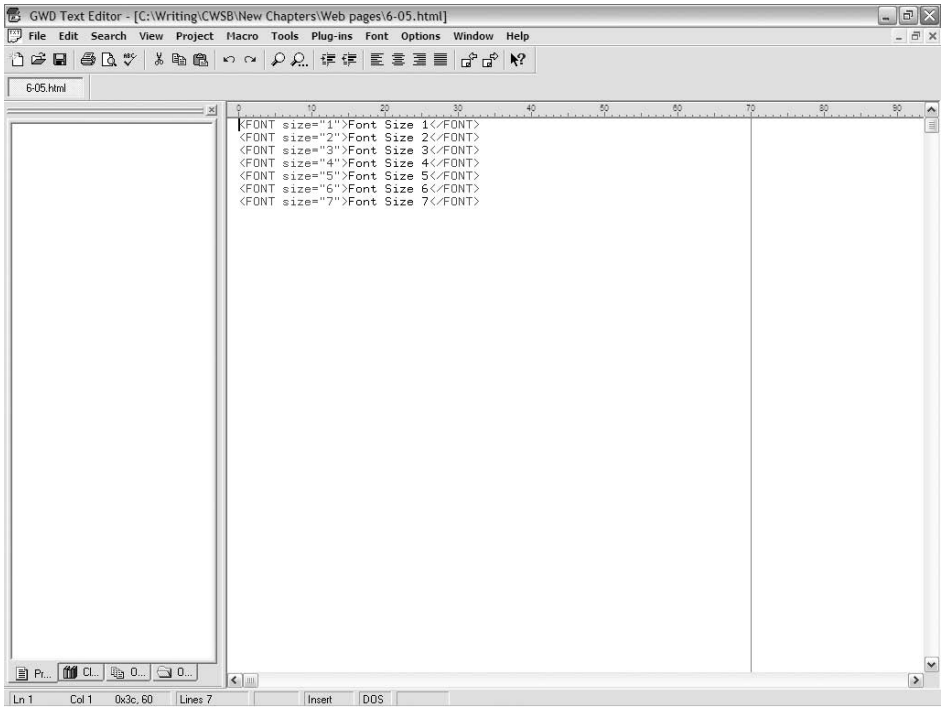


Figure 3-8: The GWD Text Editor is a programmers' tool that can also be used for simple Web page creation.

www.it-ebooks.info

For Windows, the native Notepad is the best-known text editor. If you use the Linux operating system (or any other flavor of Unix), you doubtless have your own favorite—vi, Emacs, kedit, or whatever is the latest rage. And if you use a Mac, you're already familiar with SimpleText (OS 9) or TextEdit (OS X).

Tables 3-4 through 3-6 provide the World Wide Web addresses of several sites where you can find a variety of high-quality text editors.

Table 3-4
Windows Text Editor Sources

Program	*Web Address*
Boxer 99	boxersoftware.com/pgb99.htm
Castillo TextEditor	castillobueno.com/ctexte.htm
Editeur	studioware.com
GWD Text Editor	gwdsoft.com
UltraEdit	ultraedit.com
WinEdit	winedit.com

Table 3-5
Linux Text Editor Sources

Program	*Web Address*
Cooledit	cooledit.sourceforge.net
JOE	http://trevormarshall.com/joe2.8.tar.Z
gedit	gedit.sourceforge.net
NEdit (Nirvana Editor)	nedit.org
TCL TextEdit	http://sourceforge.net/projects/tcltextedit/
Tk Notepad	www.mindspring.com/~joeja/programs.html
XEmacs	xemacs.org

Table 3-6
Macintosh Text Editor Sources

Program	*Web Address*
Alpha	www.kelehers.org/alpha/
BBEdit	www.barebones.com/products/bbedit/index.shtm
Jedit	www.artman21.net/ShowCase_E.html
Style	merzwaren.com/style

www.it-ebooks.info

HTML editors

These are specially enhanced text editors that are designed specifically for creating Web pages. The emphasis is still on typing code, but a wide variety of features are built in to make the process a lot easier. These are the best choice for many people, as they balance the power of hands-on code work with quite a bit of automated support.

Because many text editors today provide some degree of HTML support, it is sometimes hard to draw the line between a plain text editor and an HTML editor. The programs in Tables 3-7 through 3-9 are, however, far superior to normal text editors and are specifically designed for writing the code for Web pages.

Table 3-7
Windows HTML Editor Sources

Program	*Web Address*
Aardvark Pro	ntmg.net/aardvark
Arachnophilia	www.arachnoid.com/arachnophilia
Coffee Cup HTML Editor	coffeecup.com/editor
EZPad	www.mmedia.is/~hagur/ezpad2
HomeSite+	www.macromedia.com/software/homesite/trial
HotDog Pro	sausage.com/hotdog-professional.html
HTMLed Pro	www.ist.ca/htmledpro.html

Table 3-8
Linux HTML Editor Sources

Program	*Web Address*
August	lls.se/~johanb/august
Bluefish	bluefish.openoffice.nl
Quanta Plus	quanta.sourceforge.net

WYSIWYG programs

"What You See Is What You Get" (WYSIWYG) programs fully isolate their users from having to work with code. Typically, objects that are commonly used on Web pages are represented by icons that are grouped together in different *palettes*—floating windows that can be brought up or set aside as needed. Web pages are constructed by using your mouse to drag the objects from the palettes onto the working screen. Other floating windows enable you to customize the settings for the objects, choose

www.it-ebooks.info

Table 3-9
Macintosh HTML Editor Sources

Program	*Web Address*
BBEdit	barebones.com/products/bbedit.html
Creative Page	pinehillproducts.com/creativepage
PageSpinner	spinnerworld.com
World Wide Web Weaver	www.miracleinc.com/Products/W4

and apply colors, and add special capabilities without having to learn programming. Macromedia Dreamweaver and Adobe GoLive (see Figure 3-9) are two popular WYSIWYG programs that both take this approach. Other approaches to WYSIWYG programming use menus and toolbars exclusively.

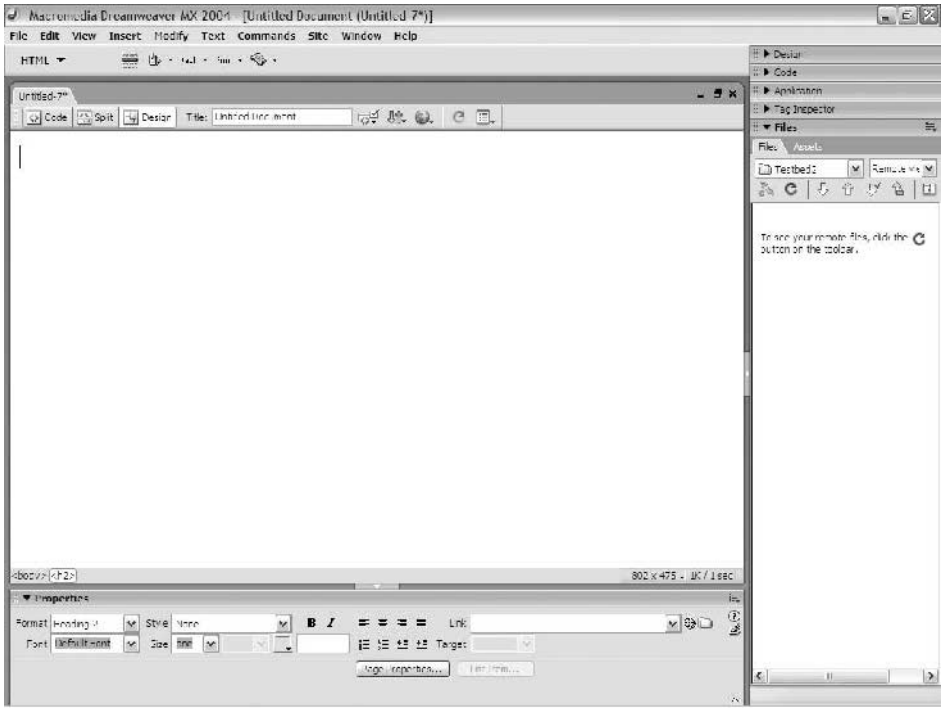


Figure 3-9: Dreamweaver uses an object-oriented approach.

Despite the fact that they are simple enough so that just about anyone can use them, these kinds of programs also appeal to professional Web designers because the best WYSIWYG programs have more than one level to them. Yes, they can be used by a child who knows no programming to create a single page, but they can also be fully exploited by a skilled expert. Most of these programs, in addition to letting you work

www.it-ebooks.info

with a simple interface, also give you direct access to the code so that you can bypass the way the program works and manually override any decisions it makes. Even people who regularly use a text editor for their work will turn to a good WYSIWYG program when a client wants a prototype Web site done "yesterday."

Tables 3-10 through 3-12 show the Web addresses for some of the best WYSIWYG software.



Some of these programs come in both Windows and Macintosh versions or both Windows and Linux versions, but the URL (Web address) is not always the same. NetObjects Fusion, which used to be available in both Windows and Macintosh versions, has discontinued Mac development, at least for the time being. At the time of this writing, the Windows version of Freeway was not yet available, but you should check the Web site for any updates on its status. WebSphere Home Page Builder, from IBM, was formerly known as TopPage.

Table 3-10
Windows WYSIWYG Software Sources

Program	*Web Address*
Amaya	w3.org/Amaya
Dreamweaver	macromedia.com/software/dreamweaver
Freeway	softpress.com
FrontPage	microsoft.com/frontpage
GoLive	adobe.com/products/golive/main.html
IBM WebSphere Home Page Builder	www-4.ibm.com/software/webservers/hpbuilder/win
NetObjects Fusion	netobjects.com/products/html/nf5.html

Table 3-11
Linux WYSIWYG Software Sources

Program	*Web Address*
Amaya	w3.org/Amaya
IBM WebSphere Home Page Builder	www-4.ibm.com/software/webservers/hpbuilder/linux
Siteseed	http://sourceforge.net/projects/siteseed/
Wisewig	sourceforge.net/projects/wisewig

www.it-ebooks.info

Table 3-12
Macintosh WYSIWYG Software Sources

Program	*Web Address*
Dreamweaver	macromedia.com/software/dreamweaver
Freeway	softpress.com
GoLive	adobe.com/products/golive/main.html
NetObjects Fusion	netobjects.com/products/html/nf5.html

Word processors

If you can find a word processor that doesn't claim to be a Web authoring tool these days, I'd be shocked. All you have to do is type in your document, drag in a few graphics, and save your document as an HTML file instead of as a normal document. It sounds good, but it doesn't usually work out well in practice. Word processors are designed for producing paper documents. They're wonderful for writing books and reports, but Web output format is something that was slapped on at the last minute as an afterthought. No word processor can possibly compete with programs that were designed from the ground up for the sole purpose of creating Web pages.

Word processors are also notorious for creating bad code. No professional Web designer I've ever heard of uses one in his or her work—other than for writing proposals and letters—but many of them do make a lot of money fixing up the problems created by clients who create Web pages in word processors.

Microsoft Word, shown in Figure 3-10, is the most popular of all word processing programs. If you work in a modern office, the odds are pretty good that you have it on your computer or, at least, someone in your office does. It's a very good word processor—I'm using it right now to write these words, and I have no complaints. It is, however, notorious for producing poorly written HTML code that produces some amazingly ugly results when viewed in anything but Microsoft Internet Explorer. (For more information, see the amusing article at fourmilab.ch/webtools/demoroniser or the more technical one at www.cs.tut.fi/~jkorpela/unicode/greek.html.)

Word code gives HTML validator programs fits. However, most Web browsers perform a great deal of error checking and correction, so using Word to create Web pages is not a totally impossible option. If you want to work in a WYSIWYG environment on the Windows operating system and you have Word, you might as well give it a try before you buy a serious Web page creation program, but I'd advise against using it to do anything significant or important.



Make sure that you always check out any Web project you do with more than one company's Web browser. If possible, test with different versions of the same browser, too, and view the page on as many platforms as you can. If you're using Windows, for example, and don't have a Linux box or a Mac, ask

www.it-ebooks.info

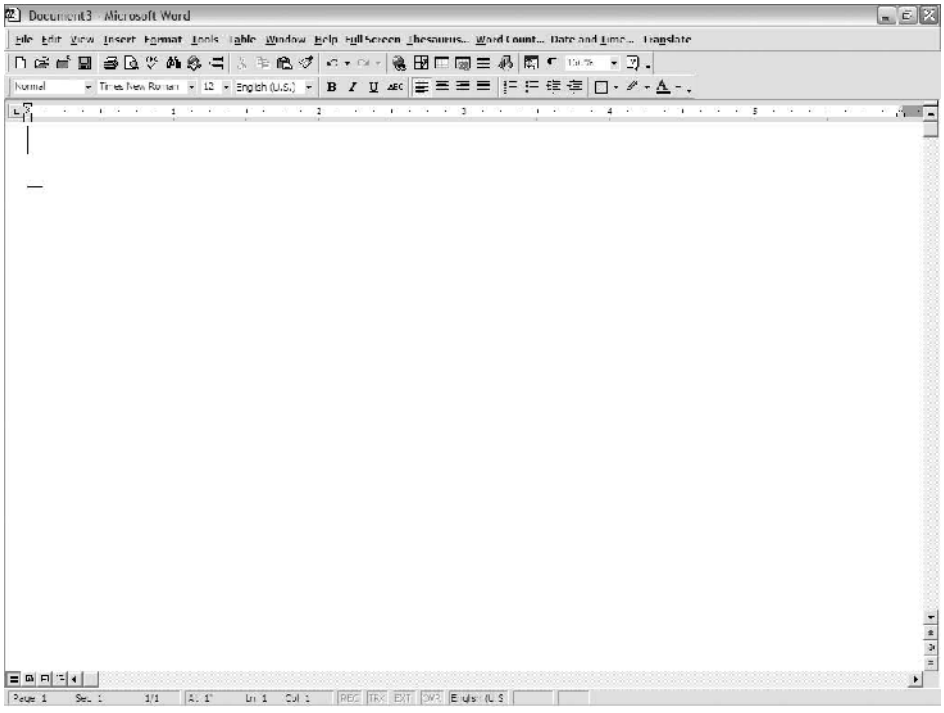


Figure 3-10: Microsoft Word is a fine word processor, but not a serious Web design tool.

a friend to surf to your pages and report on them. If you don't know anyone who has a different system, try asking for help on the `alt.www.webmaster` newsgroup.

Exceptions to the rule

As with practically any "definitive" categorization, there are no doubt exceptions to this one, and it could easily be argued that there are some unusual Web authoring programs that fall between the cracks. What is one to make of Caligari iSpace software, for instance? Although technically a 3D Web graphics program (see Chapter 9) that's usually employed in conjunction with a traditional Web authoring venue such as Dreamweaver, it can be used all by itself to create Web pages complete with text and an impressive amount of bells and whistles. Almost all Web page creation programs, however, do fit into one of the four categories listed earlier in this chapter in the section "Understanding the Types of Authoring Tools."

Choosing Your Tools

When it comes to choosing what to use for creating your Web pages, the question isn't "Which program should I use?"—it's "Which programs should I use in various

www.it-ebooks.info

situations?" Unless you're going to limit your activities to developing the simplest kind of Web site, such as a personal resume page, you'll want to have a variety of programs. Why? For the same reason that you have more than a hammer in your toolbox. That's fine for driving nails, but what if you need to tighten a bolt or cut some wood? You're out of luck if you don't have the right tool. Most serious Web designers develop a fondness for a set of programs rather than settling for the limitations of just one (and all programs have some kind of limitation).

Of course, you can always dig deep into your pockets and pop for one of the major programs, such as Dreamweaver or GoLive, that has just about every tool you'd ever want built into it. Even with that kind of power on hand, though, there are times you'll want to use something else. What if you find a single misspelling on one page in your entire site, for example? You don't need to haul out a vast and powerful array of software to solve that little problem.

Or you may simply find that you prefer the way one program handles some particular process—such as modifying tables or making cascading style sheets—even if you don't use that program for anything else and even if you can do the same thing in a different way in your regular program. That kind of decision is much more than a personal preference, by the way. Your comfort level with your software affects both the speed and quality of your work, so you should always make sure that you use the programs that you're happiest working with.

You can rely on magazine reviews, statements from the software manufacturers, and the advice of friends when choosing your programs. The only way to really know how good a particular program is *for you,* though, is to actually use it. The magazine reviewer may have a different viewpoint from yours, or a particular prejudice that you're unaware of—either for or against a particular company. The same goes for any other people, including your friends, coworkers, professors, or fellow students. A software company's claims, of course, are the most useless of all when it comes to honestly evaluating the worth of a product. All software companies say their product is the absolute best one, none of them advertise their bugs, and some of them have been known to lie outright about the features and capabilities of their products.

Trying demos

Almost every software company has some way for you to try their product out before you invest your hard-earned cash. Considering that even a small suite of programs for Web page creation can cost a small fortune, that's a very good thing. Some companies only produce *demos*, which are usually nothing but computerized slide shows that extol a product's virtues. Most, however, enable you to get your hands on the actual program.

Sometimes, the version you get will not have the full range of features available (this is called *crippleware* in the trade). For example, you may not be able to save a file. More often, the trial version of the software is only *time-limited,* meaning that it is fully functional in every respect but will stop working after a particular amount of

www.it-ebooks.info

time has passed. Typically, this gives you 30 or 45 days to work with the program from the time you install the trial version until you have to make a decision about whether or not to buy the software.

Experimenting with shareware

Another approach—more common with small companies and individual programmers than with large firms—is *shareware*. With this method, you get a working copy of the program with no limitations on it, and it's up to you to pay for the program on the honor system. The cost is usually much less than for a comparable commercial program. Technical support may be spotty, and there's no guarantee that the programmer will keep working on future versions. However, these things also happen with major software companies. Shareware producers are usually better at providing bug fixes than big companies are, too.

Shareware often keeps track of whether you have registered it and reminds you that you need to pay for it. Some of these *nagware* programs remind you every time you start them, while others do so on a periodic basis. When you register and pay for the program, you get either a code that you can enter to put an end to the automated nagging or a new version of the program that doesn't include the nag.



Try out the Web site at `nonags.com` to find shareware without automated nags.

Using freeware

Many programs, especially in the Linux world, are available for free. These are called, of course, *freeware*. With most freeware, there are no limitations at all. Some freeware programs, however, are only free for personal use; commercial operations have to pay for the use of the same programs.

Generally, you are able to download the trial version from the manufacturer's Web site. Sometimes, you have to request that a CD-ROM be mailed to you instead.

The tables earlier in this chapter list several sources of software, and the CD-ROM that accompanies this book has many different Web page creation tools and utilities on it. I encourage you to try them all and see which ones you like best.

Going Beyond the Basics

Once your Web pages are created, you're done. Right? Wrong. Flat out wrong, which is why, in addition to programs that are specifically designed to create Web pages, every Web designer keeps several utilities handy. Most of these are available as either standalone programs or as online services.

www.it-ebooks.info

Link checkers

There's not much that can frustrate visitors to your site more than running into broken links. The Web changes so quickly that anyone who's been surfing for more than a day or two will expect to click some link and find the dreaded "404 - Web Page Not Found" error now and then. One or two dead links here and there are acceptable to any but the most high-strung people. But nobody likes clicking on a whole series of dead ends—it's an annoying waste of time for them. If you let your visitors suffer this way, they'll remember your site for all the wrong reasons and they won't be coming back in a hurry or recommending it to their friends.

Link checkers are programs or online services that perform the tedious task of following every single link on every page in your entire site and reporting back to you on which ones worked and which ones didn't. They can check them all in a few minutes—even on a very large site—saving you a lot of time and aggravation. The alternative is to stare at Web pages and click and click until your vision gets blurry and your mouse finger feels like it's about to fall off.

If you have lots of links to other sites, you're particularly vulnerable because other Webmasters have no obligation to keep you informed of their actions. Even if your site has no links to the outside world, you should still check it periodically because many times, when you make changes to your site, you risk disconnecting a link. Simply changing an image file (remember, images are linked into pages just as much as if you said "click here to see the picture") can result in a broken link.

Code validators

Whether you use plain vanilla HTML or toss in CSS, XML, and all the other alphabetical wonders of the Web, there's bound to be something somewhere in any large site that isn't right. *Code validation* programs like the one at `http://validator.w3.org` read through your creations and check for compatibility with published standards. Most of the time, the differences that are found and reported are negligible—no major Web browser has ever perfectly followed any published standard—but code validators do find real problems as well.

Compatibility testers

If you happen to have a dozen different versions of every kind of Web browser on your system and you've got plenty of time on your hands, you might want to look at every page in your site with every one of them. Really dedicated professional Web designers do just exactly that.

If you're not in that situation, then it's nice to be able to run your pages through a program called a *compatibility tester* that's trained to look for the common trouble spots. You'll get advice and warnings whenever the program spots things such as obsolete tags or company-specific features. It'll also tell you which elements on your page are limited to only newer browsers. It's up to you whether you change those features, but you should at least know about their existence.

www.it-ebooks.info

The version nightmare

Why should you care which version of a Web browser somebody's using? After all, everybody keeps up-to-date with the latest browsers, right? Nope. In fact, nobody anywhere, no matter what figures they toss at you, has any idea how many of which brand of Web browsers are in use on the Internet, nor which versions of those brands people have.

The fact is that, for the average Web surfer, it's a pain in the mouse to keep up with all the version changes in even the two major Web browsers. The natural inclination of most people is, if the browser works, keep using it, and if it doesn't work there must be something wrong with the Web site. The only time when an average Web surfer will pay attention to upgrading is when a browser gives them enough trouble at a lot of Web sites.

It's different for us, and maybe for you. But we get paid to keep up-to-date with the latest happenings on the Web. And things like patches for newly discovered security holes matter to us. That's just not the case with most people. Mention "frame spoofing" to your typical Webnaut and you're not likely to get the reaction, "But wasn't that fixed in Netscape 4.51 back in early '99?" No, you'll get a blank look.

The people who claim to know exactly how many people are using Internet Explorer versus Netscape Navigator or Opera or Lynx are talking through their hats, plain and simple. Because anybody anywhere can use any Web browser without having to tell anyone about it, there's no way of tracking what's on all the computers out there. You can, indeed, check to see which browser brand and version the people who visit any particular site are using, but that's all. And the sites that claim to track browser usage for many other sites won't admit that, until they can track every single Web site on the entire Web, they can't give honest, hard figures. Yes, you can argue that a statistically significant number of sites are being tested, but you need to keep Mark Twain's famous line in mind: "There are three kinds of lies: Lies, damned lies, and statistics."

There are people who will visit your site using an antique version of Mosaic, the Web browser that grew into Netscape Navigator and "inspired" Internet Explorer. There are people with vision problems who will use a browser you never heard of to translate your text into speech. There are people who will come to you using a major brand name, but such an old version that you thought it didn't exist anymore.

Does this mean that you should program for the lowest common denominator? Does it mean that you need to put up a sign saying, "This site best viewed in Internet Explorer 6.0 or greater"? Does it mean that you have to create a whole series of nearly identical Web sites, each of which caters to a different Web browser or version of that browser? Different Webmasters have chosen all of these possible options. It's up to you whether you join the bandwagon or ignore the problem.

Server monitors

Unless you run your own Web server, you're at the mercy of other people. Your Web site may even be physically stored on a machine on the other side of the planet. A *server monitor service* (I know of no standalone programs that do this) periodically checks to see if the Web server that holds your site is up and running. Simply sign up with the service and they'll automatically keep track of your server and notify you if it goes down. If the site is down, the service will notify you right away. Usually, this is

www.it-ebooks.info

by e-mail, but the more sophisticated monitor services can even automatically trigger your beeper to let you know it's time to call up your Web space provider and demand action.



There are lots of legitimate reasons why a Web server might be down temporarily, so it's a good idea to ask your provider what's happening before launching into a tirade.

Working with Specific Tools

In the succeeding chapters of this book, I show you how to create a variety of different Web sites. In order to provide you with a fully rounded experience, I demonstrate how the same site could be created with different tools and discuss the limitations and benefits of using each one.

So many programs could be used that it's impossible to demonstrate all of them. I've chosen a sampling of popular and easily available programs that cover the whole range of Web page creation techniques from working with raw code to dragging object icons onto canvases.

I have also paid particular attention to the fact that people develop Web pages on different computer platforms. While it wasn't possible to have all programs work on all platforms, I have chosen a spread of programs so that anyone—whether working on Windows, Linux, or Macintosh systems—will be able to construct any of the example Web pages and sites.

Notepad

I'll be using Notepad for the plain text editor in the examples. It is the simplest and plainest of plain text editors. The biggest advantage to Notepad is that, if you have Microsoft Windows, you already have Notepad. If you're using a Macintosh, the closest program on your system is SimpleText (OS 9) or TextEdit (OS X). If you're using a Linux box, then you should have at least vi or Emacs available. Depending upon which installation you used, you may have a whole slew of different text editors lying around on your system.

HomeSite+

HomeSite+, shown in Figure 3-11, is one of the most popular Web page creation programs ever written. Like Notepad, it's a text editor and comes in a Windows version only, but there the similarities come to a screeching halt. It has color-coded syntax, built-in HTML references, automated hints and helpers, and a battery of wizards to assist you in writing code—and that's just scratching the surface.

I'll be using HomeSite+ to fill in the gaps in some situations where the limitations of one of the other programs would create a difficulty for you. If you're using Linux or a Mac, one of the HTML editors from the earlier tables will do as a substitute.

www.it-ebooks.info

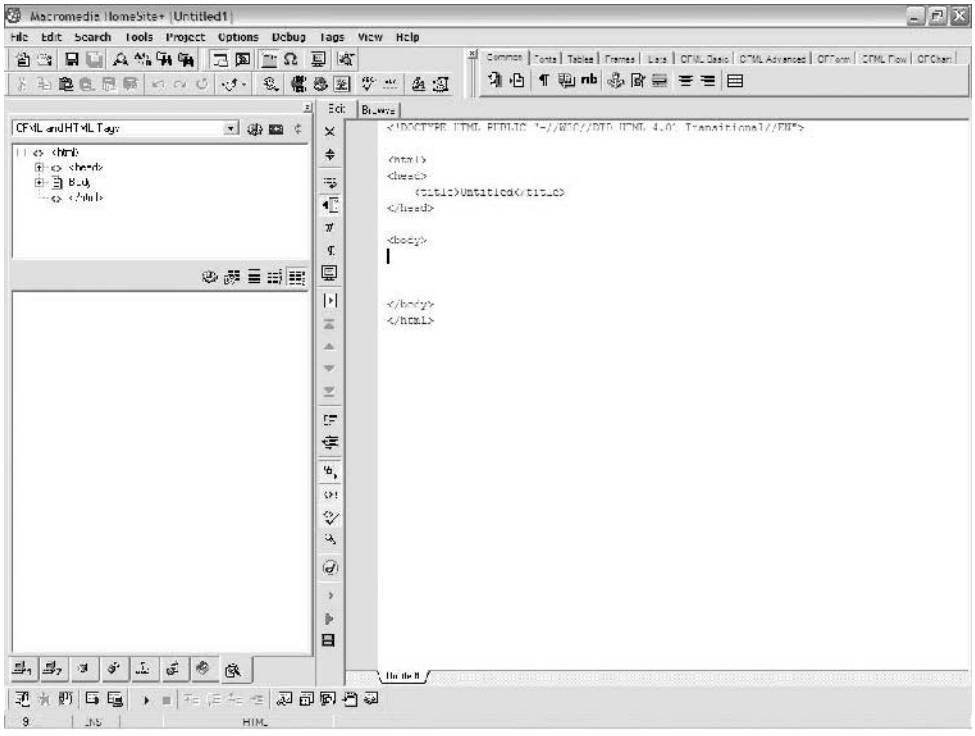


Figure 3-11: HomeSite+ is an excellent HTML editor with advanced capabilities.

Dreamweaver

In my opinion, Dreamweaver (shown in Figure 3-12) is the best of the WYSIWYG programs, and it just keeps on getting better and better with every new version. One of the most common laments in the Linux world is the fact that Dreamweaver is only available on the Windows and Macintosh platforms. Macromedia, the company that produces it, has won a loyal following among both professional and amateur Web designers because it takes great care to listen to the people who use the program. While it's true that all software companies solicit input from their users, Macromedia goes out of its way to actually pay attention to what they say. Sad to say, many others do not. The end result of this open feedback process is that Dreamweaver is one honey of a program and the most useful tool for people who regularly work with the Web.

This program packs just about everything you can do with a Web page into one package. In addition to the WYSIWYG interface and powerful table management, it has its own sophisticated text editor for those who want to get under the hood (with an optional split-screen view so you can see the effects of your changes as they're made), a built-in JavaScript debugger, and a variety of utilities ranging from version checking to its Clean Up Word HTML command (see Figure 3-13). And, because it's made by the same company that produces Flash, it's a piece of cake to integrate Flash animations into your pages in Dreamweaver.

www.it-ebooks.info

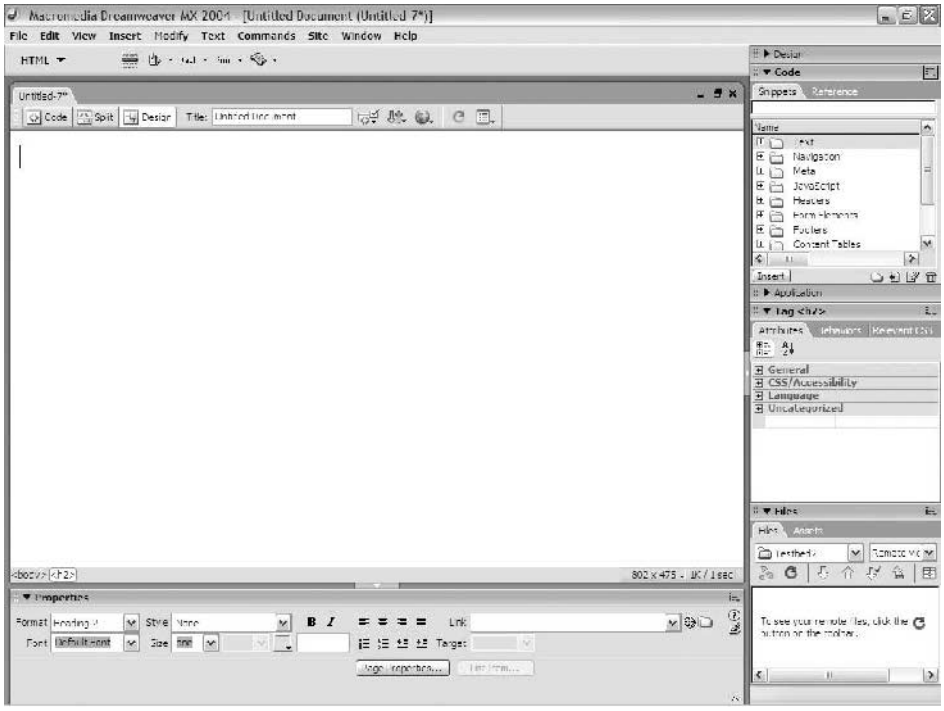


Figure 3-12: Dreamweaver is a full-featured Web site creation program.

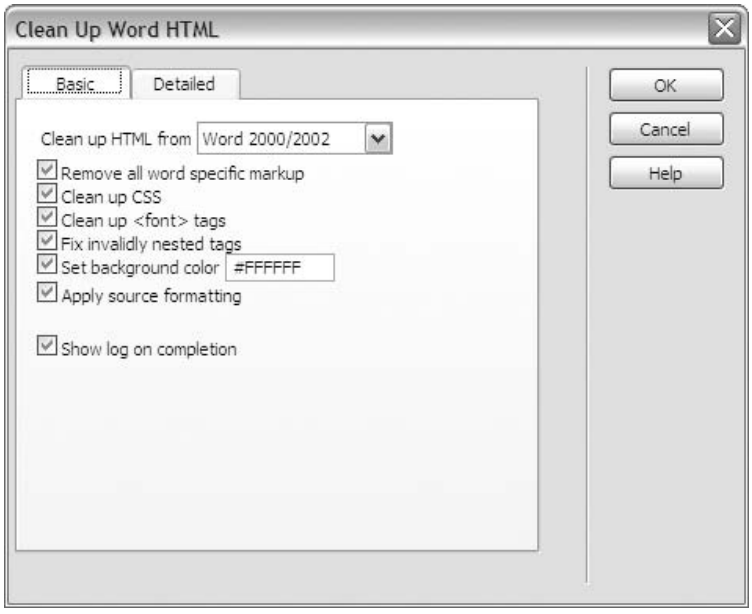


Figure 3-13: Macromedia's Dreamweaver quickly fixes Microsoft Word's HTML errors.

www.it-ebooks.info

Dreamweaver works differently from Composer. Aside from the fact that it has a lot more features and capabilities, it uses an object-oriented approach. You drag icons from floating palettes into its working area and drop them there. Once the objects are in place, you then modify their properties in a dialog box.

Summary

Although the latest generation of Web page creation tools has insulated designers from the most basic levels of markup language, a basic understanding of what is going on "under the hood" is still necessary. This chapter introduced you to the following concepts and topics:

- ✦ Web pages are composed of elements.
- ✦ Elements are modified by setting the values of attributes.
- ✦ All elements are contained within other elements.
- ✦ WYSIWYG authoring tools show the Web page as it will appear in a Web browser.
- ✦ Code authoring tools show the underlying HTML code.
- ✦ In addition to the basic Web page creation tools, you should also be familiar with link checkers and validators.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 4

Organizing Your Site



In This Chapter

Addressing resources with URLs

Adding links

Comparing page design to site design

Breaking up your site



HTML's foundation is built on the capability to create links between pieces of information. Web pages without links are dead-end streets—offering little to users and potential customers. When creating your Web site, you'll use links to organize your content into manageable chunks, as well as to reference other points of information elsewhere on the Internet.

The links you will create to lead visitors through your site will take a little bit of planning up front. However, the effort expended in doing so will be more than repaid in the ease of use your site will offer in the aftermath.

Comparing Page Design to Site Design

While many people often refer to Web site design as Web page design, the two have distinct meanings within Web development. In fact, Web page design really can't be achieved without first designing the Web site. To see what we mean, consider the Web site at `www.choppoint.org` (see Figure 4-1).

Even though the main page and the interior pages of the site are from the same Web site, you can tell the designer had the entire site in mind when she developed them. For example, the color choices remain constant across all pages on the site. This makes it easier for users to recognize elements of the page and also helps to remind visitors they are still within the same Web site.

The next few sections outline a few things to keep in mind when designing the overall look and feel for an entire Web site.

www.it-ebooks.info

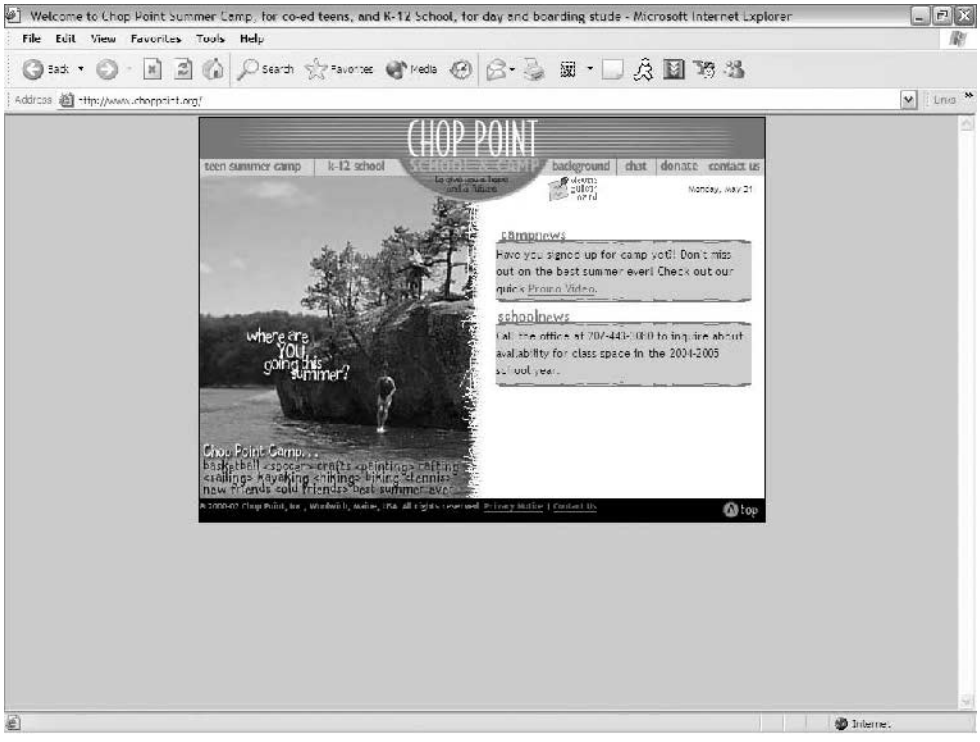


Figure 4-1: The Chop Point site is a good site to study for design.

Color

One of the easiest ways to ensure uniformity across Web pages is through the use of a consistent color palette. Identify the colors you want to use up front, and stick with them throughout the site, whether it's background colors, title colors, or link colors.

Use colors that blend well together, and avoid those that might cause certain aspects of your page to be unreadable. For example, placing red text on a green background may not be the best idea, given that color-blind visitors may not be able to distinguish between the two.



For more tips on using color, see Chapter 10.

Another useful technique involves first selecting a color palette with the same number of colors as there are sections of your Web site. Then highlight each section of your site with one of those colors and use the others in a supportive role within those sections. If you use this technique, be sure to use all of the colors in some way, shape, or form on the home page to tie the pages together.

www.it-ebooks.info

Text style

Maintain a consistent set of styles to format the text on your pages. For example, using the same font face (such as Arial or Verdana) for the headlines on all your pages can go a long way toward maintaining uniformity on your site.



For more information on using fonts, see Chapter 6.

Style information for your pages may include some or all of the following:

- ✦ Font faces
- ✦ Font weights
- ✦ Font sizes
- ✦ Colors
- ✦ Word spacing
- ✦ Line spacing
- ✦ Letter spacing

So, if the links on your home page are bold, underlined, and blue, then continue that styling on all other pages of your site.



One of the best ways to maintain consistency in styles across multiple Web pages is to use a single cascading style sheet (CSS). For more information, see Chapter 13.

Navigation

Keep the navigation constant, both in actual links and in location on the page. Consistency in links can often be accomplished with tiered navigation. For example, if the main sections of your site are accessed through a bar across the top of the page, perhaps the secondary sections or pages are accessed through a bar just below the first one. This enables the user to easily return to a section or page on the top level of your Web site, even when he is viewing a page much lower in the site structure.

You can see an example of tiered navigation if you return to `www.choppoint.org` (see Figure 4-1). The top-level navigation is continued across the top of the page, while the secondary navigation, specific to the Camp section, is displayed down the left side. Not only does this allow for easy access to other areas of the site, it also helps to orient the user to her place within the site.



For more tips regarding navigation, see Chapter 17.

www.it-ebooks.info

Understanding the importance of content

One of the buzz phrases in Web development helps explain the importance of content within a Web site: *Content is king*. This is definitely a statement you should keep in mind when creating your Web site. While it's true that beautiful designs can bring visitors to a Web site, I argue it is the content that keeps them coming back. So before you begin designing the spectacular interfaces that will enable your visitors to move around your site, make sure you have the content to engage them in the first place.

Instead of simply re-purposing old content from other sources (such as printed brochures or newsletters), scrutinize the content to be sure it is something online users might be interested in seeing and suitable for online reading. Put yourself in the shoes of your audience. Would you be interested in reading this content on a Web site? If not, consider leaving it out of your site.

It is interesting to note that many newspapers with Web sites require their reporters to write two versions of each story, one for print and one for the Web. The reason for this is simple: we read Web pages differently then we read printed pages.

Making your site functional

After you've decided what content will be on your site, determine how to best make it functional. Web sites aren't meant to be static pages of old information. One of the best ways to keep visitors coming back is to offer up-to-date content in a timely manner.

So if you are creating a site for a marina, in which you list boats for sale, you'd better update that list as often as possible. For example, if you would normally update the list printed in your local newspaper once a week, the list on your Web site better be updated at least as often as that, if not more often.

Other ways to make your site functional include the following:

- **Avoid dead ends.** On the Web, a dead end is a page without any links. Always provide, at the very least, a link back to your home page.
- **Always link to related information**, even if it may take users off your Web site. While you may lose some visitors to those other sites, you'll undoubtedly win even more with your thorough scope of information, particularly if your site is the one that wins the bookmark.
- **Provide multiple ways for users to find the information.** We all do things differently, and two people will likely surf your Web site in quite different manners. To assist each of them, provide at least three ways to navigate your site. For example, you might provide graphical buttons across the top of your site, and alternative text links across the bottom, plus a search function or a site map (index) or both.
- **Make your pages scannable.** Use design elements such as bullets and blank lines to draw attention to pieces of information. The majority of usability

www.it-ebooks.info

experts agree that users scan Web pages more than they read them. To this end, avoid long-winded sentences, in favor of short, concise bullet points. If you can help users find what they're looking for more quickly, they'll appreciate it and continue to give your site business.

- **Avoid the fluff.** Marketing jargon, including words like "great" and "huge," may be catchy on a printed advertisement, but they only add to the immense amount of "stuff" on Web pages. In general, the fewer words the better when it comes to Web pages. As many technical editors have said, if a sentence works without a particular word, take it out. Even better, if a page works without a sentence or paragraph, take that out, too!
- **Keep content current.** Nothing's worse than a Web site advertising a sale for Christmas 2004, when viewed in May of 2005. It is important to always keep Web content current, in order to keep your site functional. If you don't think you can keep a time-sensitive article current, consider leaving it off your site altogether or posting a notice stating it is provided for research purposes only and will not be updated.
- **Pull content from other sources**, as needed, to avoid reinventing the wheel. Adding a good search capability to find things on your site can greatly improve the usability of your Web site. However, instead of spending a lot of money to write your own search program, consider using freeware or shareware versions already available. The same thing holds true for things such as news and weather feeds. Both of these services can provide added value for your visitors, if geared specifically for their needs. For example, a Web site targeting athletes in Maryland might provide sports news for Maryland teams by partnering with a local newspaper or TV station.



When using material from other sites, make sure that you have the copyright holder's permission to do so.

Planning Your Site's Layout

At some point in the development of your Web site, you'll have to decide how to break up the various pieces of information within your site. Suppose you've identified the following key pieces of information for the site you're creating about a marina:

- Hours of operation (four lines of text)
- Directions (map and 10 lines of text)
- Sales staff (five photos, each with three lines of text)
- Services (one paragraph)
- Brief background or history (one paragraph)
- Contact information (10 lines of text)

www.it-ebooks.info

- ✦ 5 to 10 photos of the marina, to be used as needed
- ✦ List of 10 to 50 boats for sale; some have photos, all have details such as price, size, color, features, and so on (content is within an Excel spreadsheet)

Given this list, can you imagine how you might organize this Web site? There are certainly several possibilities, and here's one of them:

- ✦ **Home Page.** Brief background or history, list of services, one or two photos of the marina
- ✦ **About the Marina.** Hours of operation, directions, sales staff, one or two photos of the marina
- ✦ **Boats for Sale.** The details of the boats for sale.
- ✦ **Contact the Marina.** Contact information, directions, one or two photos of the marina

The key to organizing your site is putting yourself in the position of the visitor. So if you were visiting this Web site and you wanted to find directions to the marina, where would *you* look for them? If you thought your visitors might conceivably look for them either in the About the Marina section or the Contact the Marina section, you might place them in both sections. Or, you could create a separate section just for the directions, and cross-reference the directions from each of the other sections.

Whatever you decide, it is usually worthwhile to ask someone in your target audience to confirm your decisions. Be careful about asking the client, in this case, the owner of the marina. While this person might know the business inside and out, he probably doesn't think about it the same way as his customers do. Given his experience with the content, he might know to look for some piece of information in one place simply because he put it there. Instead, ask to talk to one of those customers, or try asking a friend who fits the profile of a potential customer.

This type of activity is called *usability testing* because you are confirming the usability of your Web site with members of the target audience. It is best to confirm these decisions before spending the time and money to develop your Web site because it is easier and more efficient to make changes in the beginning stages. You should do another round of usability testing after developing your site.



For more information about testing your Web site, see Chapter 27.

After grouping the pieces of information into logical sections, it's time to consider the actual flow of that content.

Creating topic-specific Web pages

Writing for the Web is much different than writing for the printed page. To test this theory, consider how you read a book as opposed to how you read a typical Web

www.it-ebooks.info

page. Most people sit down in a comfortable chair to read a book word for word, page by page. By contrast, Web users often sit in stiff office chairs staring at a screen that may or may not be at the appropriate eye level. And how many times have you actually read a Web site word for word, page by page? I'd venture to say, probably never.

This means that as a Web designer, you must create topic-specific Web pages that quickly and clearly imply the meaning and intent of each page. Did you ever have an English teacher tell you to restrict yourself to one main point per paragraph? I'm going to be even stricter with a Web page and suggest you stick to one main point per page. This doesn't mean each page should only include a few sentences, but rather that each page should focus on providing the appropriate amount of information for a single key point, as well as links to additional sources of related information.

While most books are written to be read in a *linear* fashion, the very nature of the Web invites us to move around Web pages and sites quickly, searching for bits and pieces of information. While many people do surf the Web for pleasure, a large number of us use it to *find* things, as if we're on a mission.

Web sites, unlike books, are written or coded in such a way that visitors can move from a page about antelopes to a page about zebras in a single click. So even if you intend for the pages of your site to be read in a linear fashion, you can't count on it. Instead, many Web sites are developed in a more *hierarchical* fashion, in which the site's structure looks much like a flow chart or diagram map.

By adding links back to the top of the hierarchy, you can achieve what many designers refer to as a web of information. In this type of structure, you provide links to and from all major sections of your site, as well as links to related information elsewhere on or off your site. In the following diagram, only a portion of the possible link relationships are shown. In fact, diagrams like this one can quickly become unruly and difficult to decipher. For this reason, it is not uncommon for site maps to be broken up according to section.

Finding natural breakpoints

Indeed, on the Web it is best to keep things short and sweet. If you ask 10 people how long they view most Web pages, the answers will probably come in terms of seconds, not minutes. Because you have such a short amount of time in which to grab the user's attention, make your point very clear in the title and header blocks of the page. Don't bury such important information below three navigation bars, an ad banner, and your company's logo.

The next question: What is an appropriate amount of information for a Web page? The answer depends on the type of content you have for your site. Unfortunately, you can't really think about it in terms of printed pages. Instead, consider how long it would take you to read a certain chunk of information on a page. It's probably reasonable to expect users to read no more than three to five paragraphs of information about a specific topic on a single Web page. If you have more information than that on a particular topic, consider breaking it up over several pages.

www.it-ebooks.info

Another consideration regarding the amount of content displayed on a single Web page is the length of each line. While it may be tempting to allow the text lines to run across the entire width of the screen, that is not the easiest to read. Just as newspaper columns remain short in width, so should Web page columns. In general, columns of no more than 300 to 350 pixels in width are best for optimal viewing pleasure.

When reviewing your content in preparation for Web development, look for natural pauses in long pieces of information, where you might easily break the content over two or more pages. When doing so, remember to always provide links forward and backward within the linear flow of the article, as well as links to related content.



Interested in learning more tips regarding Web site usability? Visit `www.usableweb.com` and `www.useit.com` for excellent resources on the topic.

Addressing Resources with URLs

A *Uniform Resource Locator* (URL) is the Internet equivalent of a building's street address. It tells where to find a particular item such as a Web page in the vast muddle of all the things that make up the Internet. A URL can refer to any resource, including physical equipment. However, we're mainly concerned with Web pages and the files linked to them (such as graphics or sound files).

URLs are used in various ways on the World Wide Web, depending on the type of connection between the Web page and another resource. Each of these separate approaches is technically referred to as a *scheme*. The most common use of URLs, of course, is to link one page to another one via the Hypertext Transfer Protocol (HTTP). This scheme is one that is doubtless familiar to you already, even if you don't know its details. You've certainly seen a Web address like this one that tells how to find a home page:

```
http://puzzles.usatoday.com/index.html
```

This scheme breaks down as follows:

http:// specifies the protocol to be used for transferring information.

puzzles.usatoday.com specifies the server that holds the resource being sought.

index.html specifies the name of the file to transfer from the server to the Web browser.

When you want to link to a file on another Web site, you need to give the full URL beginning with `http://` because there's no way to find the file until you first find the server it's on. This is known as an *absolute URL*, and such an address will always work no matter where it's invoked from. For example, linking to `http://puzzles.usatoday.com/index.html` can never do anything but bring up the home page of John Wiley & Sons. It doesn't matter if the link is on your Web page or anybody else's.

www.it-ebooks.info

If you're linking to other files within your own Web site, however, you don't have to use absolute URLs. Say your site is called georgesdream.com. When coding the pages on your site, you could code every link to your home page as `http://www.georgesdream.com/index.html`—or you could use a *relative URL* instead and just go with `index.html`.

This is possible because a Web browser, in the absence of an absolute URL, will automatically assume that you mean to link to another file on the same site. Although the process is *transparent*, meaning you don't see it happen, here's how it works.

Say a visitor is looking at your Web page located at `http://www.georgesdream.com/resume.html`. You have a link to `index.html` on that page. The visitor clicks that link, and their Web browser strips everything to the right of the final slash in the current address to arrive at a base URL. In this case, that's `http://www.georgesdream.com/`. Next, the relative URL `index.html` is added to the base URL to get the full URL, so the browser knows to go to `http://www.georgesdream.com/index.html`.



Relative URLs are sometimes also known as *partial URLs*.

So how do you choose which method to use? It depends on your needs and purpose. The functional difference is that relative URLs make it easy to move a site from one server to another, while absolute URLs lock you into a particular server's address. If you don't have your own domain name, then your Web site is hosted on someone else's Web server and their domain name is a major part of your pages' addresses. You may, for example, have a free Web site on Yahoo! Geocities, in which case your pages' URLs would start with `http://www.geocities.com/`. If you used absolute URLs for your internal links and later decided to move your site to one of their competitors such as Angelfire (`http://angelfire.lycos.com`), then you'd have to go through all your links and change them to reflect your new Web space provider's address.



Many URLs contain the prefix `www`—as in `www.wiley.com`. The `www` was originally used to denote that the site was located on the World Wide Web, but it is not a required part of a URL. In reality, it is up to the server's administrator to define the prefix. While many people do still use `www`, others change the prefix to signify various sections of a company Web site. For example, Yahoo! has `travel.yahoo.com` set up for its travel services, and `chat.yahoo.com` for its chat rooms. Other sites may leave off the prefix altogether and simply use the domain name, as in `amazon.com`. If you're unsure which prefix to use for your site, check with the server's system administrator.

Adding Links

Links to other pieces of information are added with the `A` (anchor) element, as in the following example.

```
<A href="http://www.yahoo.com">Visit Yahoo!</A>
```

www.it-ebooks.info

As with other container tags, you surround the content you wish to link with opening and closing `A` tags. The `href` attribute gives the specific location of whatever you are linking to. So in the preceding example, the value of the `href` attribute tells the browser to link the words *Visit Yahoo!* to `http://www.yahoo.com`. As discussed in the previous section, this is accomplished using an *absolute* URL.

A relative link back to Yahoo's home page from within the Yahoo! site might look like this:

```
<A href="index.html">Visit the Yahoo! home page</A>
```

By default, the user's cursor changes to a hand when it is moved over a link. This helps the user identify content that is clickable. For this reason, it is important to select carefully what is actually linked. For example, consider the section of text content shown in Figure 4-2.

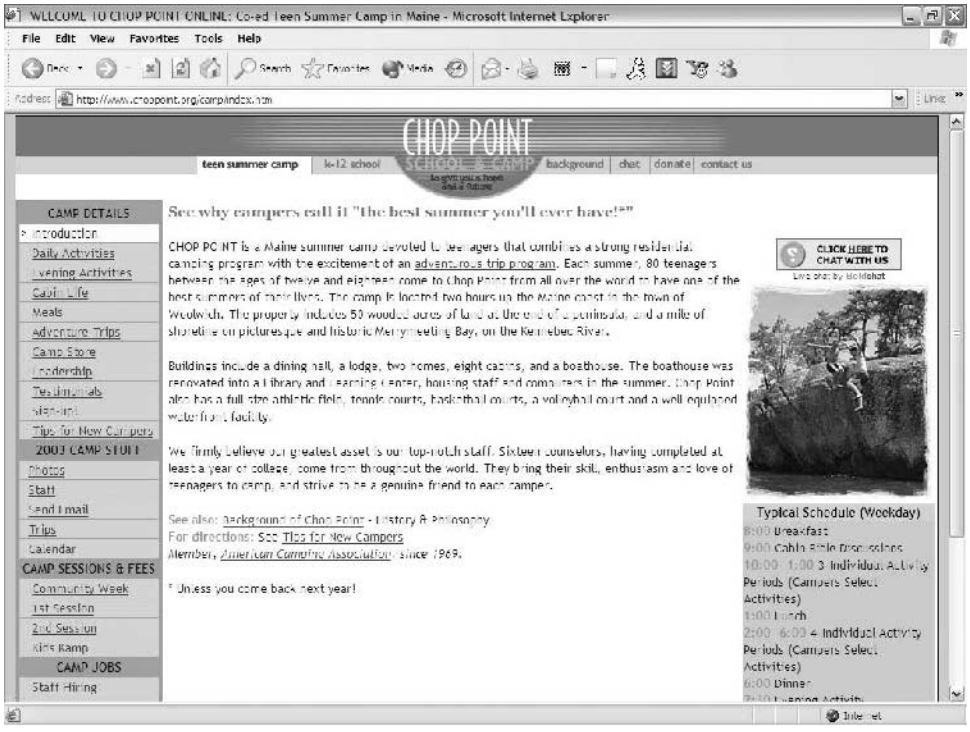


Figure 4-2: Well-worded and obvious links make a site more user-friendly and easier to navigate.

The site shown in Figure 4-2 avoids using "Click Here" to label links, often referred to as the Click Here syndrome, which is a common problem in Web sites. In addition, all the links are pulled out into a brief list, except for one in the text.

To avoid confusing link errors, give yourself a few seconds to quickly scan your Web page. If you can't pick out the links and identify what they link to within a few

www.it-ebooks.info

seconds, you may need to rethink those link labels. A few simple changes can go a long way toward making the page much easier to navigate.

Getting hypertext references right

One of the biggest challenges in coding links lies in getting the hypertext references right. This means you have to specify the appropriate path within the directory structure to properly link to a file. To better understand this issue, consider Figure 4-3.



Figure 4-3: This sample directory structure helps explain how to properly code hypertext references.

The directory structure shown in Figure 4-3 includes three folders, each containing two files. If you were coding a link on the page entitled `jobs.html` (located in the *employment* folder) that was supposed to take users to the *Contact Us* page, can you imagine what the hypertext reference would look like? Here's an example:

```
<A href="../aboutus/contactus.html">Contact Us</A>
```

When moving around a Web site, you have to tell the browser when to move up and down in the directory structure. Prefacing a folder or filename with `../` tells the browser to move back, or up, one level within the directory structure.

If I had left off the `../` in the previous code example, and simply coded the hypertext reference as `href="aboutus/contactus.html"`, the browser would have looked for a folder called *aboutus* within the *employment* folder. So when a user clicked the reference, the link would appear broken.



Many servers are also set up to recognize a single slash (`/`) as a shortcut back to the root level, or top directory. This can make coding the same thing on multiple pages of your site, such as a navigation bar, much easier because all the hypertext references can be the same, regardless of where the page lies within the directory structure. If you are unsure as to whether your server is set up to do this, contact your server's administrator.

www.it-ebooks.info

Creating internal links with local anchors

Sometimes, you may want to link to a specific section on a Web page, as opposed to just the top of it. You can accomplish this through the use of internal links and local anchors.

Suppose you created a very long Web page. You might want to provide a quick link—*Back to Top*—to give users easy access back to the top of your page without having to scroll. To do so, first give the section you want to link to a name, or *anchor*, using the `A` tag and `name` attribute, as shown here:

```
<HTML>
<BODY>
<A name="top"></A>
...Rest of page content goes here...
```

Then, add a link to that section name anywhere on the page:

```
<A href="#top">Back to Top</A>
</BODY>
</HTML>
```

Because I'm linking back to the top of the page in this example, I'd most likely add the link at the bottom of the page. Note that when you link to a section name on a page, as opposed to an entire page, you don't include an `.html` extension. Instead, place a hash mark (#) in front of whatever name you gave the section. In this case that was "top."



While you're free to create as many section names and links as you'd like on a page, keep in mind that some users may become confused if their clicks take them to a lower section on the page, where the main navigation and page title are no longer visible. So if you do break your page up into multiple sections, consider duplicating the navigation and page title at each section, or providing links back to the page's navigation and title.

If you find your internal links to sections aren't working, first confirm the section name matches the link name perfectly. Because section names are case sensitive, "top" and "Top" mean different things. In addition, it's best not to include any spaces or punctuation in your section names because they can easily be confused in the browser.

So far you've only created internal links within a single page. What happens when you want to link to a specific section of a different page? As in the previous example, you must first know the exact name of that section. Then, add a hash mark and that section name after the page name in the `href` attribute of your `A` tag.

```
<A href="shoes.html#size9">View our size 9 shoes</A>
```

This tells the browser to first look for the page called *shoes.html*, and then search on that page for a section named *size9*.

www.it-ebooks.info

Sending e-mail with mailto links

While the majority of links you create will take users to other Web pages, you might also want to allow visitors to quickly access your e-mail address. To do so, preface your e-mail address with `mailto:` as in the following example:

```
<A href="mailto:name@emailaddress.com">Email Me!</A>
```

When a visitor clicks a `mailto` link like the previous one, a new e-mail message will be created with your e-mail address in the `To:` field. Note that in order for your `mailto` link to work, your visitors must have an e-mail program available on their system.



You can further customize the e-mail message by adding values for fields such as the subject and cc list. Here's a sample: `<A href="mailto:name@emailaddress.com? Subject=Hello!&cc=anothername@emailaddress.com">Email Me!</A>`. Note that an ampersand (&) separates fields if you're defining more than one.

Other types of links

In some cases, it might be necessary to link to other types of information, such as downloadable files or newsgroup articles. When doing so, you continue to use the `A` tag and `href` attribute, but change the value of the attribute accordingly as shown in Table 4-1.

Table 4-1
Other Types of Links

Link Type	*href Value*	*Sample Code*
Downloadable file on an FTP site	ftp://sitename.com/filename.exe	`<A href="ftp://sitename.com/filename.exe">Download from FTP site</A>`
Newsgroup	news:group.name	`<A href="news:alt.html">Visit the HTML newsgroup</A>`
Specific newsgroup article	news:12345.67890@news.com	`<A href="news:12345.67890@news.com">View this news article</A>`



In order for links to newsgroups or news articles to work, users must have a browser and Internet connection capable of accessing newsgroups.

Locking relative URLs with the BASE element

You can use the `BASE` element to force all links on a page to use the same domain name. For example, say you wanted to create a page with links to your favorite books

www.it-ebooks.info

on the John Wiley & Sons Web site. Because all of the links on this page would be coded with the same starting part of the URL (www.wiley.com), putting that part of the URL in the BASE element could potentially save a lot of time when coding the page as in the following code:

```
<HTML>
<HEAD>
     TITLE>My Web Page</TITLE>
   <BASE href="http://www.yohansoldebookshoppe.com">
</HEAD>
<BODY>
<P>Yohan's Olde Bookshoppe has many great books. Here's some links to my
favorites:</P>
<P><A href="/burpee/burpee_index.html">Burpee</A> is the how-to resource,
from seeds to salads<BR>
<A href="/look&learn/index.html">Deke McClelland's Look and Learn</A>
series cuts-to-the-chase to get you up and running quickly on a new design
application.<BR>
<A href="/bc/bc_index.html">Betty Crocker</A> is America's most trusted
name in the kitchen. This series includes cookbooks from family classics to
fine dining.</P>
<P><A href="http://www.mywebsite.com/index.html">Return to my home
page</A></P>
</BODY>
</HTML>
```

Notice that instead of typing http://www.yohansoldebookshoppe.com at the beginning of each link, I only coded the folder and file names, as if these were relative links to pages within my own Web site. Then, to override the BASE element, I included the entire URL (http://www.mywebsite.com) when linking to the home page because the domain was different than that listed in the BASE element.

Summary

This chapter introduced you to the following concepts and topics:

- ✦ Absolute URLs restrict the link using them, while relative URLs allow the same link to be used with different servers.
- ✦ Local anchors enable you to add links to specific locations within a page.
- ✦ mailto links enable you to set up e-mail messages that your visitors can send.
- ✦ The majority of predevelopment decisions can best be made with the user in mind. Thinking like a member of your target audience will help you identify the best course of action when breaking up the pieces of your information to build a functional, user-centered Web site.
- ✦ Consistency is key to site design. Use the same font faces and colors to represent important page elements on all pages.

✦ ✦ ✦

www.it-ebooks.info

CHAPTER 5

Putting It on the Web

In This Chapter

- Getting your domain name
- Finding a Web space provider
- Testing responsiveness to your needs
- Avoiding common provider scams
- Exploring bells and whistles
- Uploading your pages

Once you've created your Web site, you'll naturally want to put it online, right? Although you might have some Web space through your Internet service provider (ISP) or through a free Web service, a superior Web site deserves superior Web space. And picking that Web space takes just a little forethought, some planning, and even a little bit of fun.

In this chapter, you'll find a new home for your Web pages and move them in. Everything from picking your domain name to uploading your pages with an FTP program is covered here. Whether you just need a home for a personal home page or an expandable site for your e-commerce solution, you'll be ready to go when your Web site is finished.

Getting Your Domain Name

Your Web site's URL is the most visible part of your online identity. It's what you put on your business cards, in your advertisements, and on company T-shirts and letterhead. It's how people remember where you are on the Web, so an easily remembered URL is as important nowadays as an easily remembered company name.

Although you can use a URL offered by free Web space providers or your own ISP (such as `www.geocities.com/xerexes/coganth.html`), most businesses and many individuals buy their own domain name to make it easy for people to find them on the Web. If your online presence is `www.myname.com`, people can remember and find your site more quickly and easily. A domain name is generally a worthwhile investment, once you're ready to get serious about your Web site and go beyond a simple personal page.

Choosing a domain name

Dot-com has become synonymous with *online business*. That's because the .com ending of many domain names stands for

www.it-ebooks.info

commercial. Originally, the domain name system was set up to categorize online entities by their types. The top-level domain names are the last part of a domain name (such as .com). Each top-level domain stands for a different category. Thus, there's .com (for commercial), .org (organization, usually non-profit), .net (network services such as ISPs), .gov (government agencies only), .edu (educational), and .mil (military). Unless you're a public agency, you can't get a domain name in one of the last three top-level domains, but you can get them in .com, .org, and .net.

Recently, ICANN (Internet Corporation for Assigned Names and Numbers) approved additional top-level domains: .biz, .info, .name, .pro, .museum, .aero, and .coop.

As you think about possible domain names, you might want to follow these guidelines for making a decision:

- **.com** or **.biz**—Company or corporation
- **.org** or **.info**—Non-profit organization
- .**net**—Network service
- **.name** or **.coop**—Personal Web site. You may also want to use a country-specific domain name

You should also consider the following questions:

- Can I pronounce the name easily? Will people know how to spell it just by hearing it?
- Can I spell the name clearly? Many letters (especially *f* and *s*) sound similar when spelled out over the phone.
- Is the name easy to remember? Does it have something to do with the content of my Web space, or the product or service I'm providing?

These days, most people realize that they're probably not going to get their first choice in domain names. Let's face it, most of the shorter domains are taken, and many of the longer ones are hard to pronounce or to type. So, when you go looking for a domain name, remember that it's hard to find a good domain—but not impossible. Have a list of at least 10 possible domains that would be acceptable, but don't go register them all. Domain-grabbing is one reason there just aren't enough domain names to go around anymore.

Once you have your list, the first thing to do is check out the WHOIS lookup at `http://networksolutions.com/cgi-bin/whois/whois`, shown in Figure 5-1. Type in the domain name you'd like and click Search. Don't include the `www`—that's not part of the domain name.

If the results look something like Figure 5-2, go back and try a different name. The domain that you wanted is already taken by someone else.



If you've got some money to spend and you really want that domain name, you can always send a letter to the administrative contact offering to buy the domain. Popular domain names can run upwards of a couple million dollars, so be sure to get your credit limit raised before you call!

www.it-ebooks.info

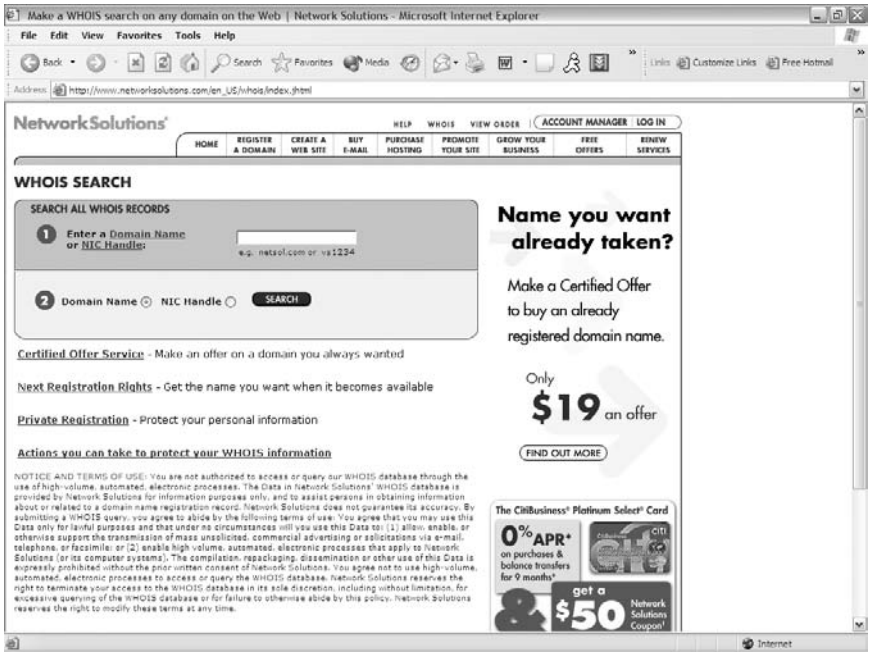


Figure 5-1: The WHOIS lookup at Network Solutions enables you to look up a domain name quickly and easily.

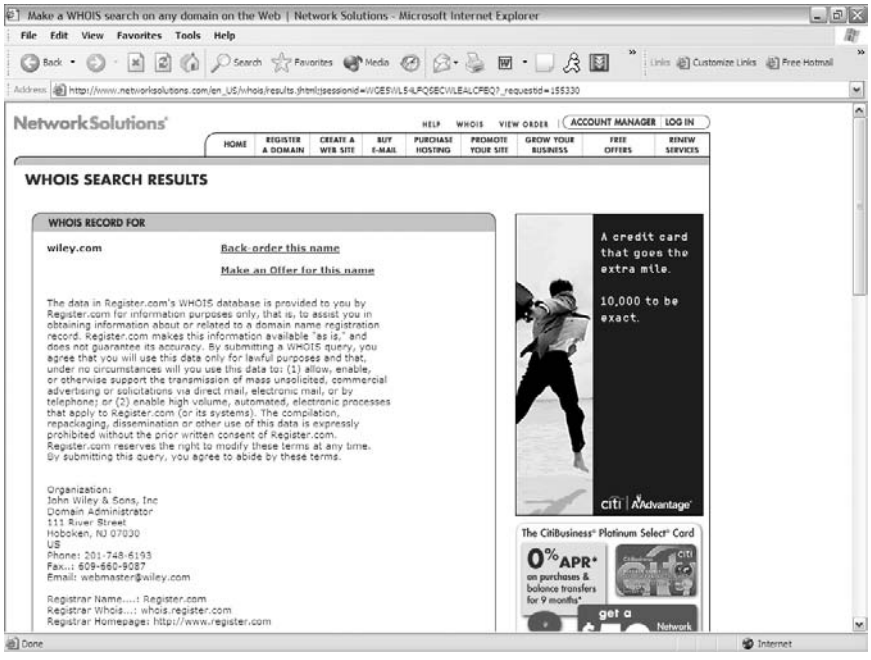


Figure 5-2: WHOIS reveals the contact information for every domain name registered with Network Solutions.

www.it-ebooks.info

If the results of your domain name search come back empty, you'll get a response similar to Figure 5-3. Here's your chance to grab that domain! Click Register Now to register the domain using Network Solutions, or find a different registrar below.

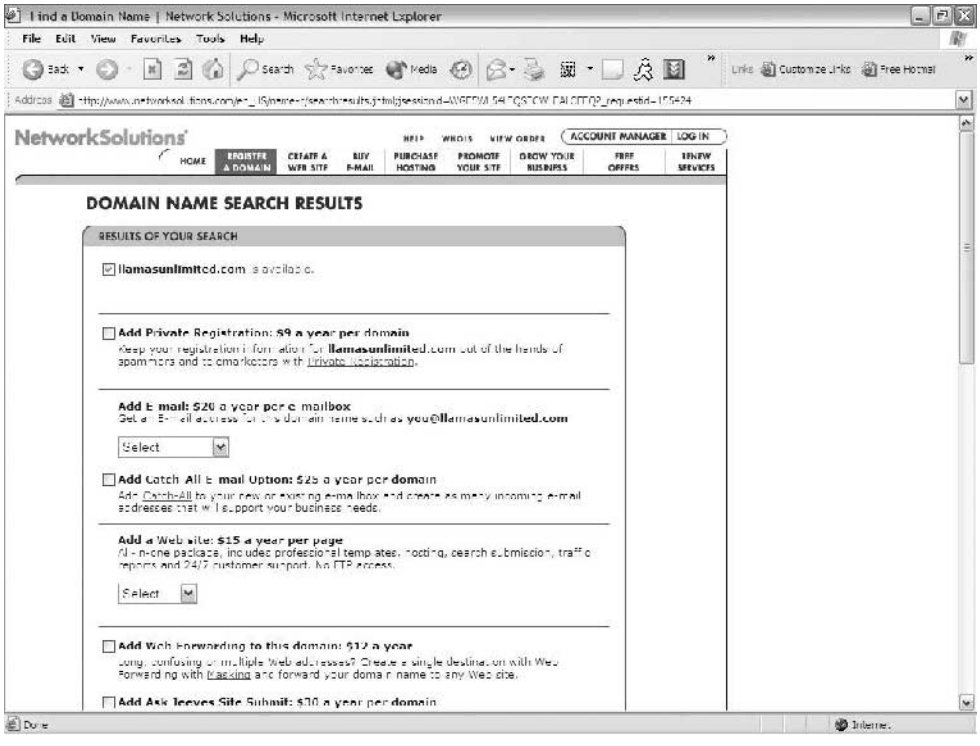


Figure 5-3: When a domain hasn't been registered yet, WHOIS lets you know.

Picking a registrar

When you go to register your domain name, you'll need to find a registration company, also called a registrar, to register with. Registrars vary in price and services, ranging from $8 a year to $35 a year. Few registrars are more than $35 per year, although several of them are less. A good online guide to picking a registrar is at `www.thecheapdomain.com`. Some registrars require you to pay a fee if you want to transfer a domain name. Others don't. The transfer fee shouldn't be more than a year's worth of domain name registration. Also, you shouldn't have to pay a sign-up fee for registering your domain name, and you shouldn't sign up for more than two years at first. Be willing to sign up for those two years, though—a domain name is a commitment. Many registrars require you to create a DNS server (domain name server) entry within a specific amount of time. This policy prevents name-grabbing by companies and individuals who just want to "hold onto" a domain name and prevent others from getting or using it.

www.it-ebooks.info



If you don't have a Web space provider or a domain name yet, hold off on registering your domain name. Most Web hosting companies offer reasonable domain name registration as part of their services.

There's another domain name registration option available to individuals, organizations, and many businesses. Although the registry is pretty much tapped-out, the country-specific domains are not. Every country has its own rules on registering a domain. Many countries will let non-residents register domain names for their country-specific codes. That's how all those .tv Web sites got their domain names—they registered with a domain registry at `www.tv` for Tuvalo, a small Pacific Island nation near Fiji.

The United States will let you register any `.us` domain, as long as you provide complete geographic information in your domain name.

For example, Wiley Publishing, the publisher of this book, could register `wileypublishing.new-york.ny.us`. While that might not be the easiest domain name to remember for a corporation, it works pretty well if you're an individual or organization and just want your own domain.

To register a domain name in the `.us` domain, go to `www.nic.us`. To register a domain name in another country, check the list at `www.domainit.com/country-domains.htm`.

Coping with registration pitfalls

There aren't a lot of things that can go seriously wrong when registering your domain name, but there are a lot of little hang-ups that can happen when you register or transfer a domain name. Just remember that even though "Internet time" is a thousand times faster than the real world, there are still people behind the process of domain name registration: be patient, and be wary.

Signing up for multiple years at a discount

Domain name registrars often offer a bulk discount if you sign up and pre-pay for several years' worth of service at the outset. Although this may seem like a good idea, be aware of how much time you're actually signing up for. Network Solutions offers new domain names at $70 for the first two years. That's not really a bad deal, although it also isn't a discount. It keeps people from creating fly-by-night Web sites and encourages long-term stability on the Internet.

Other registrars might offer up to five years with some type of discount. In general, you should avoid being locked in to a registrar for more than two years. Domain name registrars may change the terms of service agreements in the middle of your contract. With a five-year contract, you could end up committed to new service terms that you don't agree with. In most cases, you can leave the registrar, but if you break your contract early you probably can't get a refund.

www.it-ebooks.info

Why does it take so long?

Some domain name registrars work very efficiently. They have an online sign-up system where you just enter in your information and credit card number, and everything goes very quickly and smoothly, automatically processed with a computer. Others are a little slower. If they're slow, it usually means that there's a human being somewhere holding up the process. Usually, that translates into more careful customer service.

Even in the fastest systems, there's a technological barrier to registering a domain name. After you enter (and pay for) your domain name registration, your Web service provider must send a Domain Name Service (DNS) entry to all the DNS servers in the world. Essentially, your registrar is sending out a change-of-address notice to a million different machines. And then, for security reasons, those machines often respond with a verification request. Even in Internet time, the whole process can take several days. During that time, your Web site might be available in one geographical area, but not in another as one server recognizes it ahead of the time when another does.

Finding a Web Space Provider

A Web space provider is like an ISP (Internet service provider), except that it doesn't give *you* access to the Internet; it gives the Internet access to you. Specifically, the Web space provider makes your Web site available and visible to the public.

Web space providers fall into three categories:

- ✦ Virtual hosts
- ✦ Dedicated servers
- ✦ Co-located servers

These providers generally offer different amounts of Web space, customer support, and technological expertise. The server itself might run on a variety of software and hardware, some of which vary in reliability and ease-of-use. Naturally, more choices means more prices; as you choose a Web space provider, make sure you pay attention to how much it costs, compared to how much you're getting.

Picking the right Web server

A *Web server* is the combination of hardware and software that holds and displays your Web site to the Internet. Many people will try to connect to your server all at the same time, so the machine itself should have a powerful processor and lots of memory (RAM). Your server should also use a strong server software program to deliver your Web pages quickly and with few errors. For the same reason, your Web server should have a very fast connection to the Internet, either a T1 or T3. You should put as much thought into your Web server as you do into your domain name. It's a key part of your Web site—without it, there's no site at all.

www.it-ebooks.info

Finally, the Web server software should be secure enough that you don't have to worry about hackers getting in and changing your pages or hijacking your site without your permission. If you're going to be accepting or storing financial information, including credit card data, your server should use a high level of encryption. If you're inside the United States, use 128-bit for domestic connections and 40-bit for international connections (the higher the bit rate for encryption, the better the encoding is).

Although a Web server can generally run on almost any computer, most commercial Web servers will have a sufficient amount of RAM, hard drive space, and processor speed for a small- to medium-scale Web site. Any site holding more than 100MB of data or transferring more than 500MB a day should have its own dedicated server, either onsite or in a co-location facility. Dedicated servers and co-location are discussed later in this section.

Web servers run a variety of server software programs on all operating systems (OS). Some of the most common and reliable server combinations are the following:

- ✦ Sun Solaris OS with Netscape or Apache as the Web server
- ✦ Linux (any flavor) with Apache as the Web server
- ✦ Windows NT with Internet Information Server as the Web server.

Other combinations include Macintosh with Netscape or another MacOS Web server, BeOS, and even DOS. There's even a tiny Web server program available for the PalmPilot!

If you're shopping for a Web server combination, and you'd like to know what your competitors or partners use, go to the Netcraft Web site at `www.netcraft.com` and enter the Web site's domain name in the field. Click What's that site running? to see the results. You'll be given a short description of what operating system, server, and extensions are installed on the server, as shown in Figure 5-4.

If the information is available, you can also see at a glance how long the server has been running and when any data on it was last changed (see Figure 5-5).

One other thing you should look for are server extensions, APIs, and installed languages that enable you to install scripts to automate your site. An example of a Web-server API is MSAPI, Microsoft API for Windows NT. Programmers can write scripts and programs to use MSAPI features and automate the site.

FrontPage server extensions enable you to develop sites using Microsoft FrontPage. FrontPage has several *bots* and automated scripts built in. These scripts are already available once you install the server extensions.



FrontPage extensions are version-dependent. If you're using FrontPage 2000, make sure you have the server extensions for FrontPage 2000. If you're using FrontPage 2003, make sure the server extensions are for FrontPage 2003.

www.it-ebooks.info

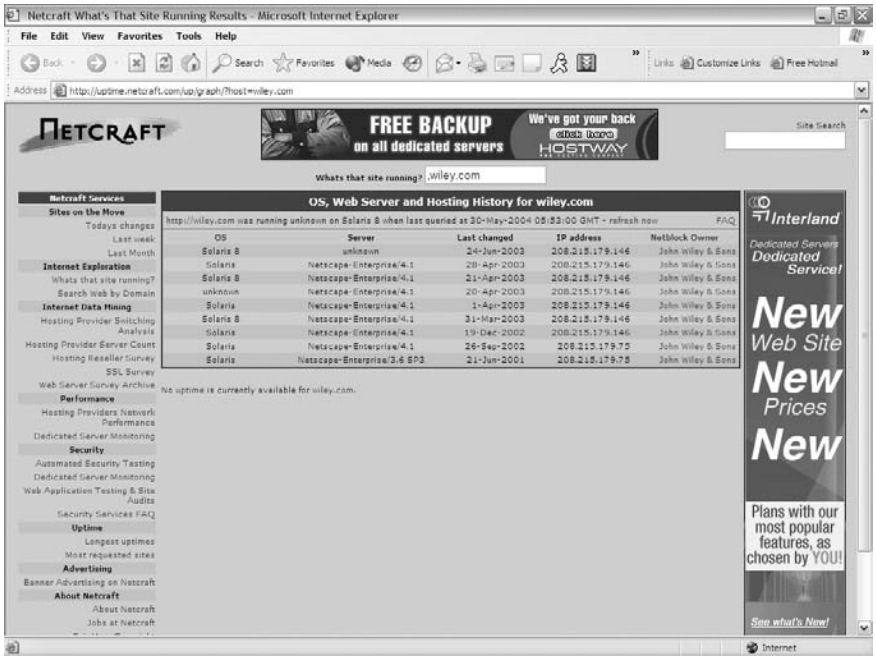


Figure 5-4: Find the operating system, Web server, and server extensions and APIs (such as PHP) for any Web server on Netcraft.

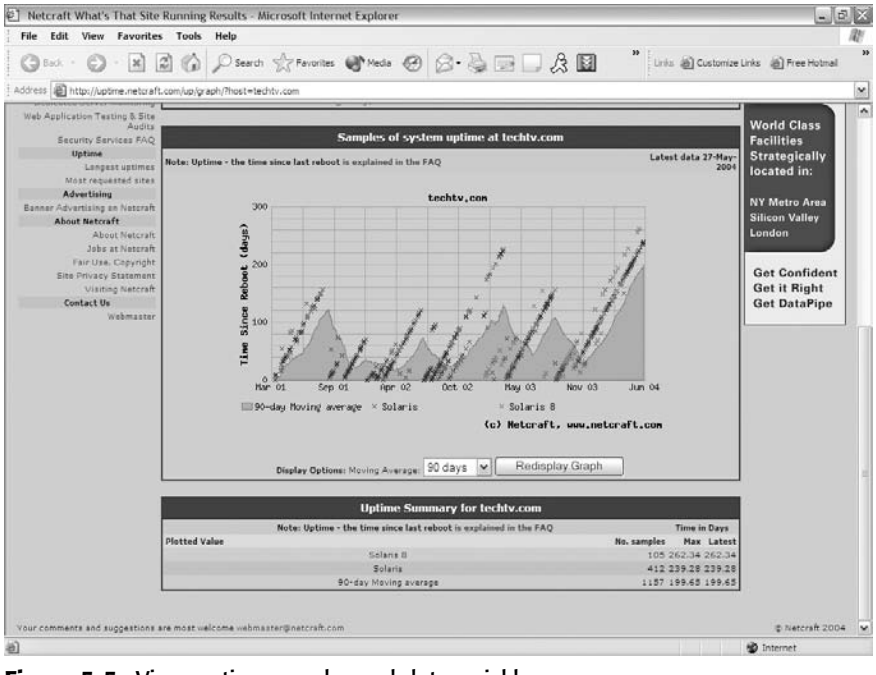


Figure 5-5: View uptime graphs and data quickly.

www.it-ebooks.info

The free, open source options include Perl, a scripting language that has been used extensively on the Internet. In addition, PHP is a server-side HTML scripting language. It's probably the most effective way to create browser-independent dynamic Web pages, and it is an open source project, so no single company has control over its development.

Determining your space needs

The next question you'll probably be asking is, "How much space do I need?" Well, that depends on a few things. First, are you just hosting a couple of HTML pages with a few graphics? Will you have a downloadable program on your site? Are you including audio files? Are you planning to serve Napster audio and video files?

The more complicated your media, the larger the files. The larger the files, the more space you'll need. As you know, HTML is just plain text, the simplest type of file. Compressed graphics such as GIFs and JPEGs are the next simplest type of file, followed by PDF files, audio files, and finally video. Executable programs can take up a little bit of space or a lot, depending on the size of the program. In general, however, you should plan for a lot of hard drive space if you want your users to be able to download programs and audio/video files from your site. If you're just providing text information, then you can get away with 30 to 50MB of storage space.

There is one other reason you might need more storage space. If you plan to run a database on your Web site, or to deliver database-driven content, you'll need to have a database available on your server. A database can be a simple Perl script and take up very little room on your hard drive, or it can be a full-blown relational database like Oracle, in which case you'd better invest in a multigigabyte dedicated drive.

Choosing a non-virtual, virtual, dedicated, or co-located server

As you look at different Web hosting options, you'll notice that there is a very large variety of ways to put a Web site online. In order of technical complexity (and price), these are non-virtual hosting, virtual hosting, dedicated servers, and co-located servers.

Non-virtual hosting

The first option is *non-virtual hosting.* Non-virtual means that you don't have your own domain name. Instead, you either host your Web site on someone else's server, as in `www.theirdomain.com/myspace`, or you *park* your site on another domain: `www.myspace.theirdomain.com`. Sometimes these parked sites are also called subdomains. There are countless free Web space providers, such as Yahoo!, Geocities, Tripod, TheGlobe, and others, and these usually offer non-virtual hosting only.



The 100 Best Free Web Space Services list at `http://100best-free-Web-space.com` is a good quick reference to choosing a free Web host.

www.it-ebooks.info

Many ISPs offer Web space to their customers at no additional charge, and most Web hosting companies let you park your site on their domain at no additional charge. Because you don't have to pay for the domain name registration fees, the main advantage to this model is price. Sometimes domain availability is also a factor. If you can't find a domain name that suits you, you might find a Web hosting company with something similar where you can park your site and still have an easy-to-remember name.

Virtual Web hosting

The second option is called *virtual Web hosting*, or using a *virtual domain.* Here, you keep your Web pages and files on someone else's Web server, but you have your own domain name. People viewing your site don't know that your server isn't actually at your business or in a dedicated server. Depending on the size of your Web site, virtual hosting can cost anywhere from $5 per month to about $100 per month. If you have enough Web material that you need more space or bandwidth, you should consider a dedicated server.

Dedicated servers

Using a *dedicated server* involves leasing the Web server software and Internet connection, as well as the hard drive space. The hosting company maintains the Web server itself, so the hardware is maintained by someone else, and any software or technical problems are usually covered by the hosting company. These services generally cost $150 and up per month. They're good if you have a lot of files, a database, or if you're expecting a lot of hits to your site.

Co-located servers

Finally, you can purchase *co-located servers* for yourself. You own the hardware, the software, and everything in the box. You rent space on a rack in a specialized facility for computers. These facilities are generally climate-controlled for computers, have dry fire suppression systems (so they won't fry your machine), onsite security ranging from someone who checks passes to special, locked rooms with bullet-proof walls. Co-location facilities often offer technical support options. The most basic is the ability to request a technician to physically restart your machine if you're not able to go to the facility itself to fix a problem. If you choose to co-locate your server, you'll probably need to know a lot more about system administration than this book will cover, or hire someone who does.



Search for a co-location facility online at `www.webhosters.com/tools/search/co-location`.

When you select a co-location facility, make sure it's close enough that you can reach it in an hour or so, in case of server trouble. Check their security options, tech support options, and find out if they monitor their own network connection 24 hours a day. How quickly do they respond to network problems? Ask their current clients about their customer response and network response times. How fast is their network connection, and can they guarantee redundancy? Do they have more than one connection to the Internet in the building? Don't neglect the physical space, too:

www.it-ebooks.info

Is the room cool when you walk in? Are the aisles between the racks neat and free of clutter? Do you detect any dust when you look at their facility? Do they have a fire suppression system that uses dry materials, such as gas, to put out fires? Dust, heat, and water are computer-killers, so be picky about what you will and won't accept for these answers.

Balancing price-service ratios

As you can see, when it comes to Web space and service, you can get what you pay for, if you do a little research. Non-virtual Web space should be free or included with your Internet service from your ISP. Virtual Web space should cost from $5 per month to $100 per month, depending on how much bandwidth and storage space you get. A dedicated server is about $150 per month and up, and space in a co-location facility will cost anywhere from $300 per month to $1,000 per month and up.

But what does that $5 or $100 or $1,000 include? We've created a little table to help you understand what services are typically included or available in a virtual Web hosting package. Non-virtual Web hosting companies usually offer the same services, unless they provide Web space for free. Free Web space companies offer a minimum of services, but you just can't beat the price. Because dedicated and co-location models put you in control of what's available on your server, we left those out of the table, too.

Typically, a Web hosting company offers two or three packages for your Web hosting needs. They might be called Silver, Gold, Platinum or Personal, Business, Executive, but they're basically three scales of Web services. Not all Web hosting companies offer the same package deals of course, but Table 5-1 gives you a quick look at what you might find in a typical package.

Table 5-1
Virtual Web Hosting Services

Service	*Low-Level Plan*	*Mid-Level Plan*	*High-Level Plan*
Hard drive space	5MB	10MB	20MB
Connection speed	T3	T3	T3
Bandwidth	250MB/Mo.	500MB/Mo.	1000MB/Mo.
Telnet accounts	1	1	1
E-mail accounts	1	2	5
FTP accounts	1	1	1
Access or per-hit charges	$0.08	$0.08	$0.08
Server-side image maps	Yes	Yes	Yes

Continued

www.it-ebooks.info

Table 5-1 *(continued)*

Service	*Low-Level Plan*	*Mid-Level Plan*	*High-Level Plan*
Pre-installed CGI scripts	Yes	Yes	Yes
User-installed CGI scripts	Yes	Yes	Yes
Access reports	Yes	Yes	Yes
Secure server (SSL)	Yes	Yes	Yes
Server Side Includes (SSI)	Yes	Yes	Yes
Database available	Yes	Yes	Yes
Commercial use	Yes	Yes	Yes
Automated mailing list (listservs, Majordomo, and so on)	No	No	No
FrontPage extensions	Yes	Yes	Yes
Perl	Yes	Yes	Yes
PHP	No	No	No
Setup fee	$29.95	$29.95	$29.95
Monthly fee	$9.95	$14.95	$24.95

Ultimately, you want to have enough storage space, a reliable network connection, and enough user accounts to handle however many people will be working on your Web site or using your domain for e-mail or Web space. If you're the only one who will be working on the site, then you only need one FTP or telnet account. If you're one person on a 10-person team, you'll need more accounts.

If you're looking for some way to give Web users accounts, make sure you have CGI scripting, PHP, or a server module such as FrontPage available to create that kind of online community. Web user accounts aren't the same as telnet, FTP, or e-mail accounts; they're a function of the Web site programming, not the server.

Investigating Web space providers

The best way to research Web space providers is to look at what you already have (through ISPs, work, school, or for free) and then hit the Internet. You want to look at as many Web hosting companies as possible before you make your final decision. One great resource for finding low-cost virtual and non-virtual hosting companies is `www.budgetweb.com/budgetweb`.

Run a query to find a Web service provider that offers what you want for the price you're willing to pay, and look into them until you narrow your search down to two or three that offer similar packages at a good price.

www.it-ebooks.info



Don't get locked into a yearly contract with your Web service provider. You should be able to leave anytime you're not satisfied with their service.

Once you've picked two or three Web space providers, use the traceroute program to check how close to a network backbone the host company's servers are. In MS-DOS, type **tracert**, followed by the domain name of the hosting company. In Linux, the program is called traceroute and works like this:

```
traceroute yahoo.com
```

The output is similar to this:

```
traceroute: Warning: yahoo.com has multiple addresses; using 216.115.108.243
traceroute: Warning: Multiple interfaces found; using 206.124.164.10 @ hme0
traceroute to yahoo.com  (216.115.108.243), 30 hops max, 40 byte packets
 1  phoenix-nap.neta.com  (206.124.164.1)  0.930 ms  0.607 ms  0.573 ms
 2  33.atm3-1-0.phnx-cust1.phnx.uswest.net  (63.226.31.229)  3.934 ms
3.852 ms  3.686 ms
 3  gig2-0.phnx-gw2.phnx.uswest.net  (206.80.192.251)  3.602 ms  3.702 ms
3.990 ms
 4  sl-gw15-ana-1-1.sprintlink.net  (144.232.192.21)  11.339 ms  11.496 ms
11.467 ms
 5  sl-bb22-ana-5-0.sprintlink.net  (144.232.1.109)  11.451 ms  11.927 ms
12.076 ms
 6  sl-bb20-ana-15-0.sprintlink.net  (144.232.1.178)  11.505 ms  12.734 ms
12.016 ms
 7  sl-intsys-6-0-0.sprintlink.net  (144.228.179.226)  61.808 ms  62.052 ms
61.629 ms
 8  pos4-0-622M.wr1.LAX1.gblx.net  (206.132.112.85)  63.554 ms  63.292 ms
64.401 ms
 9  pos2-0-2488M.wr2.SF01.gblx.net  (206.132.110.85)  261.032 ms  274.364 ms *
10  so-6-0-0-2488M.wr2.SNV2.gblx.net  (208.50.169.113)  258.041 ms  260.173 ms
```

Each line is a hop, or move from one machine to another, in the network connection. Essentially, it took 10 "hops" from `phoenix-nap.neta.com` (where the tracert connection started) to yahoo.com. It also took 260 milliseconds to reach yahoo.com.



You can also use `www.tracert.com/cgi-bin/trace.pl` to run tracert through the Web.

Testing Responsiveness to Your Needs

In addition to the speed of the connection, you might want to check a few other ways your Web hosting company can respond. The first is reliability. What's the uptime of the Web server and network connection? The Web server uptime can generally be obtained from Netcraft at `www.netcraft.com`. The network reliability is a little trickier. In general, you should go to the various online Web hosting company review sites and newsgroups, such as `alt.www.webmaster`.

www.it-ebooks.info

Obtaining customer service

The second thing you want to test is the customer service. If the Web hosting company doesn't have a phone number listed on its Web site, go with another company. If it does list one and it claims to offer 24-hour customer service, call the company at 1:00 in the morning to see if it's real. If you get voicemail or an answering service, you can expect that network problems won't be solved until the next morning, something you might not be able to afford. Also be sure to call during a peak time, around 5:00 in the evening. Time how long it takes to get connected to a live person. If you get a pre-recorded message telling you to visit the company's Web site, hang up and walk away. If customer service is only available online, and a problem takes down their network, how are you supposed to ask them about it?

Many Web hosting companies start out just fine in terms of customer service. They're fast, they answer the phone right away, and they provide continuous network connection on a very reliable basis. Alas, this kind of service does not usually last. As more people learn about the great customer service, new customers flock to the Web hosting company. Eventually, the company outgrows its customer service department. If they don't have a large enough customer base to justify a larger department, the quality of service suffers. And so do the customers.

It's an unfortunate fact online that few high-tech companies recognize the importance of customer satisfaction in making or breaking their businesses. In a matter of months, these companies may fail, simply because they did not help their customers in times of difficulty. Although it's okay to try out a newly established Web hosting company, be aware that in 9 to 12 months the company may experience customer service growing pains. Watch for these, and jump ship if it starts adversely affecting your Web service.

Assessing technical support

Finally, call their technical support line to ask for help with installing a script or otherwise setting up your site. Ask them how much help they're willing to offer with these types of problems and whether or not you'll have to pay for their technical expertise. In general, tech support should be willing to help you with questions directly relating to your account and setup, such as which e-mail servers to use when setting up your e-mail account. However, programming help and installing non-supported programs will probably cost a small fee, if it's available at all. If you think you'll be installing a lot of custom scripts, and you aren't a programmer yet, try to find a Web hosting company that offers that type of technical support, even for a fee. You'll have a much more positive experience with your Web service if technical support is available to you when you need it.

Avoiding Common Provider Scams

Although most Web hosting companies are legitimate businesses, some are run by unscrupulous people who are just trying to take advantage of customers. It's like any industry; you have to watch out for the bad guys when you're looking for the good

www.it-ebooks.info

ones. Naturally, you should stay away from any Web hosting company that sends you unsolicited e-mail about their services. By spamming you, such companies show no respect for your time and are unlikely to respect you as a customer.

As you research your Web hosting companies, log on to `http://groups.google.com/` to read `alt.www.webmaster` (or use your newsgroup reader to check the group out instead) and ask about the specific company to see if anyone has had experience with it. This newsgroup is also a good resource for other Webmaster questions as you go through the process of creating the rest of your Web site.

For online reviews and rankings of Web hosting companies, visit `www.epinions.com/inet-Web_Hosting-All`.

Finally, be sure to read the Terms of Service before you agree to the Web hosting service. Not all Terms of Service agreements are identical. Some Web hosts reserve the right to change their Terms of Service agreements without notification, or to make major technical changes without notification. Changing IP addresses, though common, is a major technical change that should require that the Web hosting company notify you. Find out if they do before you sign on.

Thinking like a crook

As with any major investment, the old adage *caveat emptor* or "Let the buyer beware" still holds true when choosing a Web hosting company. All Web hosting companies will tell you they're the fastest, best, most affordable deal in the world. Beware of fly-by-night operations that have only been in service a short time. Watch out for any signs that your Web hosting "professionals" are, in fact, a couple of unqualified high school kids. (Customer service that's only offered during after-school hours is a big tip-off!) If it looks too good to be true, it probably is.

As you go into the review forums and online, remember that some unscrupulous people will post positive reviews about their own companies without providing information about their own interest in the company. So, in some reviews, you might get someone claiming that this is the best Web hosting company in the world. Make sure you pick a Web hosting company that has more than one reviewer reviewing it, and check more than one hosting reviews site as well.

Suffering domain name theft

Domain name theft rarely happens, but when it does, it's serious. Because your domain name is registered through a registration service, it's hard for someone to outright hijack it without any legal recourse.

There are three main ways that someone can steal your domain name. First, they can hack into the DNS system and transfer your domain name entries to their site. Once detected, this kind of assault is easily remedied through your domain name registrar. The second method is to illegally convince the registrar to transfer the name. This is

www.it-ebooks.info

done through forged documents, faxes, and so on. If that happens, you may find that it's a bit more difficult to clear up. You'll have to prove that the transfer was fraudulent.

There is, however, one way for someone to steal your domain name and flat out get away with it. Most Web space providers are willing to handle your domain name registration for you, and most of them are scrupulously honest about handling the process. But you must make sure to read the fine print if you let someone else do it for you. Some providers will register the domain name to themselves instead of to you. If they do, you're out of luck.

The best solution for this is to do the registration yourself. It's not a difficult process—all you do is fill out an online form at any registrar. And you're covered that way.

Getting "unlimited" traffic or space

Unlimited traffic or Web space sounds like a great deal. And it is, if you frequently use a lot of bandwidth or have large files. For example, many adult Web sites have large files and high amounts of traffic, and there's only so much processor and network bandwidth to go around. If your Web hosting company offers unlimited traffic to you, they're also offering it to everyone. Because there is no foolproof way to get unlimited bandwidth to the Internet, this is an empty promise that will leave your Web site stranded. Frequent downtime and slower speeds will haunt your site, and you'll soon find that users leave your site for greener (and faster) pastures.

The same goes for unlimited storage space. Although network speed isn't as much of a problem, unlimited storage space is an open invitation to anyone who makes large files available for download. They're also an open invite to anyone who wants to download those large files, which will ultimately lead to network slow-downs for you, too.

When it comes down to it, any Web hosting company that offers unlimited traffic or storage space isn't interested in providing customer satisfaction. Unlimited traffic and space are a setup for customer dissatisfaction. If you absolutely think you need to have unlimited bandwidth, find a Web hosting company that doesn't permit Napster-like file serving or adult content.

Rushing deadlines for special deals

"Order by tomorrow night and get $15 off!" "Order now for a free T-Shirt!" "Sign up now for cheaper long distance!" We've all heard these kinds of deals, and they can sometimes be a real bargain. When it comes to Web hosting, however, there are few deadline deals that can't be found elsewhere, later. There are so many thousands of Web hosting companies that you won't have a problem finding someone who will beat that deadline-based deal. The only time you should sign on with one of these limited-time offers is when the Web hosting company is one of the two or three that you've researched, it's beaten the tracert test, and you're satisfied with its Terms of

www.it-ebooks.info

Services agreement and customer service options. In other words, if you were probably going to sign up with them anyway, then go for it.

Exploring Bells and Whistles

Most bells and whistles you might be looking for are probably available on any standard virtual hosting account. Typically, you should receive the following services:

- ✦ One to five e-mail accounts
- ✦ Message redirection
- ✦ Autoresponders
- ✦ Web statistic program to help you track traffic to your site, and possibly provide information on your visitors
- ✦ Web-based control panel for you to use to manage your Web site and accounts.

Getting extra e-mail accounts

Most Web hosting accounts come with an e-mail account, if not several. However, because e-mail accounts are fairly easy to set up, you can usually get additional e-mail accounts added to your account by contacting customer service for your Web hosting company. These extra e-mail accounts are not very expensive—they can be had for as little as $2.00 for 10 e-mail accounts per month.

Redirecting messages

A *redirect address* is an e-mail address that forwards all mail sent to that address to another e-mail address. There are several advantages to redirecting e-mail. First, you can make your organization seem larger by having multiple e-mail addresses for user questions. For example, `info@yourdomain.com`, `pricing@yourdomain.com`, and `techsupport@yourdomain.com` can all go to `yourname@yourdomain.com` if you wear many hats in your organization.

A redirect address also helps you filter messages in your inbox. In most e-mail programs, you can set a filter to move and file messages into specific directories automatically. Then, you just go to the directory to view mail that matches the filter criteria. For example, you could have a filter that says any mail sent to `techsupport@yourdomain.com` should be filed in a folder called Technical Support. When you read mail in that folder, you know the only questions you'll be asked are about technical support. And, you won't have tech support e-mail cluttering up your inbox.

Many Web hosting companies have Web-based control panels to set up redirect addresses. If available, use these to set up as much of your site as possible.

www.it-ebooks.info

If you don't have a Web-based control panel, but you do have a telnet (or shell) account on a Unix or Linux server, you can often create redirect accounts in the shell program. Log in to your shell account using your username and password. Once logged in, at the prompt type:

```
ls -a
```

This should produce a list of all files in your directory, including hidden files. Hidden files are usually system files. On Unix-flavor systems, their file names start with a period. Thus, your directory listing might look something like Listing 5-1.

Listing 5-1: **Results of ls -a**

```
aw1:/home2/scottrell$ ls -a
./              .htgroup        .redirect       anonftp@       maillists/
../             .htpasswd       .redirect.bk1*  dead.letter    mbox
.bash_history   .lynxrc         .redirect.bk3*  du.dat         pass*
.bash_logout    .passwd/        .screenrc       infobots/      personal/
.bash_profile   .pine-debug1    .wusage/        laptop/        www@
.bashrc         .pinerc         access-log@     mail/
```

There's only one thing you really need to pay attention to here. The `.redirect` file contains your forwarding addresses. You can edit this using a text editor, such as vi or emacs online, or download it with FTP, edit it in your text editor, and upload it again.

Adding or changing redirected addresses is fairly easy. Each line in the `.redirect` file is a new forwarding address. Enter the name of the address, without the domain information (yourname, not yourname@yourdomain.com). Put in a space, and then enter the full destination address. Listing 5-2 shows an example of a `.redirect` file.



The forwarding address in a .redirect is not a full e-mail address! It's just the user name. For instance, Voicemail, instead of voicemail@scottrell.com.

Listing 5-2: **A `.redirect` File**

```
voicemail sbryant@ekno.com
travelmail sbryant@ekno.com
phone mortaine@myvzw.com
chat scottrell@scottrell.com
steph scottrell@scottrell.com
webmaster scottrell@scottrell.com
speech scottrell@scottrell.com
default stephanie@Web-writer.net
biblebook scottrell@scottrell.com
```

www.it-ebooks.info

If you don't have control panels or a shell account, or you don't know how to use telnet, contact your Web hosting company for help setting up redirect addresses.

Using autoresponders to provide information

An *autoresponder* is similar to a redirect account in that it's not a full-fledged e-mail account on the system. However, unlike a redirect address, an autoresponder sends the contents of a file back to the original sender. A user on your Web site might click a button that says "Send Me More Info!" The user is directed to an e-mail address and sends a message to your `info@yourdomain.com` account. Although you could have this info address set up to redirect to your main account, you can also provide instant access to a standard information sheet using an autoresponder. When the visitor sends an e-mail to the address, he or she receives an automatic reply with the information requested.

You can usually set up an autoresponder to also forward the original message to a separate address. This is a good scheme for setting up a tech support address. You can set up an autoresponder with a link to the most commonly asked questions, but also assure your clients that a more personal response will be coming.

Many Web hosts offer online forms for setting up and updating autoresponders on your Web site. The text of the autoresponder itself is usually saved as a file.

If you're using a telnet or shell account only and don't have a Web-based control panel, look again at the contents of your directory using the ls command. There should be a directory called `autoresponder` or `infobots` or something similar. In Listing 4-1, the directory is called `infobots`. Go to that directory and view the files available. To add a new autoresponder, simply create a text file containing the information you wish to send as your response. Give it the same name as the autoresponder's address. So, if your autoresponder's address will be `support@yourdomain.com`, name the file `support`.

You can create this file using any text editor, including Notepad, but make sure that there is no file extension when you save the file. In Windows, that may be tricky—you might have to go into Windows Explorer and change the name to remove the file extension. If you create the file on a desktop machine, use FTP to upload it to your autoresponder directory.



If you're using telnet or a shell account to create an autoresponder, add an entry for the same e-mail address in your `.redirect` file to also forward a copy of the original message to a separate address.

Getting visitor statistics

Any good Web hosting company will offer weekly server statistics telling you how many people visited your site, and when. These statistics may be as simple as a hit count, just telling you how many visitors you had. Or they can be very extensive,

www.it-ebooks.info

including regional information on where your visitors come from, which pages they viewed and in what order, the times of day they visited, and any error messages they received. Data may be presented in tables, graphs, or charts. An example of a cumulative site report chart is shown in Figure 5-6.

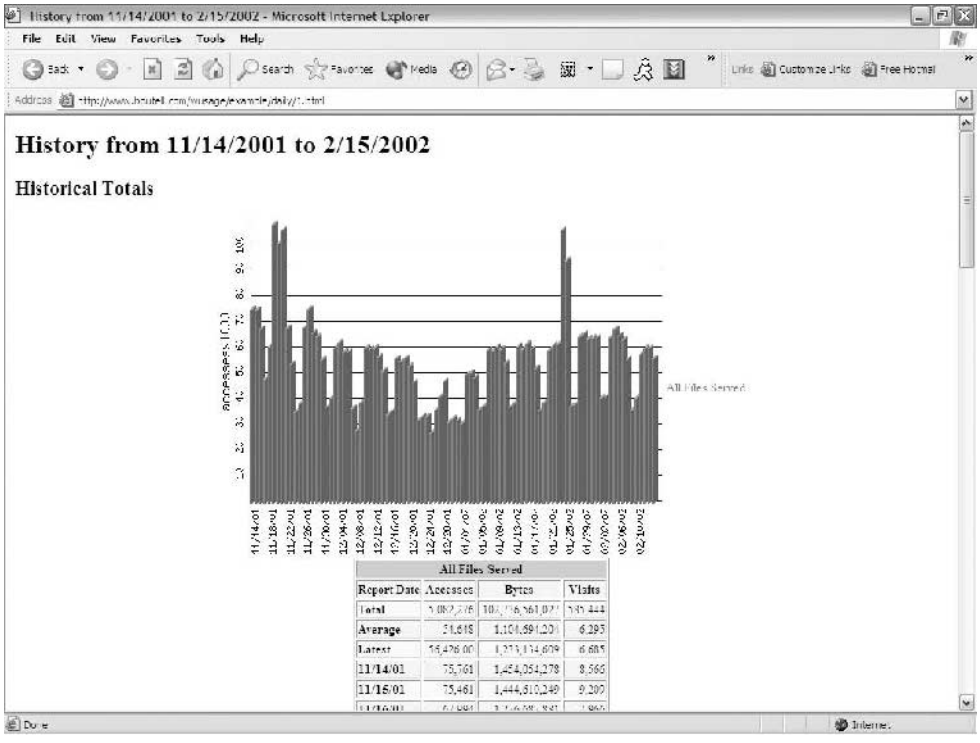


Figure 5-6: Wusage software outputs useful charts.

Most Web hosts also offer more advanced tables as well as weekly reports. These reports are a valuable way for you to track your Web site's development. Beware of any Web host that deletes reports after less than six months.



If you don't have a Web statistics program installed on your server, WebStats provides free Web statistics at `www.webstat.com`.

Using control panels to set site options

Just as you can use control panels to set up e-mail options, you can also generally use control panels or shell commands to set up site options and permissions, including password-protected directories. On Windows Web servers, a password-protected directory means you have to create a new Windows user account to access the files in that directory. In Unix and Linux, however, you can use Basic Authentication, the common method for setting new users, to set these up.

www.it-ebooks.info

If you have a Web-based program to password-protect a directory, follow the instructions of your Webspace provider. Although there will be individual variations in the process with different providers (see Figure 5-7), it essentially works like this:

1. Enter the directory you'd like to protect.
2. Type a user name and a password for that user. Set a password that's easy to remember but hard to guess. The user won't be able to change his or her own password after you create it, although you could change it if you want.
3. Click a button to add that user name and password to your list.
4. Repeat Steps 2 and 3 for as many users as you want to add passwords for.

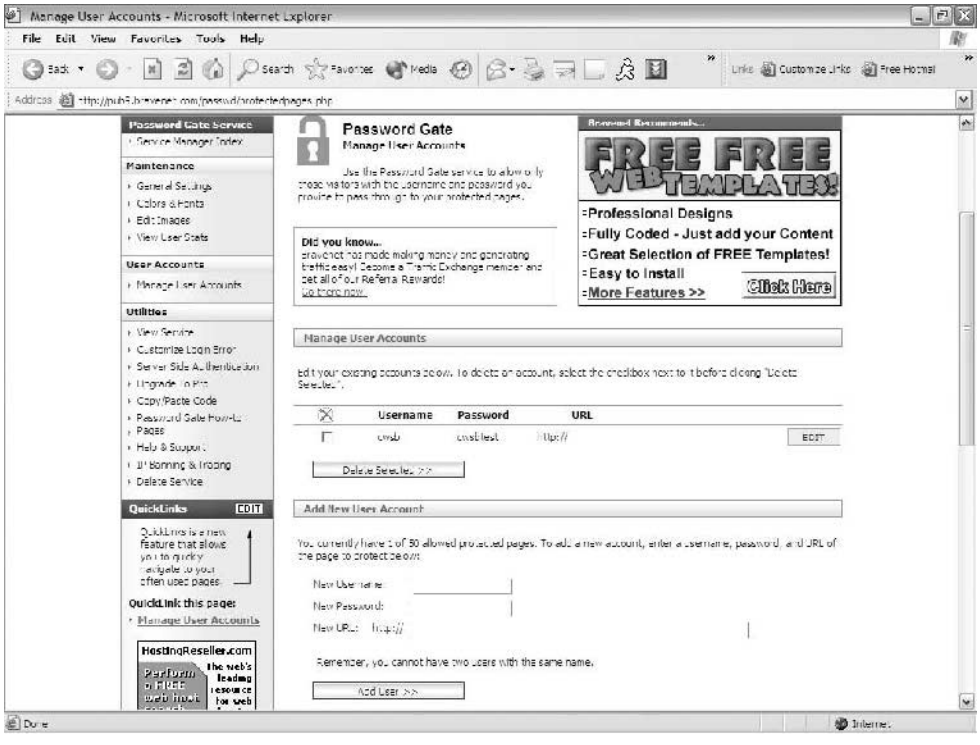


Figure 5-7: Bravenet.com uses a common form for adding authorized users.

Of course, there's another option if you're a telnet or shell account user. If you have a shell account on a Linux- or Unix-based machine, you can set up a similar Basic Authentication scheme. First, log on to your account and go to your home directory.

Type **htpasswd -c .htpasswd username** to create a new user name and password, as shown in Listing 5-3.

www.it-ebooks.info

Listing 5-3: Creating a Username and Password for Basic Authentication

```
aw1:/home2/scottrell$ htpasswd .htpasswd MySelf
Adding user MySelf
New password: MyPassword
Re-type new password: MyPassword
```



Note that the password itself ("MyPassword") won't actually appear on screen as you type.

Next, create or edit the `.htaccess` file in the directory you want to protect, as shown in Listing 5-4. Go to that directory, and put this file inside it. Replace the path `/home/` with the actual path to your home directory.

Listing 5-4: Contents of `.htaccess`

```
AuthUserFile /home/.htpasswd
AuthGroupFile /home/.htgroup
AuthName Restricted_Area
AuthType Basic
<Limit GET>
require user personalaccess
</Limit>
```

The most important thing in this file is the `AuthUserFile` reference. This is a file containing the names and encrypted passwords of users. `AuthGroupFile` is optional—use it if you want to group your users so that they all have the same authorization.

`AuthName Restricted_Area` sets the name of this restricted area. We could have as easily named it HalloweenPictures. `AuthType` is `Basic`—keep it that way.

The next bit of code limits the HTTP server from getting anything from the protected directory. In the MySelf example, you could add

```
<Limit GET>
require user MySelf
</Limit>
```

to the `.htaccess` file in the protected directory.

If you don't have a Linux or Unix server, contact your Web hosting service about setting up password-protected directories.

www.it-ebooks.info



If you're really stuck for a password-protection scheme, you can use Microsoft FrontPage (with the server extensions) to create users and passwords for your FrontPage webs.

Becoming a reseller

If you decide to resell your Web space, keep in mind the kinds of questions you asked your own Web hosting company before you signed on with it. Be ready to handle technical and network difficulties, many of which may be beyond your control. Be prepared to offer satisfactory customer service, competitive prices, and a meticulous payment tracking service. And of course, make sure your Web hosting Terms of Service permits you to resell your space.

Here's a checklist of things to do to become a Web space reseller:

- ✦ Make sure you're permitted to resell Web space through your Web host company. Some Web hosts offer a discount for signing up additional virtual domains.
- ✦ Decide on a Terms of Service agreement that is fair to both you and your clients.
- ✦ Set up a merchant account with a bank or credit card company so you can take payments by credit card.
- ✦ Install a secure Web server to protect your clients' credit card data when it's transferred online.
- ✦ Figure out how much it will cost you in costs and overhead, and set a competitive price that still covers your expenses.
- ✦ Set up a separate phone line for incoming customer support calls.
- ✦ Set up a blank account to serve as a template. Give it its own e-mail or shell account or both, domain name (if you decide to offer virtual hosting), FTP account, and Web space. Put all the files you would ordinarily add to any new account (pre-installed CGI scripts, and so on) into this account.
- ✦ When you sign up new clients, copy the contents of your template account to the new client's account, changing filenames and paths as needed.
- ✦ Follow-up with your clients. Send them a personal e-mail or call to make sure the Web space is working out. A five-minute phone call will go a long way toward customer loyalty!

Uploading Your Pages

The last step is to upload your Web pages to the server. After all, if you've got all these great Web pages sitting on your hard drive at home, they're not doing anyone any good. So, you need to put them onto your Web server using FTP or HTTP.

www.it-ebooks.info

Transferring with FTP

The most reliable way to transfer files originated over 25 years ago, in the days of the Internet's infancy. File Transfer Protocol (FTP) was a very simple system by which one computer could send a file to another computer, regardless of what operating system or platform the two computers were using. It was one of the first cross-platform file transfer types, and it quickly became the most reliable.

Today, FTP is still the most stable, reliable way to move something from your computer to another computer. FTP programs are available in every platform and nearly every operating system, and many are even built into the operating system package. If you use a PC, you can open a DOS window and type **ftp** at the prompt. If you're connected to the Internet, you'll be able to connect to any FTP server in the world.

Putting your Web pages onto the server with FTP is fairly easy, and this section shows you how, using the shareware program WS_FTP.

The first step is to connect to your Web server. Open WS_FTP, and you'll see a dialog box asking you to choose a server, as shown in Figure 5-8.

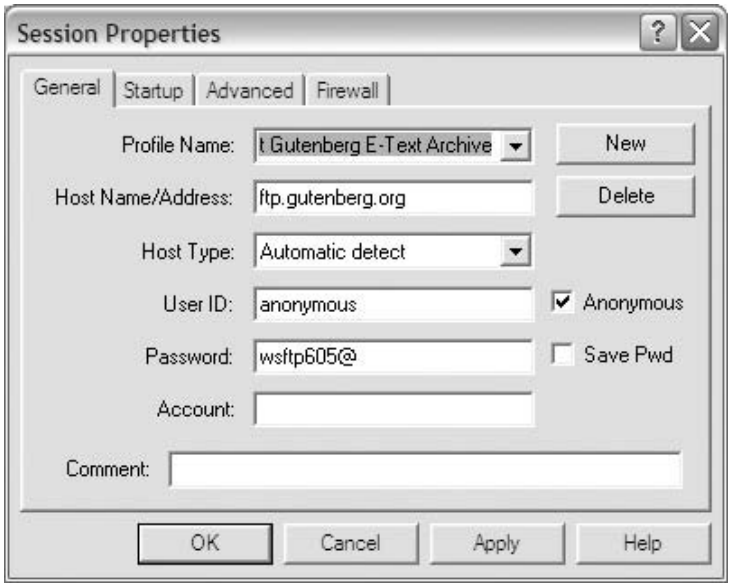


Figure 5-8: Pick a server from the Profile Name drop-down list box, or create a new one by typing in a new Profile Name.

WS_FTP comes with several preset FTP sites where you can find many files and programs available for free. However, you're not looking to download programs today. So, enter a new name for your Web server into the Profile Name text box. Type the host name, which is something like `ftp.yourdomain.com` or just `yourdomain.com`. You may need to ask your Web host provider for this information.

www.it-ebooks.info

Leave the Host Type text box set to Automatic Detect, unless you have other instructions from your Web host company. Enter your account login name in the User ID text box, and your password. For security reasons, your password is not displayed onscreen when you type it in.

Now, click the Startup tab to change your initial directory settings, as shown in Figure 5-9.

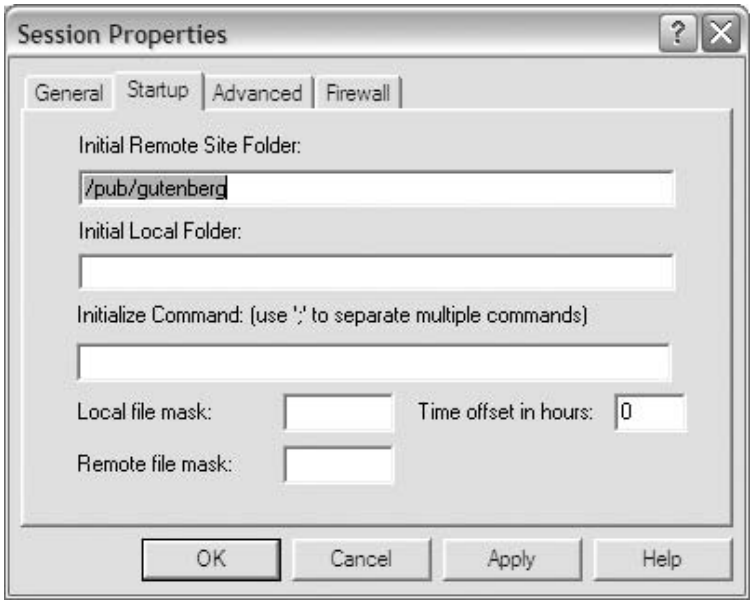


Figure 5-9: You can set the initial directories on the Startup tab.

Enter the path (from your login home directory) to your Web space on the Web server's computer in the Initial Remote Site Folder Directory text box. Enter the path to your local Web folder (on your hard drive) in the Initial Local Folder text box. Leave the rest of the fields blank; they're for advanced FTP programming and automation.

Next, select the Advanced tab, shown in Figure 5-10. Here, you can tell WS_FTP to keep trying to connect if you can't initially log on to your Web server. In addition, there's a network timeout option, so if there's no activity detected, WS_FTP will automatically close the connection.

The Remote Port is the port setting for FTP. Unless instructed otherwise by your Web hosting company, this value is always 21. Finally, Passive Transfers are required for certain network configurations, and are generally a good idea. A Passive Transfer prevents the FTP server from initiating contact during a connection—a little security feature to thwart hackers. Unless your Web hosting company is using an old version of FTP that doesn't support Passive Transfers, turn this option on.

www.it-ebooks.info

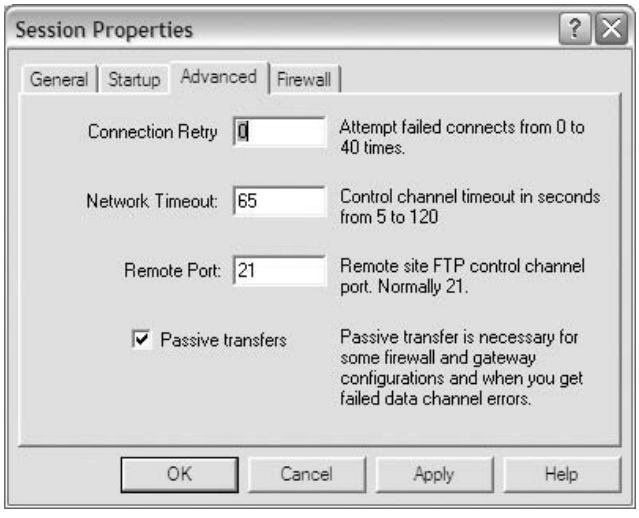


Figure 5-10: The Advanced tab enables you to set timeout times and how many times to attempt the connection.

Finally, click the Firewall tab (see Figure 5-11). You only need to specify these settings if you must go through a firewall to connect to the Internet. Your system administrator (the person responsible for the firewall) will help you fill in the required information and get through the firewall.

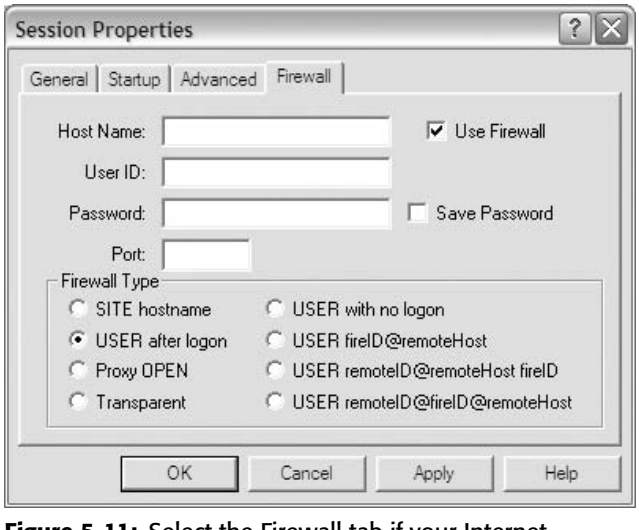


Figure 5-11: Select the Firewall tab if your Internet connection goes through a firewall server.

After you've filled in the fields and are ready to connect, click OK. As long as you've entered all the correct information, WS_FTP should connect you to your Web server and log you in. You'll be transferred to your initial directories, as shown in Figure 5-12.

www.it-ebooks.info

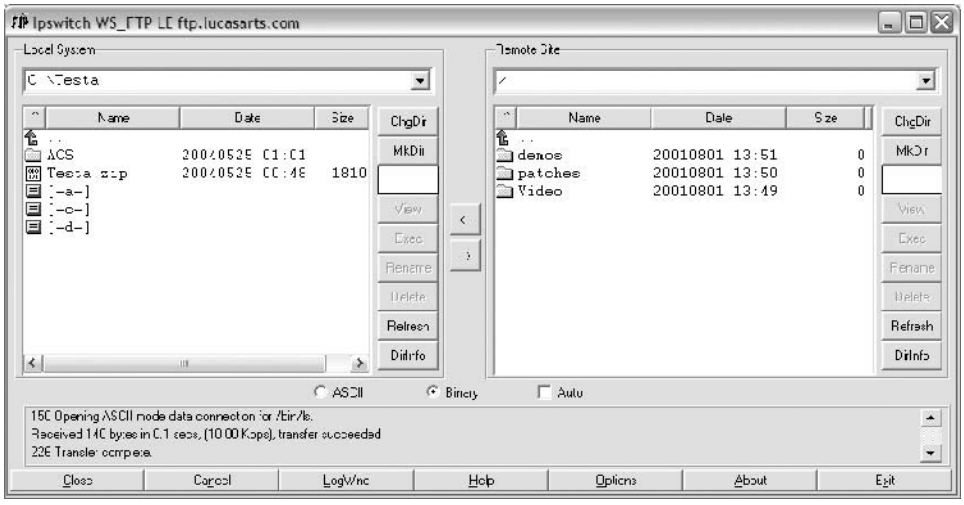


Figure 5-12: Once you log in, you're transferred to the initial directories you set up.

Look carefully at Figure 5-12. On the left-hand side is the Local System panel, which shows the contents of your hard drive. On the right-hand side is the Remote Site panel, which shows the contents of your Web server. The buttons next to each panel are explained in Table 5-2. There's also a blank field into which you can enter text.

Table 5-2
Transferring via FTP

Button	***Description***
ChgDir	Prompts you to enter the directory you'd like to change to. You can also browse directories by double-clicking the folders, or by double-clicking the up arrow at the top of the panel to go up one directory.
MkDir	Prompts you to enter the name of the directory you'd like to create. Once you create a new directory, you can move to it through ChgDir or by double-clicking it.
View	Opens the selected file in whatever program you've selected for viewing files of that type.
Exec	Runs a program if you select a program file (an executable program, often ending in .exe on Windows machines).
Rename	Enables you to rename the selected file or directory.
Delete	Prompts you to erase the selected file or directory.
Refresh	Reloads the directory listing, reflecting any changes made to the files.
DirInfo	Produces a text file containing information about the current directory and its contents.

www.it-ebooks.info

There are two option buttons below these panels: ASCII and Binary. There's also a check box for Auto. In FTP terms, ASCII means any plain-text file, such as .txt and .html files. If you can open and edit it in NotePad or TextEdit, it's probably ASCII. Binary stands for all other files—graphics, programs, compressed archives (.zip or .sit), Word documents, and so on. Select the file type that you'll be transferring—ASCII for HTML files, or Binary if you're uploading graphics. Check Auto if you want to select and send more than one type of file at a time.



The ASCII versus Binary option is very important in FTP! If you upload a file and it doesn't seem to work, try transferring it in the correct type.

There are two buttons in the center, one pointing left and one pointing right. To upload a file, select it on the left side (the Local System side) and press the arrow pointing to the right. The file will be transferred to the server (see Figure 5-13).

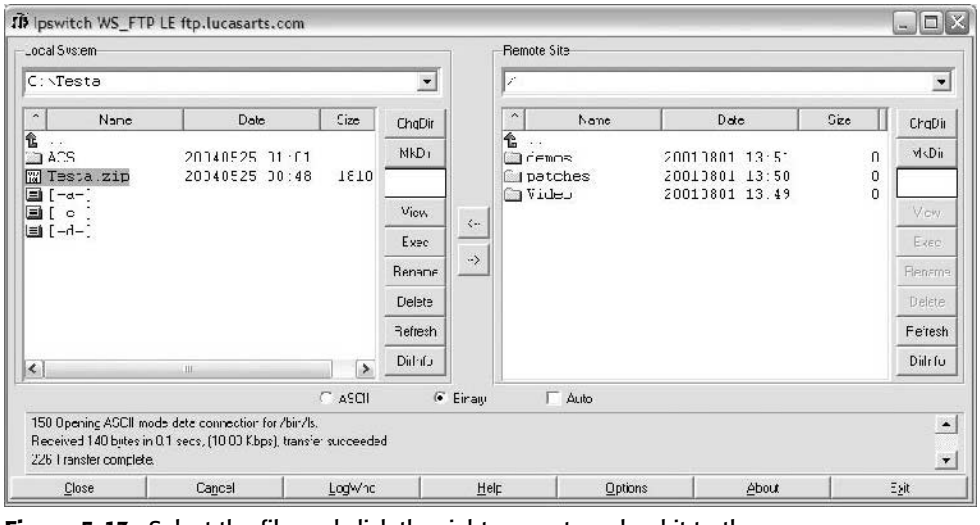


Figure 5-13: Select the file and click the right arrow to upload it to the server.



You can also double-click the file to copy it from its current panel to the opposite panel (local to remote or vice versa).

When you're done transferring files, click Close to end your FTP session, and Exit to quit WS_FTP.

There are literally hundreds, if not thousands, of FTP programs available, however, so if you're not happy with WS_FTP or CuteFTP, feel free to go online and find something better. For Mac users, the most popular by far is Fetch, which might already be installed on your Macintosh.

www.it-ebooks.info

Uploading with Web Browsers via HTTP

If your Web hosting company offers this service, you can also upload your files using an HTML form and HTTP. Many Web hosts provide a form for uploading files to the server, as is shown in Figure 5-14.

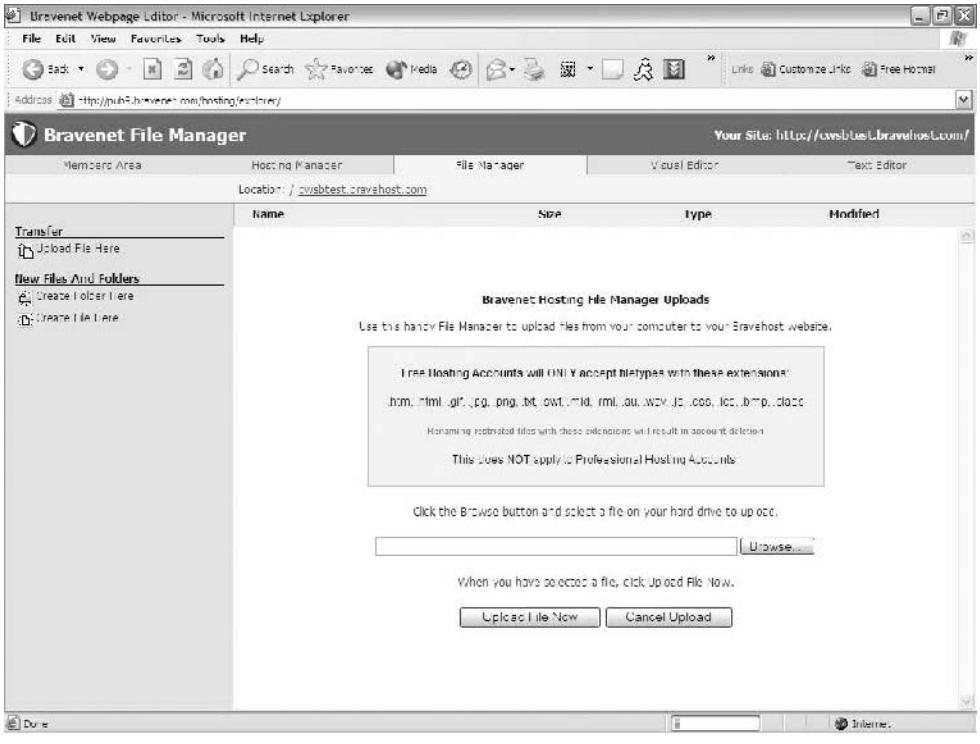


Figure 5-14: You can upload files using HTML forms.

To upload a file, enter the path and filename, or click Browse to find it on your hard drive. After you find the file, click the Submit button for the form to upload it. The file is now available on your server in your Web space or through your control panels.

Summary

As you can see, it's pretty easy to find a home for your Web pages and to put them there. You can make it more complicated, of course, but you don't have to. If you can use an online form, you can get set up with a Web server and domain name. If you can use a program such as WS_FTP, or if your Web hosting service has control panels to permit you to upload to your Web server, you can easily put your pages up on the Internet and become a published Web developer!

www.it-ebooks.info

This chapter introduced you to the following concepts and topics:

- ✦ Choosing an easily memorized domain name is one of the keys to online success.
- ✦ You may choose to use your own hardware, leased hardware, or even a software-only Web server solution.
- ✦ Whatever kind of Web server you choose should have plenty of extra space and transfer capability.
- ✦ Carefully research any Webspace provider you plan to use.
- ✦ You can upload your Web pages via either FTP or HTTP, depending on your provider's capabilities.

✦ ✦ ✦

www.it-ebooks.info

P A R T

II

Enhancing Web Pages with Fonts, Images, and Color

In This Part

Chapter 6
Getting Fancy with Text

Chapter 7
Finding Images

Chapter 8
Blending Images into Your Pages

Chapter 9
Creating and Enhancing Images

Chapter 10
Adding Color Throughout Your Site

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 6

Getting Fancy with Text

In This Chapter

Customizing text with character styles

Using the FONT and BASEFONT elements

Aligning and indenting text

Choosing character sets

Using special characters

Text is the heart of all Web pages. No matter how graphically oriented the site is, most information is transmitted by words. This chapter deals with how to use textual characters to make the most of your Web pages. At times, this chapter's information flies in the face of conventional wisdom and recommends flouting the advice of established standards bodies, but the focus is always on one thing—how to make your job as a Web page designer as easy as it can be.

Customizing Text with Character Styles

Character styles are HTML elements that modify the appearance of the text that they enclose. Character styles are most commonly used to add bold or italic styles to words. Character style elements are inline elements, which means that they do not create a new line because they are intended to be included in the middle of sentences.



See Chapter 3 for a description of inline elements versus block elements as well as a general introduction to adding text to Web pages.

Although the World Wide Web Consortium (W3C), the official standards body for the Web, has tried to phase out character styling elements in favor of Cascading Style Sheets, they are still in wide use. The reason for this is simple—they are easier to learn and more intuitive to use than CSS. CSS, on the other hand, rewards a little bit of study by giving you many more capabilities.



See Chapter 13 for details on using CSS.

www.it-ebooks.info

Italicizing text

Italics are used to place emphasis on particular words for various purposes. In this book, for instance, whenever a technical term is used for the first time, it is italicized. Words from a foreign language that have not been assimilated into your native language are also usually italicized. Thus, an English speaker would italicize *nom de plume,* which is French for pen name, but would not italicize champagne or debris, because these words have become assimilated into English through common usage, even though they are also French in origin.

To add italics to text in HTML, you enclose the text in <I> and </I> tags as in this example:

```
<P>The word <I>italics</I> is italicized.</P>
```

Figure 6-1 shows the results.

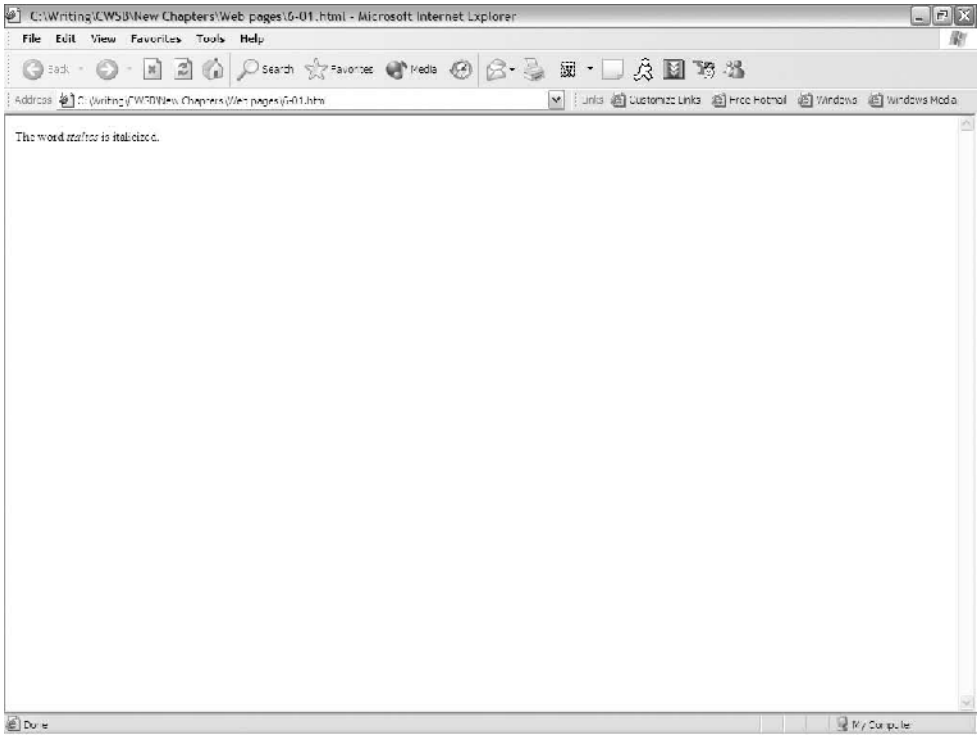


Figure 6-1: Italics are a common way of emphasizing certain words.

Bolding text

As with italics, bold is used for emphasis. Other than to make a word or phrase in a sentence stand out from the surrounding ones, the main use for bold print is to create headings. In HTML, however, heading elements are automatically bolded.

www.it-ebooks.info

To make text bold, enclose it between <B> and </B> tags as in this example:

```
<P>The word <B>bold</B> is in bold print.</P>
```

Figure 6-2 shows the results.

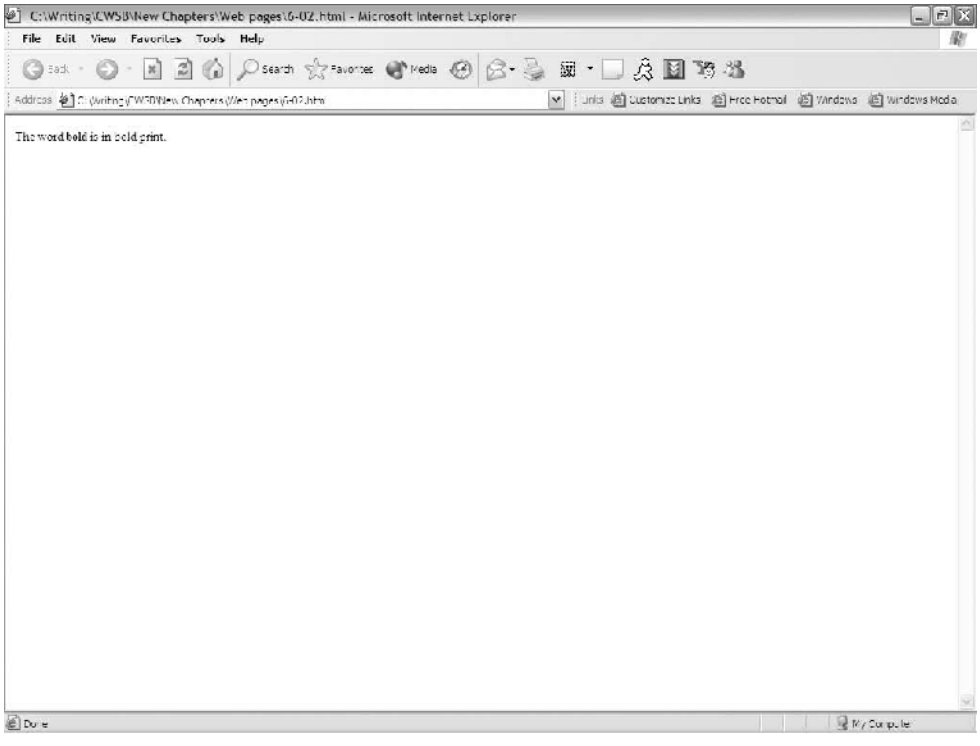


Figure 6-2: Bold print also emphasizes text.

Preformatting text

The basic text on a Web page is in a *proportionally spaced* font—that is, the amount of space taken up by a given number of characters depends on the particular letters used. The word "ion" doesn't take up the same space as the word "won," because a w is wider than an i. With preformatted text, however, each character takes up the same amount of space regardless of its size. This is because it uses a *monospace* font—one space for each character.

Other important differences occur when using preformatted text, as well. For starters, preformatted text maintains all white space (see Chapter 3) as it exists, without trying to condense it. For example, if the text contains two space characters in a row in normal HTML, one of them is eliminated by the Web browser. In preformatted text, however, both space characters are left intact.

www.it-ebooks.info

To set text as preformatted, place it between the <PRE> and </PRE> tags as in this example:

```
<PRE>Preformatted text  goes here.</PRE>
```

Figure 6-3 shows preformatted text with its effect on white space on a Web page.

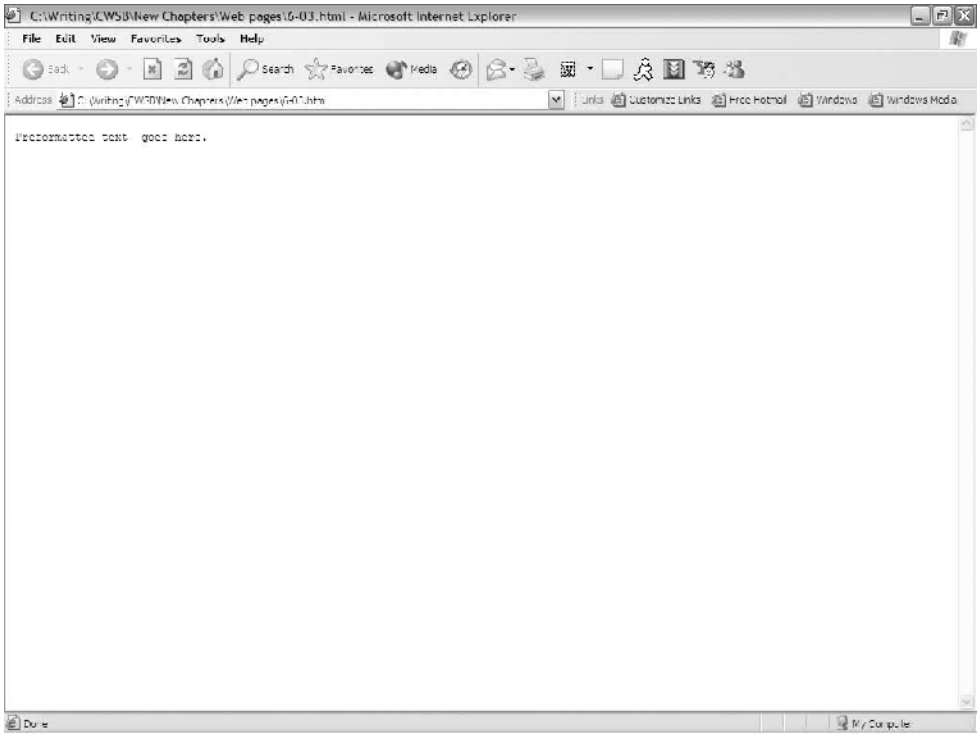


Figure 6-3: Preformatted text is in a monospace font and maintains all white space.



If you use preformatted text, it is best to keep it as simple as it can be. If you add complex text such as superscripts and subscripts (see the next section) or modify the PRE element using CSS (see Chapter 13), Web browsers won't know how to handle the situation and all bets are off.

Adding superscripts and subscripts

Generally, all characters occupy the same level on a line. However, if you need to add something such as the mathematical formula $E = mc^2$ or designate water as H_2O, you'll need to format the 2 as a superscript to raise it above the line or a subscript to drop it down a notch.

www.it-ebooks.info

You can use a few special characters for terms such as squared or cubed (see "Using Special Characters," later in this chapter), but it's better in most cases to go with a generic solution that is applicable to all your needs for raising or lowering characters.

Another common problem on the Web is that the trademark symbol is not well supported—if you add the symbol to your page, there is no guarantee that it will actually show up in a Web browser for a variety of reasons. While this problem seems tangential to the current topic, its simple and straightforward solution isn't—just superscript the letters TM. You superscript characters by putting them between the ^{and} tags as in the following code:

```
<P>We guarantee that CatMover<SUP>TM</SUP> will keep your
doorways unblocked.</P>
```

Similarly, subscripts are done with _{and} tags:

```
<P>It was nice to get some good old H<SUB>2</SUB>O after so
   long in the desert.</P>
```

Figure 6-4 shows superscripts and subscripts in action.

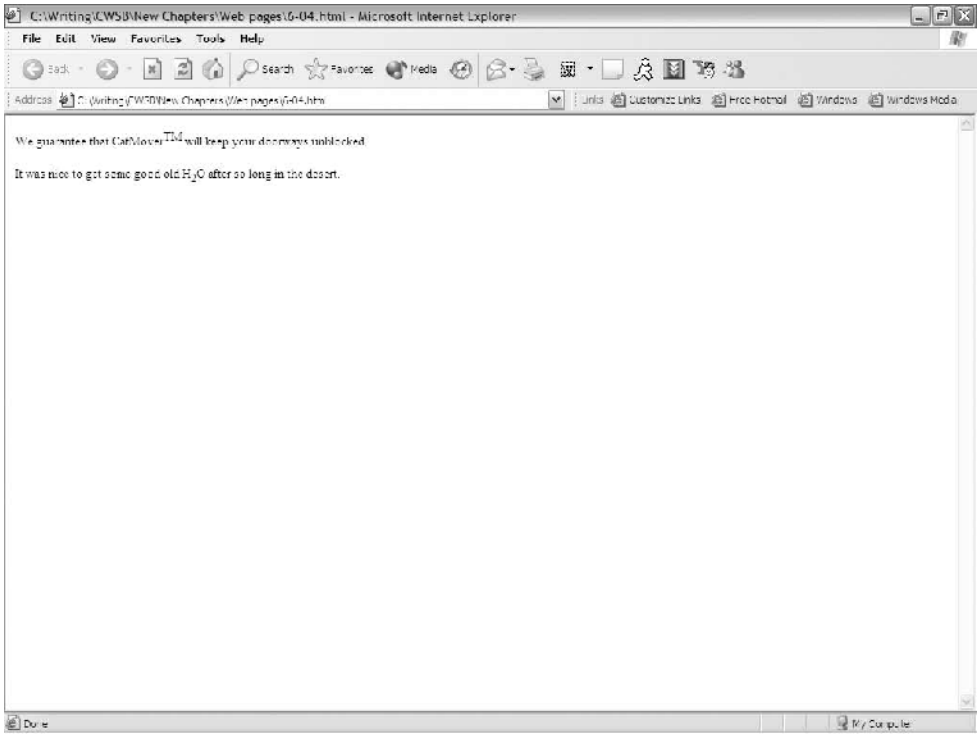


Figure 6-4: Superscripts raise characters whereas subscripts lower them.

www.it-ebooks.info

Using the FONT and BASEFONT Elements

The FONT element can be used to specify size, font face, and color for the text it encloses. The BASEFONT element, which is enclosed in the HEAD element of the Web page, serves an identical function, but for all the text on an entire Web page.

Specifying a size

For either the FONT or BASEFONT element, size is specified by the size attribute. Font sizes in HTML run from 1 (the smallest) through 7 (the largest). The default font size, if none is specified, is size 3, which generally is taken to mean a 12-point font. A *point* is $\frac{1}{72}$ of an inch, so a 12-point font is $\frac{1}{6}$ of an inch high. However, the actual size of a font on a visitor's screen is dependent on their browser preference settings, so all you can do is create a hoped-for size. Figure 6-5 shows the relative sizes of the fonts on a normal Web page.

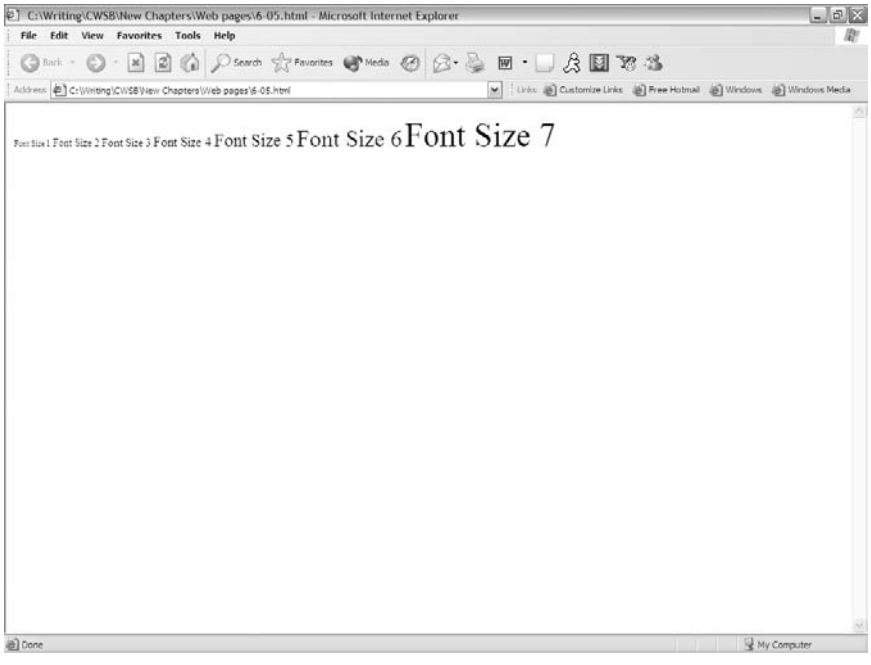


Figure 6-5: Font sizes vary from 1 to 7.



You can set an exact size for a font by using cascading style sheets. See Chapter 13 for more information.

To specify a font size of 6, write code as in this example:

```
<FONT size="6">Text goes here.</FONT>
```

www.it-ebooks.info

The BASEFONT element works exactly the same way, except that it has no end tag—there is no such thing as </BASEFONT>. To set the global font size for the Web page to size 6, you would use the following code:

```
<HEAD>
<BASEFONT size="6">
</HEAD>
```

Using relative size

Font size can also be set relative to the default font size (or to the BASEFONT size, if one exists). To make the affected characters one size larger than the default, use this code:

```
<FONT size="+1">Text goes here.</FONT>
```

To make the characters two sizes smaller than the default, use a minus sign instead:

```
<FONT size="-2">Text goes here.</FONT>
```

Remember that there are only seven font sizes possible in HTML. You can't make anything smaller than size 1 or larger than size 7. So, if you had a default font size of 3 and used the following code, you'd still have only a size 7 font as a result:

```
<FONT size="+6">Text goes here.</FONT>
```

There is no way to get to a size 9 font, because it doesn't exist.

Comparing font size with heading size

Whoever first came up with heading sizes 1 through 6 and font sizes 1 through 7 had a keen sense of proportionality. If only there were an H0, they would fall into a perfect inverse ratio. The way it works is that an H1 heading is font size 6, an H2 heading is font size 5, and so forth, all the way down to where an H6 heading is font size 1. The only difference is that the heading text is automatically bold. Figure 6-6 shows this relationship graphically.

Overriding default font faces

By using the face attribute, you can override the default fonts used on Web pages. For example, the normal font used when you display a Web page on Windows systems is Times New Roman. However, it is customary for headings to be in Arial font instead. But on Web pages, headings and text are both in the same font.

The way to change this is to set a particular element on the Web page to a different *font face*. A font face is the way the lettering looks. Times New Roman, for instance, has a series of small lines called *serifs* which many font artists believe help the human eye catch and identify a letter more effectively than fonts which do not have serifs (the latter font faces are known as *sans serif* fonts, *sans* being French for "without"). Figure 6-7 shows fonts both with and without serifs.

www.it-ebooks.info

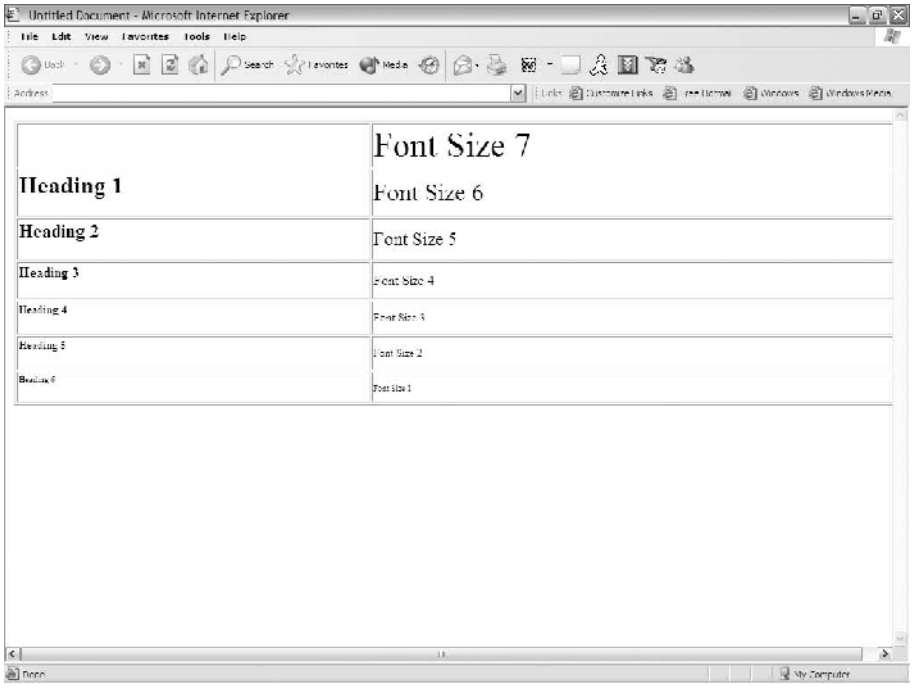


Figure 6-6: Heading sizes and font sizes are inversely proportional.

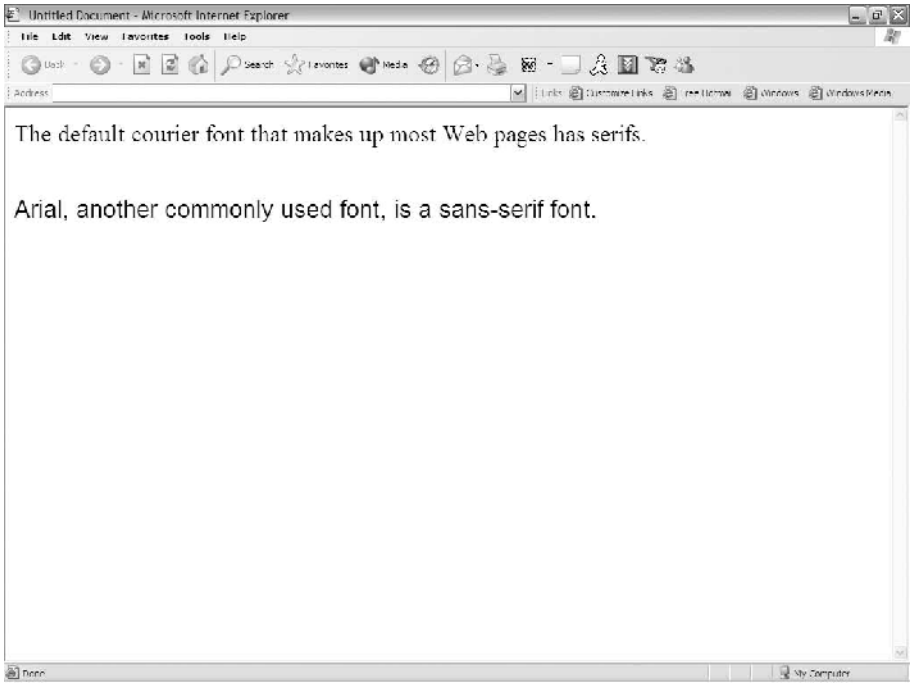


Figure 6-7: Sans serif fonts do not have small lines at the ends of the letters.

www.it-ebooks.info

To change a particular element so that it has a different font face, use the following code:

```
<H1><FONT face="Arial">Changed text goes here.</FONT></H1>
```



This is one of the places where CSS shines. Check out Chapter 13 for information on how to change all instances of headers to a different font.

It is customary to specify more than one font face because not everyone using the World Wide Web is using the same computer system. While the most popular sans serif font is called Arial on Windows computers, it is called Helvetica on Macintosh systems. To take things one step further, both Arial and Helvetica are in the sans serif family. Thus, to cover all the bases, you would need to specify the font face as follows:

```
<H1><FONT face="Arial, Helvetica, sans-serif">Changed text goes
here.</FONT></H1>
```

By doing this, you enable the system used by your site visitors the greatest latitude in choosing a display font. If Arial is not available, the system tries for Helvetica. If neither is available, a different font from the sans serif family is used instead. This approach ensures the likelihood that a visitor sees things the way you intended them to be seen.

This can be set globally for the entire Web page by using the BASEFONT element as follows:

```
<BASEFONT face="Arial, Helvetica, sans-serif">
```

Coloring text

The color of particular characters can be set with the color attribute of the FONT or BASEFONT element. To set it for a particular group of text, down to even a single character, use the following code:

```
<FONT color="red">Colored text goes here.</FONT>
```

To set all the text on the Web page to the same color, you can use the BASEFONT element as in this example:

```
<BASEFONT color="red">
```



The best way to set the text color for the whole page is to use the text attribute of the BODY element (see Chapter 3).

Aligning and Indenting Text

The P (paragraph) element has four possible alignments—left, center, right, and justify. The same is true of the six heading elements (H1 through H6, sometimes

www.it-ebooks.info

described collectively as the `Hx` elements, where *x* stands for the number). If you want to make a paragraph right aligned, you can do it as in this example:

```
<P align="right">Paragraph text goes here.</P>
```

An H1 element can be centered with this code:

```
<H1 align="center">Heading text goes here.</H1>
```

For justified text, simply use the word `justify` instead of `right` or `center`. Because left alignment is the default alignment, you don't have to specify it. Figure 6-8 shows some paragraphs and headings with various alignments.

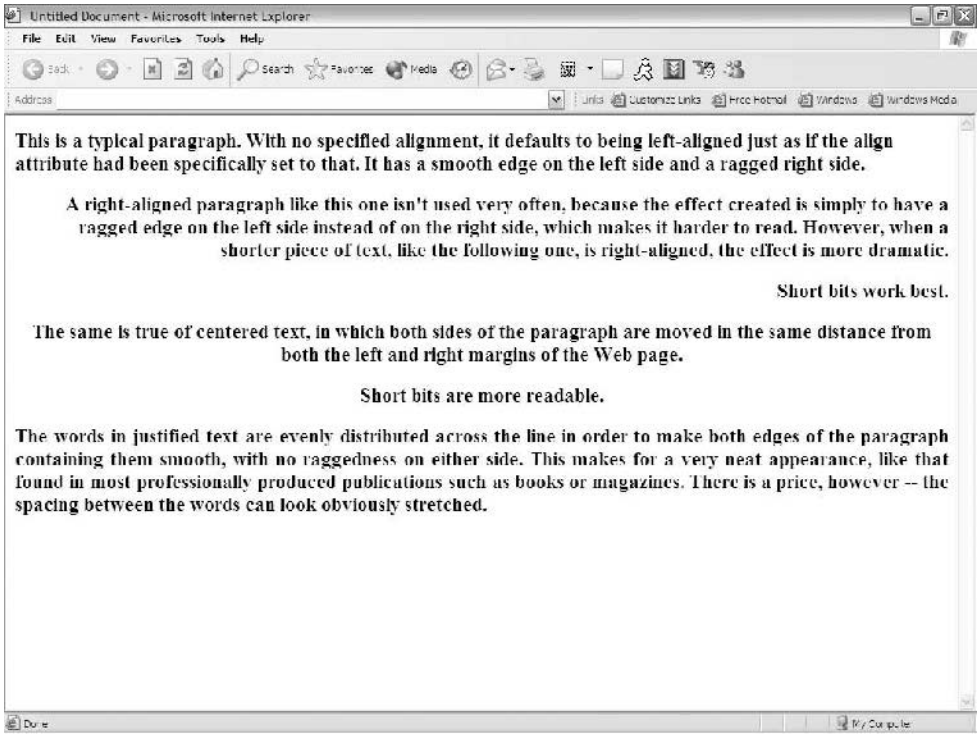


Figure 6-8: Different alignment values create different results.

Dealing with deprecation

The `CENTER` element is a simple and easy way of aligning text, but it's frowned upon by the World Wide Web Consortium, which suggests doing it in a more complicated and potentially confusing manner. According to the HTML standard, "A deprecated element or attribute is one that has been outdated by newer constructs." This would be no problem if the definition worked effectively, but in many cases, it has been applied to elements and attributes for which there is no better alternative available.

www.it-ebooks.info

The W3C's reasoning relies on some old arguments about whether or not HTML should include elements and attributes that define how things look or whether or not elements and attributes should describe the informational content of Web pages. It's very much a case of shutting the barn door after the horse is far over the horizon. Of course, these elements and attributes already exist and are already in worldwide use. It's as if you're trying to decide whether the English language should or shouldn't have adverbs in it. Nobody's going to tear down the Web and rebuild all the old pages to suit the W3C, and no sane browser maker is going to phase out backward compatibility, either. In fact, the HTML standard itself states that browsers should continue to support deprecated elements and attributes. All of which means that these deprecated methods are still fully functional, a lot easier than any alternative yet presented, and will be around for a long, long time to come.

Let's run a few examples past you to show just what we mean. Let's say you want to center some text. We've already discussed how to do it for `P` and `Hx` elements. Here's another way to go about it:

```
<CENTER>Text to be centered goes here.</CENTER>
```

Simple, elegant, and as easy as it gets. And you can put just about anything between the `<CENTER>` and `</CENTER>` tags, including images (see Chapter 8). It has the same effect as writing this code:

```
<P align="center">Text to be centered goes here.</P>
```

But you're not supposed to use the `CENTER` element any more, or the `align` attribute, according to the W3C. So, how do you do it instead? Hang on to your hats, folks. Here comes the official way.

First, you need to add the following code into the `HEAD` element of your Web page:

```
<HEAD>
<STYLE type="text/css">
DIV.whatever {text-align: right}
</STYLE>
</HEAD>
```

This establishes a CSS class called "whatever" that has a right alignment. Don't worry if you don't understand CSS at this point—we cover it in Chapter 13. Just bear with us while we show you how difficult it can be to follow the rules and why we suggest ignoring them.

Next, you need to add this code to the `BODY` element of your Web page:

```
<BODY>
<DIV class="whatever">
<P>Text to be centered goes here.</P>
</DIV>
</BODY>
```

www.it-ebooks.info

This wraps the paragraph inside a DIV element (which does nothing but hold other elements). The DIV is assigned to the "whatever" class you created earlier, and all objects of that class are right aligned.

The end result is a lot of work for nothing. Stick with the easy way. Fortunately, the W3C has no power to enforce its ideas, so you're free to develop your pages your way.



Please don't get the idea that we're against using CSS. It does have its uses, and it can be a very powerful way to control your Web sites. We do, however, believe in using the best tool for any particular job, and using something as complex as CSS to replace simple existing elements is like using an atom bomb to crack open an egg.

Indenting with BLOCKQUOTE

The BLOCKQUOTE element has a certain, specific purpose—to mark off long sections of material quoted from another source. However, it also has an interesting side effect. It causes all the text contained within it to be indented equally from both sides of the screen. Figure 6-9 shows some normal text and some text indented using BLOCKQUOTE.

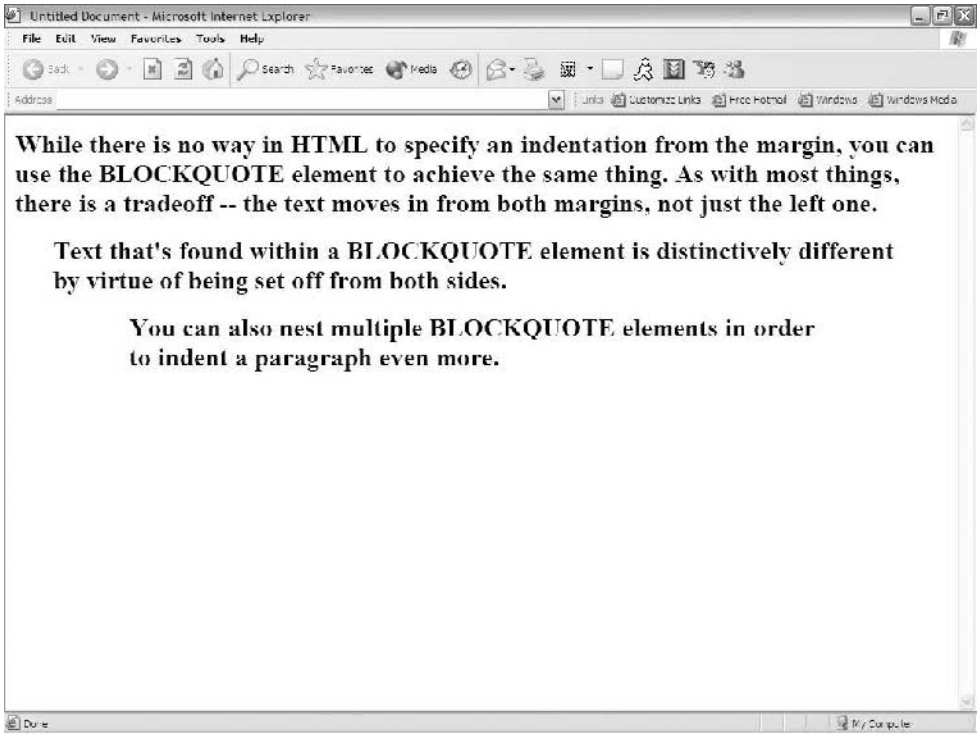


Figure 6-9: The BLOCKQUOTE element enables you to quickly indent text.

www.it-ebooks.info

The process, once again, is simple:

```
<BLOCKQUOTE>Text to be indented goes here.</BLOCKQUOTE>
```

As you might expect, this is too easy, so the W3C has deprecated this use. To the best of our knowledge, it's the only use of an element that has ever been deprecated, even though it isn't possible under the HTML standard's definition of the word. The element itself hasn't been deprecated, but using it to indent text is officially frowned upon. The suggested alternative, as you might have guessed, too, is to use complex CSS instead. Take your pick of which way you want to do it; both of them work.

Choosing Character Sets

A character set is, basically, the manner in which a group of letters or symbols are *mapped*—that is, A is followed by B, and so forth. As a Web page creator, you do not need to concern yourself with the actual mapping, but you still need to pay attention to the character set you're using because different character sets represent the characters used in different languages. HTML recognizes one very important character set, called ISO 10646, also known as Unicode, the Universal Character Set, or UCS (see `www.unicode.org` for more information).



ISO stands for the International Organization for Standardization, the group responsible for promulgating character standards (and many other types of standards as well—the next time you buy film, notice that the film speed is also set by an ISO standard). The number is simply the ISO's reference for the particular character set.

ISO 10646, however, is vast and far too unwieldy for normal use, so there are subsets of it that are normally used instead of the whole thing, the most common of which is known as ISO 8859-1 (also called Western or Latin 1). Until recently, this was the only character set used in HTML.

Character sets are specified in a `META` element with the `charset` attribute, as in the following example code:

```
<HTML>
   <HEAD>
   <META http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
   </HEAD>
   <BODY>
...
   </BODY>
   </HTML>
```



The ellipsis points (. . .) in the preceding code represent the portion of the Web page visible in a browser. It is not a part of HTML code.

www.it-ebooks.info

If you're not interested in the details of what this code means, you can safely skip the next paragraph and just copy the `META` element exactly, substituting whatever character set you're using for `ISO-8859-1`.

For those who want to know the why as well as the how, the `META` element (which must be contained within the `HEAD` element) is a generic method for adding information to a Web page that is not a normal part of HTML. In this case, it is being used to supply information to a Web server about the character encoding method being used on the Web page. The information in the `http-equiv` attribute is read by the Web server handling the page containing the code and is interpreted by the server as if it were an *HTTP header*. HTTP is the Hyptertext Transfer Protocol, the Web's native method of transferring information. Generally, the Web server itself generates the HTTP header information, but specifying an `http-equiv` value in the Web page overrides this behavior, causing the Web server to send the specified value to the Web browser when it sends the Web page to it to be displayed.



The `charset` attribute is poorly named because it does not actually define a character set, but the method of encoding those characters. This and the exact definitions of many other terms relating to character sets and character encoding are the source of furious debate among technical folks. However, for all practical purposes, all a Web page author really needs to know is to use a `charset` value that includes the language grouping he or she is writing in.

While ISO 8859-1 covers most of the characters the average American or Western European Web designer is ever likely to use, the fact that the World Wide Web is, well, worldwide, means that every language on Earth needs to be taken into account. The different characters used in countries such as Tibet, Greenland, Israel, and Russia are not included in ISO 8859-1.

Table 6-1 shows some of the other 8859-family character sets that are available and their common names. The ISO Reference column lists the character set numbers exactly as they should appear in your HTML code's `charset` attribute.

Table 6-1
The ISO 8859 Family of Character Sets

ISO Reference	*Common Name*	*Language Group*
iso-8859-1	Western or Latin 1	West European—Danish, Dutch, English, Faeroese, Finnish, French, German, Icelandic, Irish, Italian, Norwegian, Portuguese, Spanish, and Swedish.
iso-8859-2	Latin 2	Central and East European—Albanian, Czech, English, German, Hungarian, Polish, Rumanian, Serbo-Croatian, Slovak, Slovene, and Swedish
iso-8859-3	Latin3	South European, Maltese, Esperanto
iso-8859-4	Latin4	North European—Afrikaans, Catalan, English, Esperanto, French, Galician, German, Italian, Maltese, and Turkish

www.it-ebooks.info

ISO Reference	*Common Name*	*Language Group*
iso-8859-5	Cyrillic	Bulgarian, Byelorussian, English, Latin, (Slavic) Macedonian, Russian, Serbian, and Ukrainian
iso-8859-6	Arabic	Arabic
iso-8859-7	Greek	Greek
iso-8859-8	Hebrew	Hebrew
iso-8859-9	Latin 5	Albanian, Basque, Breton, Catalan, Danish, Dutch, English, Faroese, Finnish, French (with restrictions), Frisian, Galician, German, Greenlandic, Irish Gaelic (new orthography), Italian, Latin, Luxemburgish, Norwegian, Portuguese, Rhaeto-Romanic, Scottish Gaelic, Spanish, Swedish, and Turkish
iso-8859-10	Latin 6	Danish, English, Estonian, Faroese, Finnish, German, Greenlandic, Icelandic, Sami (Lappish), Latvian, Lithuanian, Norwegian, and Swedish



An alternative Cyrillic character set, KOI8-R, is more popular in Russia than ISO-8859-5.

Further extensions to the 8859 family are in the works:

- ISO 8859-11 (Thai)
- ISO 8859-13 (Celtic or Latin 7)
- ISO 8859-14 (Baltic Rim or Latin 8)
- ISO 8859-15 (Sami or Latin 9)

Oddly enough, the number 12 was skipped—perhaps someone, someday, will propose an ISO standard numbered 8859-12, but it apparently hasn't happened yet.

One other oddity—the 8859 series of standards (at least through ISO 8859-10) was not developed by the ISO, but by the European Computer Manufacturers Association (ECMA). In a further bizarre development, the ISO refuses to put the actual standards on the Web for free access, but sells them, while ECMA makes them available for free at `www.ecma-international.org/publications/standards/Standard.htm`. However, you have to dig through all of ECMA's documentation by topic, because the original versions, of course, don't have the ISO number. To save you the trouble, we've done it for you, and the results are shown in Table 6-2.



These are PDF files, which means that you need to have Adobe Acrobat Reader installed on your Web browser to read them.

In addition to the ISO 8859 series of character sets, several others can be used for the same language groups, but if you stray from the ISO standards, you risk creating

www.it-ebooks.info

Table 6-2
The Original ECMA Versions of the ISO 8859 Standards

ISO Reference	*ECMA Designation*	*URL*
ISO 8859-1 through ISO 8859-4	Latin 1 through Latin 4	www.ecma-international.org/publications/files/ecma-st/ECMA-094.pdf
ISO 8859-5	Cyrillic	www.ecma-international.org/publications/files/ecma-st/ECMA-113.pdf
ISO 8859-6	Arabic	www.ecma-international.org/publications/files/ecma-st/ECMA-114.pdf
ISO 8859-7	Greek	www.ecma-international.org/publications/files/ecma-st/ECMA-118.pdf
ISO 8859-8	Hebrew	www.ecma-international.org/publications/files/ecma-st/ECMA-121.pdf
ISO 8859-9	Latin 5	www.ecma-international.org/publications/files/ecma-st/ECMA-128.pdf
ISO 8859-10	Latin 6	http://www.ecma-international.org/publications/files/ecma-st/ECMA-144.pdf

a Web page that will not display properly on all systems. The Windows-specific character sets created by Microsoft are particularly prone to wrecking a well-designed Web page because they violate the standard encoding methods in order to include some extra characters (see ppewww.ph.gla.ac.uk/~flavell/charset/internat.html for further information on this topic). This can cause serious problems on non-Windows browsers. Table 6-3 shows the Windows-specific character sets and the ISO character sets that are roughly equivalent to them.

Of course, if you're going to be doing a Vietnamese Web page, you have no option but to use windows-1258. In fact, all of the ISO 8859 family is limited to Western hemisphere language groups, although the Unicode project has gone further with its new work (see www.unicode.org for more information). However, there are several other Oriental language character sets, the most common of which are listed in Table 6-4.



For a more extensive list of character sets supported by major Web browsers, visit the World Wide Web Consortium's listing at www.w3.org/International/O-charset-list.html.

www.it-ebooks.info

Table 6-3
Windows-Specific Character Sets

Microsoft Reference	*ISO Equivalent*
windows-1250	iso-8859-2
windows-1251	iso-8859-5
windows-1252	iso-8859-1
windows-1253	iso-8859-7
windows-1254	iso-8859-9
windows-1255	iso-8859-8
windows-1256	iso-8859-6
windows-1257	iso-8859-4
windows-1258	None (Vietnamese language)

Table 6-4
Eastern Hemisphere Character Sets

Language	*Character Set*
Chinese (Simplified)	gb2312
Chinese (Traditional)	big5
Japanese	Shift_JIS
Japanese (JIS)	iso-2022-jp
Japanese (EUC)	EUC-JP
Korean	euc-kr

Using Special Characters and Entities

You may need to use certain characters from time to time that are not found on your keyboard. How, for instance, would you type a nonbreaking space? How would a Web designer in New York type a pound or yen sign? Two methods are available for adding these types of characters—by character code number and by character entity.

When special character standards were first created, they were simply numbered by their order in the list. Thus, character number 160 was followed by character number 161, by character number 162, and so forth. Each special character was added to a Web page by typing a numeric value into the HTML code—` ` for character number 160, for instance (all special character codes begin with an

www.it-ebooks.info

ampersand and end with a semicolon). The problem with this approach is that it's a bit difficult for everyone to remember that character number 160 is a nonbreaking space while character number 8226 is a solid bullet.

That's where *character entities* come into the picture. The vast majority of character entities are simply shortened forms of the character you want to use. It's much easier to remember that you need to type ` ` for a nonbreaking space or `•` for a bullet than it is to memorize the numerical position of every character in the listings. Figure 6-10 shows how some special characters look on a Web page.

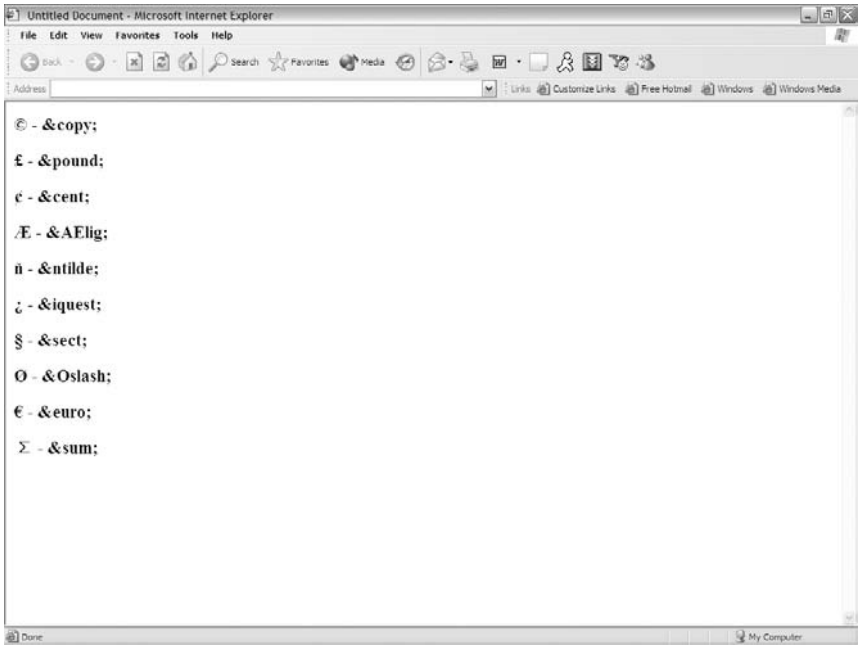


Figure 6-10: This figures shows some special characters and their respective codes.

Tables 6-5 and 6-6 list the character codes and character entities from the HTML standard but are rearranged into alphabetical order instead of numerical order for easier reference.

Table 6-5
Common HTML Character Entities and Codes

Character Entity	*Character Code*	*Character Description and Appearance*
Á	Á	Latin capital letter A with acute (´) accent
á	á	Latin small letter a with acute (´) accent
Â	Â	Latin capital letter A with circumflex (^)
â	â	Latin small letter a with circumflex (^)

www.it-ebooks.info

Character Entity	*Character Code*	*Character Description and Appearance*
´	´	Acute accent (´)
Æ	Æ	Latin capital ligature AE (Æ)
æ	æ	Latin small ligature ae (æ)
À	À	Latin capital letter A with grave (`) accent
à	à	Latin small letter a with grave (`) accent
Å	Å	Latin capital letter A with ring above (Å)
å	å	Latin small letter a with ring above (å)
Ã	Ã	Latin capital letter A with tilde (Ã)
ã	ã	Latin small letter a with tilde (ã)
Ä	Ä	Latin capital letter A with diaeresis (Ä)
ä	ä	Latin small letter a with diaeresis (ä)
¦	¦	Broken vertical bar (¦)
•	•	Bullet (•)
Ç	Ç	Latin capital letter C with cedilla (Ç)
ç	ç	Latin small letter c with cedilla (ç)
¸	¸	Cedilla (¸)
¢	¢	Cent sign (¢)
♣	♣	Black club suit (♣)
©	©	Copyright sign (©)
¤	¤	Currency sign (¤)
°	°	Degree sign (°)
♦	♦	Black diamond suit (♦)
÷	÷	Division sign (÷)
É	É	Latin capital letter E with acute accent (É)
é	é	Latin small letter e with acute accent (é)
Ê	Ê	Latin capital letter E with circumflex (Ê)
ê	ê	Latin small letter e with circumflex (ê)
È	È	Latin capital letter E with accent grave (È)
è	è	Latin small letter e with accent grave (è)
Ð	Ð	Latin capital letter ETH
ð	ð	Latin small letter eth

Continued

www.it-ebooks.info

Table 6-5 *(continued)*

Character Entity	*Character Code*	*Character Description and Appearance*
Ë	Ë	Latin capital letter E with diaeresis (Ë)
ë	ë	Latin small letter e with diaeresis (ë)
½	½	Fraction one-half ($\frac{1}{2}$)
¼	¼	Fraction one-quarter ($\frac{1}{4}$)
¾	¾	Fraction three-quarters ($\frac{3}{4}$)
⁄	⁄	Fraction slash (/)
♥	♥	Black heart suit (♥)
Í	Í	Latin capital letter I with acute (Í) accent
í	í	Latin small letter i with acute (i) accent
Î	Î	Latin capital letter I with circumflex (Î)
î	î	Latin small letter i with circumflex (î)
¡	¡	Inverted exclamation mark (i)
Ì	Ì	Latin capital letter I with grave (Ì) accent
ì	ì	Latin small letter i with grave (i) accent
¿	¿	Inverted question mark (¿)
Ï	Ï	Latin capital letter I with diaeresis (Ï)
Ñ	Ñ	Latin capital letter N with tilde (Ñ)
ñ	ñ	Latin small letter n with tilde (ñ)
Ó	Ó	Latin capital letter O with acute (Ó) accent
ó	ó	Latin small letter o with acute (ó) accent
Ô	Ô	Latin capital letter O with circumflex (Ô)
ô	ô	Latin small letter o with circumflex (ô)
Ò	Ò	Latin capital letter O with grave (Ò) accent
ò	ò	Latin small letter o with grave (ò) accent
Ø	Ø	Latin capital letter O with stroke (Ø)
ø	ø	Latin small letter o with stroke (ø)
Õ	Õ	Latin capital letter O with tilde (Õ)
õ	õ	Latin small letter o with tilde (õ)
Ö	Ö	Latin capital letter O with diaeresis (Ö)
ö	ö	Latin small letter o with diaeresis (ö)

www.it-ebooks.info

Character Entity	*Character Code*	*Character Description and Appearance*
¶	¶	Pilcrow sign = paragraph sign (¶)
£	£	Pound sign (#)
®	®	Registered trademark sign (®)
§	§	Section sign (§)
&spades:	♠	Black spade suit (♠)
¹	¹	Superscript digit one (¹)
²	²	Superscript digit two (²)
³	³	Superscript digit three (³)
™	™	Trademark sign (™)
Ú	Ú	Latin capital letter U with acute (Ú) accent
ú	ú	Latin small letter u with acute (ú) accent
Û	Û	Latin capital letter U with circumflex (Û)
û	û	Latin small letter u with circumflex (û)
Ù	Ù	Latin capital letter U with grave (Ù) accent
ù	ù	Latin small letter u with grave (ù) accent
¨	¨	Diaeresis (¨)
Ü	Ü	Latin capital letter U with diaeresis (Ü)
ü	ü	Latin small letter u with diaeresis (ü)
Ý	Ý	Latin capital letter Y with acute (Ý) accent
ý	ý	Latin small letter y with acute (ý) accent
ÿ	ÿ	Latin small letter y with diaeresis (ÿ)

Table 6-6
HTML Technical and Math Character Entities and Codes

Character Entity	*Character Code*	*Character Description and Appearance*
ℵ	ℵ	Alef symbol
Α	Α	Greek capital letter alpha
α	α	Greek small letter alpha
∧	∧	Logical and

Continued

www.it-ebooks.info

Table 6-6 *(continued)*

Character Entity	*Character Code*	*Character Description and Appearance*
∠	∠	Angle
≈	≈	Asymptotic to
Β	Β	Greek capital letter beta
β	β	Greek small letter beta
∩	∩	Intersection
Χ	Χ	Greek capital letter chi
χ	χ	Greek small letter chi
≅	≅	Approximately equal to
↵	↵	Downwards arrow with corner leftward
∪	∪	Union
↓	↓	Downward arrow
⇓	⇓	Downward double arrow
Δ	Δ	Greek capital letter delta
δ	δ	Greek small letter delta
∅	∅	Empty set
Ε	Ε	Greek capital letter epsilon
ε	ε	Greek small letter epsilon
≡	≡	Identical to
Η	Η	Greek capital letter eta
η	η	Greek small letter eta
∃	∃	There exists
∀	∀	For all
Γ	Γ	Greek capital letter gamma
γ	γ	Greek small letter gamma
≥	≥	Greater than or equal to
↔	↔	Left right arrow
⇔	⇔	Left right double arrow
ℑ	ℑ	Imaginary part
∞	∞	Infinity
∫	∫	Integral

www.it-ebooks.info

Character Entity	*Character Code*	*Character Description and Appearance*
Ι	Ι	Greek capital letter iota
ι	ι	Greek small letter iota
∈	∈	Element of
Κ	Κ	Greek capital letter kappa
κ	κ	Greek small letter kappa
Λ	Λ	Greek capital letter lambda
λ	λ	Greek small letter lambda
⟨	〈	Left-pointing angle bracket
←	←	Leftward arrow
⇐	⇐	Leftward double arrow
⌈	⌈	Left ceiling
≤	≤	Less than or equal to
⌊	⌊	Left floor
∗	∗	Asterisk operator
−	−	Minus sign
Μ	Μ	Greek capital letter mu
μ	μ	Greek small letter mu
∇	∇	Nabla
≠	≠	Not equal to
∋	∋	Contains as member
∉	∉	Not an element of
⊄	⊄	Not a subset of
Ν	Ν	Greek capital letter nu
ν	ν	Greek small letter nu
Ω	Ω	Greek capital letter omega
ω	ω	Greek small letter omega
Ο	Ο	Greek capital letter omicron
ο	ο	Greek small letter omicron
⊕	⊕	Circled plus
∨	∨	Logical or

Continued

www.it-ebooks.info

Table 6-6 *(continued)*

Character Entity	*Character Code*	*Character Description and Appearance*
⊗	⊗	Circled times
∂	∂	Partial differential
⊥	⊥	Perpendicular
Φ	Φ	Greek capital letter phi
φ	φ	Greek small letter phi
Π	Π	Greek capital letter pi
π	π	Greek small letter pi
ϖ	ϖ	Greek pi symbol
∏	∏	N-ary product
∝	∝	Proportional to
Ψ	Ψ	Greek capital letter psi
ψ	ψ	Greek small letter psi
√	√	Radical sign
⟩	〉	Right-pointing angle bracket
→	→	Rightward arrow
⇒	⇒	Rightward double arrow
⌉	⌉	Right ceiling
ℜ	ℜ	Real part symbol
⌋	⌋	Right floor
Ρ	Ρ	Greek capital letter rho
ρ	ρ	Greek small letter rho
⋅	⋅	Dot operator
Σ	Σ	Greek capital letter sigma
σ	σ	Greek small letter sigma
ς	ς	Greek small letter final sigma
∼	∼	Similar to
⊂	⊂	Subset of
⊆	⊆	Subset of or equal to
∑	∑	N-ary sumation
⊃	⊃	Superset of

www.it-ebooks.info

Character Entity	*Character Code*	*Character Description and Appearance*
⊇	⊇	Superset of or equal to
Τ	Τ	Greek capital letter tau
τ	τ	Greek small letter tau
∴	∴	Therefore
Θ	Θ	Greek capital letter theta
θ	θ	Greek small letter theta
ϑ	ϑ	Greek small letter theta symbol
↑	↑	Upward arrow
⇑	⇑	Upward double arrow
ϒ	ϒ	Greek upsilon with hook symbol
Υ	Υ	Greek capital letter upsilon
υ	υ	Greek small letter upsilon
℘	℘	Weierstrass p
Ξ	Ξ	Greek capital letter xi
ξ	ξ	Greek small letter xi
Ζ	Ζ	Greek capital letter zeta
ζ	ζ	Greek small letter zeta

So, if you're writing a Web page in English and want to include the Spanish word *mañana*, you're in a jam because the standard English keyboard doesn't include the *ñ* character—an n with a tilde. To solve the problem, simply scan down the list of character entities until you find `ñ`—at that point, you can either enter the character entity itself or the numerical code (`ñ`), whichever you prefer to use. To use the character entity to write the word, use the following code:

```
ma&ntilde;ana
```

Or you can do the same thing with the numerical code:

```
ma&#241;ana
```

In either case, make sure to always include the beginning ampersand (&) and the ending semicolon (;)—if you leave them out, you won't get the special character.



Most special characters in Table 6-5 are designed to work only with the Latin 1 (ISO 8859-1) character set.

www.it-ebooks.info

Summary

The major way for you to give information to your Web site's visitors is with text. There are plenty of things you can do to make your text stand out.

This chapter introduced you to the following concepts and topics:

- ✦ Character styling like bold and italic text is the number one way to emphasize some text over other.
- ✦ Font size, face, and color enable you to differentiate some text from the surrounding text.
- ✦ Three common font faces—Arial, Courier, and Times—are used by all Web browsers on all computer systems.
- ✦ Text can be aligned and indented to set it off from the surrounding text.
- ✦ Special characters can be used to add characters that are not found on your keyboard to your Web page.

✦ ✦ ✦

www.it-ebooks.info

CHAPTER 7

Finding Images



In This Chapter

Exploring graphics file types

Getting free images off the Web

Getting professional help

Avoiding legal problems



When it comes to Web site content, words are perhaps the most important thing. However, no site is visually appealing unless it has some sort of graphical content as well. Unless you're a skilled artist, you need to look elsewhere for help. You need to know at least the basics of the image file types that work best on the Web. And you need to know where to find them and how to stay out of trouble if you use them.

Exploring Graphics File Types

Three graphics file types are in common use on the World Wide Web:

- GIF (Graphics Interchange Format)
- JPEG (Joint Photographic Experts Group)
- PNG (Portable Network Graphics)

Each carries with it its own advantages and disadvantages. With each file type, you must consider, of course, some technical details, but there are also some legal pitfalls to watch out for.

GIF

GIF, the old standard, is a *lossless* file format, meaning that every single pixel in the image is always preserved when it is saved. There are tradeoffs with every method of saving images, and the drawback to GIFs is a fairly large file size. Nonetheless, even though the GIF file format is pretty old, it remains one of the most common ones. Part of the reason for this, other than plain old inertia (it's supported by every graphics program), is that GIFs provide a fast and easy method of adding animation to a Web site.

Back in 1989, the GIF format was beefed up to include the capability to hold more than one image in a single file, and to display those images in sequence. Voilà—instant cartoons. Although each new image in the file added to the size of the file, of course, it was still the smallest kind of animation file going.

www.it-ebooks.info

JPEG

The JPEG format, also known as JPG for short, was an improvement in many ways over the GIF file. In most cases, it allows for much smaller file sizes. The tradeoff is that there is some degree of information loss. However, JPEG images are generally much more than merely satisfactory, and the "loss" in most cases is only apparent in a mathematical formula. To the naked eye, the lossy image is as good as the original in almost every case.

PNG

One of the results of some legal difficulties with GIF licensing was the creation of a competing file format totally free of all legal complications. The PNG format was designed specifically to replace GIF images by a group of people who wanted to free the World Wide Web of the licensing problem.

While PNG files, in general, are competitive with GIFs, they do lack one major attribute—you cannot do animation with PNGs. However, a newer version known as Multiple Network Graphics (MNG) files is in the works and will include a GIF-like capability to include several images in a single file and play them back sequentially.

Getting Free Images off the Web

There are so many different sources of free images on the World Wide Web that you could literally fill several volumes with nothing but listings of their URLs. We suggest that you pay careful attention to legal details (see the section "Avoiding Legal Problems" later in this chapter), but other than that, there's such a cornucopia out there that you couldn't begin to tap it all no matter how hard you tried.

Finding public domain photographs

The major source of public domain photographs—those pictures that are not protected by any copyright—is the United States government. As you're probably aware, there is a bewildering and complex assortment of different government agencies. You may not be aware, however, that any photographs taken by the employees of those agencies in the performance of their duties are public property.



Just because a photograph is in the public domain doesn't necessarily mean you can use it any way you want to. For instance, if there are identifiable individuals in the pictures, you cannot use them for advertising purposes without the permission of the people in question. When in doubt, ask a lawyer.

Several agencies, such as NASA, have gone out of their way to make the vast array of such images easily available to the public (see Figure 7-1). With other agencies, you may have to go to some trouble to get the images, and they may not be available in digital format, which means that you'll have to scan them on your own.

www.it-ebooks.info



Figure 7-1: The NASA image archives are a prime source for free images.



See the section "Using copyrighted material" later in this chapter for more information on copyrights.

Downloading free artwork

The Web is home to some of the most gifted artists the world has to offer. Many of them make some of their artwork available for the pleasant price of zero dollars (see Figure 7-2). In most cases, such art is free only for noncommercial Web sites, but because the vast majority of sites fall into that category, that's not a bad deal. Even if you run a commercial site, you will be pleasantly surprised to find that you can generally purchase high-quality artwork with funds out of the petty cash drawer.

Finding suitable artists

There are so many artists, with so many varying styles, that the best way to find the one able to help your graphical image the most is—sorry to say—to spend a great deal of time browsing and looking at the wares. Before you start, it's best if you have a clear idea of the image you want to project on your site. Without that kind of

www.it-ebooks.info



Figure 7-2: Clipart.com is a typical free art site.

preparation, you're going to waste a lot of time and effort. With it, you can usually tell with merely a glance at the first image whether or not it's worth your while to dig into the rest of the artist's offerings.



Having a high-speed connection to the Internet will spare you from spending a lot of time waiting for images to download.

Using stock photographs

You can lay your hands on millions upon millions of photographs of everything imaginable. Easily. Cheaply. And today, most of these photographs are available in digital format, either on CD-ROM or on the Web for immediate download. Stock photo agencies have long supplied the needs of print media, and they have now turned their attention to the World Wide Web, which has provided them with a bonanza like they've never known before.

Figure 7-3, for example, shows Hemera Photo-Objects, which is a collection of thousands and thousands of high-quality images with versions for both Mac and PC. The collection comes along with a program for setting the size, file type, background color and transparency you prefer for your particular site.

www.it-ebooks.info

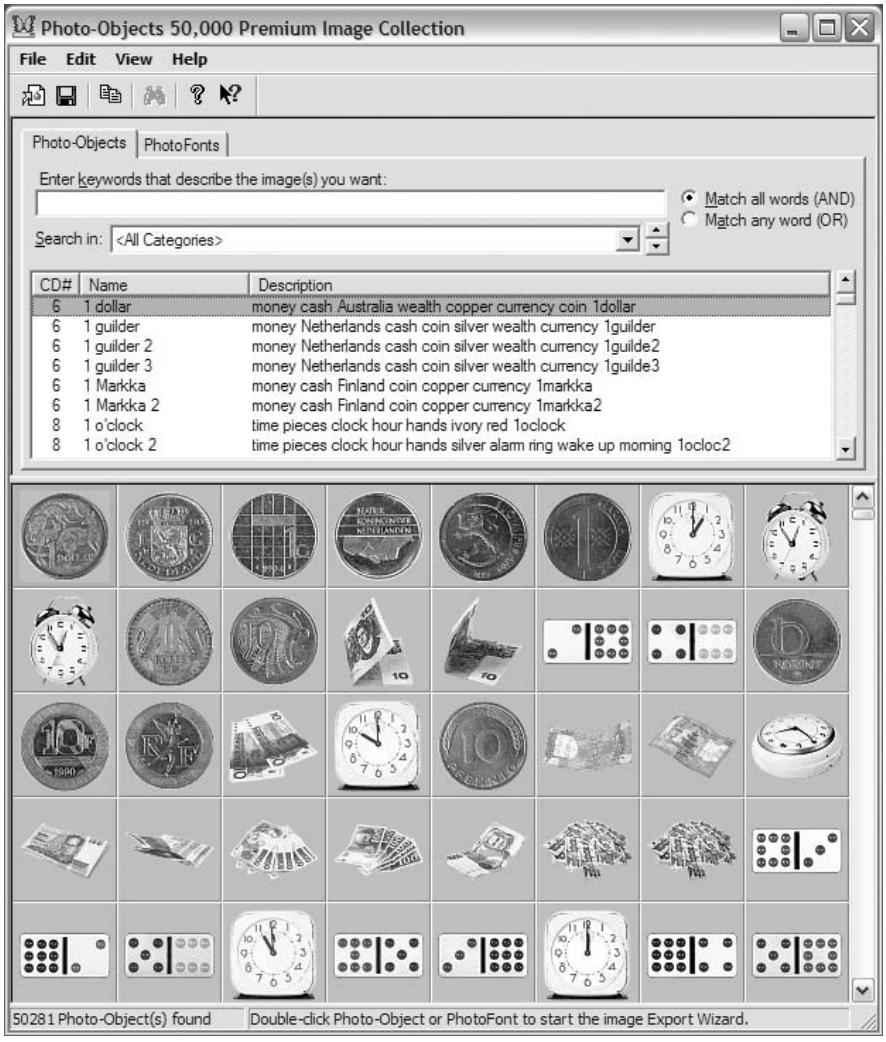


Figure 7-3: Hemera Photo-Objects is a collection of images.



You can download a trial version of Hemera Photo-Objects at `www.hemera.com/image/free_downloads.jsp`.

Avoiding Legal Problems

Nothing can be easier than getting hold of or creating digital images. The Web is full of sites offering thousands upon thousands of image files. Practically every store that sells anything related to computers has CD-ROMs by the truckload that are filled with images, too.

www.it-ebooks.info

The ability to get your hands on an image file, however, is not the same thing as the right to use it. There are all sorts of variations, ranging from limitations on the type of Web site that may use a particular image to the trickier nuances of licensing fine print.

Using copyrighted material

Copyright—the right to make copies—is usually vested in the individual who created the image. Whether that person is a digital artist or a photographer who later scans a print or transparency into a computer doesn't matter. The act of creation is what gives someone the copyright.

Modifying an existing image is not an act of creation. The resulting modified image is called a *derivative work* and is not considered to be original, no matter how major the modification is. This means that, if you use someone else's copyrighted material, it makes no difference what you do with it—it's still under their copyright.

There are few ways for the creator of an image to lose the copyright. One is for them to create the work under the terms of a contract that specifically states they are waiving their rights. This is called a *work-for-hire,* or *WFH* contract. Another is for the creator to sell the copyright. Copyright is property—intellectual property, but property nonetheless, just as a car or house is—and can be bought or sold.

Often, however, the artist in question will license the rights instead of selling them. The artist retains the copyright while allowing others to use the image. There is, sadly, no such thing as a uniform license. Using existing artwork or photographs on your Web site is a wonderful way to solve the problem of adding good graphics. If you don't take the time to read and understand the fine print in the license, however, you can end up with a hassle on your hands instead.

This is especially true with "free" graphics. Many graphic images that are available at no cost cannot be used on a commercial Web site. The artists involved do not mind if someone uses the images on a private home page, or if a nonprofit organization uses them, but most artists take umbrage if someone is using their images in a money-making situation and not paying them for their work.

Sometimes, the licensing terms can leave you wondering exactly what you're supposed to do or not do. In more than one case I've seen, a license specifies that you're allowed to use the images on your Web site but can't make the image files available for download. Maybe you can figure out some way to do both of those things, but I confess that it's beyond me.

I assume that the lawyers who wrote the license don't have the slightest grasp of how Web pages work. Or perhaps it's just a question of intent. Maybe these licenses mean that you just can't suggest that people download their own copies of the images. Who knows? Considering that no Web page image can be displayed unless it's downloaded to a Web browser, and considering that browsers automatically cache a copy of the image on your visitors' hard drives, it hardly matters what you suggest. There are two really good ways to handle a license like that. The first is to

www.it-ebooks.info

ask for clarification. If you get it, make a hard copy of the response and keep it where you can retrieve it if needed. If you don't—or maybe even if you do, depending on the answer—then exercise the second good option. Don't use those images. It's just not worth it, even if they're the greatest pictures you've ever seen.

Respecting trademarks

Whether an image is an original you drew yourself or whether it comes from a collection, the subject matter of the image may be something to be concerned about. Let's say you're a talented artist and grew up in 20th century America. You no doubt learned—and deliberately practiced—a good part of your artistic technique by carefully copying the work of established artists while you felt your way to your own approach. That means, simply, that you're probably pretty good at whipping out a perfectly recognizable Snoopy or Garfield. Even if you're no good at all at drawing, and do your artwork with 3D programs that require no physical artistic talent, I guarantee that, at some point, you have made some version of the Enterprise or something out of Babylon 5.

These are normal stages in the growth of any artist—imitation is not only flattery, it is education. However, there's a big difference between studying an artistic technique and using someone else's trademarks on your Web site.



See Chapter 26 for more information on trademarks.

Summary

Images are the key to visually appealing Web pages. While it's possible to get by without them, no page looks as good without images as it does with them. This chapter introduced you to the following concepts and topics:

- ✦ The three most common file formats on the Web are GIF, JPEG, and PNG.
- ✦ Public domain photographs and artwork do not have any copyright protection.
- ✦ Many images are available for free, even if they are copyrighted, as long as the copyright owner gives permission.
- ✦ Trademarked images cannot be used without permission even if you draw them yourself.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 8

Blending Images into Your Pages



In This Chapter

Adding images

Covering all the bases with placeholders

Aligning text and images

Using images to link files

Solving image problems

Adding background images



Images are critically important on the World Wide Web. Any Web page that offers nothing but text isn't in the running for winning any awards, that's for sure. The old saying that a picture is worth a thousand words is nowhere more appropriate than on the Web.

In this chapter, we cover the basics of adding images to your pages, but we also go on to explain why you need to use them in certain ways and how to choose the images that best suit your purposes.

Adding Images

Images are inserted into Web pages by the `IMG` element. The name and location of the file are specified in the `src` attribute. Here's the basic code:

```
<IMG src="filename">
```

If the image file is in the same folder as the Web page it's displayed on, that's all you need. However, many Web designers keep their image files in a separate folder from their HTML files. In that case, you need to specify the path to the file as well as its name. If, for example, you keep your image files in a subfolder called images, and you want to add a file called `toaster.jpg` to your Web page, you need to specify its location as follows:

```
<IMG src="images/toaster.jpg">
```

If the image file is on another Web server entirely, you need to use the full URL of the image file, like this:

```
<IMG src="http://www.wheretheimageis.com/toaster.jpg">
```

www.it-ebooks.info



See Chapter 4 for a discussion of URLs and file addresses.

By the way, if you do link to an image located on someone else's server, make sure that you have their permission first. While it's as easy to do as typing the code in the preceding example, there are two problems with doing it. The first is that you inconvenience the other Webmaster by increasing the load on their server—every time someone looks at your Web page, the other Webmaster's Web server has to serve up the linked image to your visitor's Web browser. The second is that you have absolutely no control over the contents of someone else's Web server. If they decide to rearrange files, remove the image file you're linked to, or if their server goes down or is out of business, your Web page won't work right any more. If you want to ensure that your site is always working, it's best to link only to images on your own system.

Using the height and width attributes

When a browser loads a Web page, the text on the Web page appears right away, but the image file, being separate from the Web page itself, takes slightly longer to show up (the length of time depends on the size of the image file). If the Web browser doesn't know the width and height of the image, it can't tell how much space to set aside for it until the image file has been downloaded. What happens in that case is that the browser has to rearrange the text after the image file (or files) have come in, which causes an automatic redraw of the screen, making all the existing elements jump around as the image comes in and room is made for it. This unnecessary irritation can be avoided by specifying the width and height in the HTML code as in the following example:

```
<IMG src="toaster.jpg" width="200" height="234">
```



If you use a Web authoring program such as Dreamweaver or HomeSite+, the program automatically adds the width and height information when the image is inserted.

The `width` and `height` attributes can be used to change how large the image is when viewed in a Web browser. It's simply a matter of changing the values. For instance, if you want to make the toaster image display at half its full size, alter the values like in this example:

```
<IMG src="toaster.jpg" width="100" height="117">
```

To make it show up at twice its normal size, double the values:

```
<IMG src="toaster.jpg" width="400" height="468">
```

Figure 8-1 shows the original toaster image, the half-size image, and the doubled version.

www.it-ebooks.info

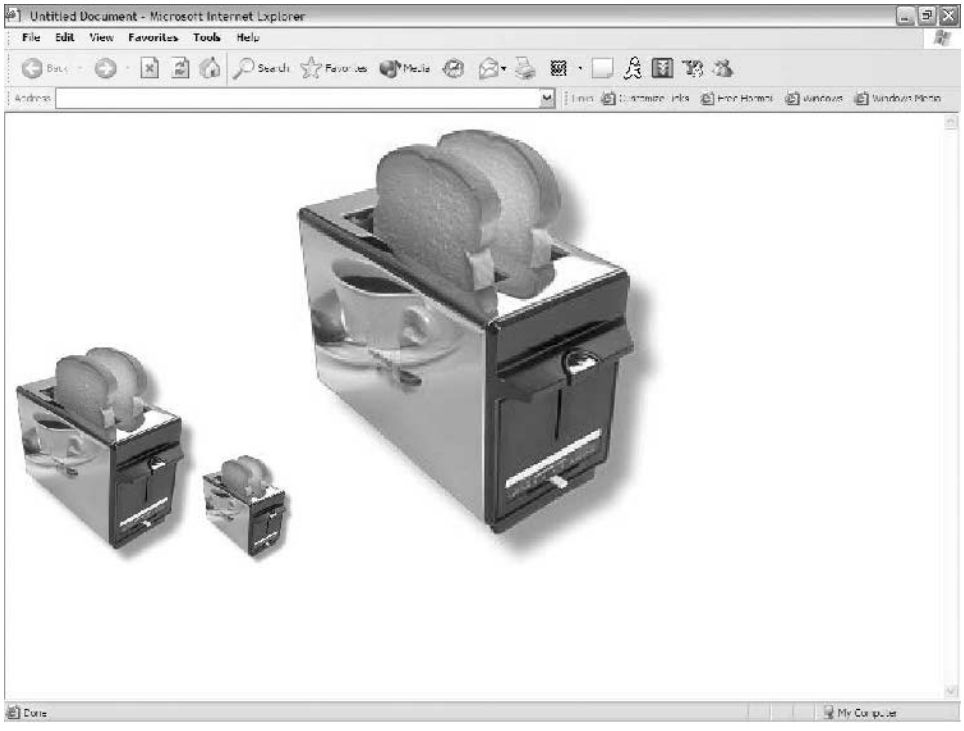


Figure 8-1: Image sizes can be manipulated with the width and height attributes.

The doubled image is noticeably blocky. This is an unavoidable side effect of enlarging images with this method. It's important to realize that the image file itself is never affected by changing the width and height settings in the HTML code. The only change is in the way the image is displayed by a Web browser, and browsers aren't geared to fine-tune image variations. If you want to enlarge an image with less distortion, you have to do it in a professional graphics program such as Fireworks or Photoshop.



Although we carefully made sure that the proportions of the image were unchanged when we altered the width and height values, you can deliberately create some weird effects by using disproportionate values.

Setting horizontal and vertical spacing

You can create a bit of breathing space—a barrier that other Web page elements cannot penetrate—around an image. The amount of horizontal space is set by the `hspace` attribute and the amount of vertical space is set by the `vspace` attribute. The following HTML code example shows both dimensions being set to a size of 20 pixels:

```
<IMG src="tractor_with_roller_2.jpg" width="329" height="285" vspace="20"
hspace="20">
```

www.it-ebooks.info

Figure 8-2 shows the image without any spacing added. Figure 8-3 shows how it is moved away from its surroundings by adding the spacing.

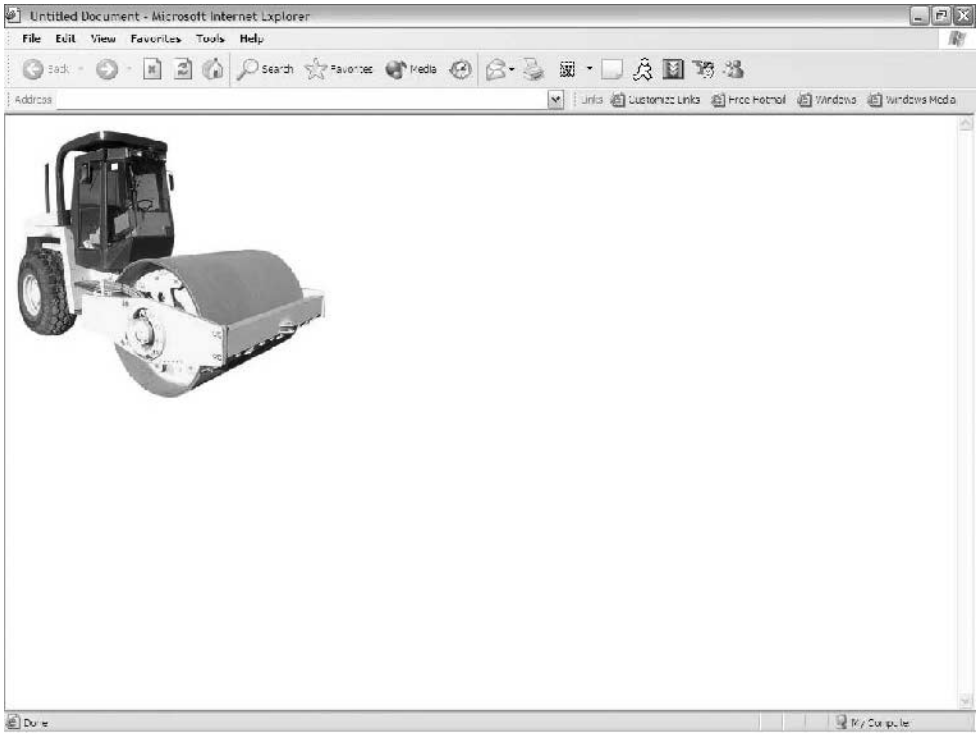


Figure 8-2: This figure shows an image without spacing.

The `hspace` attribute sets both right and left spacing simultaneously, and the `vspace` attribute does the same for both top and bottom spacing. You cannot independently set the left margin to a different value than the right margin or the top margin to a different value than the bottom one.

Cross-Reference See Chapter 13, however, for how to achieve independent margin spacing with CSS.

Setting border size

Images on Web pages can have a border—a frame for the picture, so to speak. But unlike physical artwork such as cross-stitch or a painting, there is no canvas that requires a wood or metal frame. Digital images are similar to photographs—you may or may not want to frame one.

Some images look better with a border, while others don't. It's one of those purely personal decisions that Web designers have to make, and you can be sure you will never please everybody, but there are a couple of rules you may want to keep in mind.

www.it-ebooks.info

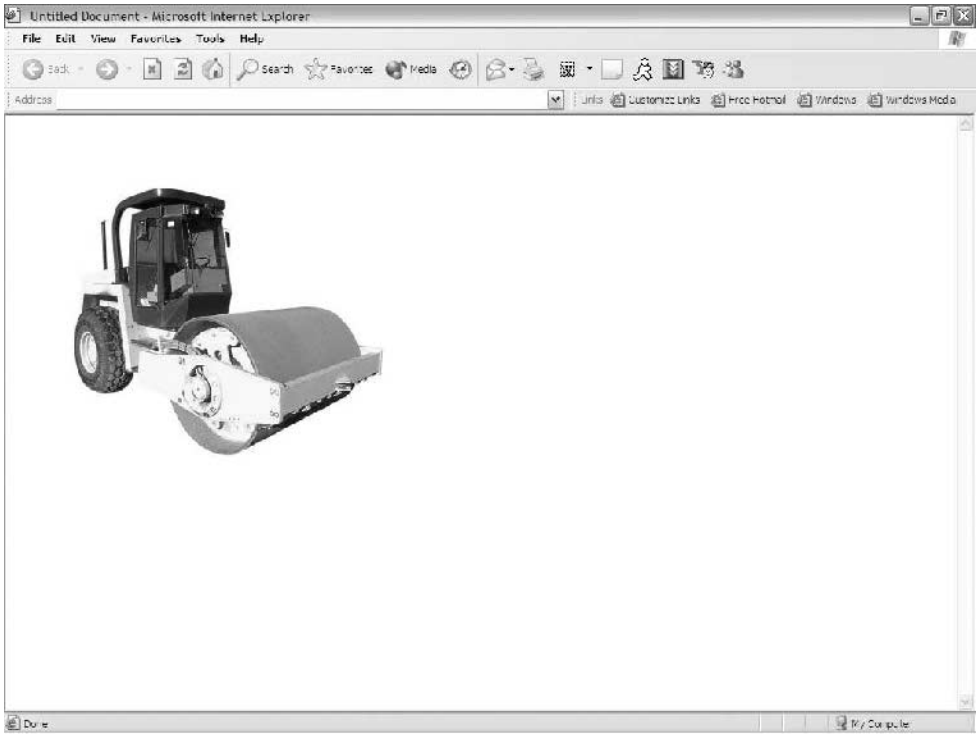


Figure 8-3: This figures shows the same image with spacing added.

The three images in Figure 8-4 illustrate the most common situations you have to deal with when deciding on borders. The first is a rectangular image where the material at the edges simply fades out into the void. The second is a rectangular image where the edges are already clearly defined within the artwork itself. The third is a freeform image with a transparent background.



See Chapter 9 for information on creating transparent areas in images.

The first image, in our opinion, fairly cries out to have a border that clearly separates its edges from the rest of the Web page. Unless you're looking to create an impression of energy dissipating from the edges, it should be reined in. For the second image, a border would be superfluous, because it already carries its own within the artwork. Adding a border around it is overkill. Adding a border to the third image, however, would disrupt its native flow, forcing the graceful curves into a rigid angularity.



All Web images are actually rectangular, but those that use transparency create an illusion that they are shaped differently. You still have to bear in mind that the transparent part takes up space on the page, however. The only exception to this is when using CSS for absolute positioning. In that case, other Web page elements can be placed behind the image and they will show through the transparent areas (see Chapter 13).

www.it-ebooks.info

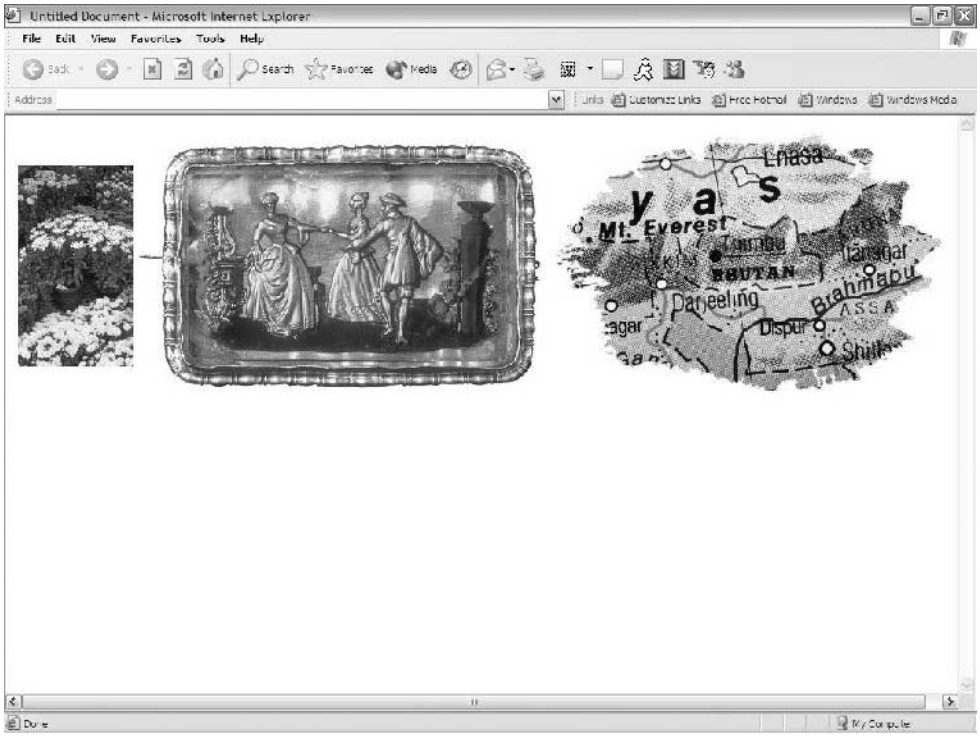


Figure 8-4: This figure shows the three most common types of images.

The size of the border is a matter of artistic choice—whether, in your opinion, the image would be best shown off with a thin, delicate border or a thick, heavy one. When you specify the existence of a border, you also specify its size. The following example of code sets a border width of 4 pixels:

```
<IMG src="assorted_vegetables.jpg" width="200" height="244" border="4">
```

Figure 8-5 shows the same image with border sizes of 1, 4, and 8 pixels.

Covering All the Bases with Placeholders

There are times when an image won't show up, for whatever reason—perhaps the file has been moved or deleted and the HTML code was never updated—but you want something more than a broken link to appear. At other times, either due to an extremely large image file or a very low speed Internet connection, an image may take a noticeable amount of time to show up. Placeholders can help to fill in the gaps.

Adding alternative text for accessibility

The `alt` attribute enables you to have alternative text show up in place of a missing image, but there's another aspect to alt text that's even more important. For people who are surfing the Web without using visual browsers, alt text is the only way they

www.it-ebooks.info

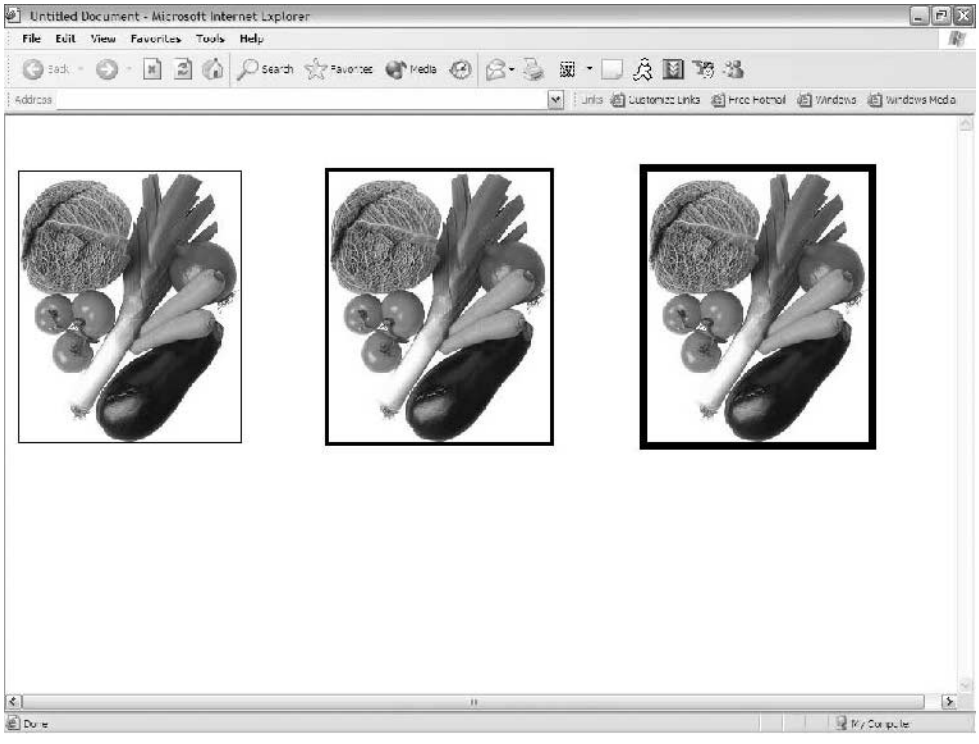


Figure 8-5: Different border sizes give various impressions.

can tell what information is included on the images on your pages. A blind person—or just someone using a text-based browser—needs alt text to fully appreciate what your page is about. For example, you might add the following description:

```
<IMG src="assorted_vegetables.jpg" width="200" height="244"
alt="picture of vegetables">
```



Some Internet search engines take alt text into account when determining your page's ranking. If you don't have any, you can't end up on top.

One drawback to using alt text is a theoretically helpful feature of Web browsers called *tool tips* or, sometimes, *help balloons*. A tool tip is a small rectangle that pops up when your mouse pointer is over an image that contains alt text. The tool tip's content is the alt text itself. Since alt text is supposed to replace the function of an image, this seems a bit silly.

Tool tips present no real problem with large images, but they can be particularly annoying with small images such as the ones that are often used to create navigation bars. Because site visitors use navigation bars by moving their mouse pointers over them before clicking on them, and because the popup tool tip can obscure the text on a small button-type image, it is tempting not to provide alt text for navigation bar images. However, alt text tooltips aren't the only things to obscure images (see Figure 8-6).

www.it-ebooks.info

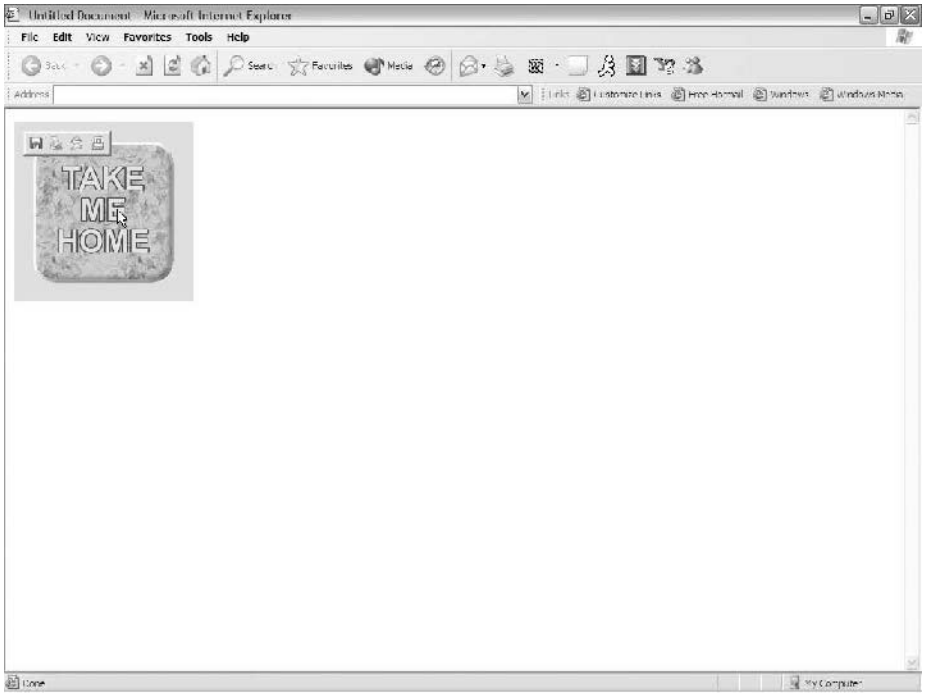
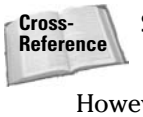


Figure 8-6: Tool tips can briefly obscure small images.



See Chapter 17 for more information on navigation bars.

However, that would mean you'd have several important links that were unexplained to users of nonvisual Web browsers. This is not a desirable situation. One possible solution is to provide a second set of links—plain text links—at the bottom of the Web page, but that's a stopgap measure at best.

The World Wide Web Consortium (or the major browser makers, acting independently) could solve this problem by adding a new attribute to the `IMG` element—something such as `notips`, which would signal visual user agents to suppress the display of tool tips for any image containing that attribute. User agents that are geared toward those with sight difficulties, such as speech synthesizers, would ignore the `notips` attribute, because the presence or absence of a visual aid is of no value to those agents.

Using low-resolution images to speed page download time

If you're using a very large image that you know is going to take forever to download, you may want to consider adding a low-resolution image to the mix as well. Low-res images are poor quality copies of an image. They may, for instance, be black and white or grayscale versions of a full-color image. The point of low-res images is that

www.it-ebooks.info

they have a smaller file size than the full version does. This means that they don't take anywhere near as much time to download.

In HTML, you can specify that a low-res image be displayed on the Web page while the full version of the image is downloading, giving your visitors some idea of what they're waiting for. The low-res image is specified via the `lowsrc` attribute:

```
<IMG src="huge_file.gif" width="1200" height="800" lowsrc="small_version.gif">
```



The low-res version must have the same physical dimensions as the full version. If it doesn't take up the same space on the Web page, the display of the full version will be messed up.

Aligning Text and Images

Although many Web designers use images as if they were block level elements, they are actually inline elements just like the ones used to make text bold or italic (the `B` and `I` elements). Thus, you can put an image right smack in the middle of a line of text and it sits there just as if it's another word in the sentence.



See Chapter 3 for more information on block and inline elements.

Of course this creates a practical difficulty because the vast majority of images are considerably larger than most text. Figure 8-7 shows how a large image can disrupt the natural flow of a paragraph.

Small images called *icons*, however, are often used on the same line as text. You often see bullets of various kinds marking the start of lines of text. And some people even use tiny, invisible images to increase spacing between sentences.

The `align` attribute for images doesn't actually align the image, but the surrounding text. For instance, the following code makes the text move to the top of the image as illustrated in Figure 8-8.

```
<p>Try our irresistible cake! <IMG src="cake.jpg" width="304" height="285"
align="top"> </p>
```

There are several possible options for the value of the `align` attribute, some of which duplicate one another's function. Table 8-1 gives you the low down.

All of the values except for left and right simply change where the text falls in relation to the image. Those two, however, mean you have to handle an entirely different situation.

Floating images out of the flow

With most alignments, an image remains right where you originally placed it. There are two, however—left and right—which cause the image to become *floating.* This

www.it-ebooks.info

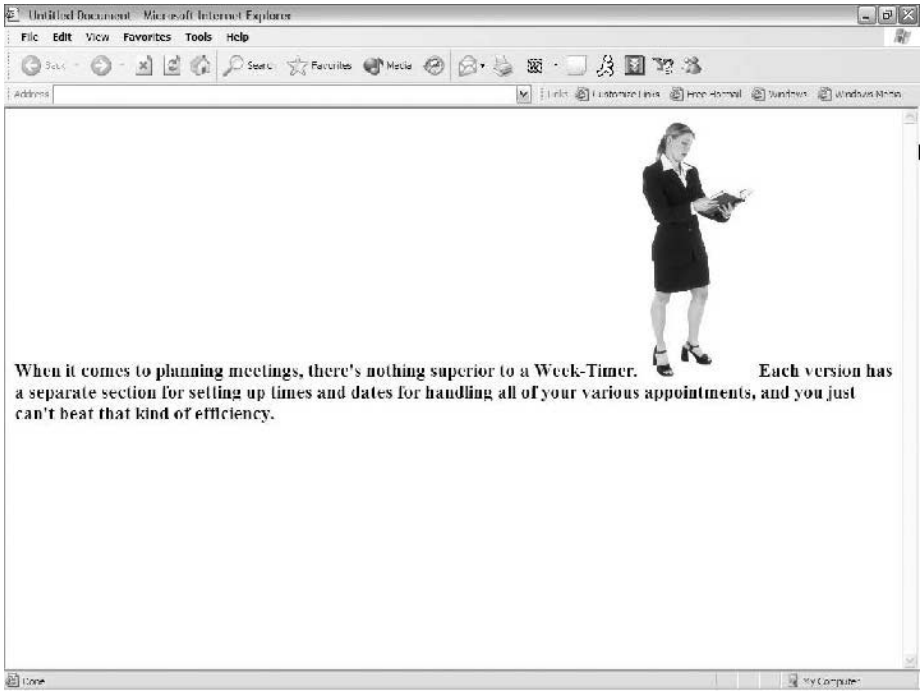


Figure 8-7: Large embedded images disrupt the flow of a paragraph.

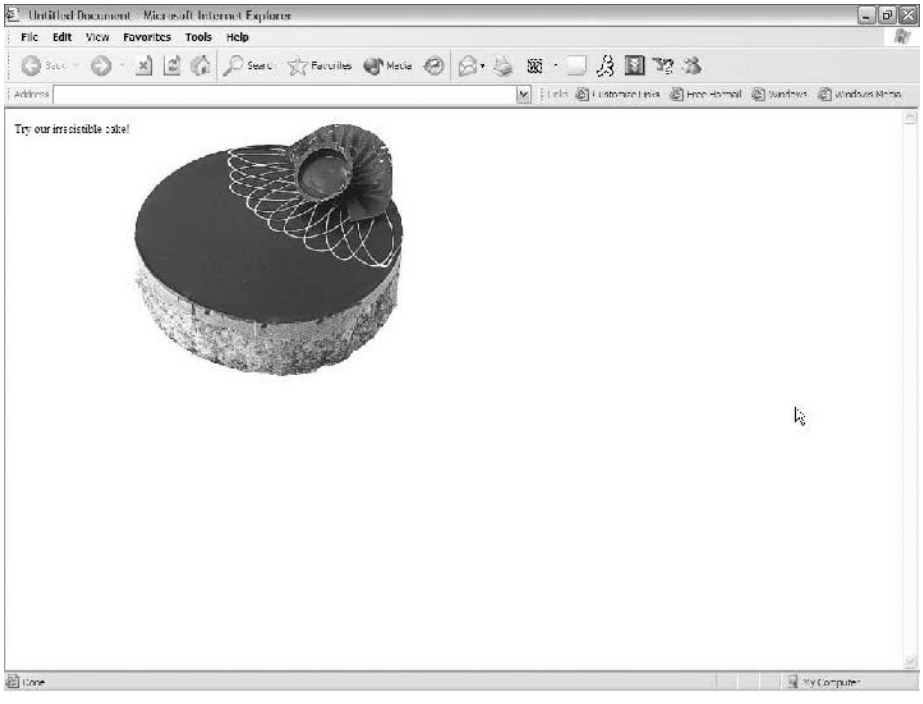


Figure 8-8: Image alignment affects the surrounding text.

www.it-ebooks.info

Table 8-1
Image Alignment Settings

Align Value	*Meaning*
absbottom	Absolute Bottom: lines up the bottom of text descenders with the bottom of the image.
absmiddle	Absolute Middle: lines up center of text with the center of the image.
baseline	Puts the baseline of the text on the same level as the bottom of the image.
bottom	Same as baseline.
left	Drops the image out of its original location and moves it to the left margin.
middle	Lines up the baseline of the text with the center of the image.
right	Drops the image out of its original location and moves it to the right margin.
texttop	Puts the top of the image on the same level as the top of the text.
top	Same as texttop.

means that the image leaves its place in the text and drops down to the next available unoccupied space on the Web page. With left alignment, the image drops down and moves to the left margin, and with right alignment it drops down to the right margin.

Figure 8-9 shows what happens if the image in the preceding section is set to left alignment.

When there is only a small amount of text, such as a caption, associated with an image, things are very simple. But when an image is involved with a large amount of text, floating images have a radically different effect on word wrap than the other alignments do.

Demystifying word wrap around images

Word wrap—the way in which the text surrounding an image reacts to the presence of the image—is also controlled by setting the image's alignment. With all alignment values except for left and right, any text that won't fit on the same line as the image wraps around to the line below the image. As shown in Figure 8-10, this can cause a disjointed flow.

When an image floats out of its surrounding text with the left or right alignments, the real reason for the difference in these values becomes apparent. The word wrap in these cases is much more natural, much more appealing, and much more conducive to good page design. Figure 8-11 shows a right-aligned image and how it affects the word wrap of a paragraph.

www.it-ebooks.info

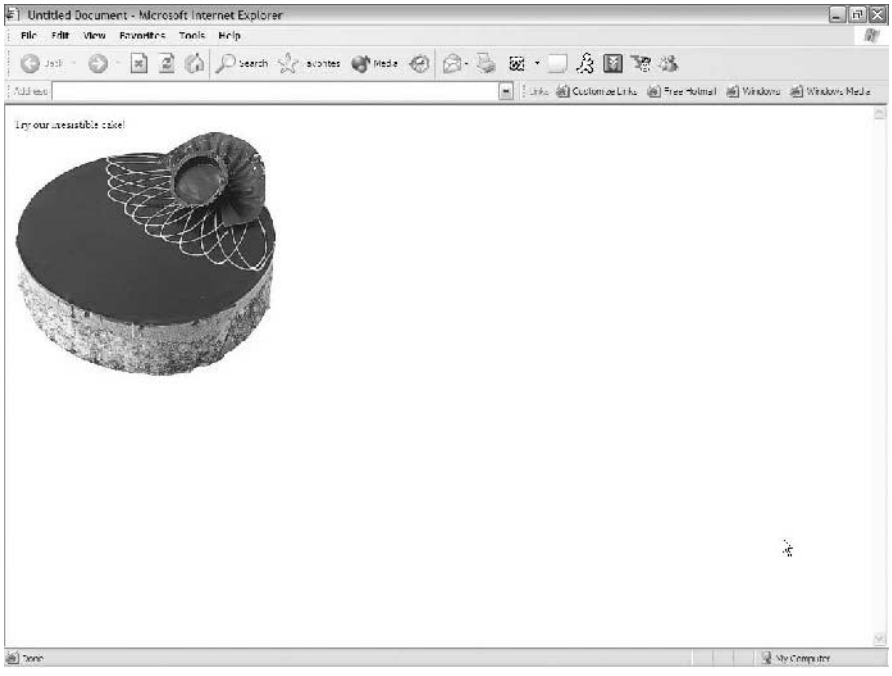


Figure 8-9: Left alignment drops the image to the next available space on the left margin.

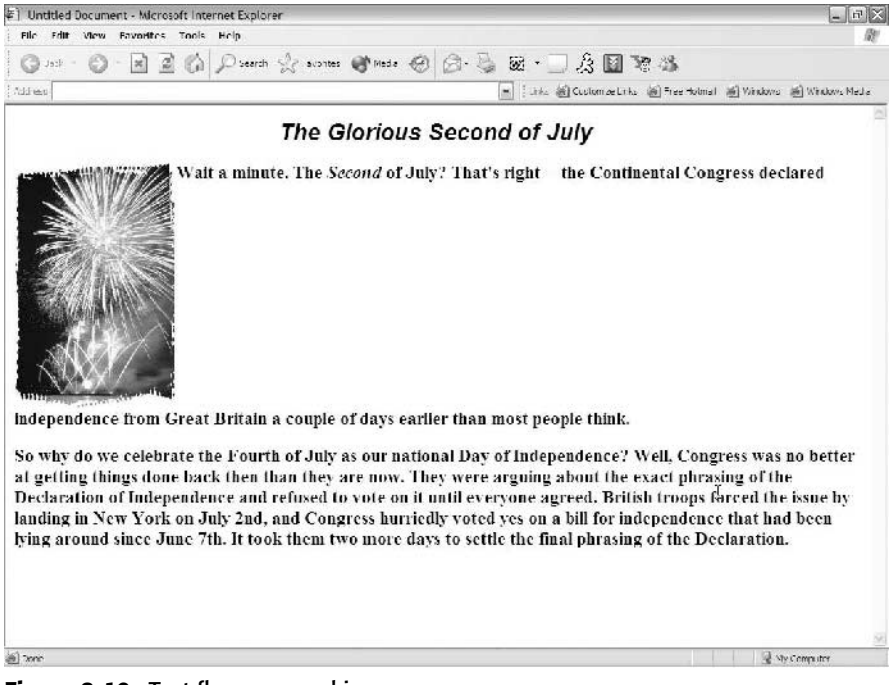


Figure 8-10: Text flows around images.

www.it-ebooks.info

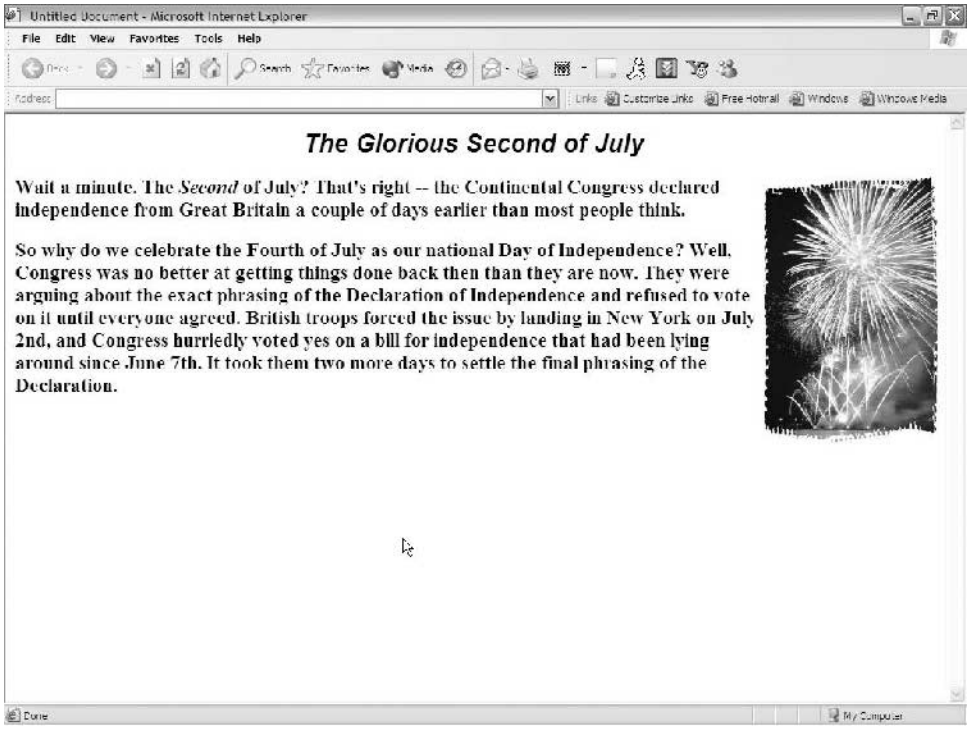


Figure 8-11: A right-aligned image flows text smoothly.

A left-aligned image would have the same effect, except that it would, of course, be on the other side of the page.

Aligning multiple images

When there is more than one image in a single line, the alignment becomes increasingly complex. All of the images `align` values affect both the text preceding and following them. The following code sets up a situation where the first image is set to top alignment and the second is set to bottom alignment.

```
<p>Try our veggies <IMG src="assorted_vegetables.jpg" width="200" height="244"
align="TextTop">and then our irresistible cake! <IMG src="cake.jpg"
width="150" height="142" align="bottom">You can't beat either one.</p>
```

All the text is top aligned with respect to the first image, and all the text is also bottom aligned with respect to the second image. This means that the first image moves down to make room for the text to fit into the specified alignments. Figure 8-12 shows what happens in a case such as this.

Using Images to Link Files

Image links are created in pretty much the same way as plain text links are created. The only difference is that an `IMG` element is used instead of text for the contents of the `A` element:

www.it-ebooks.info

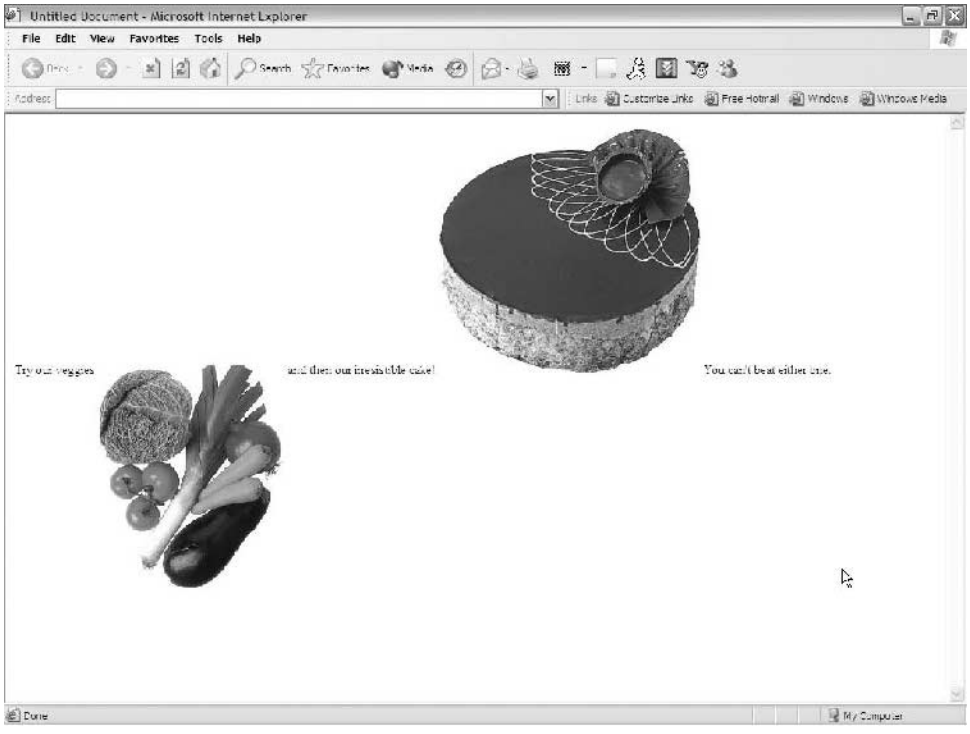


Figure 8-12: Multiple images can be difficult to align.

```
<A href="url"><IMG src="filename"></A>
```



See Chapter 4 for details on creating links.

Using borders

Borders are optional for images (see "Setting Border Size" earlier in this chapter), and many designers prefer to leave most images borderless. When it comes to image links, however, there's a very good reason to make sure that the image has a border.

Figure 8-13 shows two images without borders. One of them is an image link and the other is a plain image. Nothing indicates that there is any difference between them.

Figure 8-14 shows the same two images with borders. The plain one has a black border, but the image link has a blue border. This visual cue is standard on the Web as a clue for your visitors that the image link isn't just a plain old image, but has a special function. Of course, you don't have to put borders around plain images, too—you can just use them on image links and leave your plain images borderless. The important thing is to signify the importance and function of the image links on your page.

www.it-ebooks.info

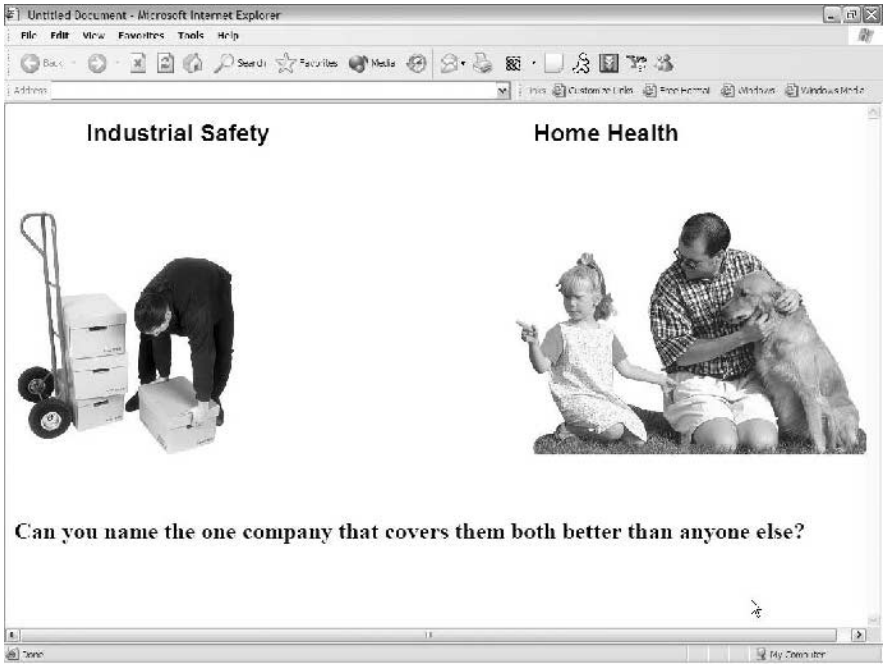


Figure 8-13: Borderless images—which is the image link?

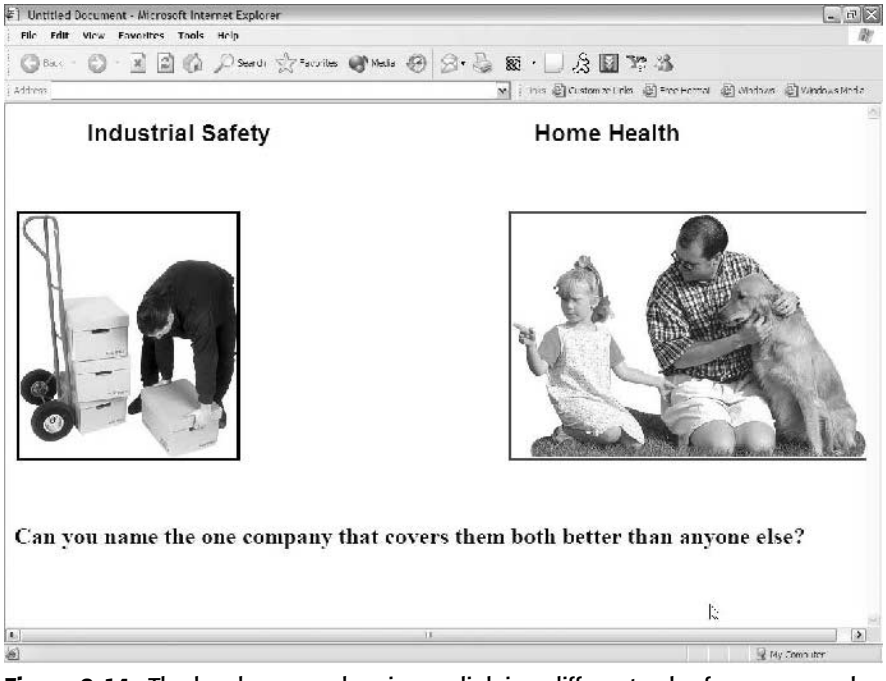


Figure 8-14: The border around an image link is a different color from a normal border.

www.it-ebooks.info



The colors—blue for image links and black for plain image borders—are the same as for the same functions in the normal text colors. Plain text is black and link text is blue (see Chapter 4). Whatever color you set for the default text and link values also becomes the border colors for images. For example, if you set the default text color to red and the default link color to green, plain images will have red borders and image links will have green borders.

Still, some Web designers don't use borders even for image links. There is no right or wrong decision. It's purely aesthetic. If there isn't a border—and, therefore, no special color marks the image as a link—how is anyone to know the link exists? Your site visitors do get two other clues. First, the mouse pointer changes shape if hovered over an image link. Instead of the usual arrow, the pointer becomes a hand. Second, the URL of the image link shows in the status bar of a Web browser when the mouse pointer is over the image.

If your site visitors are seasoned veterans, they don't need borders to find image links—they know to move their mouse pointers over the images to see what happens. However, that means they still have to check out all the images one by one to see what each one is because there's no way to just look at them and tell. If your site's artistic motif can possibly accept the existence of bordered image links, you can make life a lot easier on your visitors by including them.

Making multiple links with image maps

While a regular image link is limited to having one link per image, image maps enable you to add as many links as you want to one image. Of course, there are practical limitations—if you tried to put a hundred links into a small image, it would be pretty useless. However, there is no limit in HTML itself.

An *image map* takes an image and defines certain regions of it as specific shapes—rectangles, circles, or polygons. Each shape is associated with a different URL and, when the region defined by that shape is clicked, the Web page (or other file) at the associated URL is loaded into the Web browser.

Image maps are incredibly difficult—nearly impossible, actually—to create by hand, so it's pretty much a waste of time to even attempt typing in the code for one. However, every major Web page creation program has some sort of image mapping utility built in. Figure 8-15 shows the Dreamweaver image mapping function in action.

The type of image that's suitable for an image map varies depending on the situation. Sometimes, it's an actual geographical map such as the free ones available from `www.graphicmaps.com`. In other cases, an image composed of only text might be appropriate. If you were detailing medical information, an image of the human body would perhaps do the job where site visitors could click the parts they were concerned with to link to pages containing information about maladies. If you were working up an image map for a restaurant, you might want to use an image showing the various dishes that were available on the menu. Use your imagination when you choose, but ensure that the result is one that your site visitors will be able to use intuitively.

www.it-ebooks.info

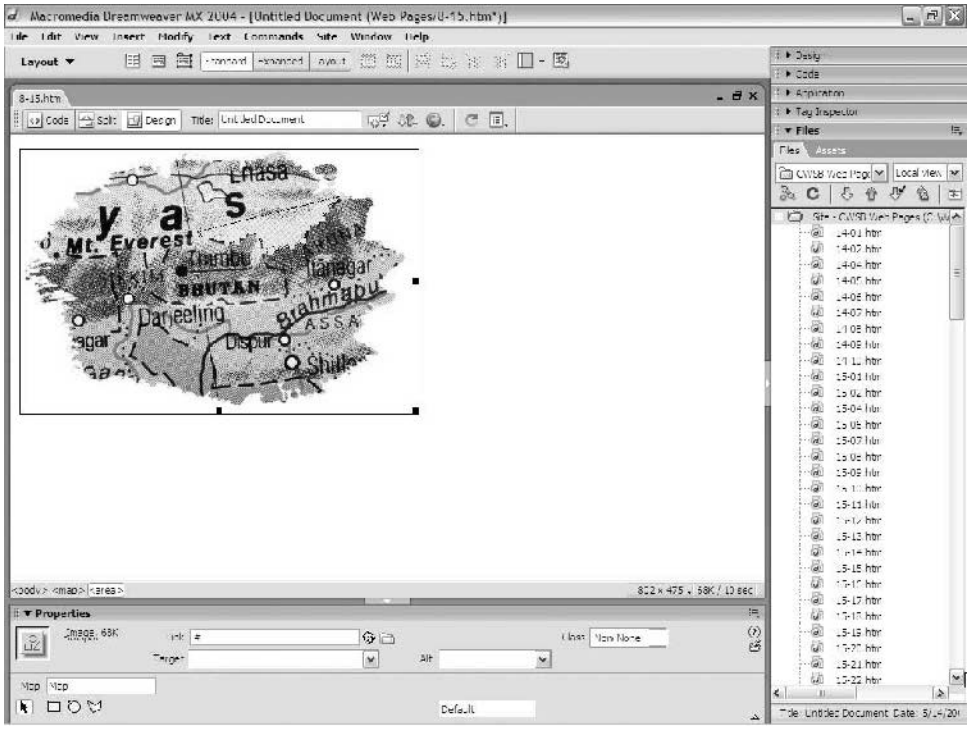


Figure 8-15: Dreamweaver has a built-in image mapping function.

Solving Image Problems

When working with images, there are some times where normal HTML methods just aren't very useful. Two of these are where there is an unavoidable delay caused by several large image files and where there is a need to use fonts that just aren't supported by usual Web page methods.

Using thumbnail images to speed up page downloads

In some cases, a Web page must deal with an awful lot of large image files. How else, for instance, can an art gallery be created online? But asking your site visitors to wait while a dozen—or a hundred—large image files download may be asking too much of their patience.

Fortunately, a simple and elegant solution was developed a long time ago: simply use image links composed of very small versions of the large images. The small image link versions are known in the trade as *thumbnails.* Figure 8-16 shows a Web page that uses thumbnail images.

www.it-ebooks.info

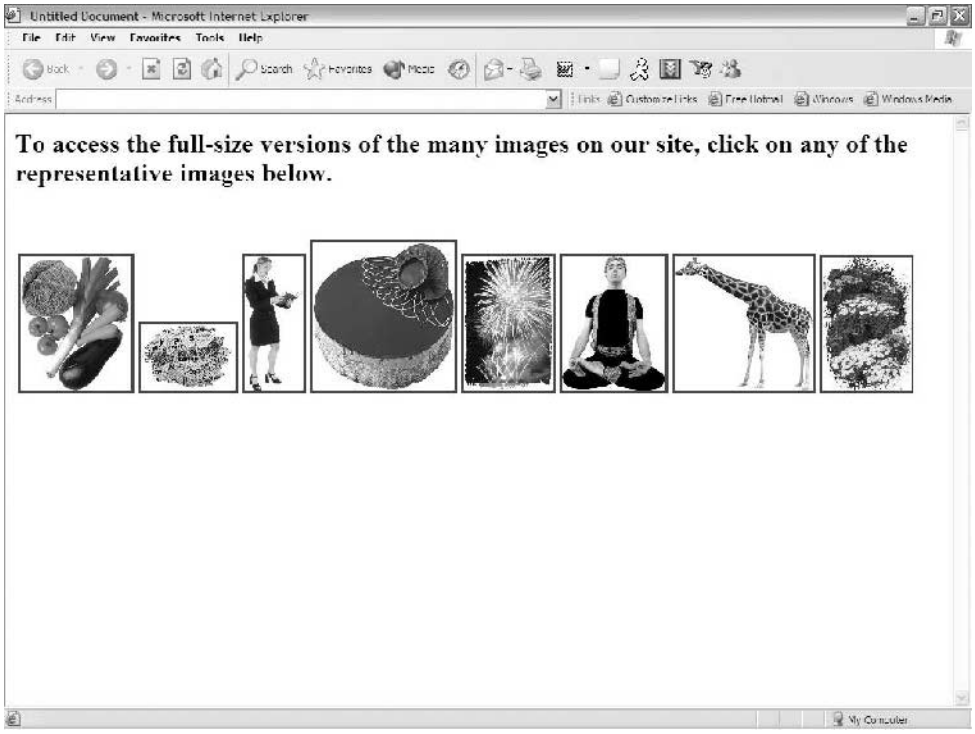


Figure 8-16: Thumbnail images can make pages with large numbers of images work better.

The advantage to using thumbnails is that they provide enough of the full image for your visitors to decide whether or not they're interested in waiting for the full version. If so, they click the link and go make a sandwich. If not, they move down to the next thumbnail.

Embedding special fonts

The three fonts commonly used on the Web (Arial, Times, and Courier) do a pretty good job of handling the majority of text needs, but they don't do much in the way of satisfying artistic desires. Various attempts have been made over the years to come up with a workable method for using other fonts. One approach, for example, was to use any font you wanted, forcing your visitors to download fonts they didn't have. This, as you might imagine, resulted in an incredible lag time before the page could even be displayed. Web surfers hate long page download times more than anything else, and the idea of downloadable fonts is now gathering dust in a computer museum somewhere.



See Chapter 6 for how to set font families.

So what do you do if you want to use a beautiful font that you have? You can embed it in an image in a graphics program and add that image to your Web page like the example shown in Figure 8-17.

www.it-ebooks.info

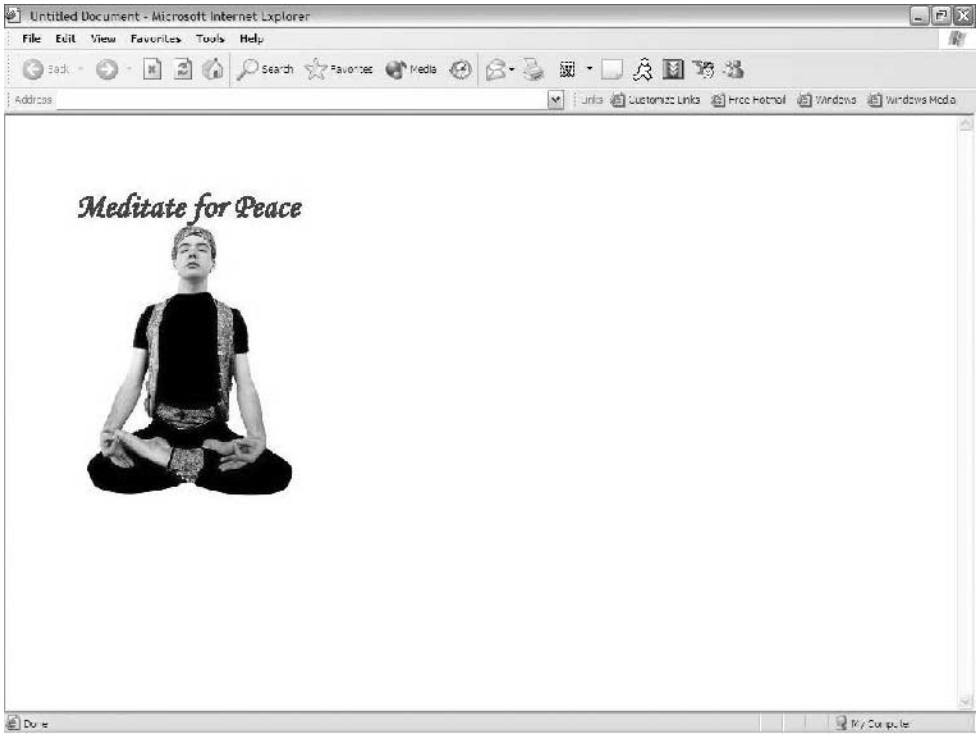


Figure 8-17: Text embedded in images can use any font.

A similar approach, but without the potential handicap of having to tie the text in with a particular image, is to create the text as an image in its own right. Every good graphics program has some method for typing onto a blank canvas or an existing image, and different ones offer various methods for setting the font color and size. Even better is the ability to apply textures and special effects to the lettering.



See Chapter 9 for details on creating graphics.

This approach is sometimes used to create *drop caps*, which are fancy capital letters used to begin paragraphs, a technique that is often seen in printed books. Figure 8-18 shows a drop cap in use on a Web page.

Adding Background Images

Background images, as with background color (see Chapter 10), can add immeasurably to the aesthetic appeal of a Web page—or they can destroy its usefulness and effectiveness. Properly chosen, a background image supplements the meaning and value of a Web page. Used improperly, it cripples it.

www.it-ebooks.info

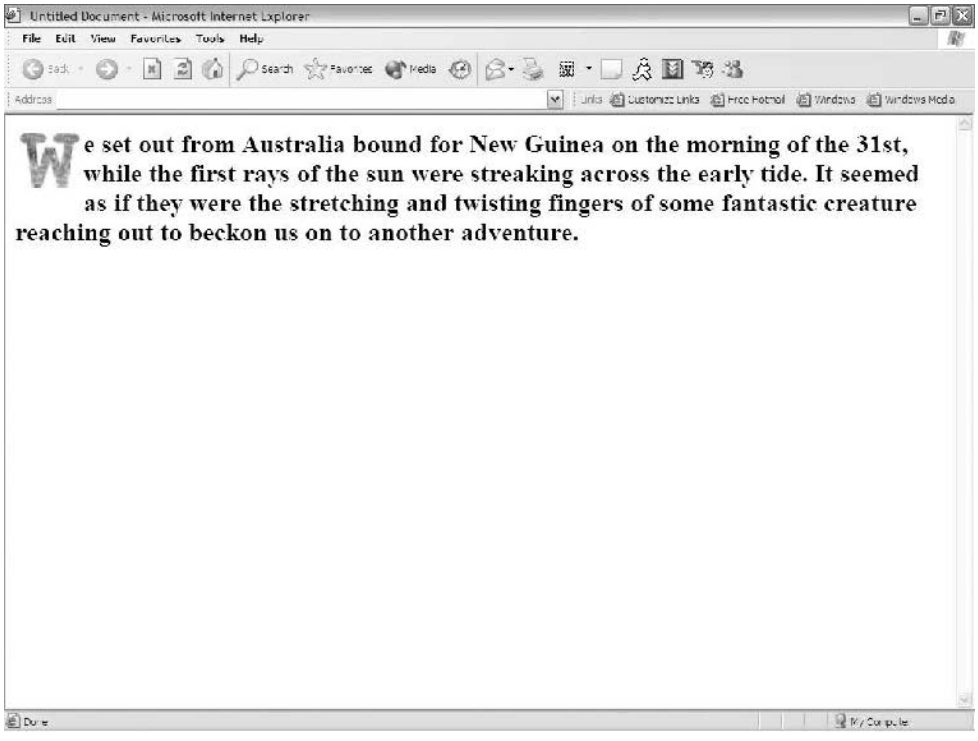


Figure 8-18: Drop caps are often used to spice up plain text.

Calculating sizes for tiling

Since the background image repeats endlessly until it reaches the right margin, you need to give some consideration to what happens to the last copy of the image on the line. Figure 8-19 shows a background image where the final copy is chopped off at the margin.

Because different visitors can have many different video settings, there is no way to avoid someone, some time, seeing a chopped-off background image. However, you can take steps to make it a rare occurrence.

The most common screen widths are 800 pixels, and 1024 pixels (although there are still some old computers around with only 640 pixels of width on their monitors). Some high-end systems can show screens as wide as 1280 or even 1600 pixels, but those settings are most often used for gaming or professional art. All of these resolutions are evenly divisible by 32, so any background image that is 32 pixels wide will tile evenly in all settings.

However, that's a pretty small size for many designs, so you might want to consider going up to 64 pixels instead. The problem with that is that it won't divide evenly into 800—you'll get 12.5 copies tiling across the screen. Somewhere along the line, you have to throw up your hands and just accept that the situation will never be perfect.

www.it-ebooks.info

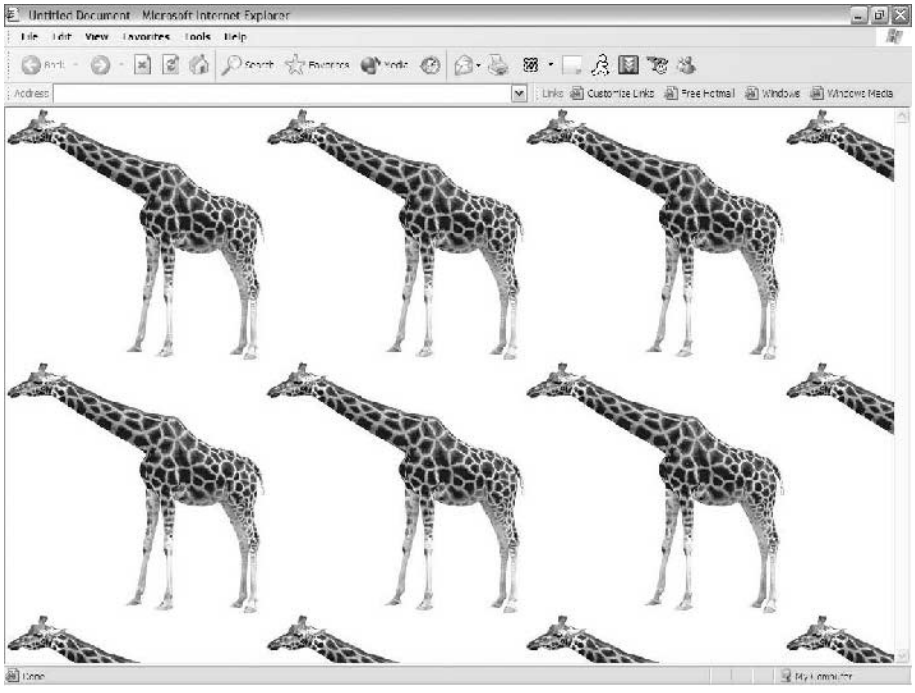


Figure 8-19: A chopped-off background image.

Dealing with "sidebar" backgrounds

Sidebar background images require a slightly different approach because horizontal tiling isn't a concern. Instead, they're meant to take up the entire width of the page with a single image. Vertical tiling still takes place, however, so the single image is repeated from top to bottom of the Web page.

Figure 8-20 shows a typical sidebar background image on a Web page.

The actual image used for the background is shown in Figure 8-21. You can see in Figure 8-20 how the vertical tiling creates the illusion of a page of notebook paper.

Sidebar backgrounds violate one of the basic rules of background images, because they create an area where the overlying text is obscured. In order to prevent this situation from occurring, pages with sidebar backgrounds must encapsulate their content within a right-aligned table that ensures it doesn't overlap with the sidebar.



See Chapter 11 for how to use tables.

One of the most common annoyances on the Web is a bad sidebar image that makes the text of the page unreadable, as shown in Figure 8-22. This problem arises when a background image isn't wide enough, and it ends up tiling horizontally. However, this situation is totally avoidable.

www.it-ebooks.info

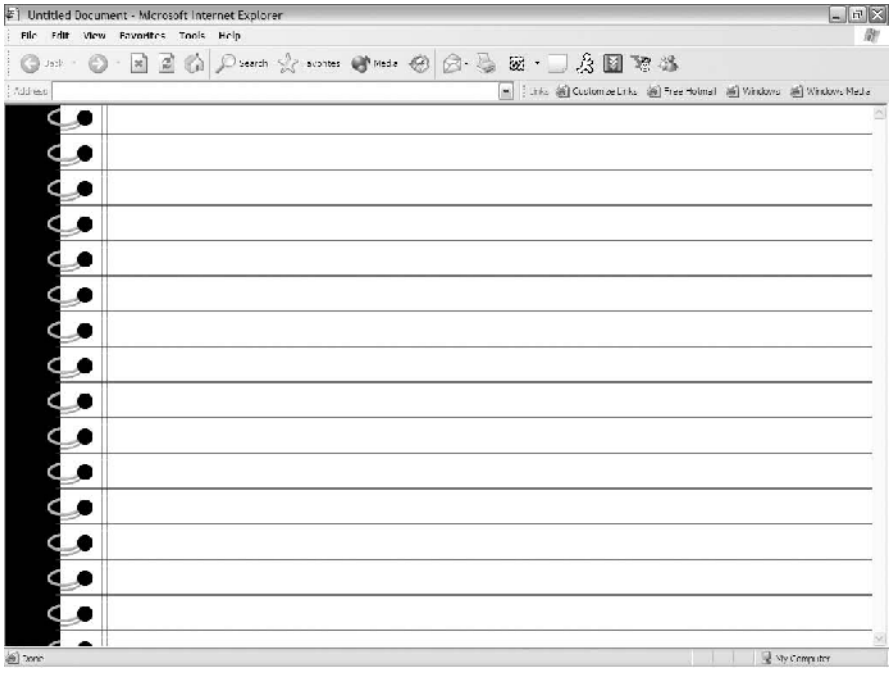


Figure 8-20: Sidebar images make a Web page appear as if it has a graphical design along the entire left side.

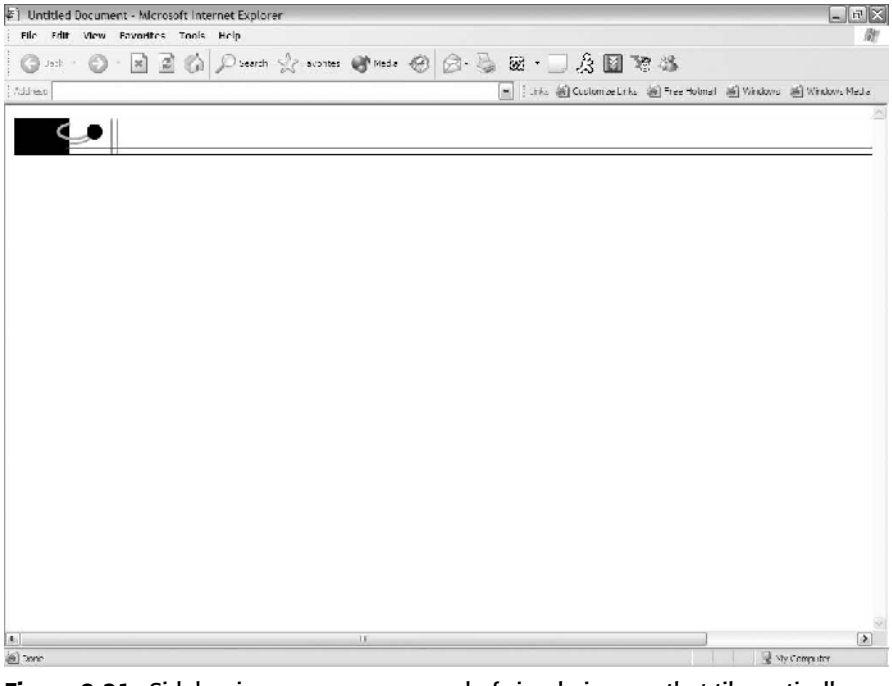


Figure 8-21: Sidebar images are composed of simple images that tile vertically.

www.it-ebooks.info

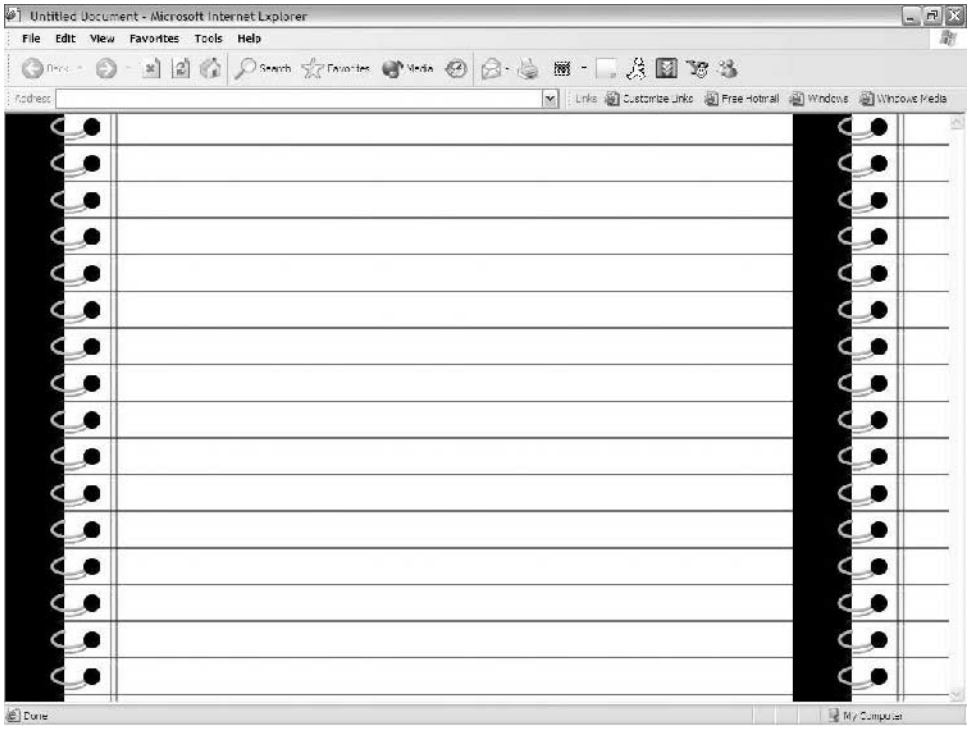


Figure 8-22: Sidebar backgrounds that are too short tile horizontally.

Failing to appreciate the wide variety of screen resolutions used on the Web results in horizontally tiling sidebar problems. What looks fine on an 800-pixel-wide screen looks terrible when viewed on a 1024-pixel screen. Solving this problem is incredibly simple—just make the sidebar image larger than you think you will ever need. A Web browser, faced with a background image that's too large to fit on a single line, simply chops off the end of the image.

When creating a sidebar image, make it as large as you think your most sophisticated visitor's screens will be. This means, in practice, that a typical sidebar must be at least 1280 pixels wide, and perhaps even 1600 pixels wide.

Checking seamlessness

The best kind of background image is one that tiles seamlessly—that is, an image that doesn't look like a bunch of repeated images, but a single, complex image covering the entire Web page. Figure 8-23 shows a seamless background.

Figure 8-24 shows the image used to create the seamless background. Note how the right side of the image matches perfectly with the left side; the top and bottom also match up. Unless the image you are using has these characteristics, a seam will be visible creating a series of squares.

www.it-ebooks.info

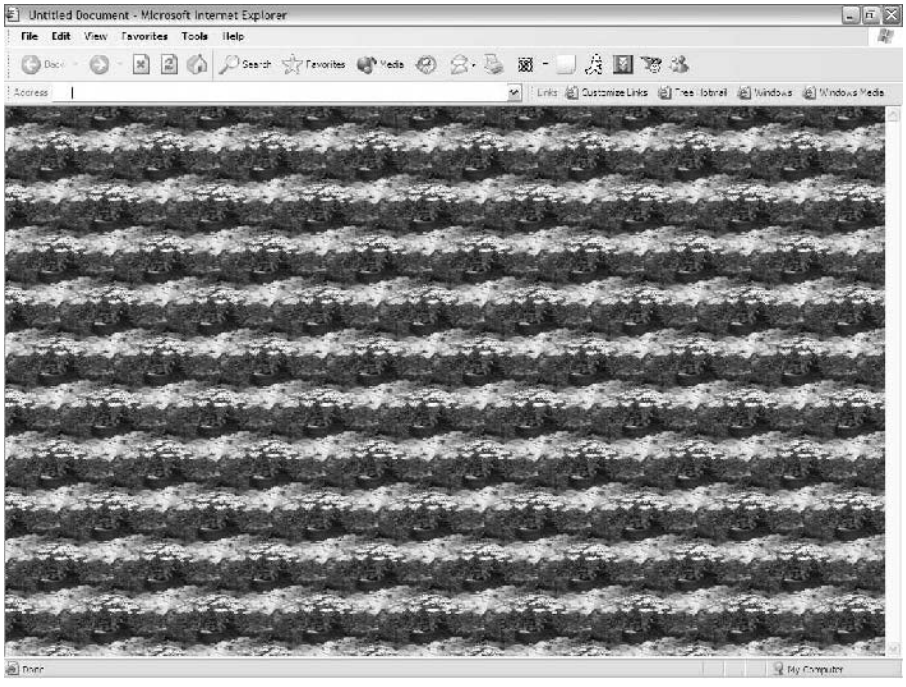


Figure 8-23: This figure shows a seamless background image.

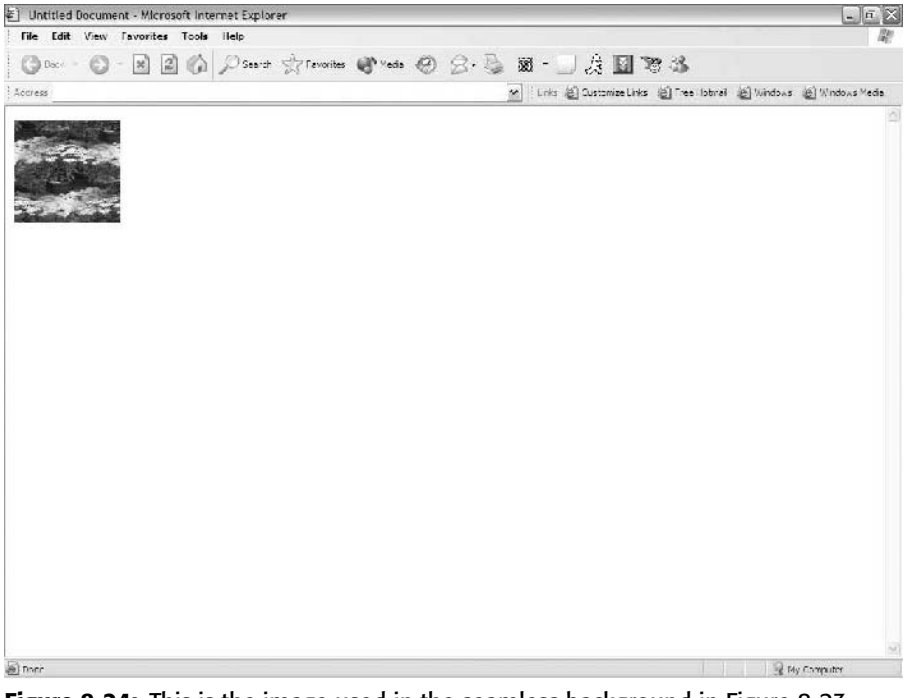


Figure 8-24: This is the image used in the seamless background in Figure 8-23.

www.it-ebooks.info

Figure 8-25 shows a seamed background image. Although the effect isn't a fatal defect, it can be distracting.

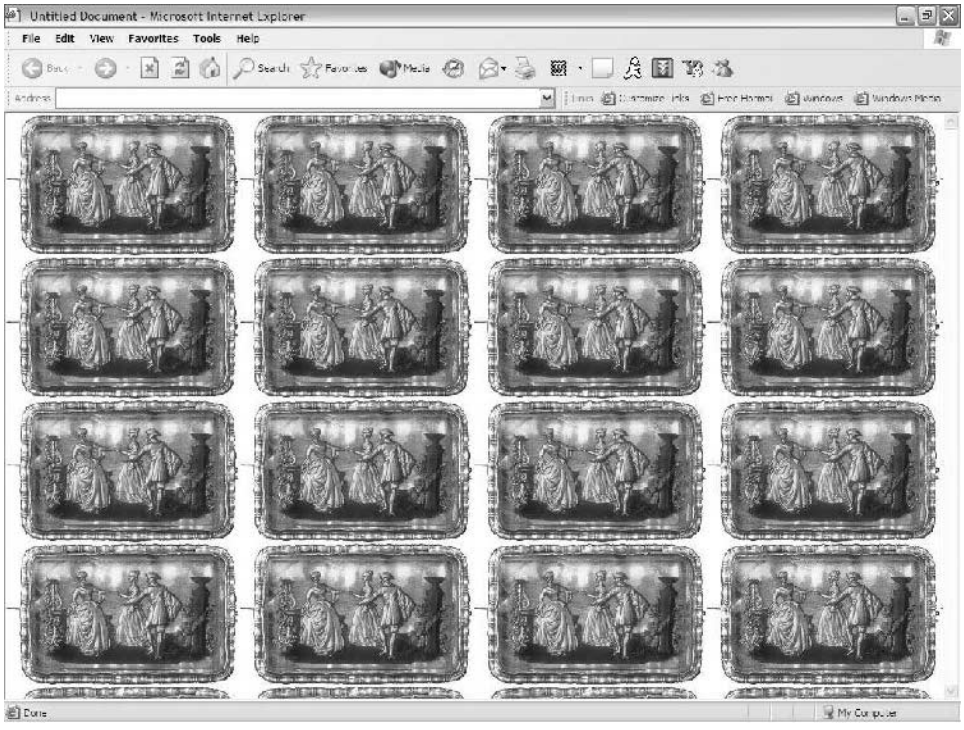


Figure 8-25: Seamed backgrounds can be distracting.

Avoiding busy backgrounds

Another common problem with background images is that some of them tend to be so complex and intricate that they distract a visitor's attention from the overlying text and images. Figure 8-26 shows such a busy background and how it detracts from the overlying elements.

Ensure that the background images you choose do not detract from the basic materials on your page.

Choosing color and contrast

A background image that is too close in color or value to the overlying text or images can be disastrous. It's amazing, though, how many times you find things like white text on a yellow background or purple on black. Pray that you never suffer a Web page with orange on red (see Figure 8-27).



See Chapter 10 for more information on using color in Web pages.

www.it-ebooks.info

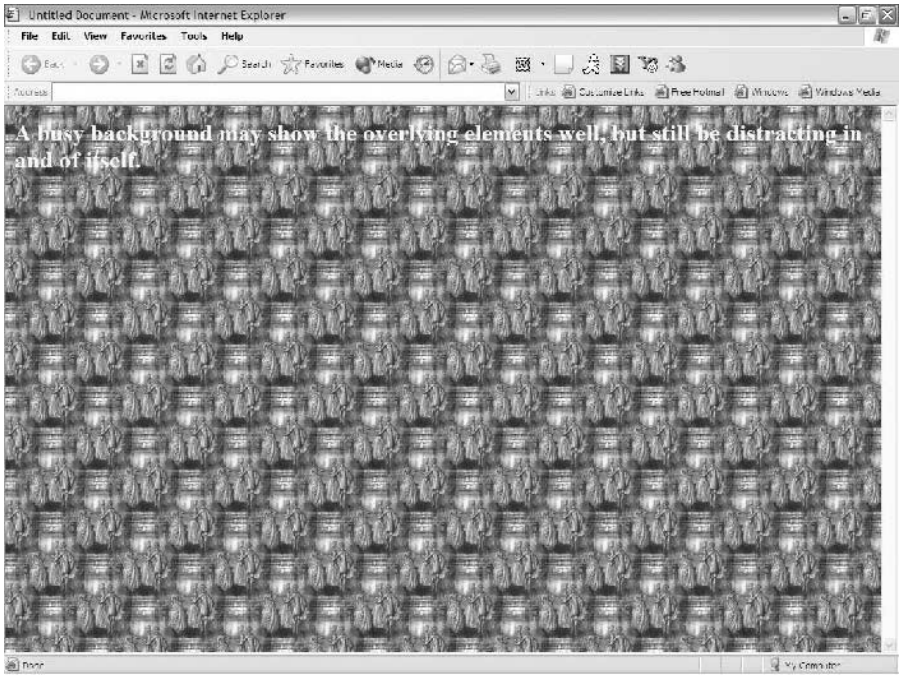


Figure 8-26: Some background images can interfere with the overlying elements.

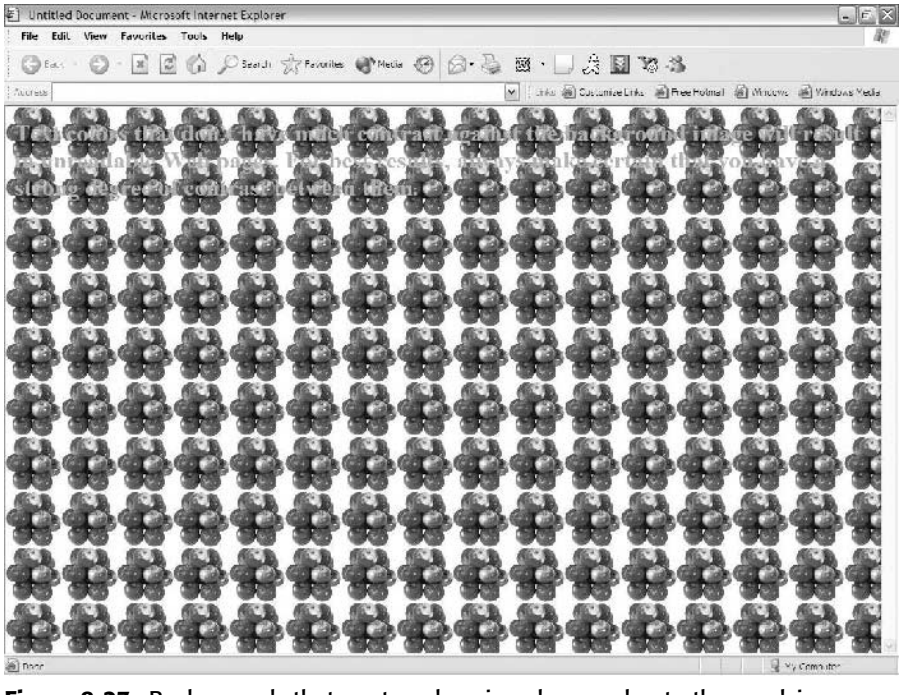


Figure 8-27: Backgrounds that are too close in color or value to the overlying elements can make Web pages hard to use.

www.it-ebooks.info

Why anyone would go to the trouble of creating a Web page that no one can read, we can't even begin to guess. Perhaps, they failed to test it on more than one system or more than one color resolution. Many Web pages that look fine in high-color or true-color settings look pretty bad when viewed in 256 colors. It is Web designers' responsibility to ensure that their work is enjoyable to as many visitors as possible.

Summary

Images are critical to the success of any serious Web site. The choice and proper use of images influences both the appearance and functionality of your site. This chapter introduced you to the following concepts and topics:

- ✦ Images are added with the IMG element.
- ✦ The width and height attributes can be used to modify the apparent size of images.
- ✦ The align attribute sets how text flows around images.
- ✦ Images can be used as links.
- ✦ Background images that contrast well with other page elements can improve the page's appearance.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 9

Creating and Enhancing Images

In This Chapter

Choosing a drawing tool

Modifying images

Using 3D graphics programs

Unless you're going to totally abandon all pretense of having anything to do with Web page graphics and hand all that mysterious stuff over to someone else, you need to have some notion of what needs to be done to make your Web pages work. Even if you don't ever touch an image yourself, you'd better be aware of what is going on under the hood so that you have an idea of what can and can't be done.

It is impossible to do more than scratch the surface of such a complex topic in a single chapter. Thus, this is an overview.

Choosing an Image Editing Tool

There are four graphics programs that anyone seriously interested in Web design should be familiar with—Fireworks, Photoshop, Painter, and Paint Shop Pro. At least one of these, and perhaps all of them, belong in your Web page creation toolkit.

Photoshop

Adobe Photoshop (see Figure 9-1) is the old, venerable workhorse of the graphics industry, familiar to every serious digital artist. Adobe Photoshop is important, not only because it has long been the industry standard, but also because of the features it lacks. When you get right down to it, all of the major graphics programs are fairly similar, and the choice of one over the other is largely a personal matter of user comfort. When it comes to adding features, however, you need to pay attention to whether or not the program you're using can handle Photoshop-compatible plug-ins. All four of the paint programs discussed here can handle these plug-ins.



You can download a trial version of Photoshop at adobe.com/products/tryadobe/main.jsp.

www.it-ebooks.info

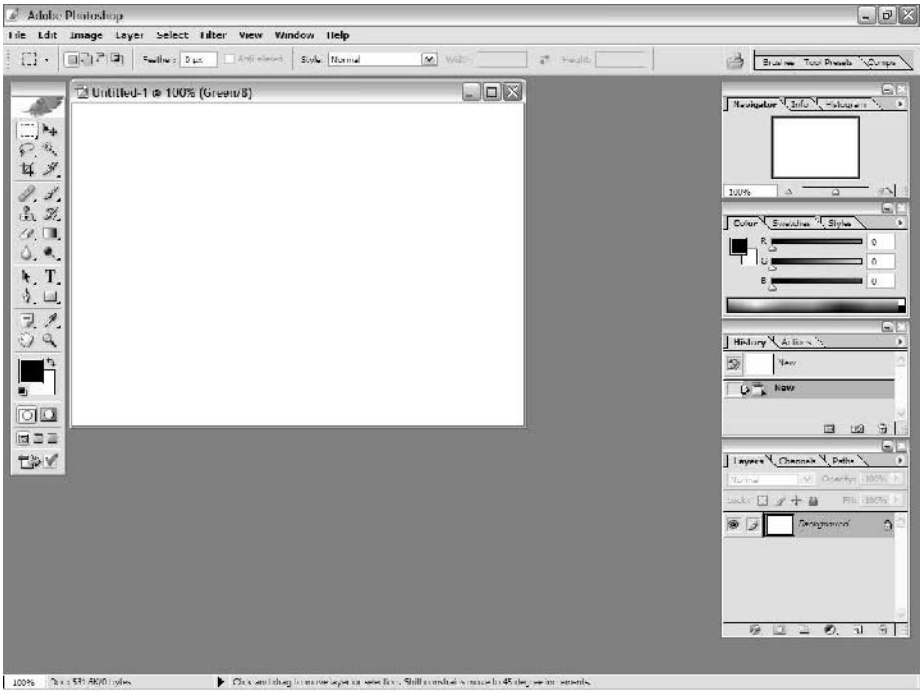


Figure 9-1: Photoshop is the most popular image editing program.

Fireworks

Macromedia Fireworks (see Figure 9-2) is a relative newcomer to the graphics program marketplace, and it is one of the most impressive programs I've ever used. It is similar to Photoshop in layout and function. Fireworks, however, is a child of the Web and was deliberately designed for the purpose of making graphics for Web pages, rather than having such functionality tacked on as an afterthought. It was planned to use in concert with its companion program, Dreamweaver, the premier Web page creation tool, but it is also a perfect standalone graphics program.

Painter

Corel Painter (see Figure 9-3) gives you the feeling that you have gone into what is almost a case of virtual reality rather than just another paint program. Working with it is like working on a real canvas with brushes so realistic that they almost drip paint onto your desk. You feel as if you need to keep a bottle of turpentine handy.

Paint Shop Pro

Jasc Paint Shop Pro (see Figure 9-4) is a delightful rarity—a shareware program that really is as good as its commercial competitors.

www.it-ebooks.info

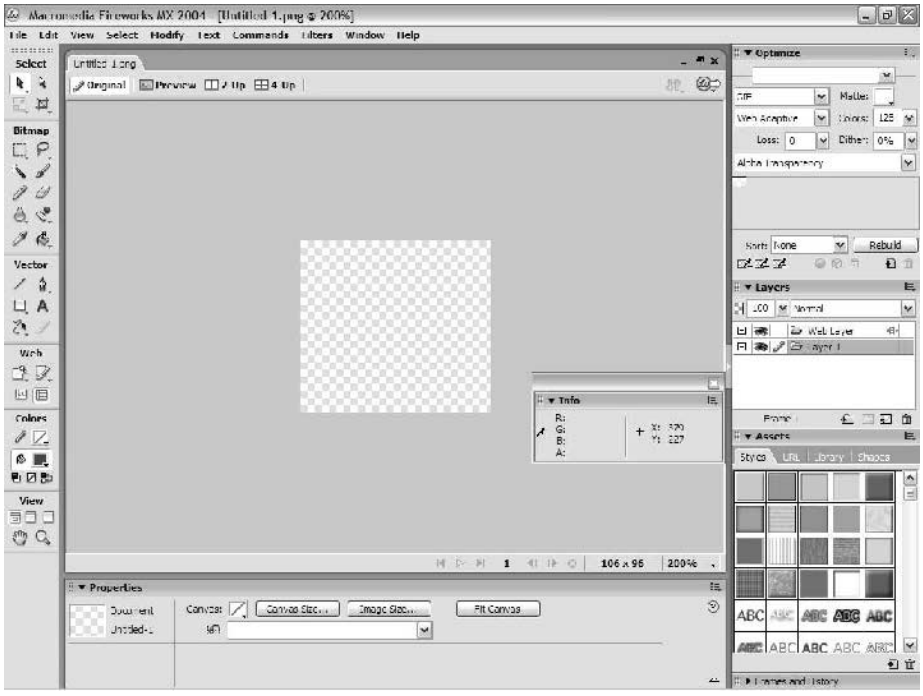


Figure 9-2: Fireworks is designed for creating Web graphics.

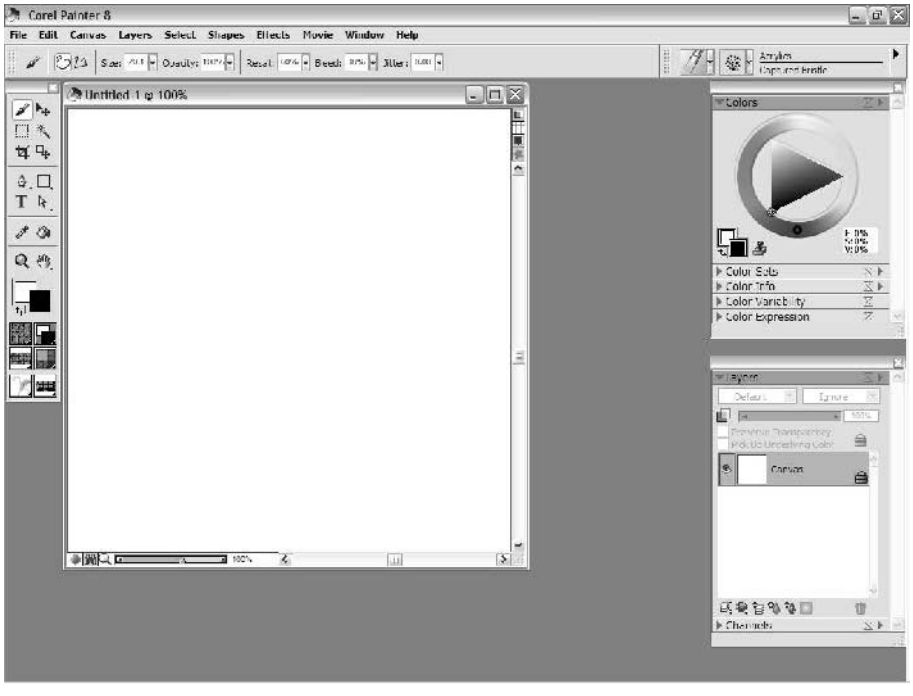


Figure 9-3: Painter is a good program for those making the transition from traditional art to digital art.

www.it-ebooks.info

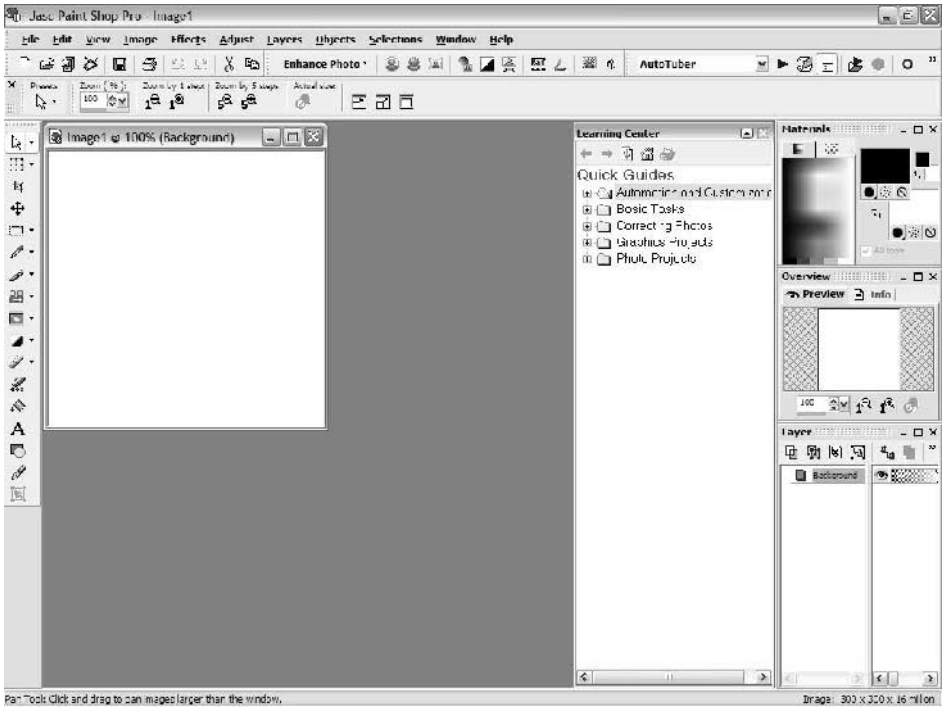


Figure 9-4: Paint Shop Pro, although shareware, is as good as any other paint program.

Modifying Images

Most Web designers aren't good enough artists to draw their own images from scratch. This has led to a large market in clip art—ready-to-use images, either photographs or drawings—that can just be dropped into your Web pages as needed.



See Chapter 7 for more information on getting clip art for use on your Web site.

However, the needs of each site—indeed, the needs of each page within a site—vary, and there's often something about an image that makes it just not quite right. Perhaps it's too large, or too dark, or whatever. But with a little bit of tweaking, it can be made to work just fine.

Every graphics program has several methods of handling these situations. If you don't want to get deeply involved in learning the ins and outs of sophisticated graphics software just so you can use a few of the features once in a while, you might want to try a handy program called IrfanView (see Figure 9-5). It's a graphics viewer that includes some nice extras such as file type conversion and most of the capabilities discussed in this section.

www.it-ebooks.info

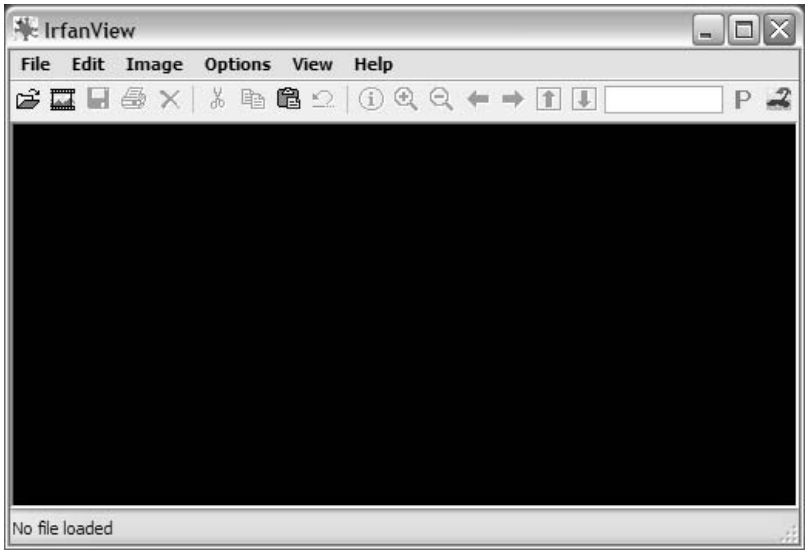


Figure 9-5: IrfanView is a good program for making small, quick adjustments to images.

Cropping

Often, an image would be very useful, but in addition to things you want, there are also things in it you don't want. For example, you have a picture of a bagel and a cup of coffee, but you only want to show the bagel. Cropping, which is eliminating all but a selected portion of an image, solves the problem. Figure 9-6 shows the original image and, underneath it, the cropped version.



Figure 9-6: Cropping enables you to eliminate the undesired portions of an image.

The process is similar, but not identical, in each of the four major graphics programs. In each case, you use the Crop tool (see Figure 9-7). Three of the programs also offer a menu-based approach.

www.it-ebooks.info



Figure 9-7: The Crop tool looks the same in all four programs.

Photoshop
To crop an image in Photoshop, click the Crop tool. Then drag it to select the area you want to crop. When the area is selected, double-click within it. Alternatively, select the portion you want and choose Image ➪ Crop from the menu.

Fireworks
To crop an image in Photoshop, click the Crop tool. Then drag it to select the area you want to crop. When the area is selected, double-click within it. You can also select the part to be cropped, then choose Edit ➪ Crop Document from the menu.

Painter
To crop an image in Painter, click the Crop tool. Then drag it to select the area you want to crop. When the area is selected, click within it (the cursor changes to an animated pair of scissors when it is over the selected area).

Paint Shop Pro
To crop an image in Paint Shop Pro, click the Crop tool. Then drag it to select the area you want to crop. When the area is selected, double-click anywhere on the image. Alternatively, select the area, then choose Image ➪ Crop to Selection from the menu.

Resizing and resampling

One of the most common problems with images on Web pages is dealing with the physical size of the image. There's just a certain amount of real estate available on a computer screen, and a large image can quickly eat it up and overwhelm the rest of the Web page. While it's true that you can easily set the height and width attributes of a Web image, changing its apparent size on the Web page, this is not the ideal solution because you aren't actually changing anything but the way the image is displayed in a Web browser.

The original file is still the same size as it originally was, which means that its download time is still longer than it has to be. Also, Web browsers are not as sophisticated as graphics programs—you can get far better results by modifying the image in one of them instead, like the one in Figure 9-8, which was resized in Fireworks.

When an image's size is altered, it runs into an information problem. Although you might not think about it very often, each pixel in an image is data. If the image is made smaller, there can't be as many pixels in it, meaning that some of that data has to go. If it's made larger, it has to have more pixels than it originally did, and that data has to come from somewhere. Both processes are handled by *resampling*, which is the process of figuring out what to do about the wayward pixels.



Making an image smaller is called *downsampling*. The reverse process is called *upsampling*.

www.it-ebooks.info

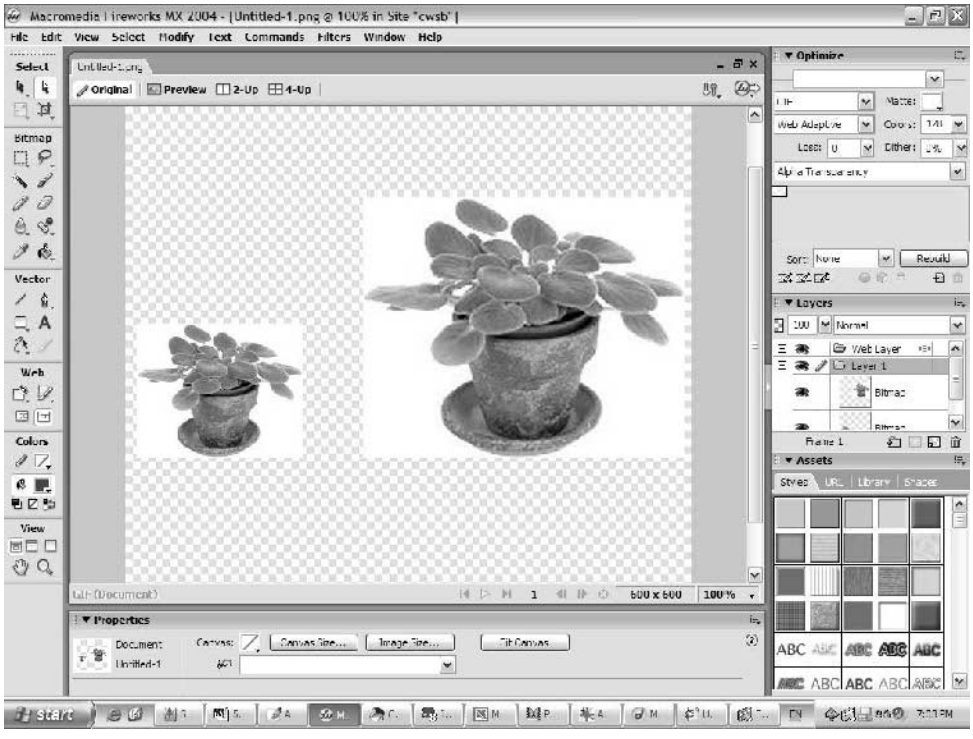


Figure 9-8: Resizing in a graphics program produces better effects than simply resetting the width and height attributes in HTML.

Photoshop

To resize an image in Photoshop, follow these steps:

1. Choose Image ➪ Image Size from the menu.
2. In the resulting Image Size dialog box, shown in Figure 9-9, enter the new figures for either the Width or Height. You can use pixels or percentages.



If the Constrain Proportions check box is selected, you only need to enter one of the figures. The other is automatically calculated to keep the image from being distorted.

3. If desired, you can specify the type of resampling calculations to be performed by choosing one from the Resample Image drop-down list. The default is Bicubic sampling, which is the best.
4. Click the OK button.

Fireworks

In Fireworks, you resize an image exactly as you do in Photoshop, except that you choose Modify ➪ Canvas ➪ Image Size from the menu in Step 1.

www.it-ebooks.info

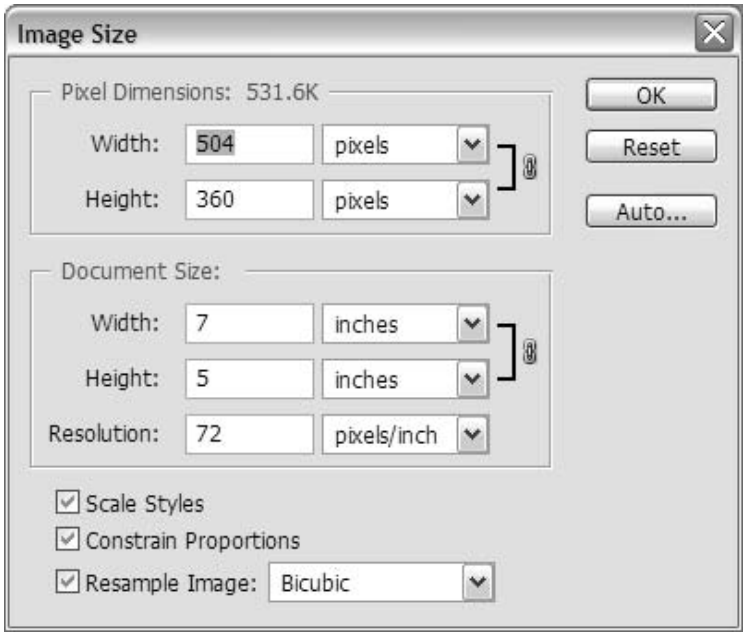


Figure 9-9: This figure shows the Image Size dialog box in Photoshop.

Painter

Painter also follows the same routine as Photoshop, except that you choose Canvas ➪ Resize from the menu in Step 1.

Paint Shop Pro

Paint Shop Pro's resizing works similarly, but the dialog box is quite different.

1. Choose Image ➪ Resize from the menu.
2. In the Resize dialog box, shown in Figure 9-10, you can either enter the new figures for the Width or Height manually, or you can use the two arrows to the right of the figures to scroll up and down. The third, larger, arrow brings up a slider bar that you can use to more quickly change the size.



You can set the size in either pixels or percentages.

3. To keep the image's proportions intact, make sure that the Lock Aspect Ratio is set to 1.0000.
4. Click OK.

Rotating and flipping

The orientation of an image can be altered by just about any graphics program. Some of them have only simple choices such as *flipping* an image horizontally or vertically.

www.it-ebooks.info

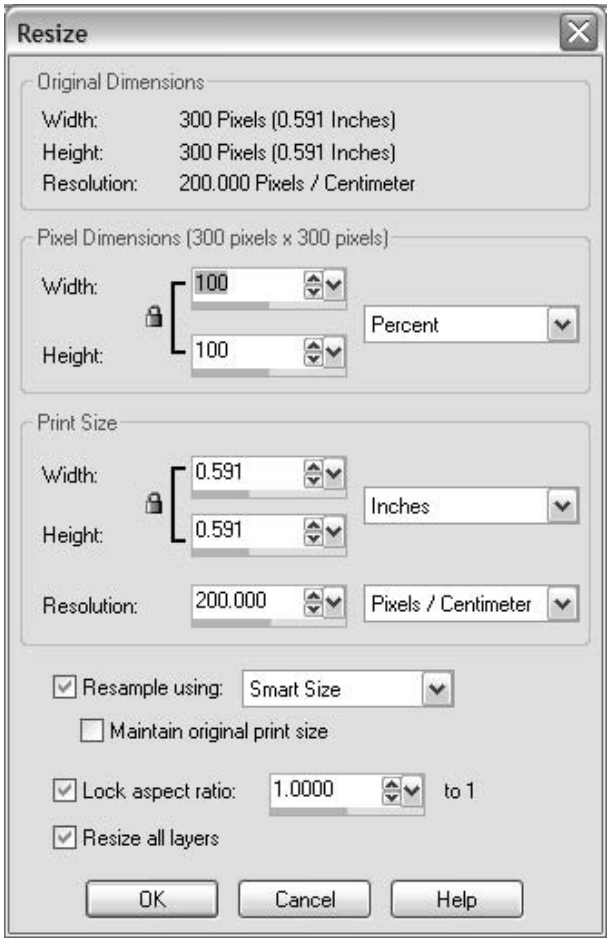


Figure 9-10: The Resize dialog box in Paint Shop Pro is different.

Flipping means to create a mirror image. Thus, if you have a portrait in which the person is facing right and you flip it horizontally, you end up with an image in which they are facing to the left. If you flip an image vertically, you are simply turning it upside down.

Rotation, on the other hand, can be done in either a simple or complex way. Most programs enable you to rotate by some fixed amount—usually in multiples of 90 degrees. The better ones, however, enable you to specify a particular degree of rotation—anywhere from 1 to 360 degrees. Figure 9-11 shows an image in its original orientation and rotated as well.

Photoshop

Photoshop enables both coarse and fine rotation. For coarse rotation, simply choose Image ➪ Rotate Canvas from the menu, then choose 180, 90 CW, 90 CCW, Flip Canvas Horizontal, or Flip Canvas Vertical from the submenu.

www.it-ebooks.info

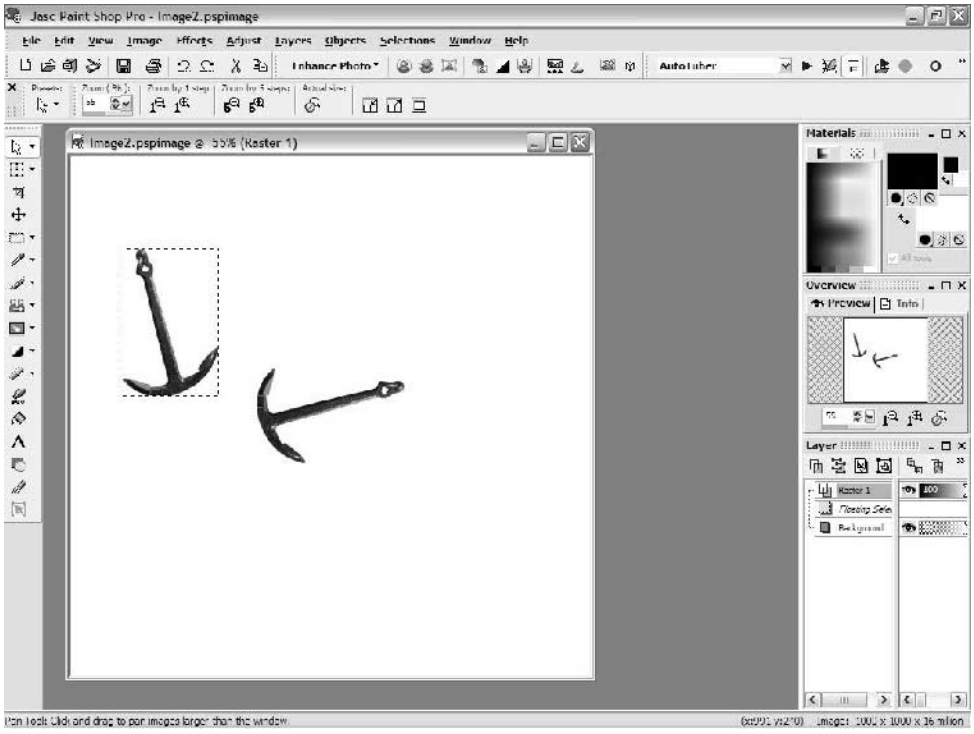


Figure 9-11: This figure shows a rotated image.

For fine rotation, follow these steps:

1. Choose Image ➪ Rotate Canvas ➪ Abitrary from the menu.
2. In the Rotate Canvas dialog box, shown in Figure 9-12, enter the number of degrees by which you want to rotate the image.

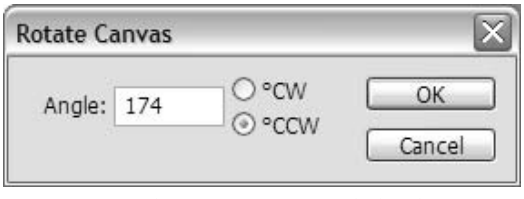


Figure 9-12: The Rotate Canvas dialog box speaks for itself.

3. Click either the CW or CCW radio buttons to select clockwise or counterclockwise rotation.
4. Click the OK button.

www.it-ebooks.info

Fireworks

For simple 90-degree rotation and flipping in Fireworks, select Modify ➪ Transform from the menu, then choose 180, 90 CW, 90 CCW, Flip Horizontal, or Flip Vertical from the submenu.

For finer rotation, the method isn't as obvious, because it isn't called rotation.

Here's how to do it:

1. Choose Modify ➪ Transform ➪ Skew from the menu.
2. Place the mouse pointer anywhere but on the image (the pointer changes to a circular arrow), hold down the left mouse button, and move the pointer in the direction you want the image to rotate.
3. When you are satisfied with the position of the image, release the mouse button.

Painter

1. Choose Effects ➪ Orientation ➪ Rotate from the menu.
2. In the Rotate Selection dialog box, enter the value for the number of degrees you want to rotate. This is the only option.



In Painter, all rotation is counterclockwise. To rotate clockwise, enter a negative number.

3. Click the OK button.

Paint Shop Pro

To do a quick 90 degree rotation, choose Image ➪ Rotate ➪, then choose either Rotate Clockwise 90 or Rotate Counter-clockwise 90. For fine-tuned rotation, follow these steps:

1. Choose Image ➪ Rotate from the menu.
2. In the Rotate dialog box (see Figure 9-13), click the Right or Left option buttons to set the direction of rotation.

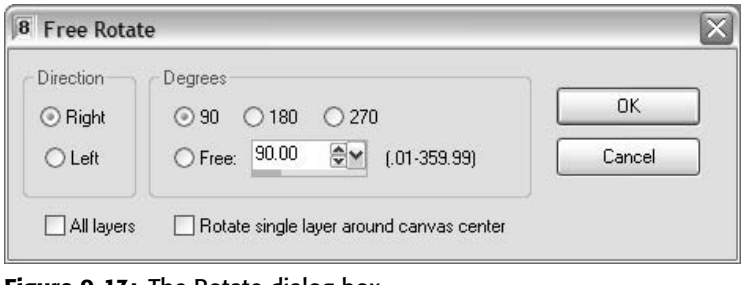


Figure 9-13: The Rotate dialog box.

www.it-ebooks.info

3. To rotate by a fixed degree, click one of the three option buttons in the Degrees panel.
4. To rotate by a custom degree, click the Free option button and enter its value in the text box next to it.
5. Click the OK button.

Sharpening and blurring

It's difficult to repair a bad image, but sharpening can sometimes do it. If the image is a bit out of focus, applying a sharpening filter, in theory, makes it appear to be more focused. We're hedging so much in these comments because, with most images, sharpening doesn't do much to improve the appearance—it simply brings out more of the little flaws you didn't originally notice but still leaves the important parts out of focus. It's, sadly, one of those things that sounds great but just doesn't work well in practice much of the time.

Blurring, on the other hand, is the reverse—it takes a perfectly clear image and deliberately makes it out of focus. Despite how bizarre that may sound at first, there are often good reasons for doing this. Say, for instance, that you were doing a Web site for an eyeglass manufacturer—you could use blurring for a before and after effect. The same technique could be used in any situation where a metaphor for achieving sight—or insight—would apply. Also, you don't have to apply a heavy amount of blur. A tiny amount of it results in a simple softening that can help disguise, or even eliminate, small flaws or graininess in an image.

Blurring and sharpening work pretty much the same in all four programs, although there are minor differences in the menu options and layout:

Photoshop

In Photoshop, choose Filter ➪ Blur or Filter ➪ Sharpen from the menu. From the submenu, select the type of blur or sharpening you wish to use.

Fireworks

In Fireworks, choose Filters ➪ Blur or Filters ➪ Sharpen from the menu. From the submenu, select the type of blur or sharpening.

Painter

In Painter, choose Effects ➪ Focus from the menu. Then select the kind of blur or sharpening from the submenu.

Paint Shop Pro

In Paint Shop Pro, choose Adjust ➪ Blur or Adjust ➪ Sharpness from the menu. Next, select the type of blur or sharpening.

Applying artistic effects filters

Plug-in filters, such as Kai's Power Tools (KPT) extend the capabilities of graphics programs. The standard for graphics filters is Photoshop compatibility—always

www.it-ebooks.info

check any graphics program you plan to use to ensure that it can use Photoshop filters. If it doesn't, you will find that your choices are rather limited. If it does, there are about 20 gazillion different options you can explore.

The basic kinds of filters that come with most image editing programs (sharpen, gamma correction, and so on) are fairly common and they are useful in a variety of situations. If you want to get really artistic, adding effects such as metallic sheens, lightning, and so forth, you'll have to go for the plug-ins.

The exact method of operation depends on the individual plug-in filter, but in order to use them at all, you have to make those filters available—the program you're using them in has to be told where to find them.

Photoshop

Photoshop keeps plug-ins in its own dedicated folder. However, you may also indicate a second plug-in folder as in the following:

1. Choose Edit ➪ Preferences ➪ Plug-Ins and Scratch Disks from the menu.
2. In the Preferences dialog box, shown in Figure 9-14, click the Additional Plug-Ins Directory check box.

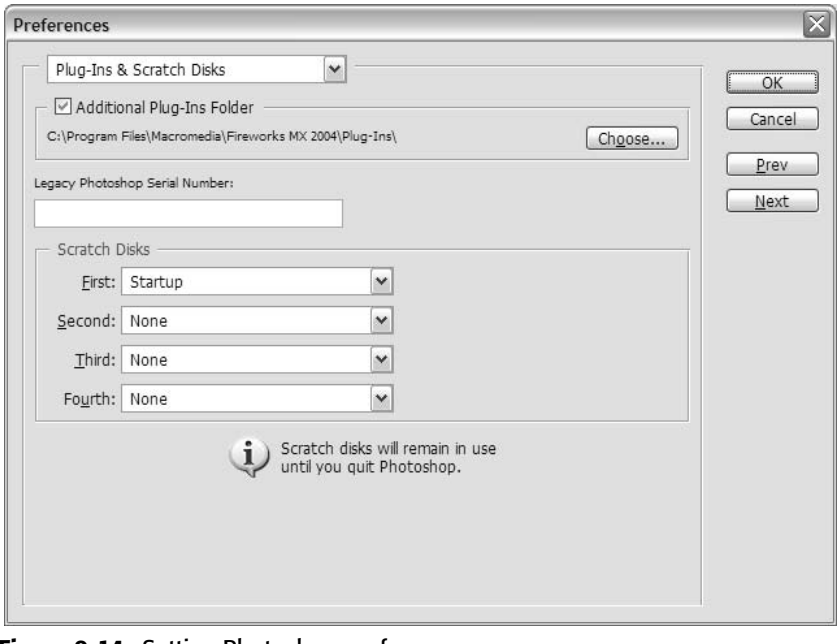


Figure 9-14: Setting Photoshop preferences.

3. Click the Choose button and navigate to the folder that holds your plug-in filters. Then select it and click the OK button to return to the Preferences dialog box.
4. Click the OK button to finish.

www.it-ebooks.info

Fireworks

In Fireworks, the process is virtually identical:

1. Choose Edit ➪ Preferences from the menu.
2. In the Preferences dialog box, shown in Figure 9-15, click the Folders tab.

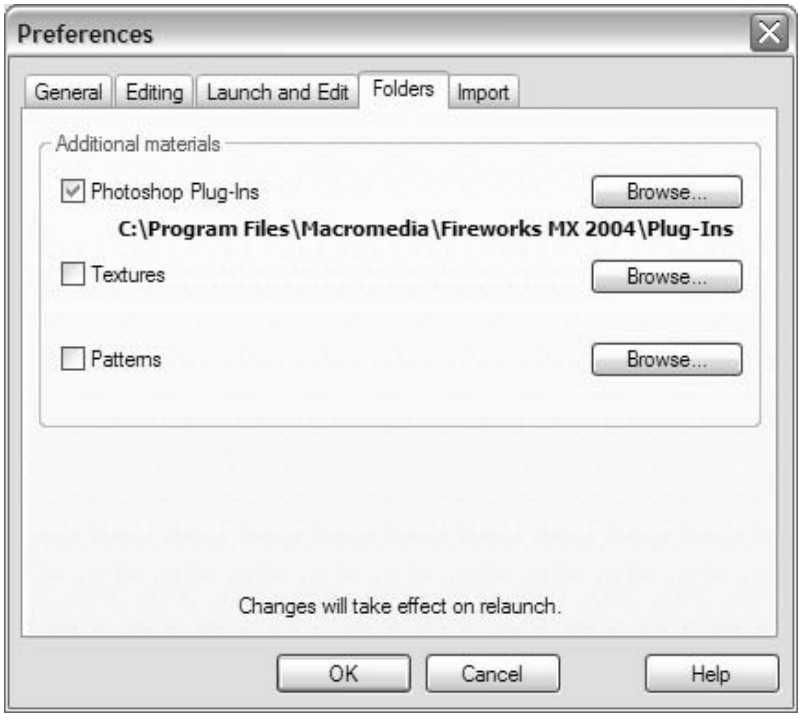


Figure 9-15: Setting Fireworks preferences.

3. Click the Photoshop Plug-Ins check box. Then click the Browse button.
4. Navigate to the folder that holds your plug-in filters. Then select it and click the OK button to return to the Preferences dialog box.
5. Click OK to finish.

Painter

Painter has a comparatively laborious process. You have to create a shortcut to the folder containing the plug-ins and add that shortcut to Painter's own plug-ins folder. In a typical Windows installation, this would be `C:\Program Files\Corel\Corel Painter 8\Plugins`. The way to avoid this is to simply install the plug-ins into that folder.

www.it-ebooks.info

Paint Shop Pro

Paint Shop Pro allows multiple plug-in folders. To show the program where your plug-ins are located, follow these steps:

1. Choose File ➪ Preferences ➪ File Locations from the menu.
2. In the File Locations dialog box, shown in Figure 9-16, click the Plug-ins option.

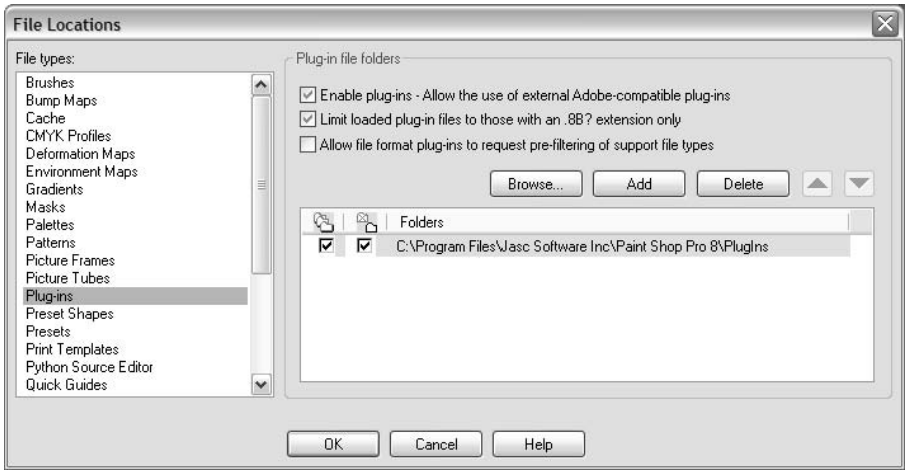


Figure 9-16: Setting Paint Shop Pro preferences.

3. Click the Add button, then click the Browse button.
4. Navigate to the folder holding your plug-in filters and select it. Click the OK button to return to the Preferences dialog box.
5. Repeat Steps 3 and 4 until you have added all desired plug-in folders.
6. Click OK to finish.

Using 3D Graphics Programs

While standard image editing software is very useful—in fact, irreplaceable—most people do not have the artistic skill necessary to use it to its fullest. Art is, well, an art. And it takes a lifetime commitment to really do it well. Most Web designers are more skilled with computers, a bit more left-brain oriented, than they are with artistic endeavors. Yet, graphics are a big part of the World Wide Web. It's difficult to imagine a serious Web site without graphics, and you can't always find just what you need ready-made and available for use.

Fortunately, there's a fabulous solution available. Graphics programs that are used to create 3D scenes are geared toward exactly those people who are least comfortable with creating art in the traditional manner. Although some degree of

www.it-ebooks.info

visual awareness is required—you must be able to imagine what you want the finished product to look like or you won't know where you want to go—no manual skill is needed. The 3D scenes can, when completed, be exported as normal two-dimensional graphics files, thus giving anyone the capability to create professional-looking images for their Web sites.



All 3D programs are capable of both importing and exporting a variety of 3D file formats, which means that you can use objects that were created in one program in another one.

Poser

Poser, from Creative Labs, solves one of the most pressing problems for non-artists—it creates people (see Figure 9-17). Well, images of people, anyway. Every single part of each character can be customized, and a variety of props—from clothing to hair to furniture—can be incorporated into the scene. As the name of the program suggests, you can position your characters any way you want to.



Figure 9-17: Poser enables you to easily create detailed and realistic human figures.

The human figure is one of the most difficult of all things to manage perfectly, and Poser does run into an occasional problem. First of all, it's pretty easy to do

www.it-ebooks.info

impossible things in Poser, such as moving an arm so that someone's hand is in the middle of their head. The program lacks rules that would prevent this. On the other hand (no pun intended), this also gives you total flexibility in character design—and Poser is often used to create nonhuman characters.

Bryce

Bryce, from Corel, is a favorite of 3D artists because it's capable of quickly and easily creating realistic settings (see Figures 9-18 and 9-19). It's also a favorite because it's capable of quickly and easily creating totally unrealistic settings. Whichever way your inclinations take you, Bryce happily generates land, sea, sky, stars, sun, and moon. It even throws in things like trees and comets if you instruct it to.

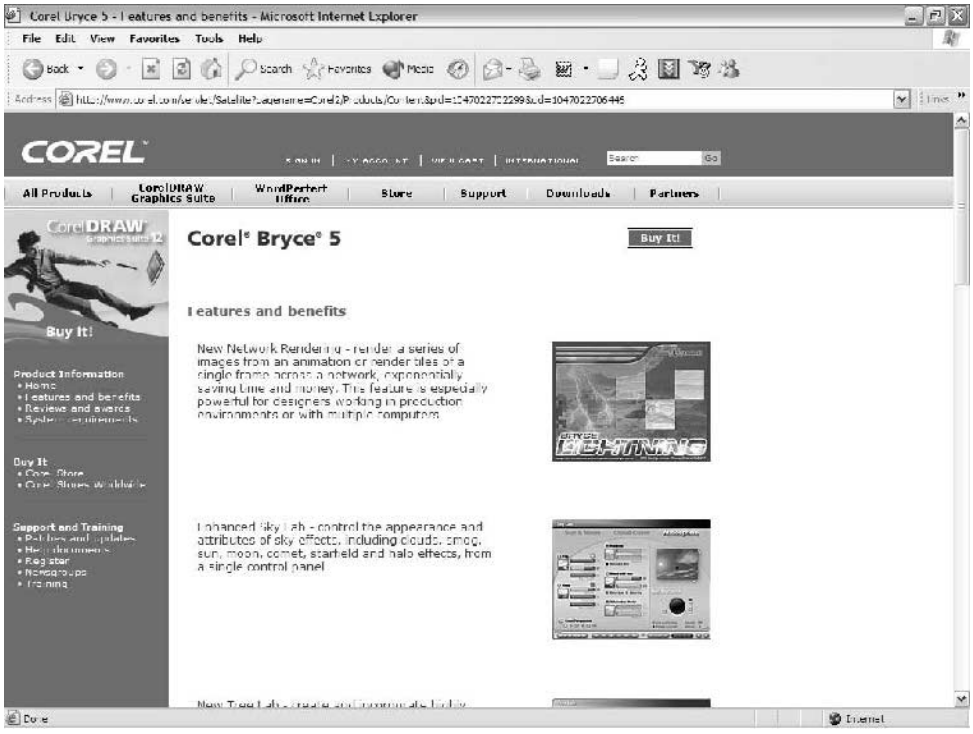


Figure 9-18: Bryce lets you make your own worlds.

Bryce's main drawback, believe it or not, is simply that it does not do trees well. Its foliage, however, is adequate for most tasks, especially for Web work, where graphics tend to be small enough so that tiny details are obscured.

trueSpace

Caligari trueSpace is a heavy-duty 3D program, requiring a much steeper learning curve than Bryce or Poser (see Figure 9-20). The interface alone is a serious

www.it-ebooks.info



Figure 9-19: A seascape created in Bryce.

challenge and, if you haven't cut your teeth on simpler 3D software, you might find this one a little bit intimidating. The old saying that anything worth doing takes some effort, however, is true here. trueSpace is more challenging because it is more flexible. Basically, if you can imagine it, you can make it in trueSpace.



Figure 9-20: trueSpace is a more powerful and flexible 3D program.

www.it-ebooks.info

This program is, in my opinion, the best of the general 3D graphics programs. It is moderately priced, yet fully capable of holding its own with software costing much more. In fact, it offers a variety of features that are fabulously advanced. The best of these is that you can model in real time, operating directly on the various parts of your 3D creations.

iSpace

Caligari iSpace is a truly unique program, and one that you must check out (see Figure 9-21). It's a 3D program specifically designed for making Web pages—not just the graphics for the pages, but the pages themselves as well. Unlike its big brother, trueSpace, the learning curve for this one is minimal. The results, however, are stunning.

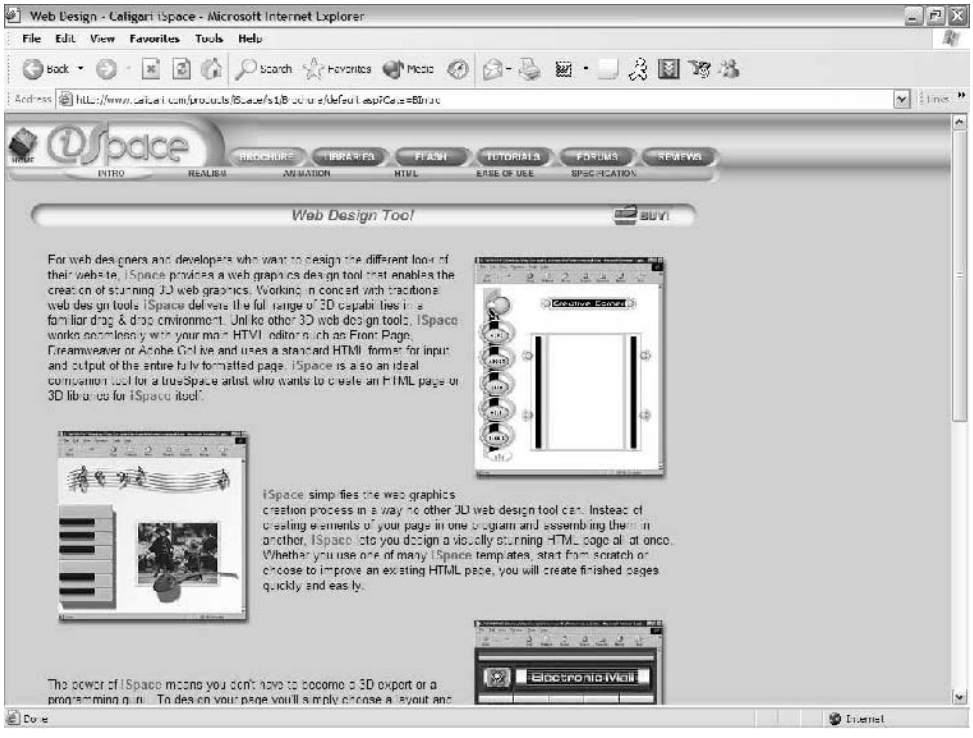


Figure 9-21: iSpace is used to create 3D Web pages.

The selection of 3D fonts, buttons, and widgets is incredible. The program enables you to create fabulously sophisticated Web pages complete with animated, interactive features that rival the best any major company could do with an unlimited budget.

www.it-ebooks.info

3D models and more

The 3D artist community is one of the most supportive you'll ever experience. If you are just getting started, you'll find that there are crowds of people willing to answer your questions and endless Web pages filled with tutorials. As you move into the intermediate stages, you'll be able to use and adapt a variety of existing models, from Poser people to spaceships. These are available both as freebies and as commercial products. Some companies, such as DAZ3D, offer free models as well as selling others (see Figure 9-22).

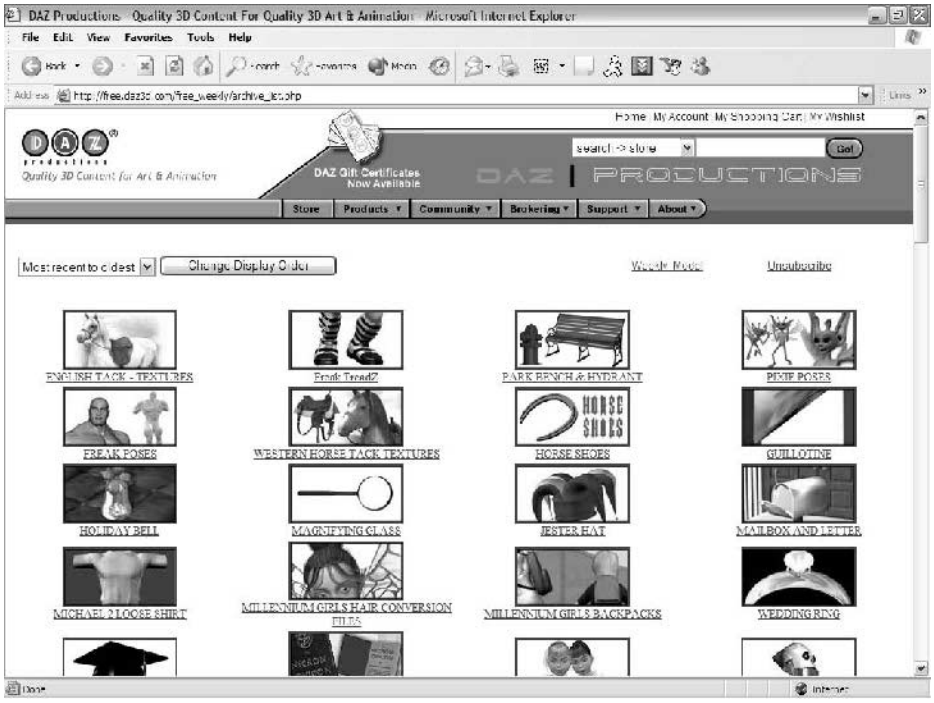


Figure 9-22: DAZ3D offers free, high-quality 3D models each week.

Caution Always carefully examine the accompanying documentation for license information, especially with free 3D models. Some artists specify that their freebies cannot be used in commercial artwork.

Once you have moved into the advanced realm, you'll find that the same companies you used to buy models from now stand ready to market your own 3D models for you.

Summary

The choice of the right tool for creating and altering images can be critical to your success. Although most drawing tools work in a similar way, each has its own quirks,

www.it-ebooks.info

and the choice is ultimately a personal one. This chapter introduced you to the following concepts and topics:

- ✦ Fireworks, Photoshop, Painter, and Paint Shop Pro are the four most important image editing programs.
- ✦ Existing images are often modified so that they fit the needs of a given Web page.
- ✦ All major graphics programs accept the addition of plug-in filters, which expand the programs' capabilities.
- ✦ 3D graphics programs can be used to quickly and easily create incredibly sophisticated scenes.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 10

Adding Color Throughout Your Site

In This Chapter

Setting color attributes

Using color names and hex numbers

Using the three C's: Complement, contrast, and coordination

Creating emotional impact

Color is your most powerful tool when you create your Web site. How you use color in your Web site determines one of two outcomes:

- Visitors will come back to your Web site and perhaps comment on what a nice Web site you have.
- Your Web site will become the object of derision by visitors and perhaps appear on one or more Web sites that are dedicated to pointing out badly designed sites.

This chapter discusses not only how to set color attributes in HTML, but also the proper use of color so you can avoid the second outcome. The latter discussion not only includes how to select background, text, and graphic colors to give your site's pages the greatest impact for visitors, but also how to select colors so visitors view the same colors on their monitor screen no matter what type of computer and browser they use.

Setting Color Attributes

When you start your Web browser for the first time, the browser has default information about how the colors on a page will appear on the screen. Your site visitors can customize the colors on their screen in Internet Explorer, Netscape Navigator, and other browsers no matter what computer or operating system you run.

The two major browsers in computing today are Microsoft Internet Explorer and Netscape Navigator. No matter what computer and operating system you use, chances are that you will be using one of these two browsers. So, this chapter discusses color issues that apply to these two browsers. If you

www.it-ebooks.info

use another browser such as Opera or Mosaic, check your browser's documentation for specific color information.

In Internet Explorer, you can access color settings by following these steps:

1. Click the Tools option in the Internet Explorer menu bar.
2. Click Internet Options in the Tools drop-down menu.
3. On the General tab of the Internet Options window, click the Colors button in the lower left-hand corner.
4. The Colors dialog box appears, as shown in Figure 10-1.

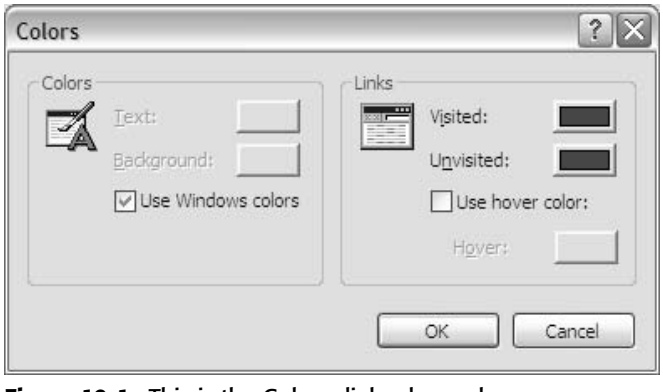


Figure 10-1: This is the Colors dialog box, where you customize colors in Internet Explorer.

When the Colors dialog box appears, you will notice that there are two separate control areas: Colors and Links. The Colors area enables you to change the text, or foreground, color and the background color. The Links area lets you change the color of links. When you click OK in the Colors dialog box, Internet Explorer reflects the changes on the page you're viewing, provided that the page does not have defined colors. If the reflected changes aren't what you're looking for, you can click the Colors button in the Internet Options windows General tab again.

Setting background colors

The *background color* is the color that appears in the viewing area of your browser window. The background color does not apply if the Web page you are viewing has a specific background color that it displays, but if there is no specified color on a Web page, then the background color you see will be the one you select in the Colors dialog box.

The default text and background color settings are the same as those in Windows. If you want to have a different background setting for Internet Explorer, click the Use Windows Colors check box to deselect it, and then click the color button to the right of the Background: label. As Figure 10-2 shows, the Color dialog box appears, and selecting colors from this dialog box is the same as selecting a color in Windows:

www.it-ebooks.info

select a color from the Basic Colors sample color display (called a swatch) or create your own custom color by clicking the Define Custom Colors button.

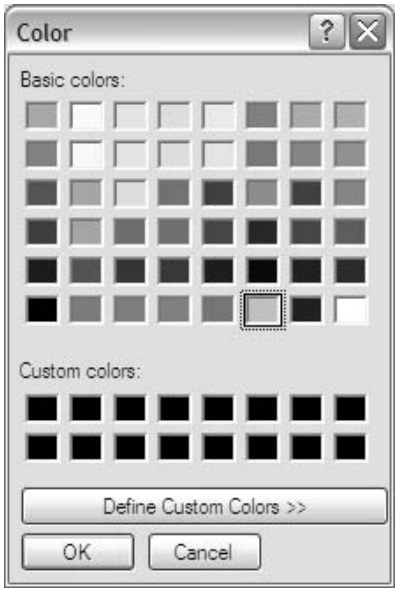


Figure 10-2: Select a basic background color in the Color dialog boxs.

Specifying foreground colors

Specifying a *foreground,* or *text,* color in Internet Explorer is as easy as selecting the background color. You can choose a text color by clicking the color to the right of the Text: label. The Color dialog box will appear, as shown in Figure 10-2, and you can select a color from the Basic Colors swatch or create your own custom color.

Changing link colors

Internet Explorer can change the color of three types of text links (visited, unvisited, and active) that appear on the page.

Links you have already visited reflect the Visited color that appears after the Visited label in the Links area. You can change the color, just as you can with the background and foreground colors.

Links you have not visited before reflect the Unvisited color that appears after the Unvisited: label in the Links area. You can change the color, just as you can with the background and foreground colors.

A link reflects the hover color when you move the mouse pointer over a link before you click it. The default setting in Internet Explorer is no hover color at all. If you want a hover color, click the Use hover color check box, and then click the color to

www.it-ebooks.info

the right of the Hover label, as shown in Figure 10-3. You can change the hover color, just as you can with the background and foreground colors.

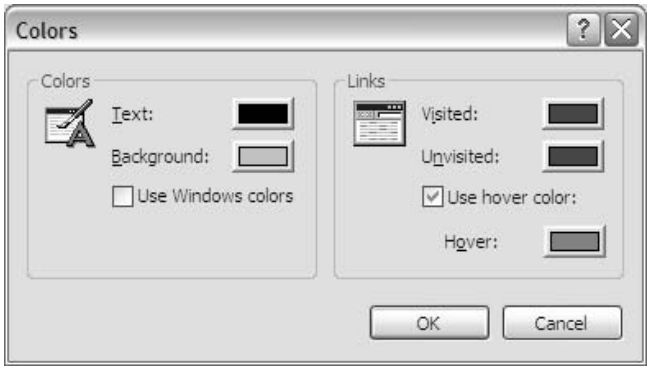


Figure 10-3: The Hover color becomes available when you click the Use hover color check box.

Using Color Names and Hex Numbers

When a monitor displays colors on the screen, it uses the three primary colors of light: red, green, and blue (RGB). The monitor takes different amounts of each color, combines them, and projects them onto an area of the screen called a pixel. If a pixel must be white, then the monitor displays 100 percent of the red, green, and blue light in that pixel. If a pixel must be black, then the monitor displays zero percent of the red, green, and blue light in that pixel.

HTML manages these RGB color combinations by using hexadecimal, or base-16, notation. Harking back to the days when you learned about mathematical bases in elementary school, hexadecimal notation (which is referred to more often as hex notation) is shorthand for the binary (or base-2) notation that computers use to communicate.

Hex notation has a number set from 0 to 15, unlike the 0 to 9 numbers in the decimal (base-10) notation we use every day. Because we can use only one digit in each number place, hex notation uses A through F to denote the numbers 10 through 15. For example, when you translate the number 60 into a hex number, the hex equivalent is 3C. Writing down 3C is much easier than writing 111100 in binary notation.

When you pronounce a hex number that looks like a decimal number, such as 43, you don't say "forty-three" but "four three." This precise pronunciation avoids confusion; it's even better if you say "four three hex." (However, saying this may require you to reassure the person you're talking to that you're not casting a spell on him or her.)

Decoding RGB triplets

HTML specifies colors by using a six-digit hex number, called a *color code.* The color code begins with a pound sign (#) and then the hex number follows. The six-digit hex

www.it-ebooks.info

number actually is a combination of three two-digit hex numbers that denote the RGB values. These RGB "triplets" enable you to set the red, green, and blue values of a color, and the color code combines these values and produces the color.

For example, the color code #FF00FF denotes the strongest red (FF), no green (00), and the strongest blue (FF). This color code produces a magenta color on your screen. Here are some additional simple color code examples:

- #FFFFFF: White (red, blue, and green)
- #000000: Black (no red, no blue, no green)
- #FF0000: Red
- #00FF00: Lime (a very bright green)
- #0000FF: Blue

If you want a less intense color, you can lower the red, green, or blue color value in the color code. For example, if you want to display a lighter blue you can use the color code #0000AA.

Facing limitations of named colors

Knowing what color code value combinations produce the desired color can be a process of trial and error. Once you do find the color, the number of sticky notes to remind you of the correct color code can cover the entire perimeter of your screen. Fortunately, HTML does have common colors named so you can enter the name of the color instead of the color code. Table 10-1 summarizes the named colors in HTML and their corresponding color codes.

Table 10-1
English Color Names in HTML

Color Name	*Hex Value*
Aqua	#00FFFF
Black	#000000
Blue	#0000FF
Fuchsia	#FF00FF
Gray	#808080
Green	#008000
Lime	#00FF00
Maroon	#800000

Continued

www.it-ebooks.info

Table 10-1 *(continued)*

Color Name	*Hex Value*
Navy	#000080
Olive	#808000
Purple	#800080
Red	#FF0000
Silver	#C0C0C0
Teal	#008080
White	#FFFFFF
Yellow	#FFFF00

So, for example, if you want to have the color blue as the color for the ubiquitous computing phrase "Hello, world!" you would enter the following HTML code:

```
<font color="blue">Hello, world!</font>
```

As you can see, the number of HTML color names is limited. If you don't want to go down the sticky-note path, you have other alternatives. One such alternative is the style sheet. Style sheets enable you to associate names with color codes so you only have to enter the color code once. You can learn more about style sheets in Chapter 13.

Using color pickers

There are 216 *Web-safe colors,* which are colors that display solid colors on any computer, monitor, and Web browser. Colors other than these 216 exhibit *dithering* — geometric patterns that appear in the color that try to give the appearance of more colors.

What happens instead is the dithered colors give the appearance of a freakish-looking quilt or tartan. What's more, a dithered color will look different in a different browser or on a different computer than the one you are using. So, Web-safe colors are the only way to ensure that your Web site has a uniform look no matter who accesses it.

One of the other alternatives to the sticky-note path, and an effective way to ensure that your site has Web-safe colors, is to choose your colors from a color picker. If you use a Web authoring program such as FrontPage or Dreamweaver, you can select from Web-safe colors directly in the program. If you choose to program directly in HTML, there are several sources on the Web for you to explore.

The Microsoft Developer Network Web site has a safety palette page at `http://msdn.microsoft.com/library/en-us/dnwebgen/html/colorpick.asp`. This page, shown in Figure 10-4, contains all the Web-safe colors displayed in two ways: from beginning to end and grouped by related colors. When you move your mouse

www.it-ebooks.info

pointer over the color you're looking for, the color code appears in a box next to the color.

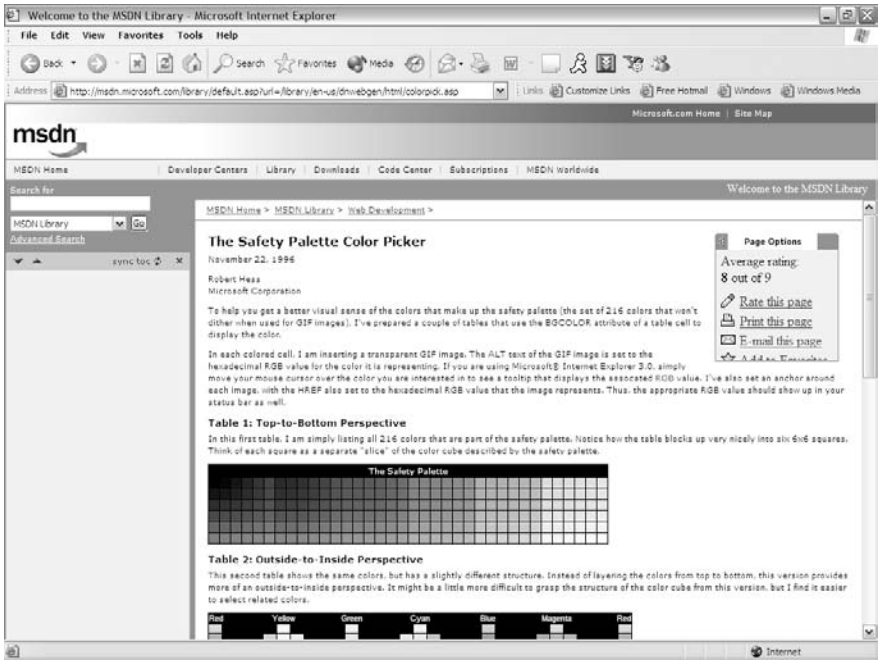


Figure 10-4: The Microsoft Developer Network Web site has a safety palette with all the colors displayed.

The Visionquest Images VQ Color Site (`www.visionquestimages.com/index.htm`) contains information about Web-safe colors and the free VQPalette program, as shown in Figure 10-5. This program displays the Web-safe colors for Internet Explorer and Netscape Navigator (although the only difference is that the color swatches display differently for each browser). You can then copy the color codes from the program and paste them directly into your HTML code. If you have Windows, you can download the VQPalette program for free, or you can use the program directly within Internet Explorer.

The VisiBone Webmaster's Color Lab at `www.visibone.com/colorlab` (see Figure 10-6) displays its 216-color Webmaster's palette graphic on its site, and if you click a color in the graphic, the site lists the color name, the HTML color code, the RGB value information, and the CMYK (cyan-magenta-yellow-black) print color value information. This site has several unique features, including

- ✦ Web-safe swatch collections that you can download for free and use with different Web-related graphics programs such as Adobe Photoshop and Macromedia Fireworks
- ✦ The capability to access the site in one of 16 languages
- ✦ The Webmaster's Palette available for purchase as a poster for easy reference (or as wall art if you prefer), not to mention some pretty nice mouse pads

www.it-ebooks.info

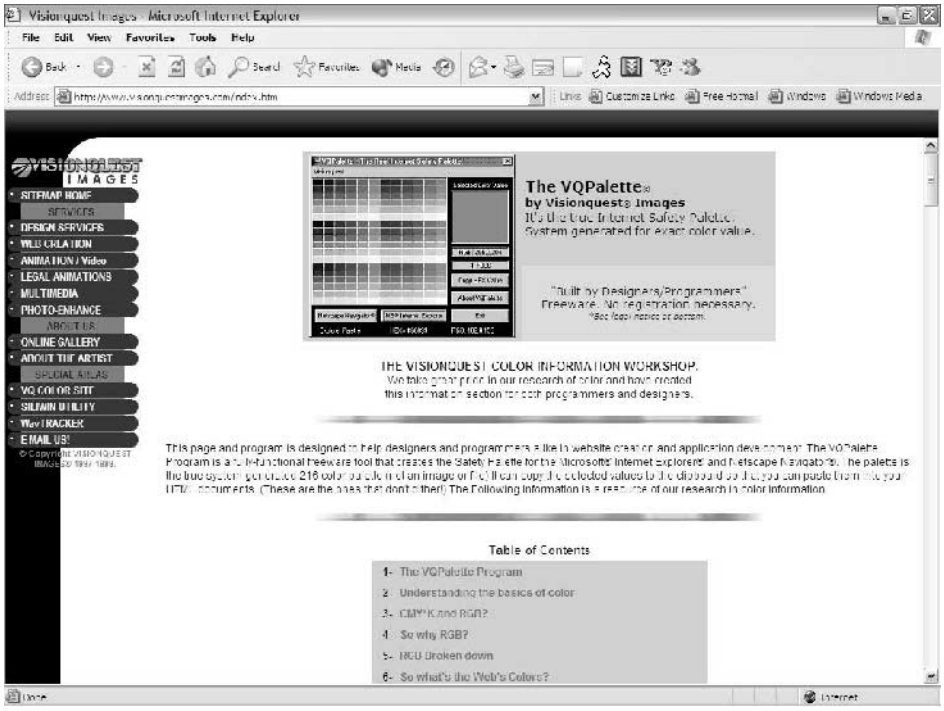


Figure 10-5: The Visionquest Images Web site contains a lot of color information and the free VQPalette program.

Table 10-2 provides the URLs of some other Web sites where you can also find color pickers.

Do "Web-safe colors" matter anymore?

The whole issue of Web-safe colors is based on the 256-color video display. Such video cards are more than a little bit outdated today, and new computers are typically capable of displaying millions of colors instead. The only computers still displaying such a limited set of colors are rubbing elbows with the dinosaurs.

However, even though such systems are no longer sold (at least on the mass market), a lot of them are still in operation. Not everyone upgrades to the latest technology all the time, and companies, in particular, are still using older systems that are incapable of competing with the new computers.

It's the bugaboo of all Web designers—what audience are you creating your site for? If you want to cover all the bases, you have to use Web-safe colors so everyone can visit your site. If you don't, you run the risk of losing visitors and, if you're promoting a business, potential customers.

www.it-ebooks.info

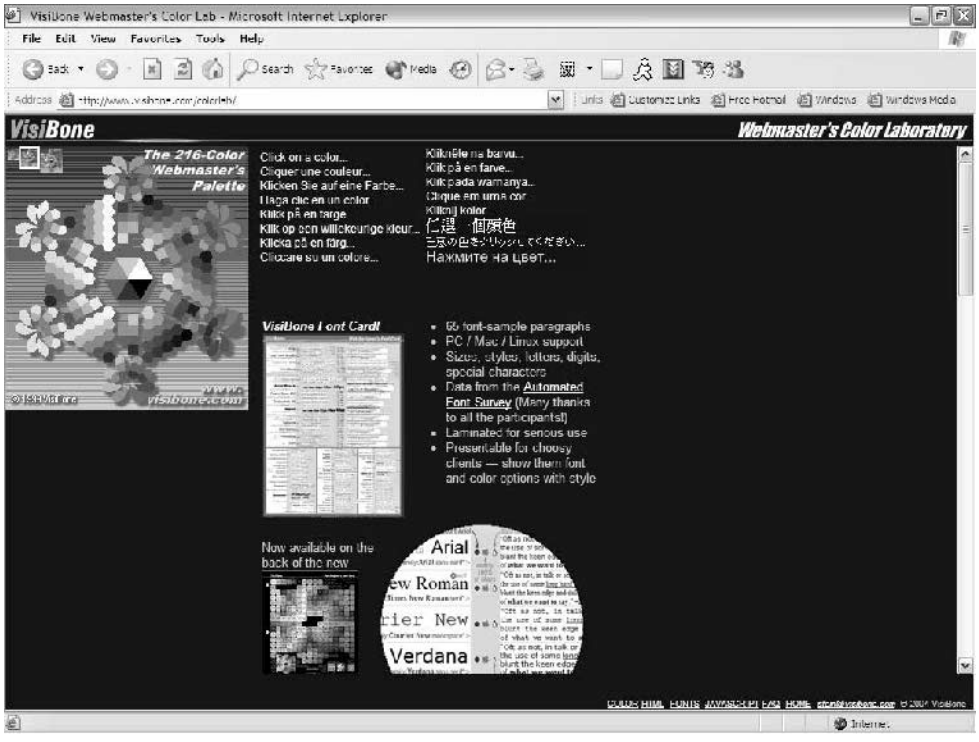


Figure 10-6: The VisiBone Webmaster's Color Lab.

Table 10-2
Color Pickers

Web Site	*URL*
ColorMaker	www.bagism.com/colormaker
Dougie's Color Picker	hypersolutions.org/pages/colorSel.html
TomaWeb Hex Color Picker	colors.tomaweb.com
HTML Color Chooser	geocities.com/colorchooser
HTMLcolor	bluefive.pair.com/htmlcolor.htm
Kira's Web Toolbox	lightsphere.com/colors
RGB/Hexadecimal Color Converter	www.eons.com/rgb2hex.htm

www.it-ebooks.info

Using the Three C's: Complement, Contrast, and Coordination

One Web site that has garnered media attention is called Web Pages That Suck (`www.webpagesthatsuck.com`). This site, which every serious Web designer should pay at least one visit to, displays and discusses poorly designed Web sites, and one issue that this "hall of shame" site always focuses on is the use of color.

Color is what visitors first notice about your site. If you don't use color effectively, then at best it can be a hindrance to visitors and, at worst, the site will be so uninviting that no one will want to come back except the Web Pages That Suck people.

How do you use color effectively? Effective color usage begins with the three C's of color management:

- ✦ Complementary colors are pleasing to look at.
- ✦ Contrasting colors make your elements stand out, not wash out.
- ✦ Coordinating colors throughout all the parts of your pages and site makes your work look like part of a whole, not several different elements or pages jumbled together.

Planning for color should be one of the first things you do when planning your Web site. By keeping the three C's in mind, you will be able to create a site that's inviting, consistent, and memorable.

Choosing complementary colors

The World Wide Web is a semi-static medium. The Web is semi-static because you can add live motion and other dynamic elements to your Web pages. However, most elements on a Web page are stationary much like a page you read in a magazine or view on a billboard.



See Chapter 18 for more information about adding dynamic page elements.

If you've read any magazines or seen any billboards on the highway recently, the ones that probably caught your eye (though hopefully not for too long in the billboard case) were the ones that had good complementary color. That is, the colors look like they go well together so the ad is pleasing to look at.

The same concept of complementary colors also applies to Web pages. If you have colors for your background, graphics, and text that go well together, your page will not only be more readable but more pleasing to look at. The more pleasing your Web page is to look at, the more visitors (and more returning visitors) your page will receive.

www.it-ebooks.info

But what are complementary colors, exactly? To illustrate the principle, take a look at Figure 10-7. Even though this book is in black and white, you can view the figure in full color on the accompanying Web site (www.wiley.com/compbooks/crowder).

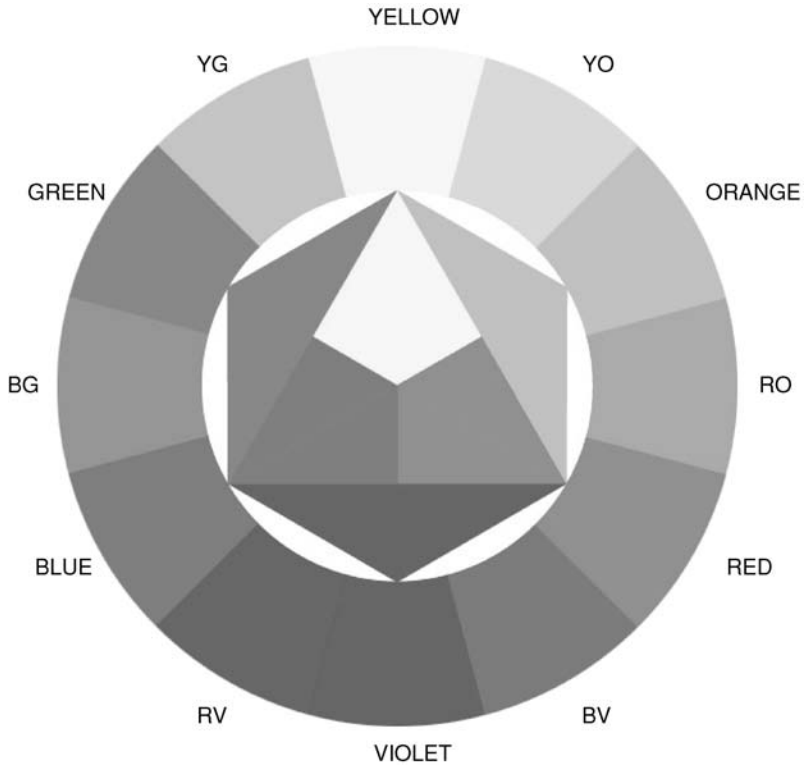


Figure 10-7: The color wheel is a basic artist's tool.

In the strictest sense, *complementary colors* are two colors that are opposite each other on the color wheel, a tool long used by traditional artists.

Analogous colors are ones that are right beside one another on the color wheel. Blue and blue-green, for instance, are analogous to one another. In addition to the strictest definition of complementary colors, you can also consider any colors that are analogous to a complementary color as complementary. For example, orange is complementary to blue. The analogous colors for blue are blue-green and blue-violet, so both of those colors would also work well as complements to orange.

This color model provides basic information about how to use color on your pages, but it does not guarantee that different colors, or the colors in your background and text, will appear correctly. For that to happen, you have to have the right color contrast.

www.it-ebooks.info

Ensuring contrast and visibility

Complementary colors are also called contrast colors because they let the lighter colors stand out more against the darker colors. The most familiar contrasting colors are black and white. Books are printed with black ink on white (or light-colored) pages because it's easy to read. The same is true for the Web; it's much easier to see black text on a white background or, for example, a light gray wallpaper that you use as a background graphic.

If you try to read white text on a black background, you can't read the text as well. Black tends to mask the outer boundaries of the white text, and if you see any sites with black backgrounds, the text is usually in bold face to make the text easier to read. Some sites, such as the TrekWeb site (`www.trekweb.com`), use blue fields as the background to make the white text stand out even better.

If you have a triad or split complementary color scheme, the same contrast rules apply. For example, a triad color combination that has a background of red and green with blue text makes the text hard to read. If you change the red to yellow, then the blue text stands out from the green and yellow background so you can read the text more easily.

Coordinating color schemes

Coordination of your Web site's color schemes goes hand-in-hand with planning your Web site. (Refer to Chapters 1 and 4 if you need to brush up on planning your Web site.) Now that you have a firm grounding of how to complement and contrast colors on your site, it's time to coordinate how those colors will appear on your screen.

The audience that you are developing the Web site for must determine your color schemes. For example, if your site is designed for seniors, then it may be a better idea to have a high contrast color scheme (such as black text on a white background). If your site is designed for children, then having less contrast and more colors may be the best approach.

Understanding Color Warmth

If you want your site to be memorable, you must use color to create an emotional impact for your audience. Your visitors will come back to your site and encourage others to visit your site as well. You can manipulate colors by changing their values, thus producing strong, flamboyant colors or softer colors.

The color wheel shown in Figure 10-7 contains so-called warm and cool colors. *Warm colors* consist of red hues, from yellow to purple. Warm colors exude energy, so they stand out from the cool colors. These warm colors are better used as foreground colors.

"Cool" colors, on the other hand, consist of blue hues, from green to dark blue. *Cool colors* exude strength and calm, so these colors are best used for the background so the warm colors can stand out and do their work.

www.it-ebooks.info

Although contrasting warm colors with cool colors is a good general rule, some colors contrast better than others. For example, if you use light blue and orange together, they won't contrast well because both colors are light. If you change the blue to a dark blue, or if you change the orange to black or brown, the colors will contrast much better and visitors will have a much easier time viewing your page.

If you want more information about color theory, check out some of the Web sites in Table 10-3.

Table 10-3
Color Information

Web Site	*URL*
Color Codes	`hotwired.lycos.com/webmonkey/reference/color_codes`
Color Theory	`www.bway.net/~jscruggs/notice2.html`
Poynton's Color FAQ	`www.icaen.uiowa.edu/~aip/Misc/ColorFAQ.html`
The 16,777,216 Colors of the Web	`the-light.com/16m/16m.html`
Web Color Reference	`webreference.com/html/reference/color`
Wheel of Color	`http://msdn.microsoft.com/workshop/design/color/hess08142000.asp`

Summary

Color is critical to your site's success. The proper use of color can make or break it. This chapter introduced you to the following concepts and topics:

- ✦ Despite what you do, your visitors can specify their own colors in their Web browsers.
- ✦ Web-safe colors theoretically display the same on any computer system.
- ✦ The color wheel, a basic artist's tool for choosing complementary colors, is important to Web designers.
- ✦ Colors carry emotional messages—warmth, coolness, harshness, softness, and so forth.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

P A R T

III

Designing Web Layouts



In This Part

Chapter 11
Harnessing the Power of Tables

Chapter 12
Organizing Your Site with Frames

Chapter 13
Styling Web Pages with Cascading Style Sheets

Chapter 14
Positioning Elements with DIVs



www.it-ebooks.info

www.it-ebooks.info

CHAPTER 11

Harnessing the Power of Tables

In This Chapter

Adding tables and setting table size

Managing borders

Using padding and spacing

Aligning tables and cell contents

Spanning rows and columns

Working with images and color

Although tables on Web pages can be incredibly dull, they also can be the single greatest tool at your disposal. It all depends on how you use them. If a table does nothing but list row after row of numbers or words, then it's pretty hard to make it an exciting addition to your site. Of course, setting up neat and well-structured tabular data is a useful and often necessary way of presenting information. We're not exactly against it—you notice that we use tables of tabular data in the examples in this book from time to time when that's the best way to get something across.

But that's the pedestrian side of tables, and, even though it's all you can do with tables on paper, Web pages enable you to use tables in astonishing ways. Let's start with the basics: tables are groups of cells that contain information. The only difference between dull, plebian tables and powerful, delightful ones is what you choose to put in the cells of the table. Each cell in a table can hold nearly anything that can be put on a Web page — frames are just about the only exception, and that's because table cells can only contain things that also go into the `BODY` element. There's practically no limit to what you can do with a little imagination.



See Chapter 3 for a discussion of the `BODY` element and Chapter 12 for information on frames.

Adding Tables and Setting Table Size

Savvy Web designers have used tables to structure their pages for years. Some of the most exciting and unusual designs — the ones that make you wonder, "How did they do that?" — are simply tables with the cells filled with different elements such as images or links. The Web page in Figure 11-1, for example, is nothing but a table filled with various elements.

www.it-ebooks.info

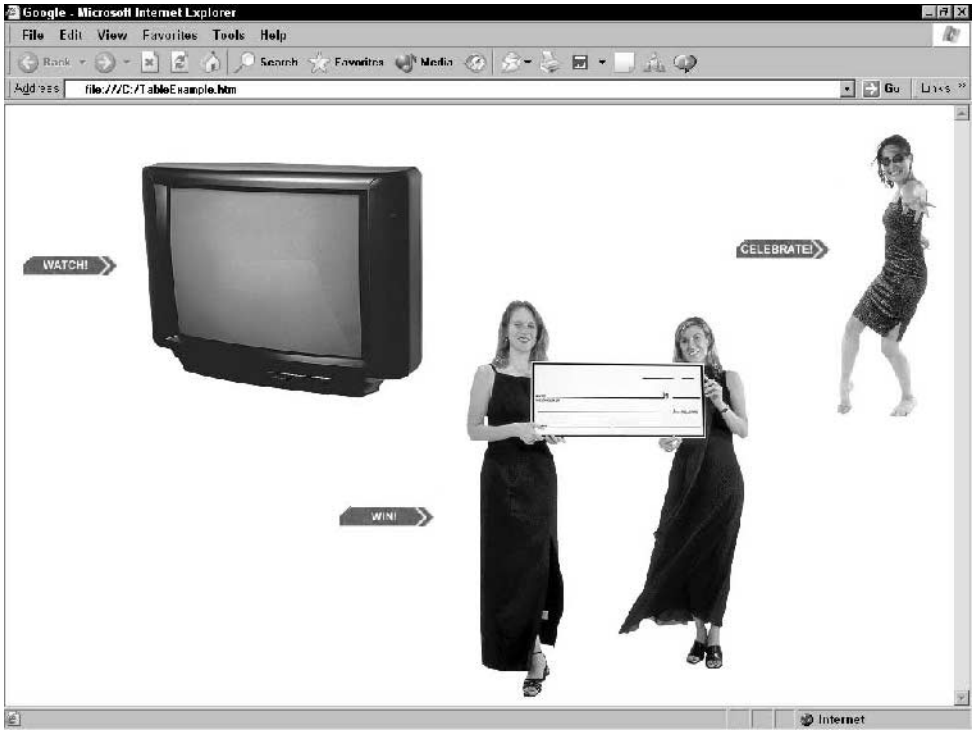


Figure 11-1: Tables can also be used to structure Web pages and hold nontextual content.

Tables, predictably enough, are added using the `TABLE` element. The `TABLE` element contains one or more `TR` (table row) elements. The `TR` elements, in turn, contain one or more `TD` (table data) elements, which are usually referred to as *cells*. The HTML code for a typical table might look as follows:

```
<TABLE border="1">
<TR>
<TD>First Cell</TD>
<TD>Second Cell</TD>
</TR>
<TR>
<TD>Third Cell</TD>
<TD>Fourth Cell</TD>
</TR>
<TR>
<TD>Fifth Cell</TD>
<TD>Sixth Cell</TD>
</TR>
</TABLE>
```

This table contains three rows, with each row having two cells. The results are shown in Figure 11-2.

www.it-ebooks.info

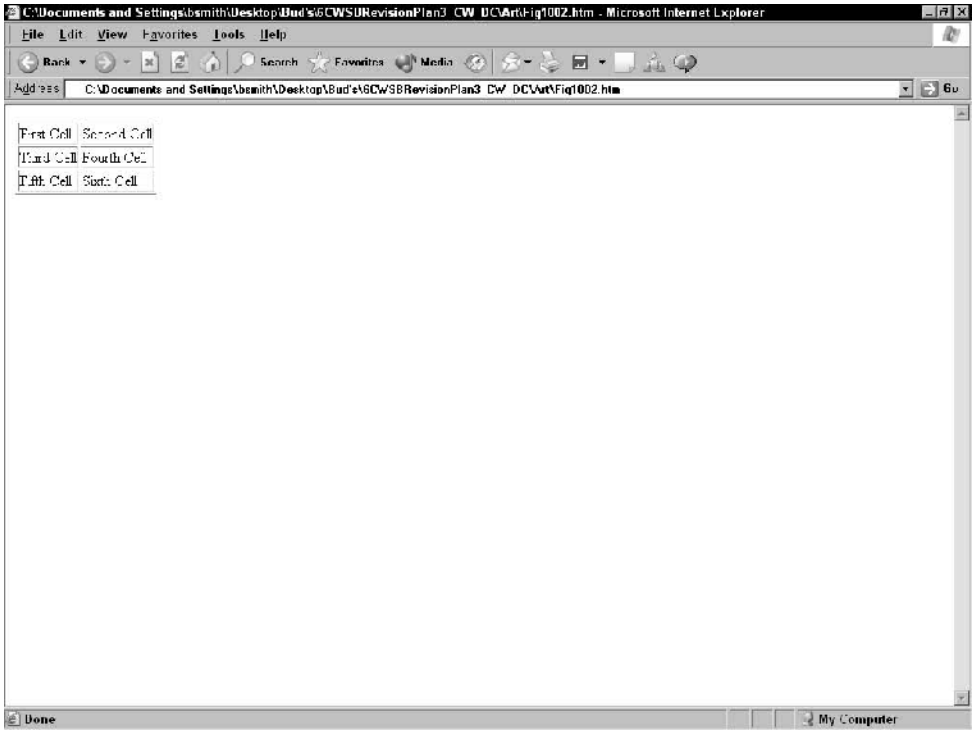


Figure 11-2: This is a basic table.

The width and height of a table, if they are not specifically stated in your HTML code, are determined by the size of the table's contents. The table automatically expands to accommodate whatever is put into its cells – even if the result means that the table is larger than the overall table size you specify in your HTML code.

To specify a particular size for a table, you use the `width` and `height` attributes. For example, to make the table take up the entire width and height of the screen as shown in Figure 11-3, you would use the following code:

```
<TABLE width="100%" height="100%">
```



Of the two, `width` is far more often specified than `height` because Web pages effectively have infinite height, and people are used to scrolling to view whatever is below the current screen.

Specifying size as a percentage works no matter what screen resolution a visitor is using. One hundred percent of the screen is the whole thing, period. Fifty percent of the screen is half of it, and so forth. You can also size the table with a set number of pixels. To create a table that is 400 pixels across and 250 pixels high, for example, you would state this:

```
<TABLE width="400" height="250">
```

www.it-ebooks.info

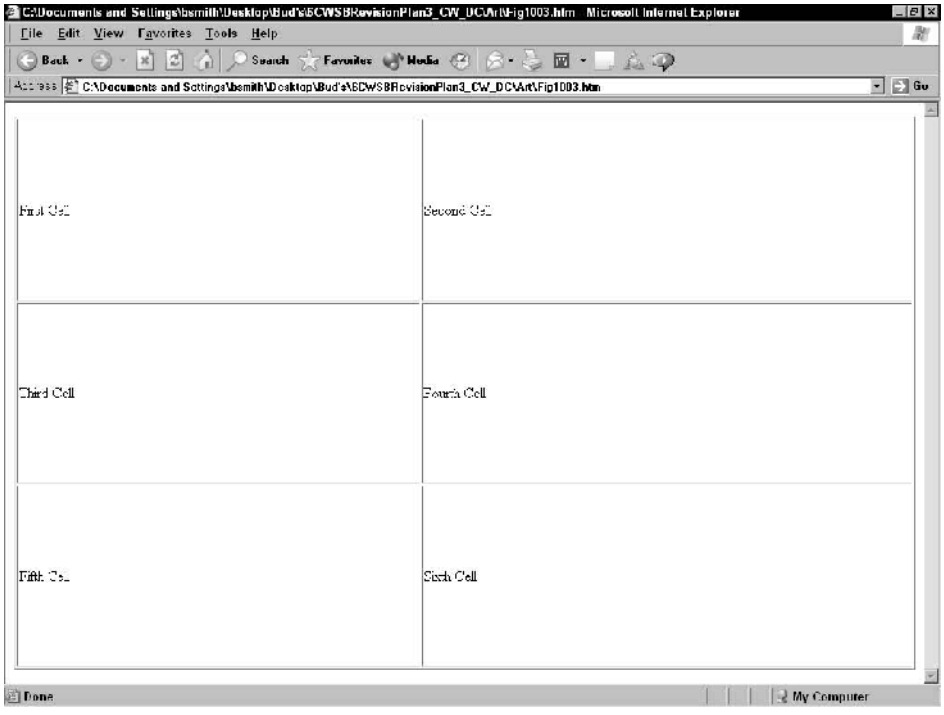


Figure 11-3: A table can take up the entire screen.

The drawback to specifying a particular width is that you may force some of your site visitors to scroll horizontally to see the entire table. For instance, if you fix your table at a width of 800 pixels and someone using a screen resolution of 640 × 480 viewed it, the right side of the table would be offscreen. It's usually better to use the percentage approach and leave the details of the display to the Web browser.

Both techniques can be combined, as well. To do so, simply use a percentage value with one attribute and a pixel value with the other. The following code, for example, fixes the width at 400 pixels but still takes up three-quarters of the screen height regardless of the screen resolution:

```
<TABLE width="400" height="75%">
```

You can also set the `width` and `height` of individual cells, although you need to be careful of the impact of this on the rest of the table. Setting the height of one cell also makes every other cell in the same row change to that same height.



If you specify two sizes that conflict with one another, the larger one wins out. For instance, if you set a cell to be 50 pixels high and set another cell in the same row to be 80 pixels high, all cells in that row will be 80 pixels high. If a third cell's contents are 100 pixels high, then all cells in the row become 100 pixels high.

www.it-ebooks.info

The `width` and `height` attributes work the same way for cells as they do for tables, with one important difference. When you set a percentage value for a cell, that's the percentage of the *table*, not the screen. Say, for example, that you have a table that's set to be 50 percent of the screen width, and you set a cell to take up 50 percent of the table width—that cell is only 25 percent of the overall screen width.

The following HTML code shows how to set the table from the preceding example so that its first column takes up one-third of the table, while its second row is fixed at 100 pixels in height.

```
<TABLE border="1">
<TR>
<TD width="33%">First Cell</TD>
<TD>Second Cell</TD>
</TR>
<TR>
<TD height="100">Third Cell</TD>
<TD>Fourth Cell</TD>
</TR>
<TR>
<TD>Fifth Cell</TD>
<TD>Sixth Cell</TD>
</TR>
</TABLE>
```

The results are shown in Figure 11-4.

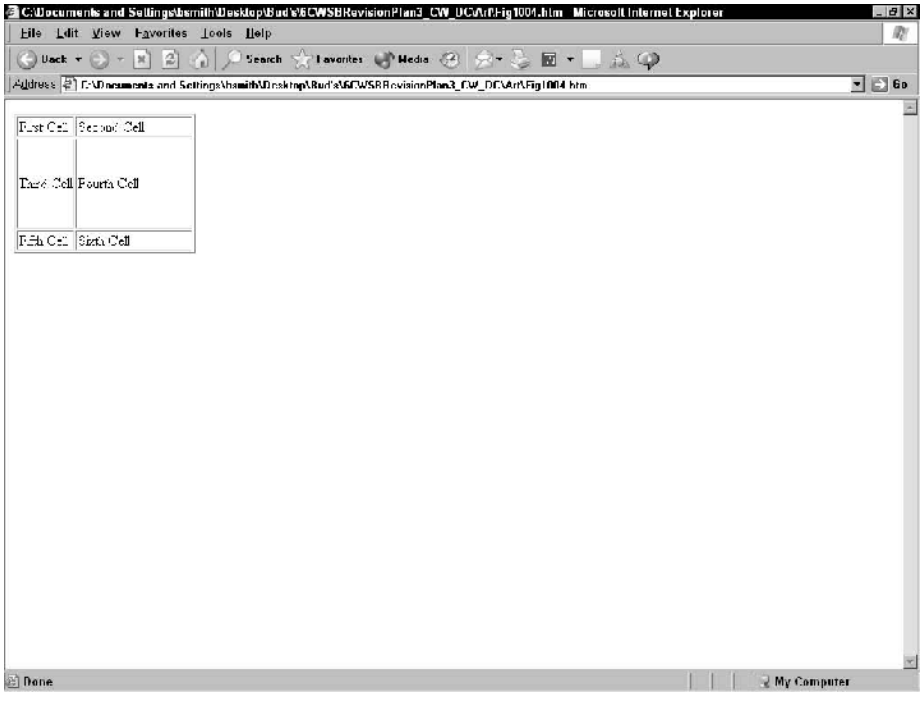


Figure 11-4: You can set rows and columns to different values.

www.it-ebooks.info



If you don't add any content to your table, the table won't show up in a Web browser.

If you start specifying sizes for different parts of tables, some bizarre situations may result. Setting a specific pixel size for individual cells tends to work best if you don't also set the table size that way. The following code, for instance, doesn't work out because things don't add up properly:

```
<TABLE width="600" border="1">
<TR>
<TD width="200">First Cell</TD>
<TD width="100">Second Cell</TD>
</TR>
<TR>
<TD>Third Cell</TD>
<TD>Fourth Cell</TD>
</TR>
<TR>
<TD>Fifth Cell</TD>
<TD>Sixth Cell</TD>
</TR>
</TABLE>
```

Fortunately, if you do end up with a situation such as this, Web browsers are smart enough to fix it so that the results are still viewable. The outcome of the preceding code sample, for instance, is that the first column ends up being 400 pixels wide and the second one ends up at 200 pixels. The browser takes a look at the situation, realizes that things just won't add up, and decides to go with the table width of 600 pixels, maintaining the proportions (2 to 1, in this case) that you wanted for the cells.

Because the results are not what you intended, it's important to ensure that you don't create a size conflict problem.

It's easy enough to come up with percentage values that don't add up to 100 percent, too. Web browsers deal with that problem in the same way as they do with absolute values—by assigning the remaining percentage proportionally among the cells. In the case of values that exceed 100 percent, the excess is subtracted proportionally from each cell.

This behavior can actually be used deliberately to your benefit. Many Web designers, faced with the problem of setting up three cells of identical size in the same row, assign percentage values of 33 percent, 34 percent, and 33 percent, sacrificing a tiny bit of precision in order to make all the percentages add up to 100 percent. However, if you make them all 33 percent, the Web browser automatically mixes in the missing 1 percent to make them all perfectly even.

www.it-ebooks.info

Managing Borders

Depending on how you're using tables, you may or may not want them to have a border. When tables are used to provide the underlying structure for a Web page, most designers prefer to leave borders out of the picture. But when you do need a border, it's easy to provide one.

To specify the thickness of a border, you set a value in pixels for the `border` attribute. The following example would create a thin border, one pixel wide:

```
<TABLE border="1">
```

Generally speaking, most people don't use very thick borders, preferring to stay in the range of from 1 to a maximum of 5 pixels. Any border size above 1 pixel makes the right and bottom sides of the border darker than the left and top sides, giving the border a shaded effect, as if it were lighted from the top-left corner. Figure 11-5 shows tables with various border sizes.

Note that the border size has no effect on the internal borders around each cell. There is only one exception to this — if you make a table with no border by setting the value of the `border` attribute to 0, this also removes the internal borders. The effect is to leave all the table's structure intact, keeping the table itself from intruding.

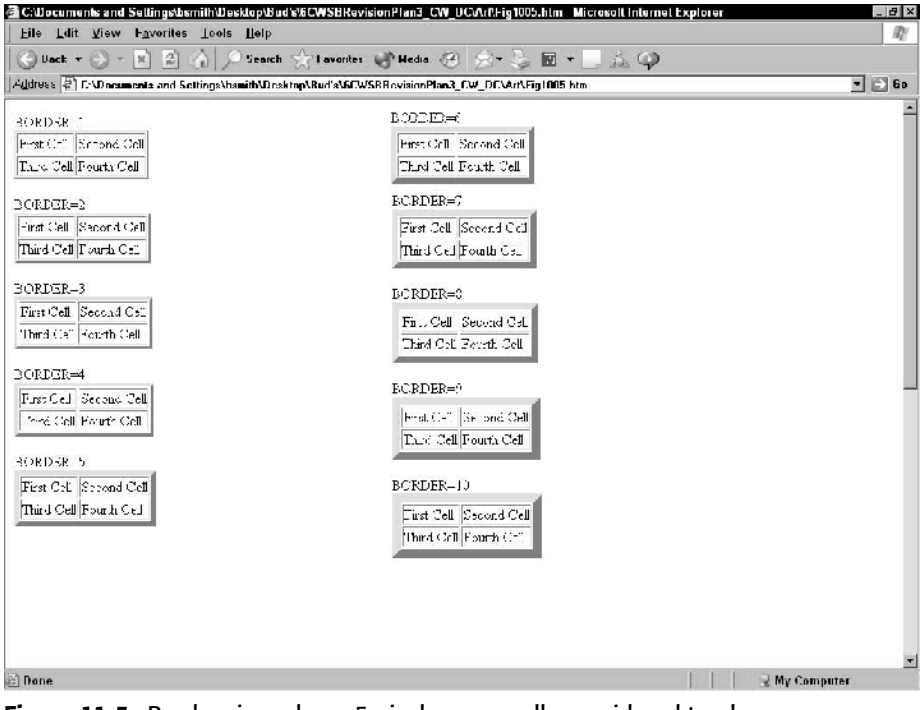


Figure 11-5: Border sizes above 5 pixels are usually considered too large.

www.it-ebooks.info

There are various other options for altering borders, including internal borders (which are technically known as *rules*). However, these are either proprietary to a specific type of browser or not widely supported, so they are of little value.



Even when you plan to use a borderless table, it's best to keep the borders intact during the design phase. This helps you to visualize the exact location of each cell and their relationships to one another. Setting the border to zero is one of the last steps before uploading the Web page to your site.

Using Padding and Spacing

Depending on the contents of a particular cell, you need to create some space between the border and the content, or avoid having any space at all. The space between the border and the content is called *cell padding*. Figure 11-6 shows two cells that have no padding at all. The first one contains an image, which fits perfectly within the cell. The second one contains text, which is rendered less readable by being crammed up against the cell's border.

Figure 11-7 shows the opposite situation, in which the cells have 2 pixels of padding that separate their contents from the border. The image now has an undesirable

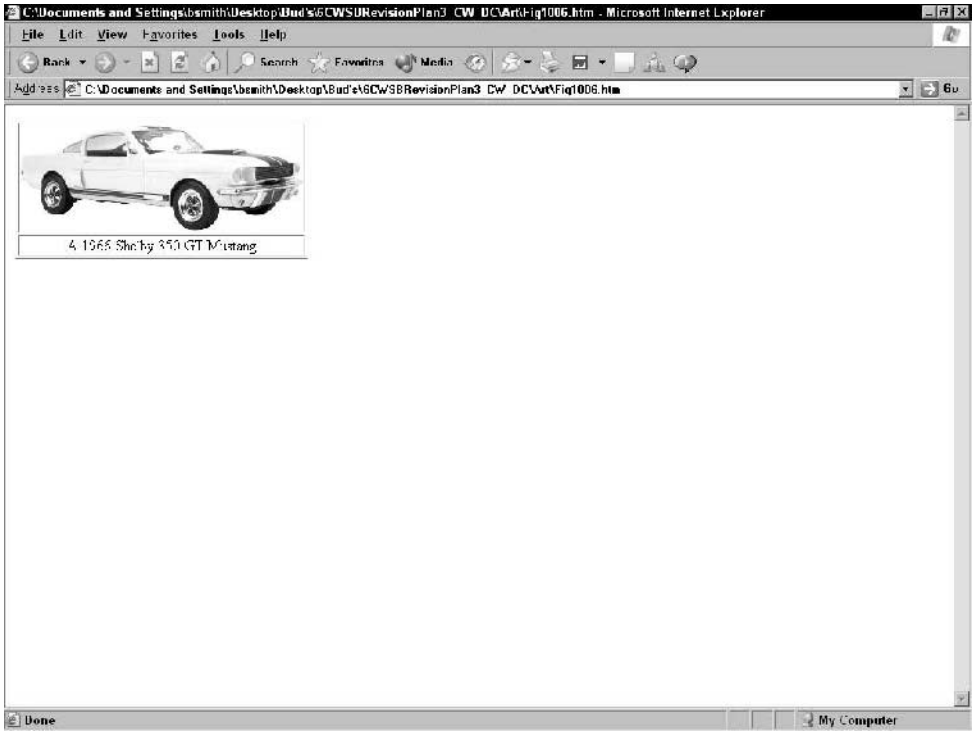


Figure 11-6: Images look best with no cell padding.

www.it-ebooks.info

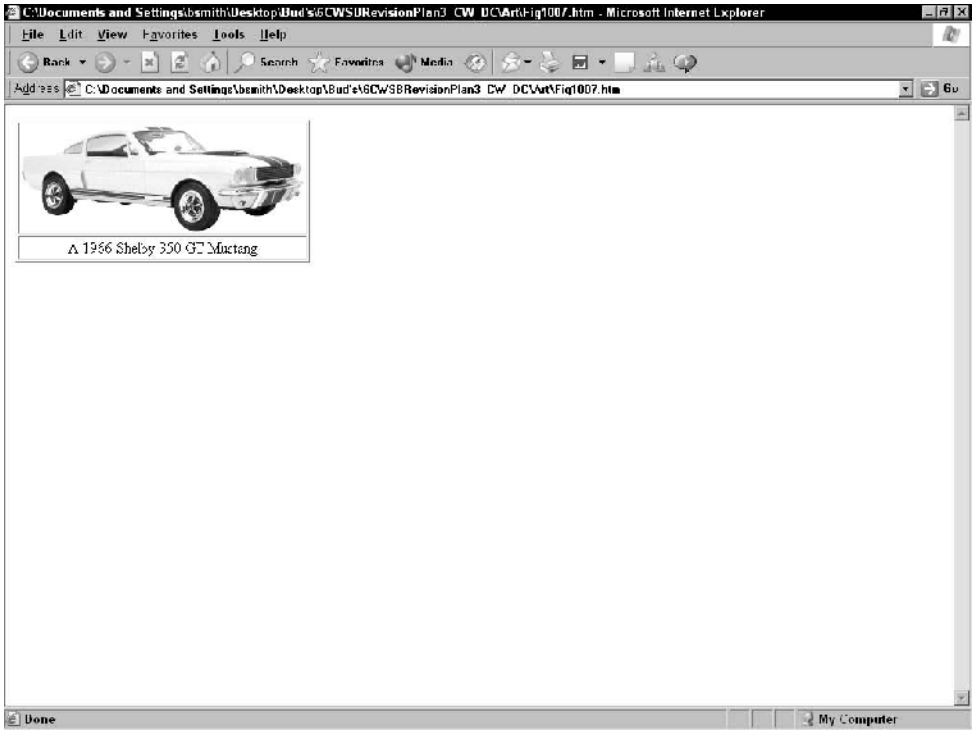


Figure 11-7: Text is more readable with some space between it and the cell walls.

space around it, but the text has become more readable by virtue of the same padding.



See "Working with Images and Color" later in this chapter for details on adding images to tables.

Cell padding is set via the `cellpadding` attribute of the `TABLE` element. You can only set padding for the table as a whole, not for individual cells. To get the 2 pixel cell padding shown in the preceding figure, you would use the following code:

```
<TABLE cellpadding="2">
```

Cell spacing is different — instead of setting the amount of space inside the cell's border, it sets the amount of space between the cells themselves. Figure 11-8 shows several tables with various cell spacing values.

Cell spacing is set, as you might have guessed, with the `cellspacing` attribute of the `TABLE` element. The following code places each cell 5 pixels away from its neighboring cells:

```
<TABLE cellspacing="5">
```

www.it-ebooks.info

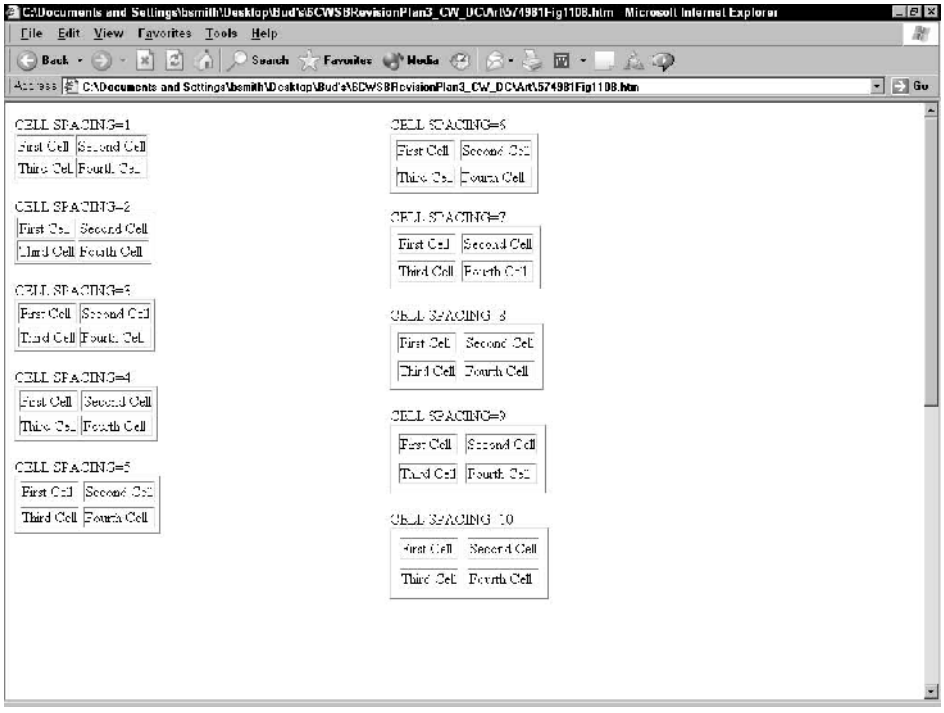


Figure 11-8: Cell spacing determines how far apart each cell is from its neighbors.

The question of exactly how much, if any, cell padding or cell spacing to use is an aesthetic choice. A table with neither value set still works perfectly from a technical point of view.

Aligning Tables and Cell Contents

The alignment of a table is a totally different thing from the alignment of rows and cells. Table alignment refers to where the table falls on the Web page — the possible values for the `TABLE` element's `align` attribute are `left`, `center`, and `right`. Left and right alignment put the edge of the table on the appropriate margin of the Web page, whereas center alignment puts it smack in the middle of the screen, with both edges of the table the same distance away from the margins. Figure 11-9 shows a right-aligned table.

Setting horizontal alignment

The `align` attribute of the `TR` and `TD` elements, on the other hand, aligns the *contents* of the affected cells, not the cells themselves. Despite the fact that the exact same terminology is used for the attribute, and the same three values are possible—`left`, `center`, and `right`—there is no connection whatsoever between the way a table is

www.it-ebooks.info

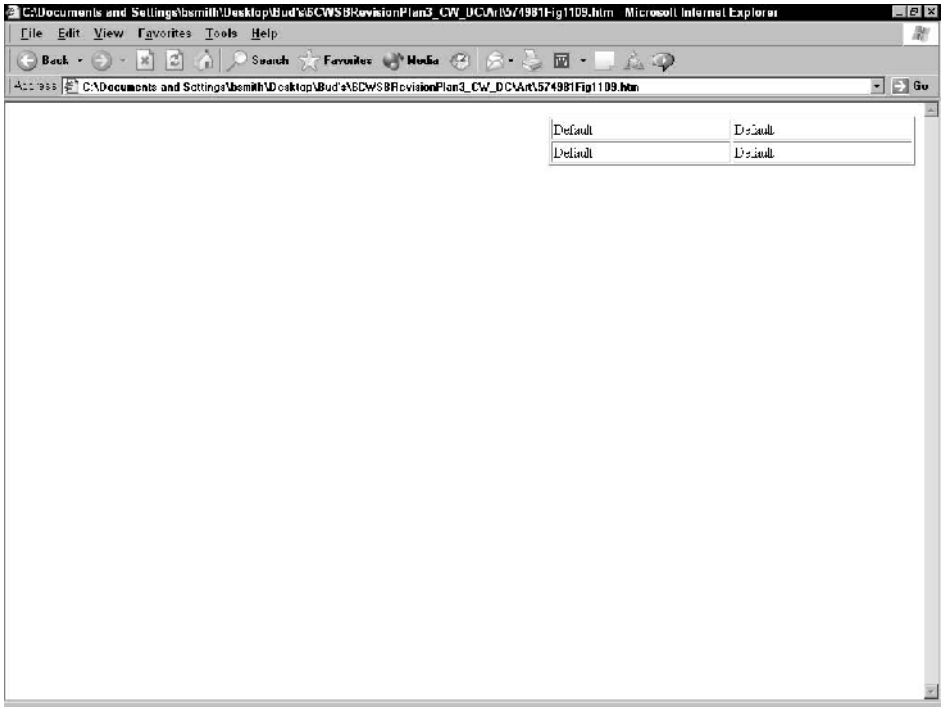
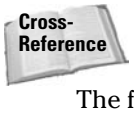


Figure 11-9: A right-aligned table.

aligned and the way rows or cells are aligned. Rows and cells, however, work exactly the same. It's only a question of how many cells are affected by the setting.

This is another one of those cases where the HTML container relationships come into obvious play. By default, the settings for the row become the settings for all the cells that the row contains. Thus, if a `TR` (table row) element is set to center alignment, then all the `TD` (table data, or cell) elements in that row automatically have that same alignment. In order to change the alignment of a specific cell, you need to override the inherited attribute values.



See Chapter 3 for more information on container relationships.

The following HTML example shows a table that is center aligned. It contains two rows, the first of which has no specified alignment. This means that all the cells in that row automatically have the default alignment, which is that their content is left-aligned. The second row is set to have center alignment, but one of the cells in it is set to have right alignment.

Figure 11-10 shows the results.

```
<TABLE width="400" border="1">
<TR>
<TD>Default</TD>
```

www.it-ebooks.info

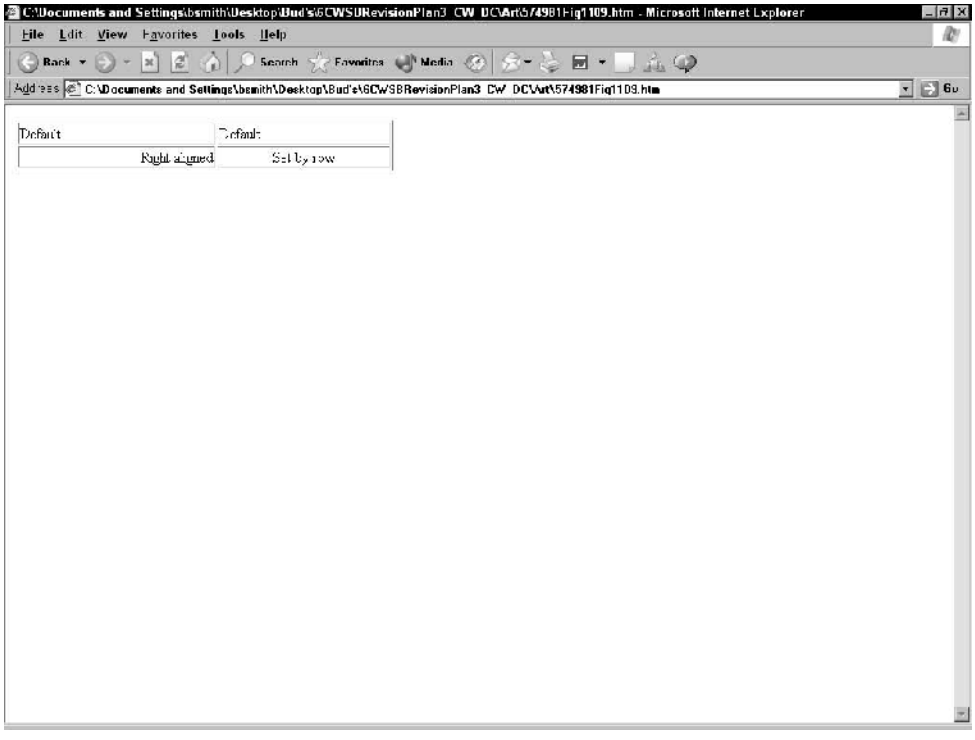


Figure 11-10: Cell alignment overrides row alignment.

```
<TD>Default</TD>
</TR>
<TR align="center">
<TD align="right">Right aligned</TD>
<TD>Set by row</TD>
</TR>
</TABLE>
```

Using tables with forms

Good Web designers know that forms, which have a deplorable visual appearance in plain HTML, can be meshed with tables to create a carefully arranged appearance. This not only gives the form more visual appeal, but it also increases the ease of understanding for anyone faced with filling out the form. Figure 11-11 shows a form done in plain HTML, which was created using the following HTML code:

```
<FORM action="cgi-bin/checkident.pl" method="post">
<P>User Name: <INPUT type="text" name="username">
<BR>
Password: <INPUT type="password" name="pw">
</P>
<INPUT type="submit">
</FORM>
```

www.it-ebooks.info

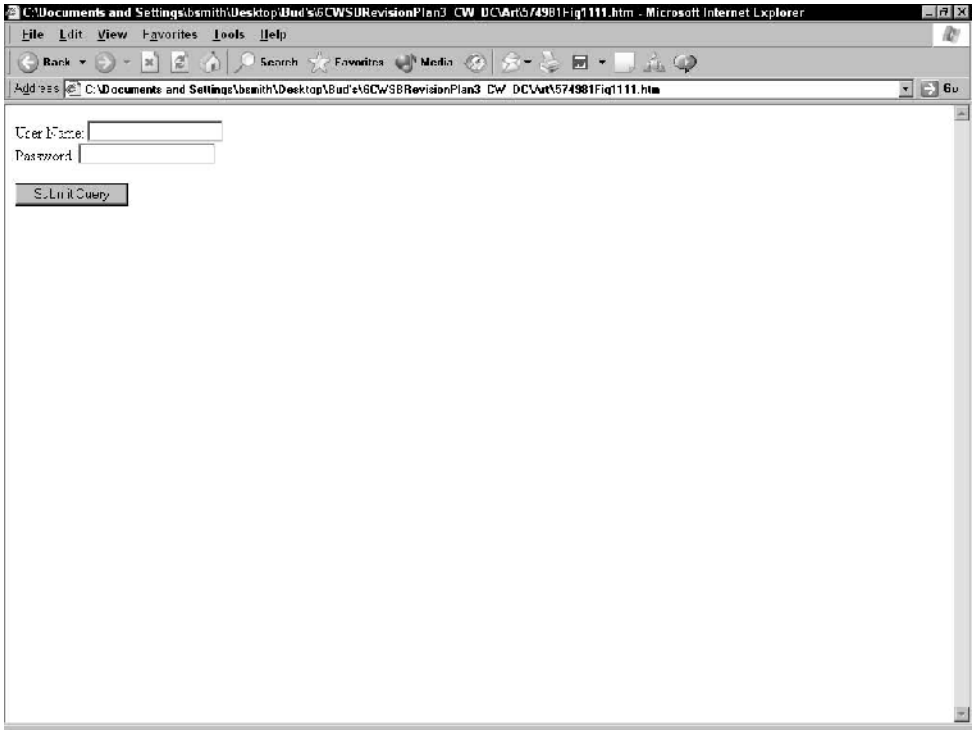
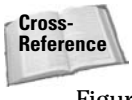


Figure 11-11: Forms in HTML are naturally sloppy.



See Chapter 15 for a discussion on forms.

Figure 11-12, on the other hand, shows how much better the same form looks when a table is added to the mix and the form controls are carefully aligned.

The following example shows how a table was added to the form so that the form controls would be meshed into the table for restructuring:

```
<FORM action="cgi-bin/checkident.pl" method="post">
<TABLE>
<TR>
<TD align="right">User Name:
</TD>
<TD align="left">
<INPUT type="text" name="username">
</TD>
</TR>
<TR>
<TD align="right">Password:
</TD>
<TD align="left">
```

www.it-ebooks.info

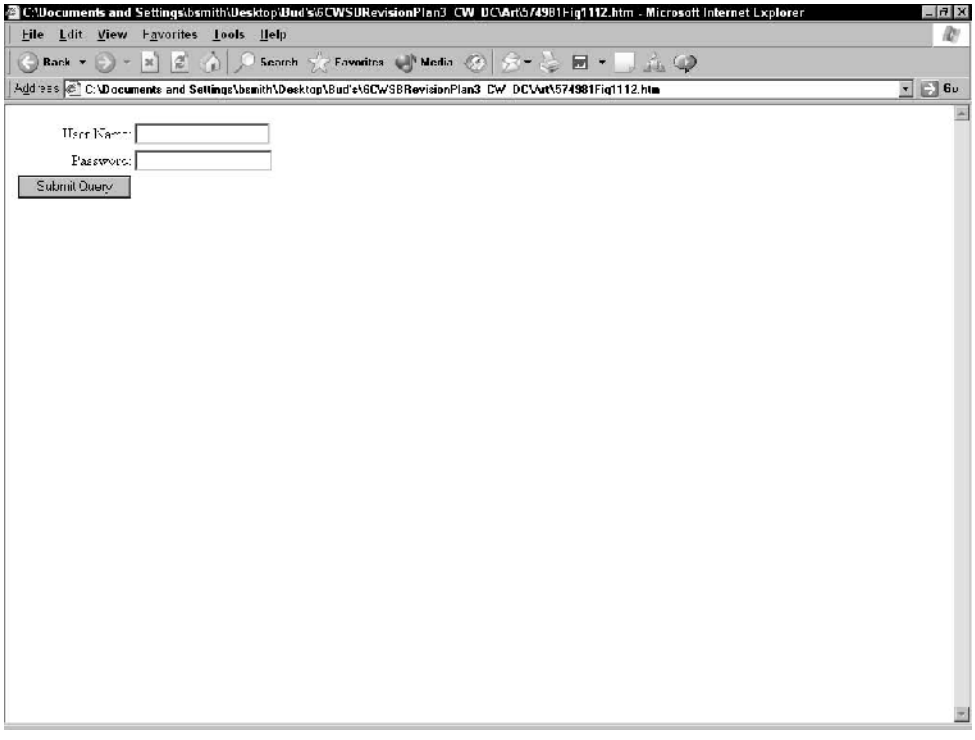


Figure 11-12: Tables can be used to make forms presentable.

```
<INPUT type="password" name="pw">
</TD>
</TR>
<TR>
<TD align="left">
<INPUT type="submit">
</TD>
</TR>
</TABLE>
</FORM>
```

To create this situation, the `TABLE` element is actually placed within the `FORM` element, rather than the other way around, because the only place you can put anything in a table is within an individual cell. Obviously, the entire form could not be put into such a small space, so the table necessarily goes into the form, not the other way around. This way, the structural capacities of the table are gained and the functional capabilities of the form are maintained.

By setting things up so that the table has two columns with the first column, which contains the descriptive text, right-aligned, and the second column, which contains the form controls, left-aligned, the form suddenly gains a nice, neat structure that is easy for the user to follow.

www.it-ebooks.info

Using tables with sidebar images

Tables are absolutely essential when you use sidebar images on your Web pages. These backgrounds fill up the left edge of the screen with artistic or cute material that interferes with the display of text and other elements, but leaves the rest of the screen in a proper condition. Figure 11-13 shows a typical sidebar background and its impact on a normal Web page's contents.

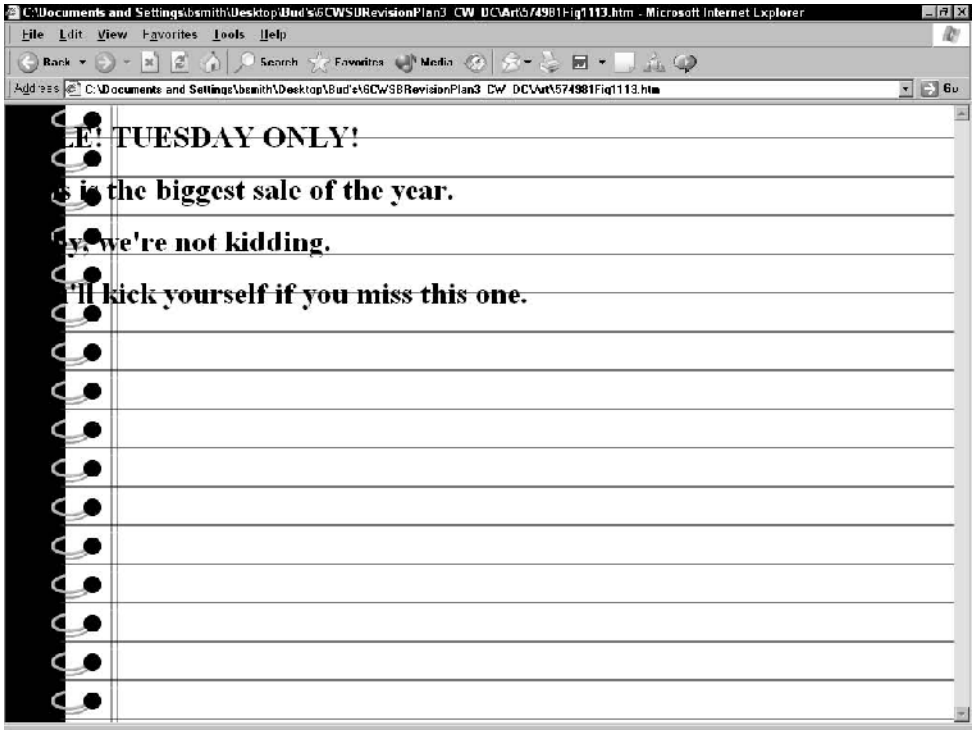
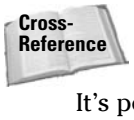


Figure 11-13: Sidebar background images interfere with page content.



See Chapter 8 for more information on sidebar images.

It's perfectly easy to use sidebar images, thus gaining the artistic look they provide, without having them mess up the rest of the page. As usual in HTML, there's more than one way to do this, but tables provide the quickest, simplest solution. At the lowest level of table usage, all you need is one cell to solve the problem. The following HTML code provides you with a basic Web page that enables you to use a sidebar background without interference:

```
<HTML>
<HEAD>
<TITLE>Sidebar Image Table Framework</TITLE>
</HEAD>
```

www.it-ebooks.info

```
<BODY background="imagefile">
<TABLE width="80%" align="right" border="1">
<TR>
<TD>
Contents
</TD>
</TR>
</TABLE>
</BODY>
</HTML>
```

Figure 11-14 shows how the single-cell table this code creates is set away from the left side that is taken up by the sidebar image. The trick is in the combination of the table's alignment and width. Setting the right alignment puts it on the opposite side of the screen from the sidebar. Setting the width, so that the table takes up less than the full screen, keeps the sidebar from interfering with the table — and its contents.

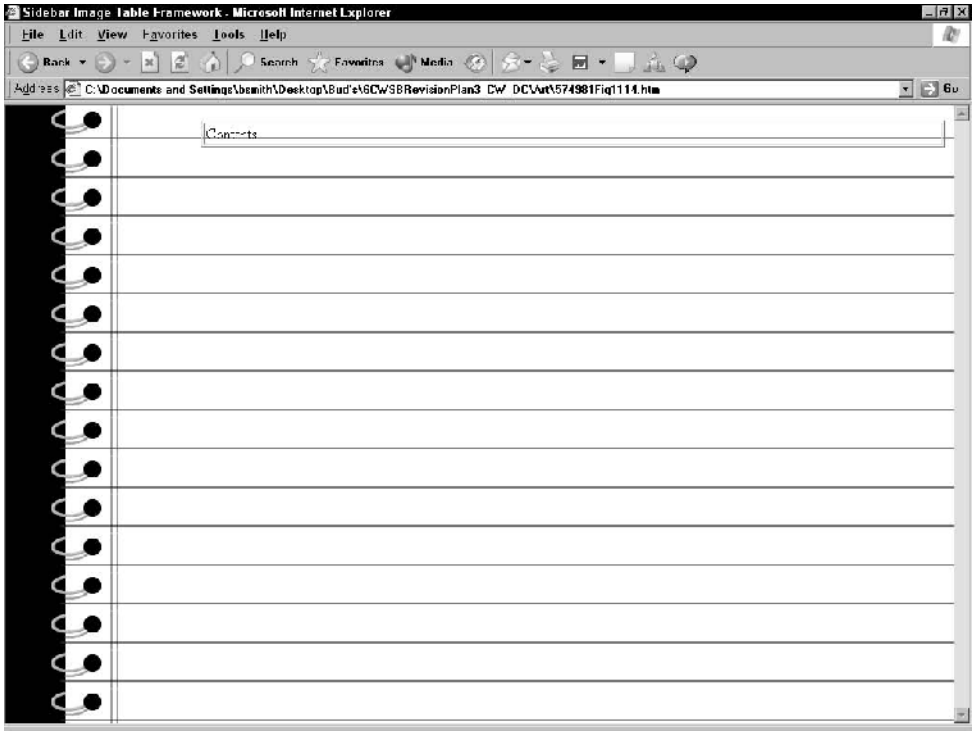


Figure 11-14: A properly sized right-aligned table does not rest over the sidebar.

To use this for your own Web pages, replace the placeholder *imagefile* with the name of your own sidebar background image file. Next, adjust the width of the table to fit the amount of space that the background image leaves open on your page. For example, you might be able to widen the table to 95 percent with some sidebar images — we're just playing it conservatively by setting the width to 80 percent in

www.it-ebooks.info

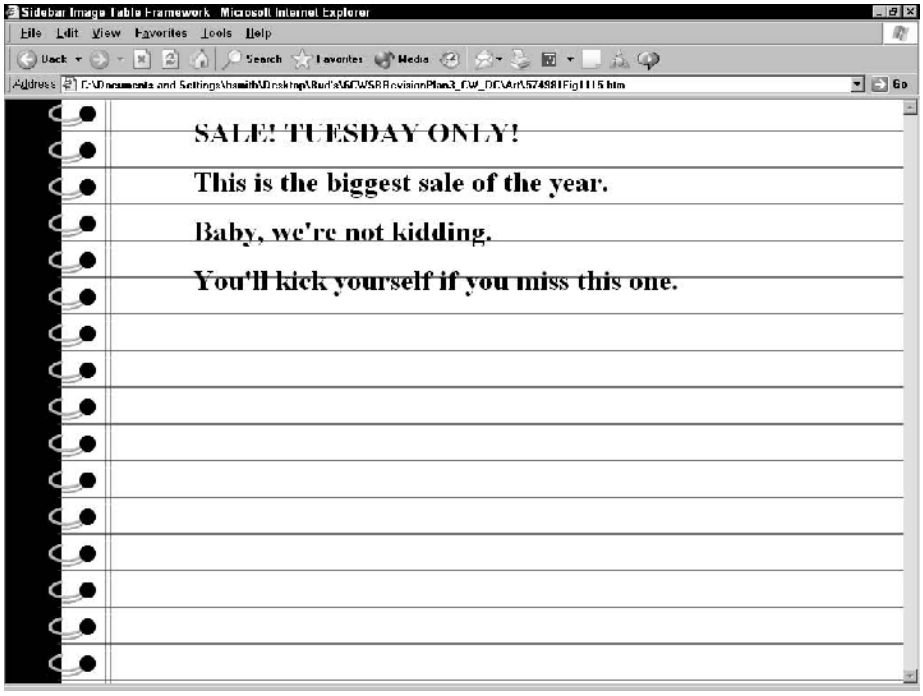


Figure 11-15: The addition of a table to the page makes the sidebar image work with the other elements.

the example. When you have the table sized accurately, you'll probably want to eliminate the border so that the presence of the table won't be obvious to your visitors. To do so, change `border="1"` to `border="0"`.

From here on in, you can forget all about tables and background images. Simply replace the placeholder word *Contents* found between the `<TD>` and `</TD>` tags with the contents of your Web page. Everything on the entire page will fit into the one cell, which expands downward to accommodate anything you put within it. Figure 11-15 shows the same Web page that started off this section, but with all its content encapsulated within the single-cell table.

There are other ways to move the elements on the page out from the edge. You could, for example, encapsulate all your text in `BLOCKQUOTE` elements. Doing that, however, would have the undesirable result of moving the text in from both margins, when you're only trying to move it away from the left side. And this solution doesn't even begin to address what to do with nontextual elements.

You could also use CSS properties to set up a margin space on the left side of the page, but CSS doesn't work with all Web browsers.



See Chapter 13 for a discussion of CSS (cascading style sheets).

www.it-ebooks.info

Setting vertical alignment

In addition to horizontal alignment of row and cell contents with the `align` attribute, you can specify a vertical alignment by using the `valign` attribute. The four possible values for the vertical alignment are `top`, `middle`, `bottom`, and `baseline`. The first three are pretty much self-explanatory. The `baseline` value, however, is one of those attributes that has just never worked. Various browsers treat it differently, and there's no telling what will happen to the appearance of your tables if you use it. Just say no to its use. Stick with the basic three.

The following example of HTML code sets up a table with four rows. The first row has no explicit `valign` setting, so it defaults to center alignment. The second row is set to top alignment. The third row is set to bottom alignment. The fourth row is set to center alignment, but two of its cells override the setting with their own `valign` settings.

```
<TABLE height="400" border="1">
<TR>
<TD>Default</TD>
<TD>Default</TD>
<TD>Default</TD>
<TD>Default</TD>
</TR>
<TR valign="top">
<TD>Top aligned</TD>
<TD>Top aligned</TD>
<TD>Top aligned</TD>
<TD>Top aligned</TD>
</TR>
<TR valign="bottom">
<TD>Bottom aligned</TD>
<TD>Bottom aligned</TD>
<TD>Bottom aligned</TD>
<TD>Bottom aligned</TD>
</TR>
<TR valign="center">
<TD valign="top">Top aligned</TD>
<TD>Set by row</TD>
<TD>Set by row</TD>
<TD valign="bottom">Bottom aligned</TD>
</TR>
</TABLE>
```

The results are shown in Figure 11-16.

Preventing word wrap

Generally, when text in a table cell reaches the border of the cell, it wraps around and continues on the next line. If necessary, the height of the cell increases to accommodate the additional text. This is the way most Web designers leave things.

www.it-ebooks.info

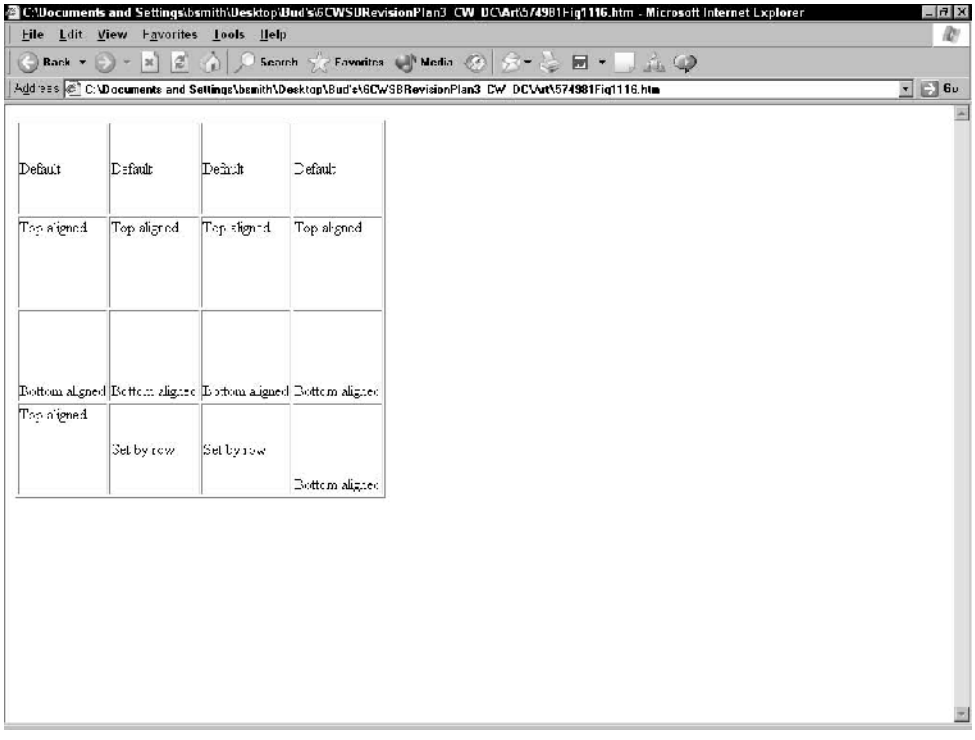


Figure 11-16: Vertical alignment in cells overrides vertical alignment in rows.

However, if you have some need to keep all the text in a cell on a single line, you can simply add the `nowrap` attribute to it, as follows:

```
<TD nowrap>
```

This causes a reversal of the normal process. When the text in the cell reaches the border, the cell expands horizontally to accommodate the additional text to keep it all on one line. Figure 11-17 shows a table where one cell has been set to `nowrap`.

Spanning Rows and Columns

The basic layout of a table is purely symmetrical — each row has the same number of cells, every cell in a column is the same size, as is every cell in a row. But this doesn't have to be the case. The `colspan` and `rowspan` attributes of the `TD` (table data, or cell) element enable you to specify that one cell should extend across the space normally taken up by several cells. This is extremely useful in typical data presentation scenarios, where you may need to set up a title cell that applies to more than one row or column. Figure 11-18 shows such an application of cell spanning.

www.it-ebooks.info

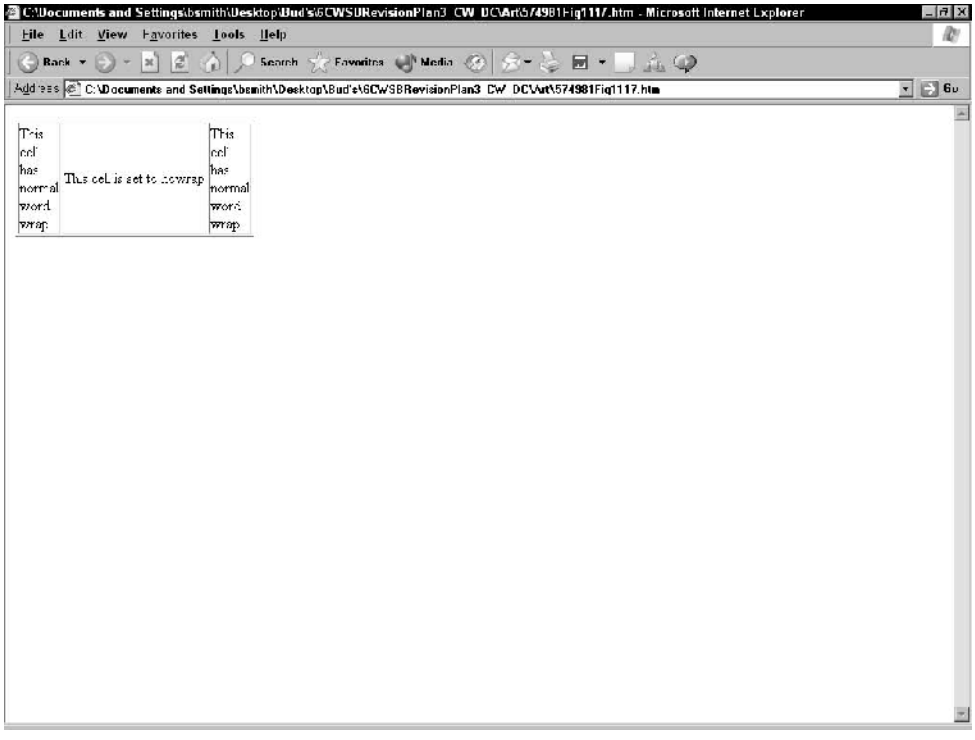


Figure 11-17: The nowrap attribute makes all text stay on one line.

The HTML code used to generate Figure 11-18 is as follows:

```
<TABLE border="1">
<TR>
<TD rowspan="2"></TD>
<TD colspan="2">Cats</TD>
<TD colspan="2">Dogs</TD>
</TR>
<TR>
<TD>Male</TD>
<TD>Female</TD>
<TD>Male</TD>
<TD>Female</TD>
</TR>
<TR>
<TD rowspan="2">
<P>Measured<BR>
Weight</P>
</TD>
<TD>17</TD>
<TD>12</TD>
<TD>40</TD>
<TD>35</TD>
```

www.it-ebooks.info

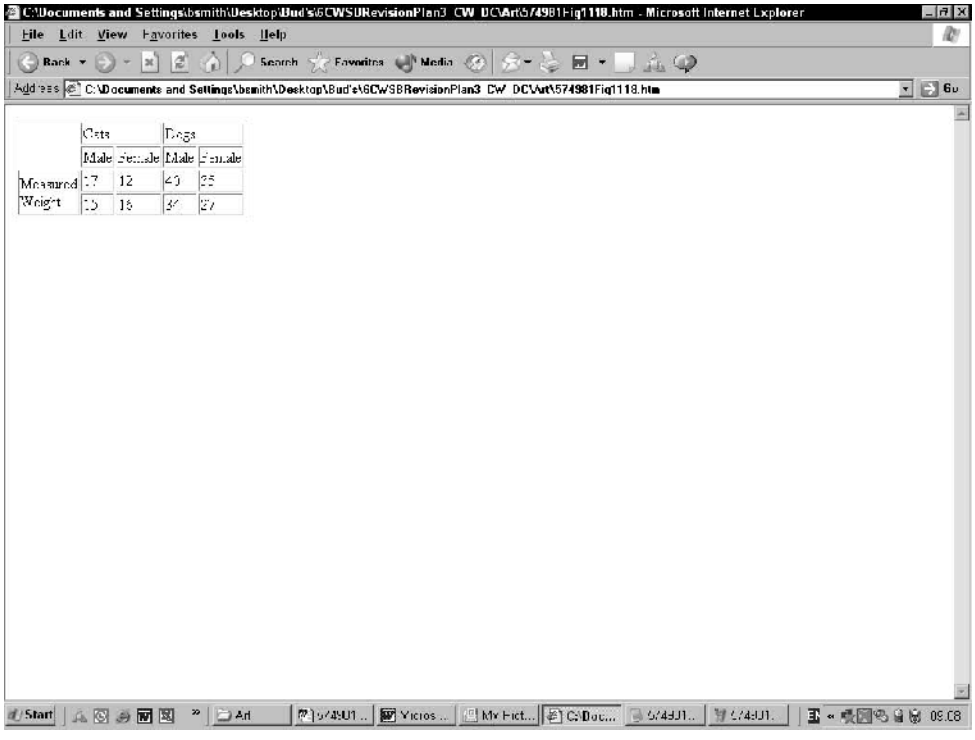


Figure 11-18: One cell can span more than one row or column.

```
</TR>
<TR>
<TD>15</TD>
<TD>16</TD>
<TD>34</TD>
<TD>27</TD>
</TR>
</TABLE>
```

This, despite the various bits of cell spanning, is still a table that is basically measured as five columns by four rows. However, the change in the number of rows or columns that a single cell can fit into makes the HTML code harder to follow than a purely symmetrical table.

The first row works like this: the first (blank) cell occupies two rows, but only one column. The next two cells each occupy only this row, but take up two columns each. Thus, there are five columns accounted for.

The second row is a bit more complicated. Only four cells are apparently in it. However, the first cell in the row is actually the blank cell from the first row, which spans both the first and second rows. Thus, the four `TD` elements, added to it, make up the five columns.

www.it-ebooks.info

The third row begins with a cell that spans the third and fourth rows, but only one column. The remaining four TD elements complete the five columns.

The final row, as with the second row, gives up its first space to the first cell from the preceding row, and adds four TD elements to finish up the table.

Working with Images and Color

Images are added to cells the same way they are added to a Web page — with the IMG element. The sole difference is that, this time, the IMG element goes in between a pair of <TD> and </TD> tags. The following HTML code shows how it's done:

```
<TD><IMG src="imagefile"></TD>
```

Figure 11-19 shows a table that includes an image. The table itself is not visible, however, because its purpose is to structure the Web page, with the image's placement controlled by the layout of the table. Therefore, the table's border value has been set to zero.



See Chapter 8 for more information on adding images to Web pages.

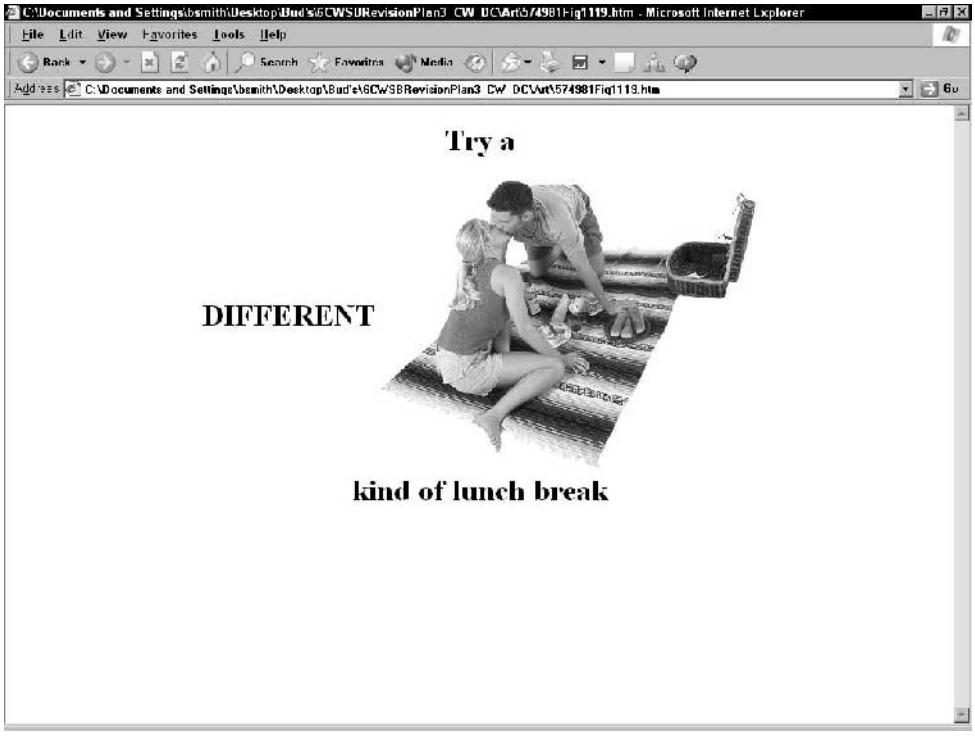


Figure 11-19: Images can be included in tables just as with other elements.

www.it-ebooks.info

Specifying background images

Entire tables can have background images, as can rows and individual cells. Candidly, there's not much real use for this ability. In fact, you can end up with some pretty ugly results. If you're using some of the more sophisticated table tricks, you run flat up against the fact that background images can undo all your work. Figure 11-20, for example, shows a table that uses a combination of background color along with both visible and invisible foreground images to create a seamless angular frame. It's a nice effect, and it takes a bit of work to get it just right. Figure 11-21 shows what happens to the table when a background image is added—it sends all that work right down the drain as the background image takes over the whole setup and wrecks the carefully designed layout. The effect isn't quite as bad in Internet Explorer as it is in Netscape Navigator, but it's still bad enough.

There are also browser compatibility problems — Netscape Navigator handles background images at the table, row, and cell levels, but Internet Explorer does not. It won't display a row background image.

If you still want or need to work with background images in tables, it's simple to do. You add the `background` attribute to the appropriate element and specify the location of the image file as the value. For example, if you wanted to add an image called `leaf_bg.gif` to the table, and `leaf_bg.gif` was in the same folder as the HTML file, you would use this code:

```
<TABLE background="leaf_bg.gif">
```

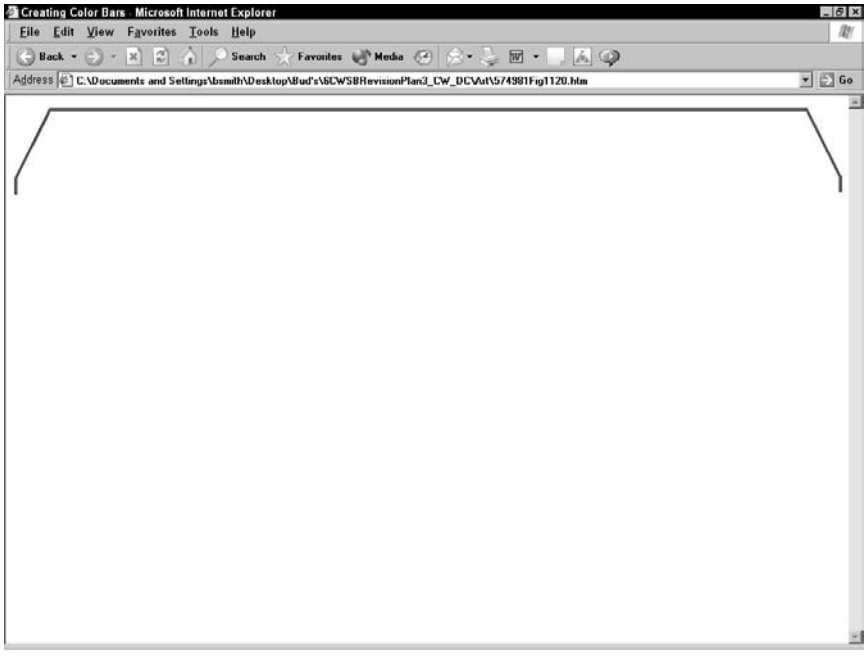


Figure 11-20: This figure shows a carefully crafted table with no background image.

www.it-ebooks.info

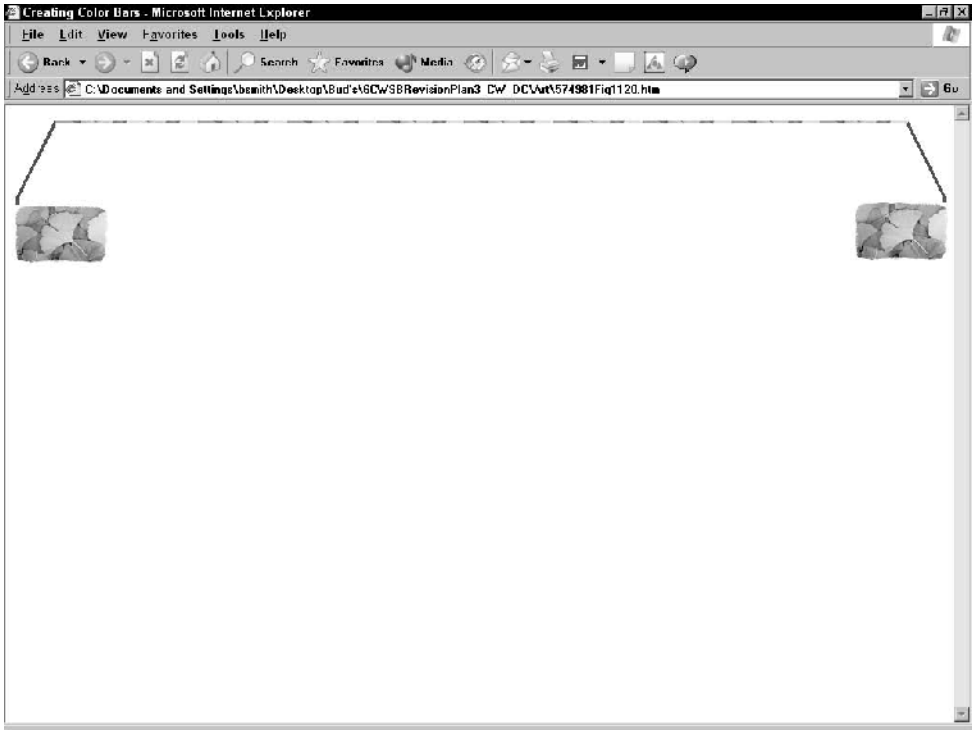


Figure 11-21: Here is the same table trashed by a background image.

To add a background image to a table row, do the same thing, but in the `TR` element:

```
<TR background="leaf_bg.gif">
```

And, finally, at the cell level:

```
<TD background="leaf_bg.gif">
```

Specifying background colors

Background colors, unlike background images, are extremely useful in tables. For plain tabular data display, they can be used to delineate different rows to increase readability, just as with the alternating lines of color on spreadsheet printouts. Background colors can also make headings stand out from the data, emphasize the bottom line, or fulfill just about any other kind of attention-grabbing need. Figure 11-22 shows a simple table containing only data, with the various parts emphasized by background colors.

www.it-ebooks.info

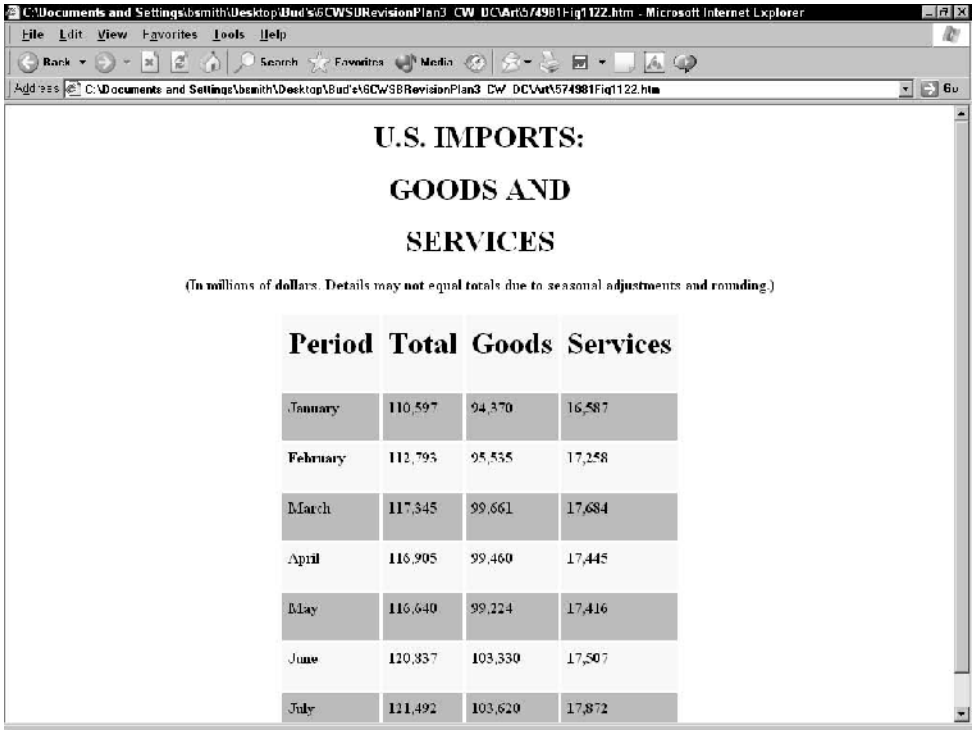


Figure 11-22: Background colors can emphasize cells.

In Web design, table background colors really shine simply because you can make any cell be any size and color you want. How does this matter? It enables you to create color bars that are an integral part of the table. This, especially in situations where you're going borderless, opens up all sorts of possibilities.

A cell's background color won't show up in a Web browser unless there's some content in the cell. So how do you get the color bar without spoiling it by covering the background color over with something else? The solution adopted by savvy Web designers has been devilishly simple — what you do is to create a special image that's composed of nothing but a single pixel and set that pixel so that it's transparent. The end result? An image that you can load into any cell without having to shrink it to make it fit into the cell, and which is invisible. The real beauty of this solution becomes apparent when you consider that you can deliberately alter the `height` and `width` values of any image, and that those values determine the size of the cell containing the image.

Here's a practical application of this technique that you can use on your own pages. Figure 11-23 shows a table composed of a series of color bars and a pair of images. The images are diagonal lines, and they're placed in opposite corners of the first row of the table. The horizontal and vertical lines in the table are the kind

www.it-ebooks.info

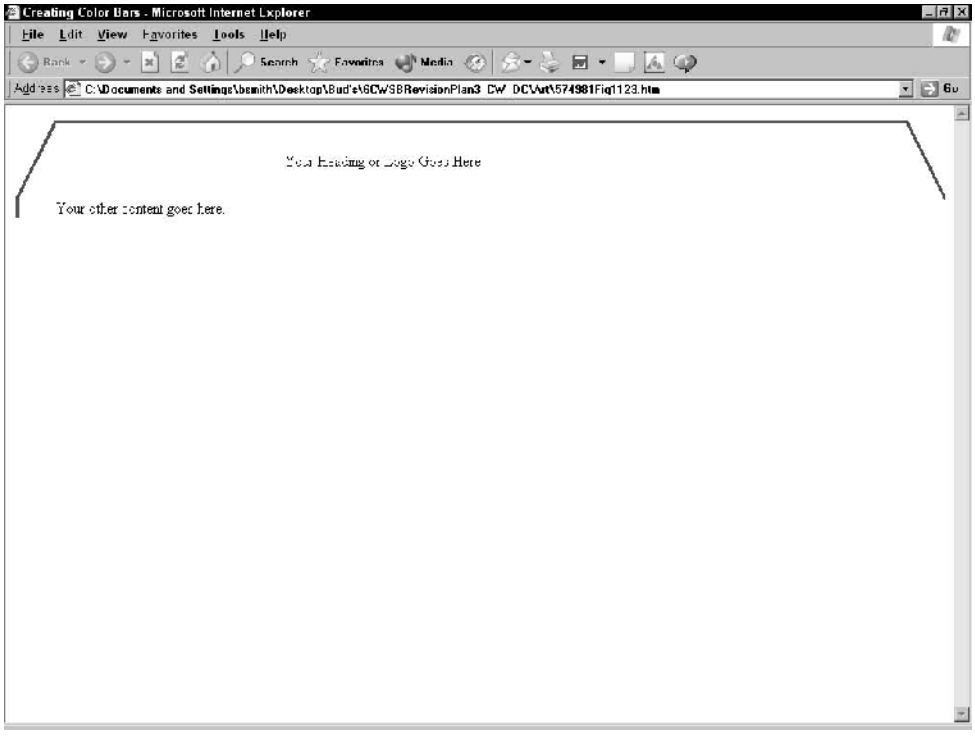


Figure 11-23: Color bars can be combined with images to produce dramatic effects.

of color bars we've been talking about, where `IttyBittyClear.gif` is doing its work.

Here's the HTML code used to create that page:

```
<HTML>
<HEAD>
<TITLE>Creating Color Bars</TITLE>
</HEAD>
<BODY bgcolor="#FFFFFF">
<TABLE border="0" cellspacing="0" cellpadding="0" align="left">
  <TR>
    <TD rowspan="2" align="right" colspan="2" valign="top"><img
src="TopLeft4.png" width="43" height="84" border="0" alt="top
left corner image"></TD>
    <TD bgcolor="#3C5897" valign="top" align="center"
width="100%"><IMG src="IttyBittyClear.gif" width="1" height="4"
border="0" alt="top color bar set to 100%"></TD>
    <TD rowspan="2" align="left" colspan="2" valign="top"> <IMG
src="TopRight4.png" width="43" height="84" border="0" alt="top
```

www.it-ebooks.info

```
right corner image"></TD>
  </TR>
  <TR>
    <TD align="center" valign="middle" height="78"
width="693">Your Heading or
     Logo Goes Here</TD>
  </TR>
  <TR>
    <TD align="left" bgcolor="#3C5897" height="100%" width="4"
VALIGN="top"> <IMG src="IttyBittyClear.gif" width="4"
height="1" border="0" alt="">
    </TD>
    <TD width="39"> <IMG src="IttyBittyClear.gif" width="39"
height="8" border="0" alt="">
    </TD>

    <TD colspan="3">Your other content goes here. <IMG
src="IttyBittyClear.gif" width="39" height="1" border="0"
alt="">
    </TD>
</TR>

</TABLE>
</BODY>
</HTML>
```

The purpose, by the way, of setting the top color bar's width to 100 percent is to ensure that the table automatically stretches or contracts to accommodate any screen resolution it's viewed at. You may recall from the section on sizing tables that, whenever the total size of all the cells in a row adds up to more than 100 percent, as it does in this case because there are other cells in this row, the Web browser automatically compensates. This keeps the table from being sized inappropriately.



You could also set the width to some outlandishly large absolute value such as 99999 pixels, to get the same effect. Because the table is already set to the screen width, this does not cause the cell to actually go up to that size.

Simply add your own content to the HTML code in the places indicated and you have a Web page that's outlined in color. To add more rows so that the table can be expanded downward, simply copy the final `TR` element and its contents, then paste it in as many times as you need (we separated it with spaces above and below it in the example code so you can pick it out easily). Figure 11-24 shows how this outline looks when it's filled with content.

Color bars don't have to sit on the fringes of the table, however. You can use them anywhere in a table. For instance, Figure 11-25 shows a newspaper style design for an e-zine (online magazine) where a thin vertical line separates one article from another. As in the preceding example, you can use this right away on your own site—just fill in the blanks and you'll be ready to go.

www.it-ebooks.info

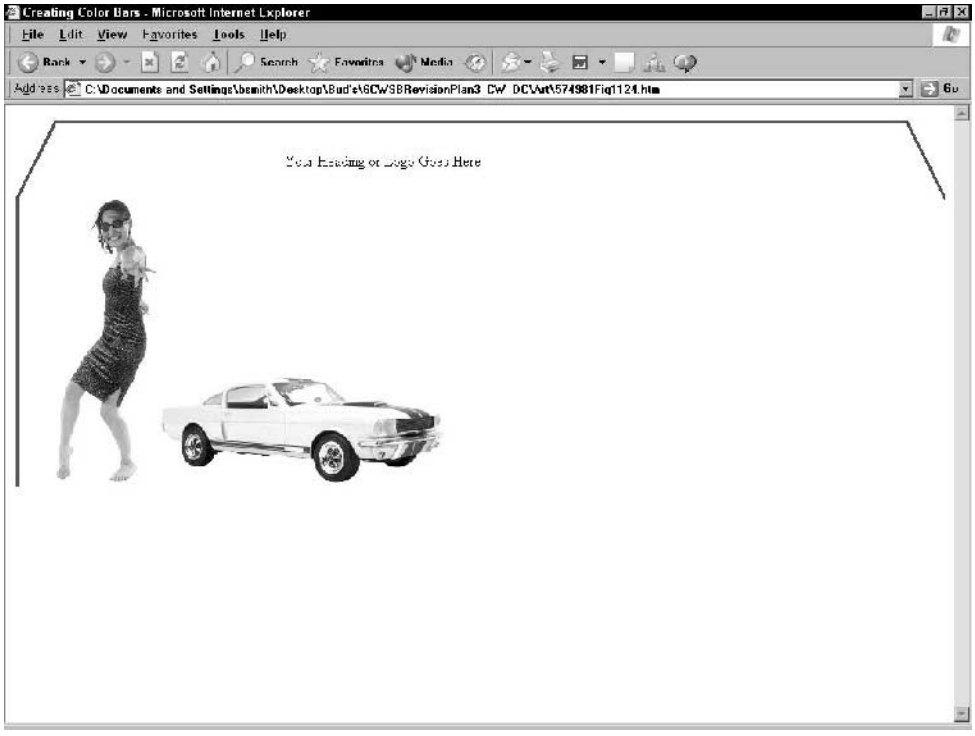


Figure 11-24: The basic outline fills out well.

Figure 11-25 shows the design where the first row, which holds the e-zine's logo, is composed of a single cell that spans all the columns. The next row is composed of three cells, with the center one holding the invisible image and having a black background color. The cells on the outside hold the articles. The bottom row, as with the top row, is a single cell that spans the other three columns. We recommend using it for links to other pages in the site, but that's up to you.

This is the HTML code that creates the framework:

```
<HTML>
<HEAD>
<TITLE>Creating a Newspaper Design</TITLE>
</HEAD>
<BODY>
<TABLE width="100%">
<TR ALIGN="center" valign="top">
<TD COLSPAN="3">Logo Goes Here</TD>
</TR>
<TR valign="top">
<TD width=50%>First article goes here.</TD>
<TD bgcolor="Black" height="100%"><IMG src="IttyBittyClear.gif"
```

www.it-ebooks.info

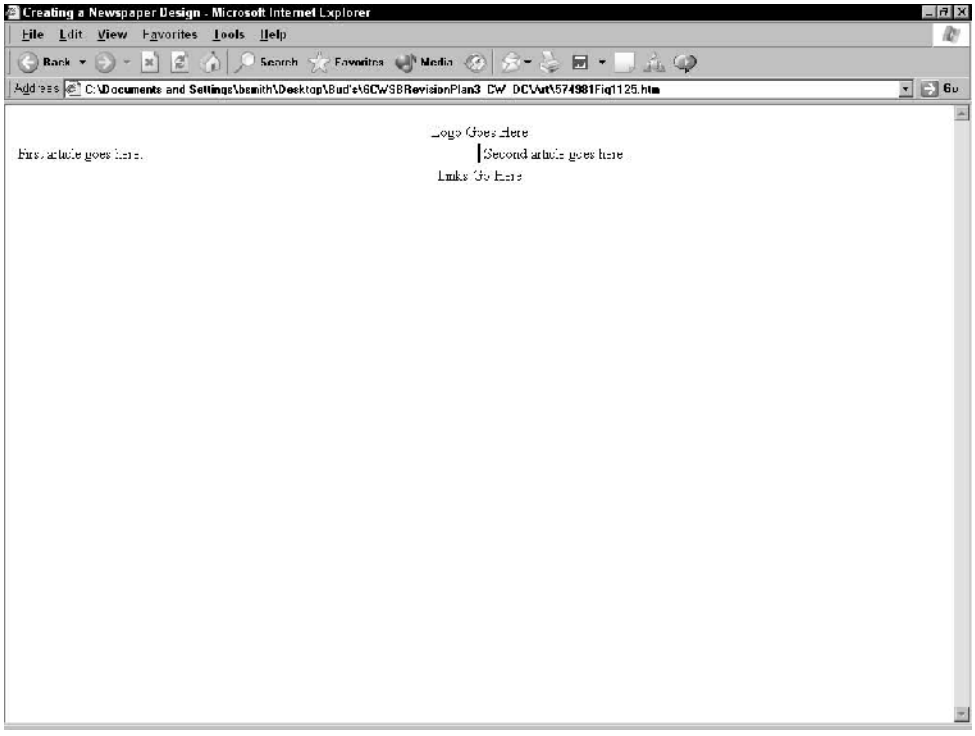


Figure 11-25: A basic e-zine layout using a color bar as a divider.

```
width="1" height="1" border="0" alt="one little dot"></TD>
<TD width=50%>Second article goes here.</TD>
</TR>
<TR ALIGN="center" valign="middle">
<TD COLSPAN="3">Links Go Here</TD>
</TR>
</TABLE>
</BODY>
</HTML>
```



Because the color bar cell is set to 100 percent of the table's height, the line extends all the way down the entire article, no matter how large or small it is.

Figure 11-26 shows the basic outline when content is added to the cells. We have also added a cell padding of 3, which increases the size of the central line in addition to moving the text out from the edges.

The same basic principles, of course, can be applied to more than just dividing two columns with a single line. Study the HTML code, let your imagination run free, and you'll find a lot of different ways to use color bars and tables.

www.it-ebooks.info

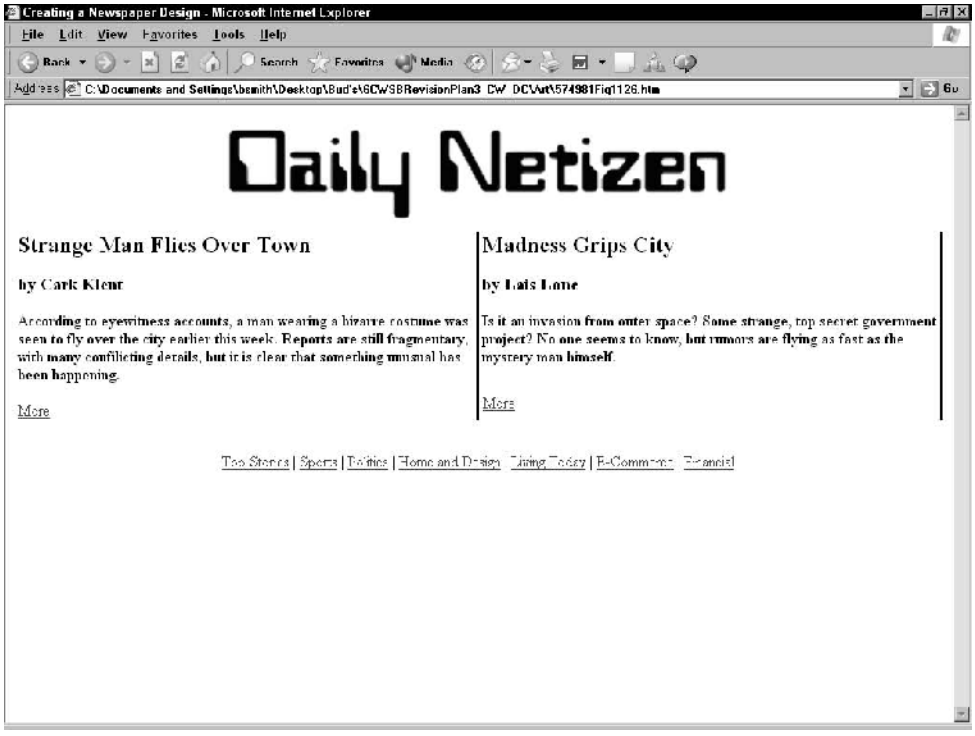


Figure 11-26: The newspaper style filled in.

Summary

If there's a secret weapon in the Web designer's arsenal, it's tables. Tables can be used for a variety of purposes.

This chapter introduced you to the following concepts and topics:

- ✦ Tables can contain any element that a Web page can contain.
- ✦ Tables are composed of rows, which are represented by the `TR` (Table Row) element. The rows are composed of cells, which are represented by the `TD` (Table Data) element.
- ✦ Tables can be set to take up either an exact size or a percentage size of the Web page.
- ✦ Borders show the edges of tables and their cells.
- ✦ Cell spacing sets the distance between cells.
- ✦ Cell padding sets the distance between the borders of cells and their contents.
- ✦ Alignment can be used within and outside of tables for special effects.
- ✦ Cells can be set to span more than one row or column.

www.it-ebooks.info

CHAPTER 12

Organizing Your Site with Frames



In This Chapter

Designing frame layouts

Creating framesets

Setting targets for your links

Customizing frame borders and margins

Setting scroll bar options

Avoiding frame problems



Frames are used to put multiple Web pages onscreen at the same time. Because each framed page resides within its own distinct and limited area of the screen, the effect is to create a seamless whole out of many parts.

All the frames together are contained within a different kind of Web page called a *frameset*. Framesets themselves are never seen by a site visitor. Only their contents—which consist of the borders of the frames and the framed pages—are visible.

The primary reason to use frames is to provide at least one area of the screen where the content remains stable while another area changes. The stable area keeps something important in front of your site visitors at all times. What this important item is varies from site to site. Most often, a special page showing a set of links to all the other pages in the site is kept in a small frame while the pages themselves are displayed in the remaining screen area. The small frame with links is commonly called a *navigation frame* because to navigate to any page in the site, visitors simply click one of the links. By keeping the links content available at all times, visitors are able to move about the site more easily than if they had to constantly return to the home page to locate a links listing.

Although navigational content is the most common use for frames, frames are sometimes used for other types of static content, such as a legal notice or graphic that you want to always keep in your visitors' field of vision.

Designing Frame Layouts

The first step in designing your framed screen is to determine how many frames you need in order to display your content. While that varies from one situation to another, a good rule of thumb is to use the minimum possible number of frames. Part of that statement is pure aesthetic opinion, and you may well

www.it-ebooks.info

disagree with it. Still, it is a simple matter of practicality that every Web designer must face—there's only a certain amount of real estate available on a computer screen. If you want to cram 20 or 30 frames into a frameset, it can technically be done, but the results won't be anything that anybody will be able to use easily. Figure 12-1 shows a Web page that has far too many frames—it's a horror that we created only so that it could serve as a bad example.

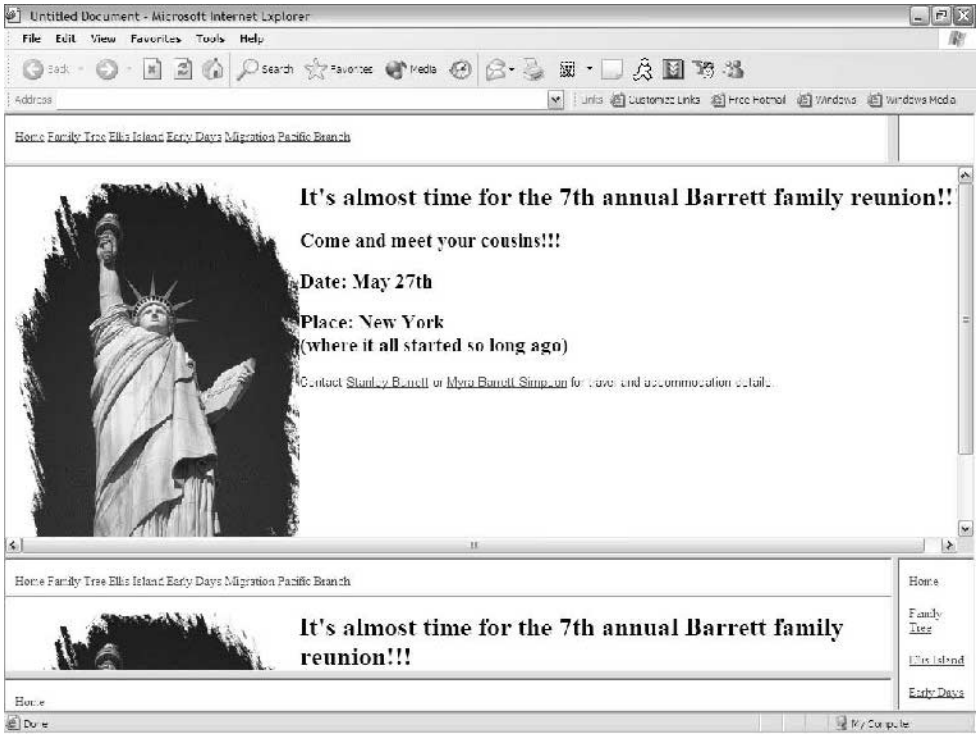


Figure 12-1: Too many frames can make a screen confusing or even unusable.

So, how many frames is right? Again, it depends on the needs of the particular Web site you're working on, but most Web designers would hesitate to have more than three to five frames on one screen. The smaller the content, the more frames you can get away with, but the key is to figure out what you really need to make the framed site work.

Analyzing functional needs

Do you have a corporate logo that you want to keep in front of visitors at all times? Frame it. Do you have a set of links that must always be accessible without users' having to scroll up and down or retrace steps via the Web browser's Back button? Frame them.

Bearing in mind that the purpose of frames is to maintain some type of unchanging content for part of the screen layout, you must take a good look at which parts of

www.it-ebooks.info

your screen you want to lock into place and which parts of it can be left free to scroll in the normal manner.

If you don't have anything that really needs to be locked into place, then you don't really need to be using frames at all.

Creating navigation layouts

The vast majority of framed pages on the Web exist for navigational purposes. With this approach, one frame holds a set of links that, when acted upon, change the contents of the other frame or frames. The most common arrangement is a two-frame setup in which the navigation frame is relatively small and the content frame takes up most of the screen. Figure 12-2 shows a left-sided navigation frame.

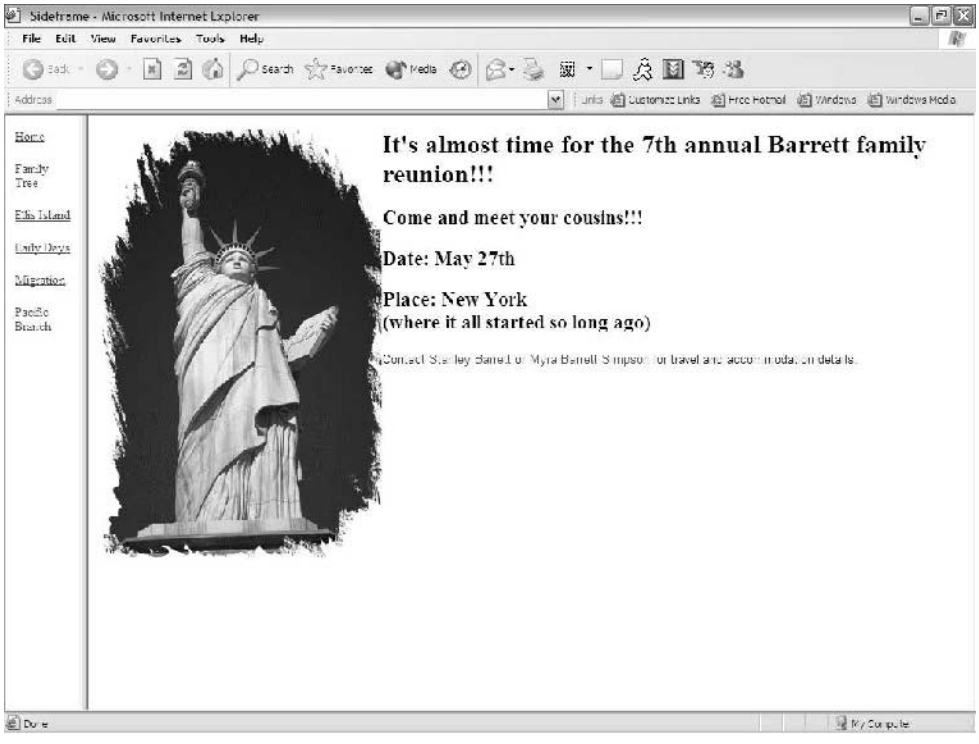


Figure 12-2: The left-sided navigation frame is common on the Web.

However, many designers use either the top or bottom of the screen to hold the navigation frame. Figure 12-3 shows the same Web site with a navigation frame on the top instead of on the left side. The right side is rarely used, but there is certainly no reason why it can't be, especially if your Web site is meant for an audience whose native language reads from right to left.

Some sites even have more than one navigation frame, depending on the needs of the material involved. It is even possible to load one frameset into a frame in another

www.it-ebooks.info

frameset. In that case, the second frameset may have its own navigation frame that works only within that smaller area (see the discussion on the `_parent` target in the section "Using reserved frame names" later in this chapter).

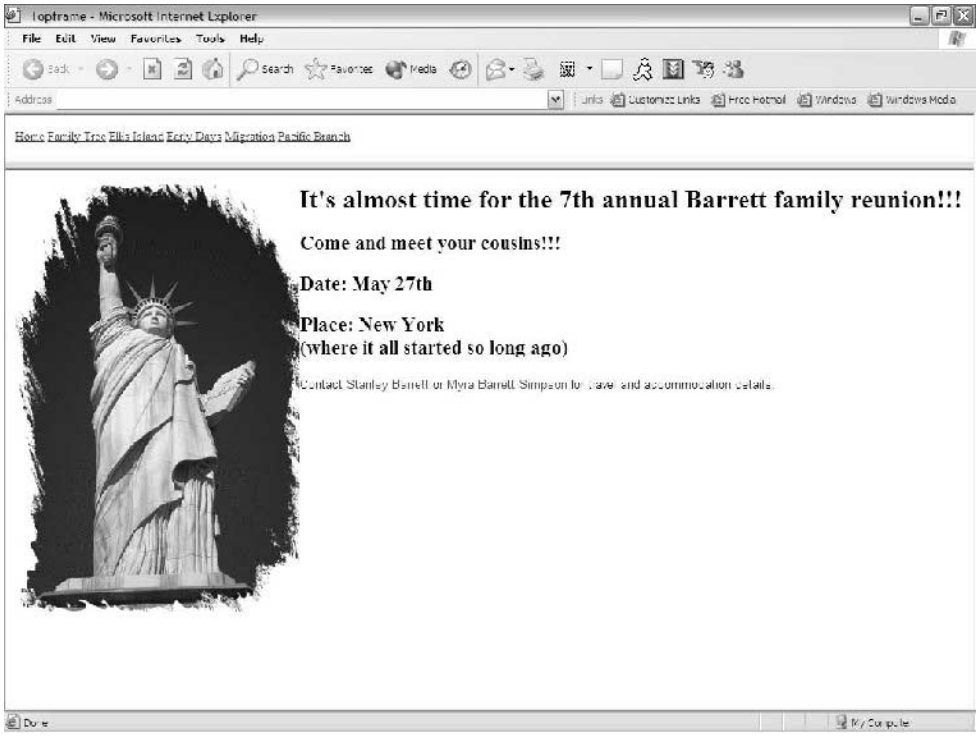


Figure 12-3: The top-sided navigation frame is almost as popular.

Setting up action or result layouts

Frames are very useful when they are teamed up with JavaScript (see Chapter 16). Although HTML links can cause only one kind of change—a new file address is called upon—JavaScript programming is much more powerful than HTML alone.

Figure 12-4 shows a Web site in which the top frame is used to enter values and the bottom frame is used to present the calculated results of those values.

Creating Framesets

Framesets are Web pages that define the areas of a Web browser's screen that will contain framed pages. In order to do this, framesets work a bit differently from the usual Web page. To save you from having to flip back to Chapter 3, here's the basic skeletal structure of a normal Web page.

www.it-ebooks.info

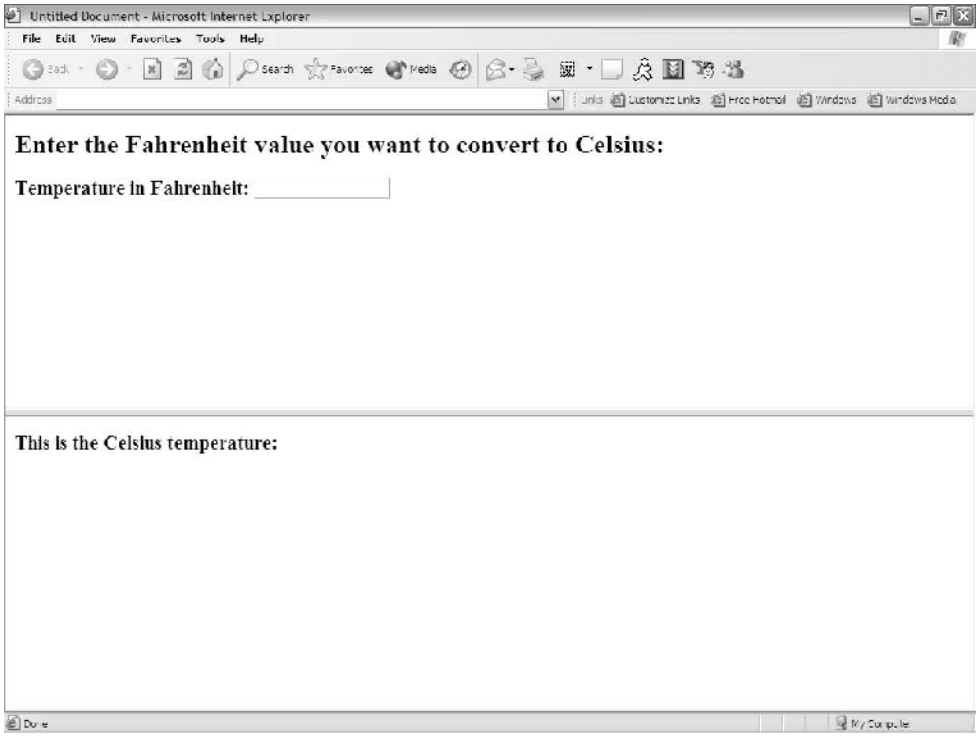


Figure 12-4: JavaScript calculation pages often use frames to accept values and respond with results in separate screen areas.

```
<HTML>
<HEAD>
<TITLE>Web Page Title Goes Here</TITLE>
</HEAD>
<BODY>
Visible Web page content goes here.
</BODY>
</HTML>
```

The most important way in which frameset pages differ from regular HTML pages is that the BODY element is replaced by the FRAMESET element. The FRAMESET element contains two or more FRAME elements. You could, in theory, have only one frame, but there's no point to doing so—the result would be just one Web page onscreen, and you don't need frames to do that. The following HTML code shows the outline structure of a frameset page:

```
<HTML>
<HEAD>
<TITLE>Frameset Title Goes Here</TITLE>
</HEAD>
<FRAMESET>
  <FRAME name="top" src="test.html">
```

www.it-ebooks.info

```
  <FRAME name="bottom" src="test.html">
</FRAMESET>
</HTML>
```

Unlike all the other elements in this listing, the FRAME element does not have an end tag. The <FRAME> tags are indented to visually emphasize their containment within the <FRAMESET> and </FRAMESET> tags, but you do not have to indent them in your HTML code if you don't want to. Because HTML doesn't recognize leading white space, it's the same to a Web browser if you indent or not.

The FRAME element needs to have an src attribute that tells it what page to display in the frame. The name attribute is optional, but to use links in one frame to change content in another frame the attribute is required (see "Setting Targets for Your Links" later in this chapter).



When a Web page is displayed in a browser, its title shows at the top of the browser. However, framesets contain multiple Web pages. Browsers solve this dilemma by showing the title of the frameset itself instead of the title of one of the contained Web pages.

The previous frameset has a problem in that only one frame is shown. That's because we need to specify the size of the columns and rows.

Setting columns and rows

The first choice you must make, as we noted earlier, is how many frames to have onscreen. The second is whether to set up vertical or horizontal frames, or both at the same time. The following example uses the rows attribute to create a pair of horizontal frames, each of which takes up half of the screen:

```
<FRAMESET rows="50%,50%">
  <FRAME>
  <FRAME>
</FRAMESET>
```

Figure 12-5 shows an evenly divided pair of horizontal frames.

To set a pair of vertical frames (columns) that each take up half the screen, you would use the same code, but substitute the cols attribute instead:

```
<FRAMESET cols="50%,50%">
  <FRAME>
  <FRAME>
</FRAMESET>
```

Figure 12-6 shows the results of this code.

www.it-ebooks.info

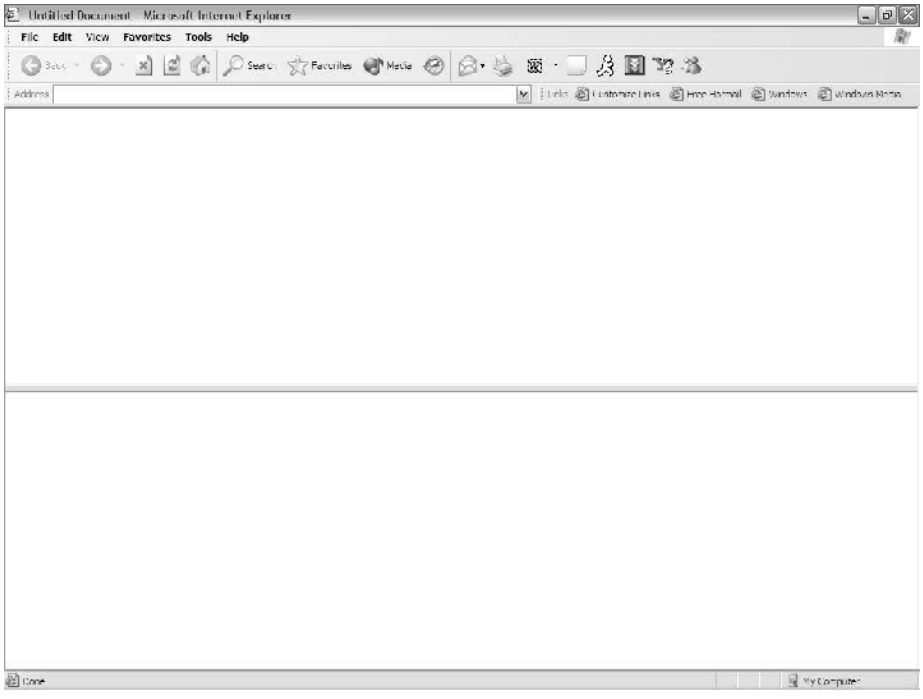


Figure 12-5: The rows attribute sets up horizontal frames.

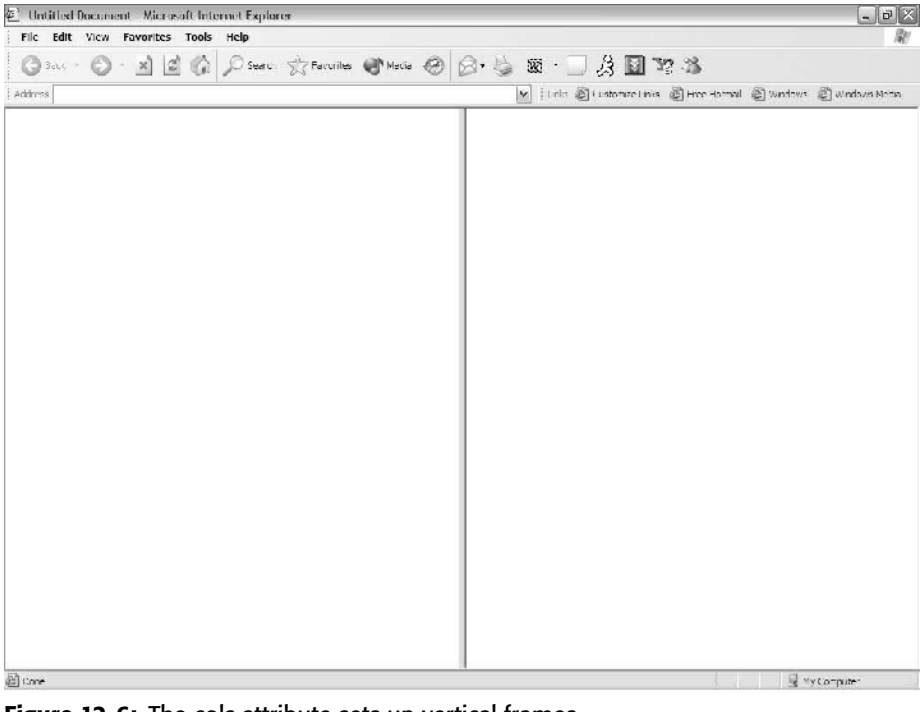


Figure 12-6: The cols attribute sets up vertical frames.

www.it-ebooks.info

If you want to have four columns, each one the same size, you could use the following code:

```
<FRAMESET cols="25%,25%,25%,25%">
  <FRAME>
  <FRAME>
  <FRAME>
  <FRAME>
</FRAMESET>
```



The number of size values in the `rows` or `cols` attribute specifies the number of frames to be created and sets how large each will be. Because we have given sizes for four frames, there must also be four `FRAME` elements.

You do not have to set up symmetrical frames. Any frame can be any size you want. You could, for example, set up the preceding four columns like this:

```
<FRAMESET cols="10%,45%,15%,30%">
  <FRAME>
  <FRAME>
  <FRAME>
  <FRAME>
</FRAMESET>
```

If you are setting up symmetrical frames, however, you can't use decimal points for your row and column sizes. The usual solution to this is to just drop any small amount of leftover pixels into one of the rows or columns. In the following code, it's impossible to create three equally sized rows and still have everything totally equal, so the final row is 1 percent larger than the first two:

```
<FRAMESET rows="33%,33%,34%">
  <FRAME>
  <FRAME>
  <FRAME>
</FRAMESET>
```

This is a simple solution and the difference is virtually unnoticeable—very few people can tell the difference between two frames that are just a handful of pixels different in size.

It's common practice—and good programming procedure—to make sure that all the column or row percentages total 100 percent. However, if you fail to get the numbers exactly right, the Web browser used to view the frameset automatically compensates for your error. If you specified each of the three rows in the preceding example as 33 percent in size, for example, the Web browser would simply make each frame equal to one-third of the screen size (for more details on error correction by browsers, see "Mixing pixel, percentage, and relative sizing" later in this chapter).

A frameset can contain both columns and rows at the same time by simply using both the `cols` and `rows` attributes. The following example, illustrated in Figure 12-7, makes eight frames by setting up two rows of four columns:

www.it-ebooks.info

```
<FRAMESET cols="25%,25%,25%,25%" rows="50%,50%">
  <FRAME>
  <FRAME>
  <FRAME>
  <FRAME>
  <FRAME>
  <FRAME>
  <FRAME>
  <FRAME>
</FRAMESET>
```

The first four frames are placed in the top row, and the next four in the second row. The frames in each row are placed from left to right.

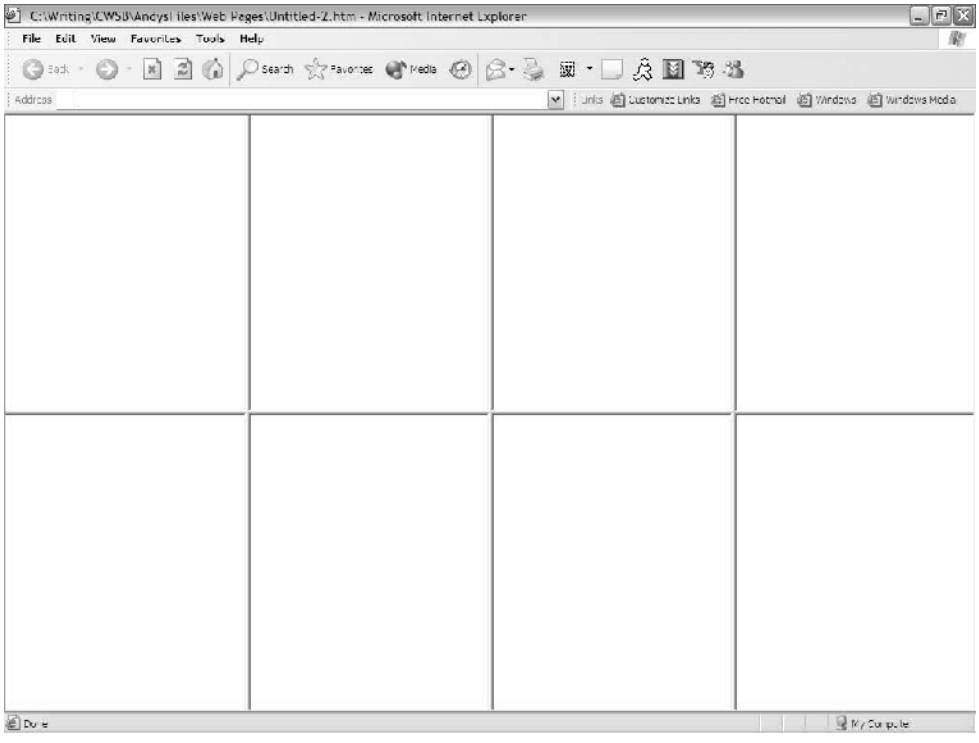


Figure 12-7: The rows and cols attributes can be combined to create a grid.



If you use both rows and columns in the same frameset, you must multiply the number of rows by the number of columns to determine how many frames are created. It's a common error to add the numbers of rows and columns instead of multiplying them.

The drawback to mixing rows and columns in this manner is that you're limited to creating symmetrical grids. Thus, if you want to divide one column in half, you have to divide all of them the same way. Asymmetrical frame layouts are possible,

www.it-ebooks.info

however, and they are discussed later in this chapter in the section "Nesting framesets."

Mixing pixel, percentage, and relative sizing

So far, we have used only percentages of screen size when setting the sizes of frames. However, there are two other options. You can state a precise and absolute size in pixels for a frame or use *relative sizing*. A frame that is relatively sized takes up all the space on the screen that isn't occupied by other frames. If there's half of the screen left for the relative frame to fill, then it takes up that half. If there's only 10 percent of the screen available, then that's how big the relative frame will be.

It's unusual to specify an exact size in pixels for all the frames in a frameset. It can be done, however, and Web browsers are smart enough to adjust for any difference between your intentions and the reality of the actual screen display. The following code, for example, sets up two columns so that they add up to an exact width of 1,024 pixels:

```
<FRAMESET cols="200,824">
  <FRAME>
  <FRAME>
</FRAMESET>
```

If this frameset is viewed by somebody using a screen resolution of 1,024 × 768, it works perfectly. However, if someone else views it using a screen resolution of 640 × 480, it will still work. The Web browser realizes the frameset can't fit on the screen and automatically resizes both frames so that everything fits into the available screen space. The resizing is done proportionally—each frame is shrunk by the same amount so that the layout remains intact. The same thing is true in reverse—if you specify a screen size that's too small, then both frames are proportionally enlarged to fit.

Although the HTML specification doesn't include percentages that are too large or too small—it only tells Web browsers how to handle errors in absolute values—both of the major browsers handle percentage errors just as they do absolute errors. If you make three frames take up only 70 percent of the screen among them, the remaining 30 percent of the space is used up by providing 10 percent more screen width to each of the three frames. If you make two frames take up 140 percent of the screen between them, the excess 40 percent is taken up by making each frame 20 percent of the screen size smaller.

Absolute values are commonly used in tandem with relative sizing. You might, for instance, specify that two columns are to be divided so that the first column is 200 pixels wide and the second one will be relatively sized, taking up all of the remaining space. To do so, you would write the following code:

```
<FRAMESET cols="200,*">
  <FRAME>
  <FRAME>
</FRAMESET>
```

www.it-ebooks.info

The asterisk is the relative size character. Exactly how much space is allotted to the second column depends on the screen resolution a visitor to your site is using. If the display is 640 pixels wide, for example, then the second column will be 440 pixels wide (640–200). Figure 12-8 shows the results of this code on a 1024-pixel-wide screen where the size of the second column becomes 824 pixels wide (1024–200).

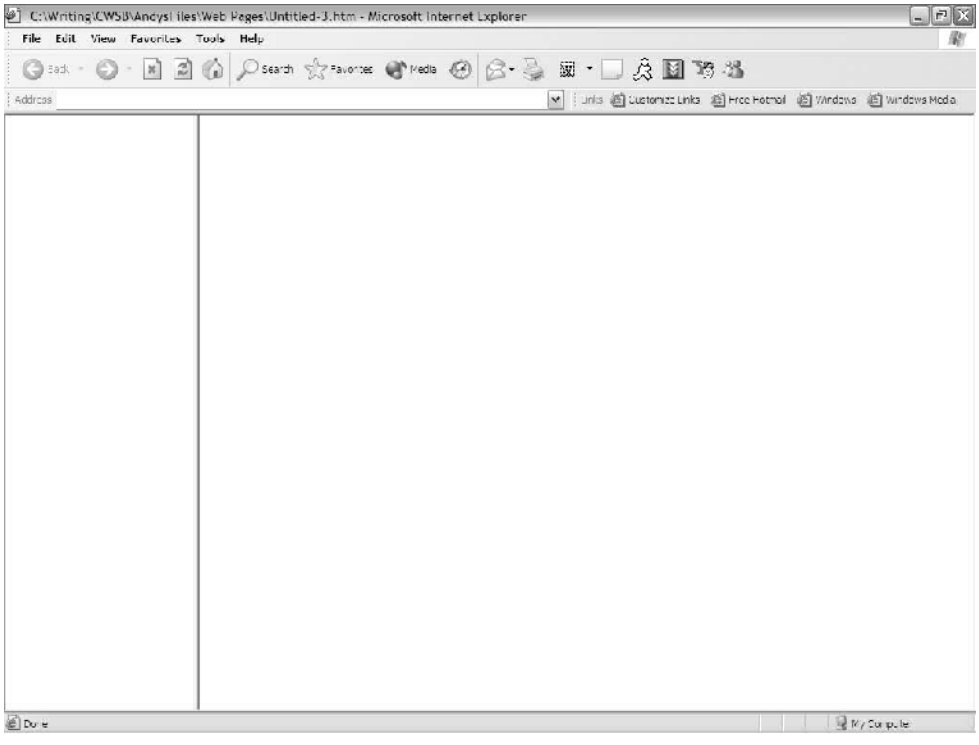


Figure 12-8: Relative sizing enables you to specify an exact size for one frame and let the remaining frame take up whatever space is left.



It's not as easy to calculate exactly how relative sizes for rows will turn out, because every Web browser takes up some of the vertical screen space with its menus and icons—and the exact amount of vertical space depends on user preference settings. Some browsers also use up some of the horizontal screen space. For instance, when you click the History button in Internet Explorer, the History window opens on the left side of the screen. When that happens, the available space for columns is also affected.

You can set relative sizes for multiple frames as well. To set up three rows in which the first is 200 pixels high and the other two each take up half of the remaining space, you would use this code:

```
<FRAMESET rows="200,*,*">
  <FRAME>
```

www.it-ebooks.info

```
  <FRAME>
  <FRAME>
</FRAMESET>
```

This means that, if the screen height is 768 pixels, the last two rows split the remaining 568 pixels, and each is thus 284 pixels high. For a screen height of 600 pixels, all three rows are 200 pixels high (400/2). If the screen height is 480 pixels, then the last two rows are each only 140 pixels high (280/2).

If you want the center row to be the one with the absolute size, the code is modified like this:

```
<FRAMESET rows="*,200,*">
  <FRAME>
  <FRAME>
  <FRAME>
</FRAMESET>
```

If none of the frames has a specific size, then all are proportionally distributed. The following code sample sets up four equally sized frames using relative sizing:

```
<FRAMESET rows="*,*,*,*">
  <FRAME>
  <FRAME>
  <FRAME>
  <FRAME>
</FRAMESET>
```

The effect is the same as if each frame had been set using a value of 25 percent.

You can also specify relative sizes by stating the proportions to be allotted to each frame. Thus, 2* is twice as much space as * and 3* is three times as much as *. To make a trio of columns in which the center column is four times as wide as the other two, write this code:

```
<FRAMESET cols="*,4*,*">
  <FRAME>
  <FRAME>
  <FRAME>
</FRAMESET>
```

Any combination of the three approaches can be used together. The following example shows how to make three rows in which the first row is 100 pixels high and the second is twice as high as the third:

```
<FRAMESET rows="100,2*,*">
  <FRAME>
  <FRAME>
  <FRAME>
</FRAMESET>
```

www.it-ebooks.info

Likewise, you can specify percentages for some frames while others are left to relative sizing. In this example, the first row takes up 35 percent of the screen height, and the second row is one-fifth of the height of the third row:

```
<FRAMESET rows="35%,*,5*">
  <FRAME>
  <FRAME>
  <FRAME>
</FRAMESET>
```

You can mix all three techniques together, as in the following code:

```
<FRAMESET rows="25%,*,80">
  <FRAME>
  <FRAME>
  <FRAME>
</FRAMESET>
```

This sets the first row to one-quarter of the screen height, sets the final row to 80 pixels in height, and leaves the center row to take up all the remaining space.

Preventing resizing

The default behavior of Web browsers is to allow users to resize frames at will by dragging the frame border to a new location. Generally speaking, it's preferable to leave your site visitors with this option intact. However, if there is some reason why your framed design would suffer if anything were changed, you can ensure that the frames cannot be resized simply by adding the `noresize` attribute to the FRAME element:

```
<FRAME noresize>
```

Specifying frame names and contents

Each frame must have a name if you are ever going to change its original content. Although you can put a Web page into a frame just by specifying it in the initial frame declaration, the only way to tell which frame to load a new Web page into is by using the frame's name (see "Setting Targets for Your Links" later in this chapter).



When we refer to the contents of a frame as a Web page, we are simply recognizing that most people use frames for displaying Web pages. However, there is nothing to stop you from putting any Web-displayable file into a frame. You could specify a JPEG image file, for example, for the contents just as easily as you could a Web page. You can even use another frameset as the contents instead of a normal Web page, although that's pushing the capabilities of most Web browsers to the limit and begging for a crash.

There are very few limits on creating frame names. There are some reserved names that you cannot use: `_blank`, `_parent`, `_self`, and `_top`. The first character of your

www.it-ebooks.info

frame name must be a letter (the reserved names are the only ones that don't follow this rule). Beyond that, the name may contain letters, numbers, hyphens, underscores, colons, or periods. Table 12-1 provides examples of legal and illegal frame names.

Table 12-1
Frame Naming Conventions

Valid Name	*Invalid Name*
secondframe	2ndframe
AboveTheOthers	_AboveTheOthers
left-Side	-leftSide
a9q:-._	a9q:-._?/

Although this means that you can create practically any names you want for your frames, it's good practice to use descriptive names. `MainDisplay`, for example, is much easier to comprehend than something like `Frame3`.

The name of the frame is an attribute of the `FRAME` element and is assigned by using the `name` attribute. So is the location of the Web page that will initially fill the frame; the location is shown by the `src` (source) attribute. The following line of code shows how the name and source are specified:

```
<FRAME name="LeftSide" src="firstpage.html">
```



The location of the Web page in the `src` attribute follows the same rules about absolute and relative URLs that the `href` attribute does in anchors (see Chapter 4).

Figure 12-9 shows a genealogy Web site with two frames. The small top frame holds a Web page whose sole content is a graphics file that shows a fancy image of the family name. The larger bottom frame holds the event announcement page for the site.

The frameset and contents were specified using the following code:

```
<HTML>
<HEAD>
<TITLE>The Barrett Family Genealogy Page</TITLE>
</HEAD>
<FRAMESET rows="105,*">
  <FRAME name="banner" src="barrettslogo.html">
  <FRAME name="mainpage" src="family.html">
</FRAMESET>
</HTML>
```

The top frame is set to a height of 105 pixels because that gives the graphic in the Web page contained in that frame a little bit of room around it. The bottom frame is set to relative sizing, taking up whatever remains of the available vertical space.

www.it-ebooks.info

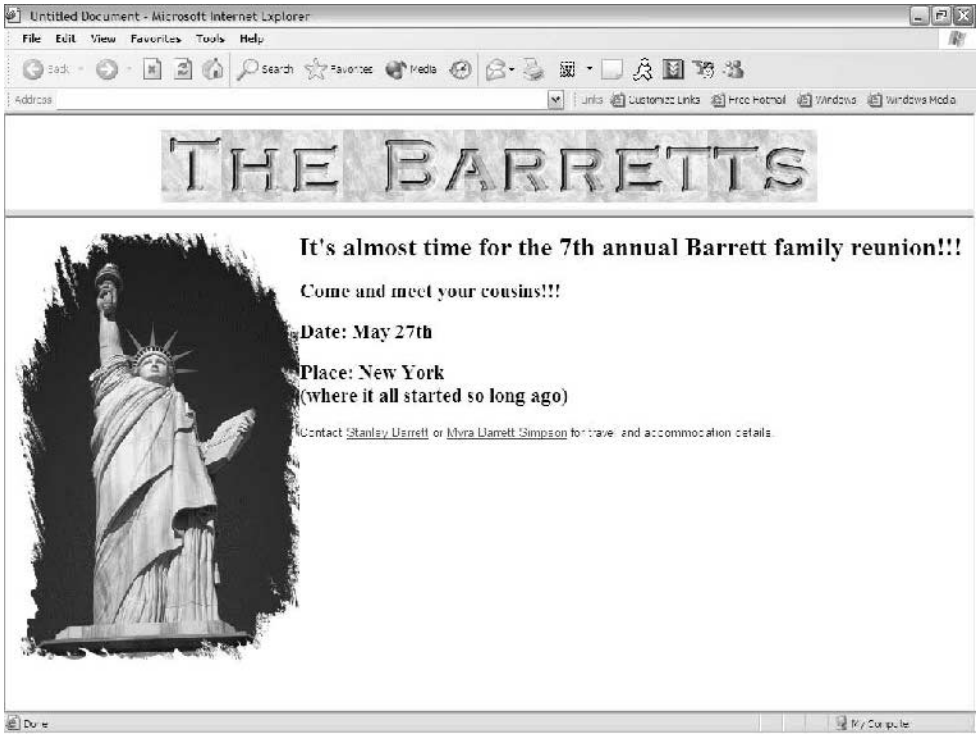


Figure 12-9: This figure shows a framed genealogy Web site.



Don't forget that, if you want the frameset to be the default page for your site, you need to save it as index.html.

Nesting framesets

Framesets, in addition to containing frames, can contain other framesets. This technique is called *nesting*. Why would anyone want to do this? To overcome the basic limitation of frames—that they must be either horizontal or vertical. By nesting one frameset within another frameset, you can set one to vertical and the other to horizontal, resulting in a much more complex and useful screen layout.

The skeleton structure of a nested frameset page shows that the internal frameset is equivalent to a regular frame:

```
<HTML>
<HEAD>
<TITLE>Frameset Title Goes Here</TITLE>
</HEAD>
<FRAMESET>
  <FRAME>
  <FRAME>
```

www.it-ebooks.info

```
    <FRAMESET>
      <FRAME>
      <FRAME>
    </FRAMESET>
  </FRAMESET>
  </HTML>
```

The genealogy page in the preceding example uses a simple pair of horizontal frames. That kind of setup maximizes the screen area that the main Web page has to sprawl out in, but it also means that frames aren't being used to their full advantage. All the links for the site will have to be contained within the Web page itself, which also puts constraints on the layout, content, and functionality of that page.

You could solve the problem by putting a set of links to other pages in the site into the top frame, but the tradeoff is that you must replace the logo graphic with the link set. If a second frameset is added, however, the top frame can remain as it is and site links can be placed to the side of the main page, as shown in Figure 12-10. This not only frees up the main page for any other type of content you want to add to it, but keeps the link listing constantly available as other Web pages are loaded into the main frame.

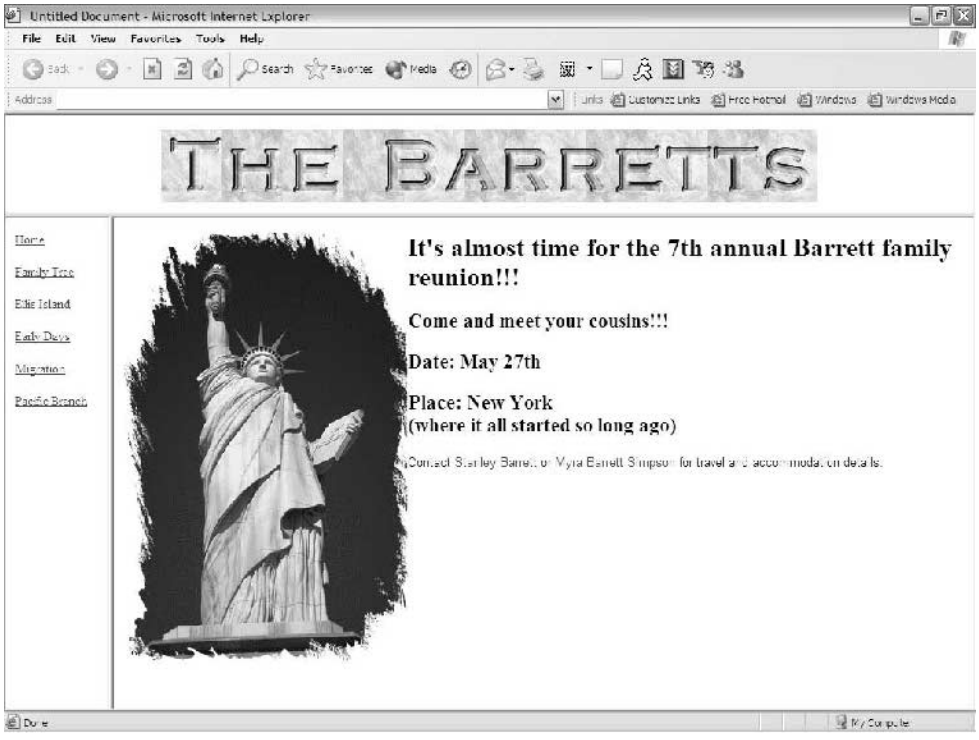


Figure 12-10: A second frameset enables you to include more content.

www.it-ebooks.info

This is the HTML code that achieves the modification:

```
<HTML>
<HEAD>
<TITLE>The Barrett Family Genealogy Page</TITLE>
</HEAD>
<FRAMESET rows="105,*">
  <FRAME name="banner" src="barrettslogo.html">
  <FRAMESET cols="110,*">
    <FRAME name="navpage" src="barrettlinks.html">
    <FRAME name="mainpage" src="family.html">
  </FRAMESET>
</FRAMESET>
</HTML>
```

The containing frameset is still the same as before—it divides the screen into two rows with the top row containing the graphic. The content of the bottom row, however, is no longer a frame but another frameset. That frameset divides the bottom row into two columns. The left column is set to a width of 110 pixels, and the right column uses relative sizing. The left column contains a new Web page that has the links to the rest of the site. The right column contains the same page that used to take up the entire bottom row.

Adding NOFRAMES content

The NOFRAMES element has only one purpose—to provide some visible content for users of antique Web browsers or alternative browsing tools that don't display framesets (the text-to-speech browsers used by blind people, for example). In this respect, it is much like supplying alternative text for images (see Chapter 8). Nobody knows how many people are using browsers that can't handle frames, so you should always make some sort of provision for that possibility.

The NOFRAMES element is not, as you might expect, contained within the FRAMESET element. Because it's not a part of the frame scenario, but an alternative, it goes outside. The following HTML code shows where it fits:

```
<HTML>
<HEAD>
<TITLE>Frameset Title Goes Here</TITLE>
</HEAD>
<FRAMESET>
  <FRAME>
  <FRAME>
</FRAMESET>
<NOFRAMES>
Noframes content goes here.
</NOFRAMES>
</HTML>
```

Within the NOFRAMES element, you enter everything you want to be displayed in case a browser that isn't capable of showing frames is used to view the frameset page. That content depends upon your own site's intentions, audience, and capacity.

www.it-ebooks.info

Many framed sites use a blunt and simple approach, as you see in this example:

```
<NOFRAMES>
<P>This Web site uses frames. Your browser does not support them.</P>
</NOFRAMES>
```

However it is better to provide a link to a version that doesn't use frames:

```
<noframes>
  <h2>Frame Alert</h2>
  <p>
  This document is designed to be viewed using the frames feature.
  You are using a non-frame-capable browser.
  <br>
  See the <a href="noframe.html">Non-frame version.</a>
  </p>
</noframes>
```

The contents of the `NOFRAMES` element can be as complex as any Web page. We'll explore this in the following project.

Setting Targets for Your Links

It is possible that you could set up a framed site in which none of the contents of any of the frames ever changes, resulting in a simple, static layout that's no different from a normal Web page except for the fact that it's composed of multiple HTML files. However, people generally use frames so that their content can be altered.

Targeting custom-named frames

Generally, when someone clicks a link on a Web page, the new Web page simply replaces the old one on his or her screen. Frames are different, however. Different frames can be specified as the *target* for the link—the frame in which the link will open. Links between frames have a variety of different behaviors depending on what target is set for the link (see Chapter 4 for more information on creating links). The syntax for setting a link's target is as follows:

```
<A href="url" target="framename">
```

When you click a link that's on a framed Web page and no target is specified, the linked Web page opens in the same frame that the link was in, replacing the linking page. If you specify another frame as the target, then the linked page opens in the specified target frame instead, replacing whatever is in that frame but leaving the linking page unaffected.

For example, let's say you have two frames, one of which is named `frameone` and the other `frametwo`, and you have a links page in `frameone`. If you used the following code in a link in `frameone`, then the linked page would open in `frameone`:

```
<A href="somewhere.html">Over The Rainbow</A>
```

www.it-ebooks.info

In this case, the Web page named `somewhere.html` would be loaded over the links page in `frameone`, and there would be no effect at all on `frametwo`.

However, if you gave an additional instruction by adding the target attribute to the code and specifying `frametwo` as the target, the results would be different:

```
<A href="somewhere.html" target="frametwo">Over The Rainbow</A>
```

This would cause the Web page named `somewhere.html` to open in `frametwo`, leaving the links page in `frameone` intact.



If you use a target name that doesn't exist, the linked page opens in a new browser window.

The following is the HTML code for the links page that was used in the navigation frame for the genealogy page in the preceding section:

```
<HTML>
<HEAD>
<TITLE>Barrett Site Links</TITLE>
</HEAD>
<BODY>
<P><A href="family.html" target="mainpage">Home</A></P>
<P><A href="familytree.html" target="mainpage">Family Tree</A></P>
<P><A href="ellis.html" target="mainpage">Ellis Island</A></P>
<P><A href="earlydays.html" target="mainpage">Early Days</A></P>
<P><A href="migration.html" target="mainpage">Migration</A></P>
<P><A href="pacific.html" target="mainpage">Pacific Branch</A></P>
</BODY>
</HTML>
```

The main display area in that site's frameset was a frame named `mainpage`, and each of the links in the navigation page is set to have that frame as its target. Without that target specification, the linked pages would all open within the narrow navigation frame.

Using reserved frame names

Each reserved frame name serves a special purpose. Table 12-2 explains what each of them does:



HTML is not case-sensitive, which means that `FRAMENAME` and `framename` are the same to it. However, that's not the case with the reserved frame names; they must be all lowercase all the time.

Three of the reserved frame names—`_self`, `_top`, and `_blank`—are clear and unambiguous. The `_parent` name, however, is a source of confusion to most Web designers. Most of the time, it acts just like `_top`. Only when one frameset is contained within another one does the difference become apparent.

www.it-ebooks.info

Table 12-2
Reserved Frame Name Functions

Frame Name	*Function*
`_self`	Opens the linked Web page in the frame that holds the link. This is the same as the default linking behavior if no target is specified.
`_top`	Opens the linked Web page in the full browser window, obliterating all the frames.
`_blank`	Opens the linked Web page in a new instance of the Web browser.
_parent	Opens the linked Web page in the next highest frame or window.

We're referring here not to the practice of creating nested frames in your HTML code, but to using a second frameset page as the contents of one of your frames. It doesn't matter if it got there as the frame's original content because you put it there using the `FRAME` element's `src` attribute, or if a link from the original content loaded the second frameset into the frame. If a link in any of the frames in the second frameset has its target set to `_parent`, then the linked page appears in the frame that holds the second frameset, overwriting the second frameset but leaving all the frames in the first frameset intact.

Customizing Frame Borders and Margins

The borders that separate one frame from another are usually left alone. However, you do have several options you may want to pursue, such as changing the width or color of the border. The amount of space around the contents of frames is also open to manipulation.

Setting border width

The width of the borders between frames (in pixels) is set with the `border` attribute. Setting the value of `border` to 0 (zero) removes all frame borders, as shown in Figure 12-11. Any setting from 1 on up changes the width and is a matter of personal preference and should be determined by the needs of a particular Web site's design.



You can create the same effect without using frames. To see how to use tables to create a layout of this sort, see Chapter 11.

If we were to change the genealogy Web site so that it had borders that were 30 pixels wide, for example, it would pretty much ruin its appearance. The following change in the first `FRAMESET` element is all that's required to mess up the site, as shown in Figure 12-12:

```
<FRAMESET rows="105,*" border="30">
```

www.it-ebooks.info

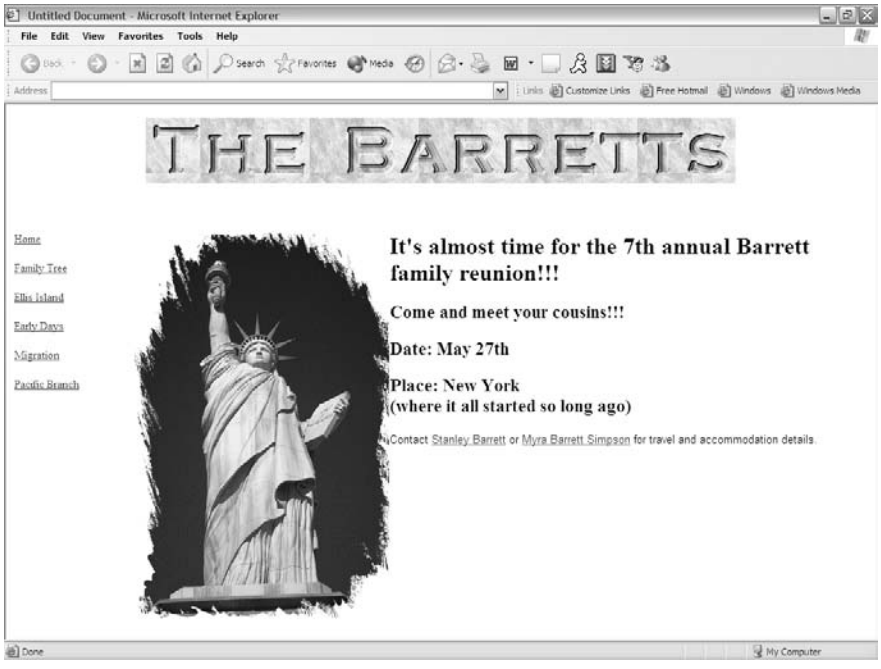


Figure 12-11: You can display frames without borders.

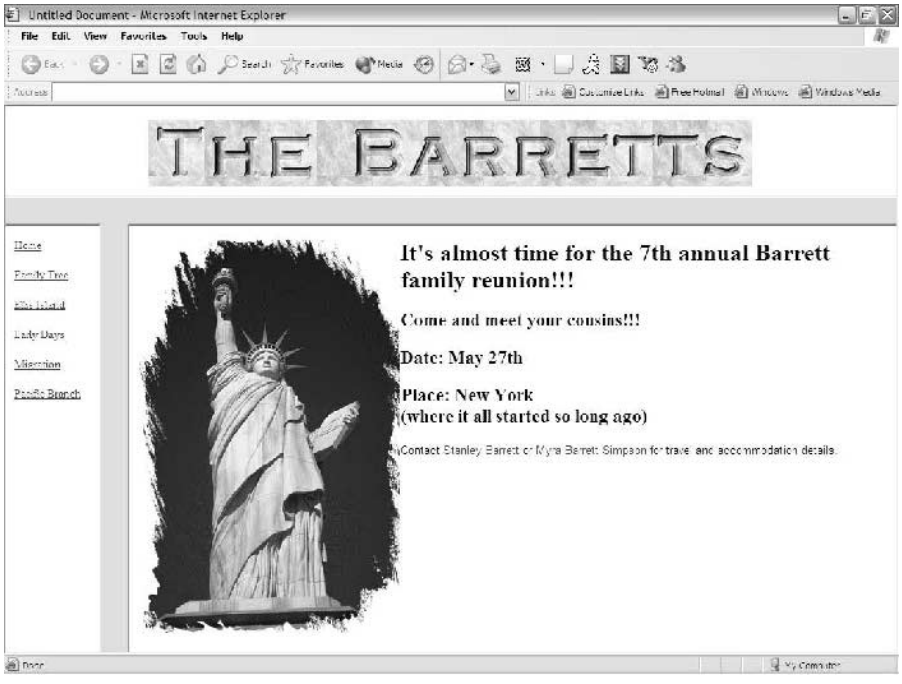


Figure 12-12: Changing the border size can have unfortunate effects on the display.

www.it-ebooks.info

If you add the `border` attribute to the second `FRAMESET` element instead of the first one, you begin to get a sense for how frustrating it can be to monkey with borders—Netscape Navigator ignores the attribute entirely unless it's in the first `FRAMESET` element, whereas Internet Explorer implements the change in border size for the specified frameset only. If you add different border sizes to both framesets, both Navigator and Internet Explorer ignore the second one.

In addition to setting the border value to zero, you can remove all borders between your frames with the `frameborder` attribute:

```
<FRAMESET frameborder="no">
```

The values for `frameborder` can be yes or no. Alternatively, you can use 1 for yes and 0 for no. There is no reason to use this attribute if the value is going to be yes, however, because that's the default setting, anyway.

The same attribute can be used with individual `FRAME` elements as well. Here, though, the going gets even rougher. Depending on the layout of frames in your site and which browser is used to view it, the appearance can be anything but what you intended. Any border can be viewed as belonging to both of the frames it divides, and various browsers take different approaches, so make sure to check the resulting screen in all browsers you intend to serve.

Handling border colors

The `bordercolor` attribute is the one that handles border colors:

```
<FRAMESET bordercolor="fuchsia">
```

The same attribute works with the `FRAME` element as well.

Border colors between frames can be tricky. Unless you want all your frame borders to be the same color (which we would generally recommend), you again run up against the fact that a single border touches on two frames. If you set the first frameset in the genealogy site to one color and the second one to another, Internet Explorer will go along with your desires, but Navigator sets all the borders to the second color.

If you set the first frame in each frameset to yet another color, you'll finally find Navigator and Internet Explorer in agreement—the frame colors override the frameset colors.

As with border sizes, it's worth a little bit of experimentation to get the hang of it.

Setting margin width

Margins—the amount of space between the content of a frame and the borders of that frame—are set with the `marginheight` and `marginwidth` attributes of the `FRAME` element. The first sets how much vertical space (in pixels) there is between the top and bottom of the frame and its content. The second does the same for the horizontal space between the left and right sides of the frame and its content.

www.it-ebooks.info

Thus, if you want to specify a 20 pixel space horizontally and a 40 pixel space vertically, you would use this code:

```
<FRAME marginwidth="20" marginheight="40">
```

Setting Scroll Bar Options

The default behavior of scroll bars in frames is `auto`, which means that, if they're needed, they'll appear. Basically, that means that if the contents of the frame are larger than the space allotted for the frame itself, you get scroll bars. However, you can specify that they must appear regardless of need or that they can't appear under any circumstances. In our opinion, the default setting should be left intact in any but the most unusual situations. Just in case you're facing one of those unusual situations, here's how to manage it.

To set scroll bars so that they cannot appear in a frame, use this code:

```
<FRAME scrolling="no">
```

Just change the "`no`" to "`yes`" and you have made it so that they must appear.

Alternatives to framesets and frames

There are alternatives to the use of frames to show fixed content on each page of a Web site iframes, server side includes, and XML with XSL transformations. Each option including frames has its advantages and disadvantages. The main disadvantage of frames is that you have to specify the size of the frame yet you don't know the size of the users window so your page may not be displayed as you wish. With server side includes and XML with XSL transformations, you can take advantage of the formatting properties of standard HTML and let the browser decide how to layout the pages. When using iframes the size doesn't need to be specified.

iframes

Using frames and framesets you define a template and include Web pages inside the frames. However, sometimes you may have to do the opposite and put a frame inside a Web page. Using an iframe (internal frame), you can embed a Web page in the current page in much the same way as you display an image in a Web page.

The advantages of using iframes are that you have control over how they are aligned within surrounding text. They can move around as a user resizes the page, and they can appear in any part of the page. And so they don't have to divide your harmoniously designed Web page in squares.

The way to do this is to use an iframe tag:

```
<iframe src="encrusted.html" width="300" height="300"
frameborder="0">
```

www.it-ebooks.info

```
Your browser does not support internal frames. See the
<a href="encrusted.html">non frame content</a>.
```

The width and height are in pixels or can be percentage values relative to the parent element. Writing `frameborder="0"` specifies not to draw a border. If you don't specify the attribute, the default is a border of 1 pixel.

Programs in JavaScript sometimes use iframes to mimic a slide show or reload information that changes such as stock quotes.

Similarities between iframes and frames:

- ✦ Both are scrollable
- ✦ Both show external HTML pages

Differences between iframes and frames:

- ✦ You don't have to specify the size of an iframe.
- ✦ iframes are not resizeable by the user.
- ✦ Instead of a noframes tag the iframes content is displayed if the browser does not support iframes.

Server side includes

You may have noticed that some pages on the Web have the extension .shtml. That is because these pages have content that is generated or included by the Web server.

Most people use frames because they want a standard header, footer, or navigation bar on each page. You can accomplish the same thing with server side includes (SSI).

For example File: ssi-example.shtml

```
<table>
<tr><td><!--#include virtual="header.html" --></td></tr>
<tr>
<td><!--#include virtual="navegationbar.html" --></td>
<td>Content goes here</td>
</tr>
<tr><td><!--#include virtual="footer.html" --></td></tr>
</table>
```

The contents of the files are included just as if you had done a copy-and-paste operation and so shouldn't contain the `<html>`, `<header>`, or `<body>` tags, just the content that you want, as shown in the following example:

```
If you want to get really flashy, you can automatically include the
modification date of the file.
```

Just as a word of warning, you should be aware that the SSI support depends on your Web server and its configuration. You should check before designing your whole

www.it-ebooks.info

Web site using them. The apache Web server does support them but has to be configured to do so.

XML and XSL

Another alternative is to write the page content in XHTML and use XSL to form the template for the pages. This will be covered in greater detail in Chapter 29.

Avoiding Frame Problems

Even if you do everything perfectly and avoid all typographical errors and syntax blowups, there are still a few "gotchas" you need to be on the alert for.

Making sure you have enough frames

If you've been banging your head over your HTML code and just can't figure out why your frames won't display as intended, take the time to count the number of frames you've specified. Remember that the number of sizes stated in your FRAMESET elements must match up with the number of FRAME elements that it contains. If you have code like this, you'll have problems because you have an extra frame that won't display:

```
<FRAMESET rows="200,100">
<FRAME>
<FRAME>
<FRAME>
</FRAMESET>
```

And, if you have code like this, you'll have problems because you'll end up with an empty third frame:

```
<FRAMESET rows="200,100,*">
<FRAME>
<FRAME>
</FRAMESET>
```

Adding foreign elements

If you put any element that might be found in a normal Web page's BODY element—or the <BODY> tag itself—before the FRAMESET element, the frameset will not work. This should not be an issue if you create every page from scratch, but many Web designers use template pages for all their work. If the template page you're using to create your framesets contains anything at all that shouldn't be in a frameset, you're risking making the whole thing not work. If your frames aren't showing up, check the frameset page's HTML code carefully before you do anything else.



The same problem can also be caused by failing to include the </FRAMESET> end tag.

www.it-ebooks.info

What size is the user's screen?

One serious problem when using frames is that you have no idea what the dimensions are on the users' screens—640 × 480, 800 × 600, 1024 × 768, 1152 × 864, 1280 × 1024, or 1600 × 1200.

But how can you design your site so that it looks good at any screen resolution? What if you've put in a frame that isn't resizable and only displays half of the menu?

Some sites try to standardize things by giving users instructions to change their resolutions to 800 × 600. But it's unrealistic to expect users with high-end graphics workstations to halve their screen resolution just to see your site.

Most HTML is designed to resize the layout of the page depending on the size of the content and the size of the screen. Frames are an exception, You have to specify a fixed size or a size relative to the size of the browser window. However, since you don't know the size of the content of your page nor the size of the window, it's impossible to get it right every time.

Many experts on Web page design discourage the use of frames for this reason. They advocate the use of non-frame HTML. The choice is yours.

Using too many frames

It's easy to get carried away with frames, especially for those people who like to put everything into neat little boxes. And that's what frames are, right? Well, not exactly. Only a limited number of frames make a workable Web site—generally, we'd recommend no more than three to five frames on one screen.

It's important to remember that even though a framed Web site looks like a single Web page, in reality every frame is holding a different Web page. That means that, with only three frames, you're making a visitor to your site download three Web pages at once. And if each of those pages has its own graphics as well . . . You begin to see the problem? It's an issue of download time. Not everybody has a broadband connection to the Internet, and tripling or quadrupling, or quintupling the download time is enough of a burden without unnecessarily adding even more Web pages to the display.

Providing backup navigation

You've designed your Web site to use frames and, specifically, a navigation page tucked over on the side, top, or bottom of your main entry point. However, people may come into your site from other sites that have linked to an individual page instead of your entry point. This generally happens when visitors come to your site using the results from a search engine.

In that case, such visitors are never going to see your nice navigational setup, and, unless you provide at least a minimal alternative, the single page is all they'll ever see of your site. At rock bottom minimum, make sure that every page on your site has a link to your home page (the framed entry point) or you'll lose some visitors for sure.

www.it-ebooks.info

Legal challenges to framing

Framing your own pages does not create any kind of legal problem, but when you frame somebody else's pages on your own site, a bit of a risk is involved. Some commercial Web sites have objected to becoming part of another site through the use of frames, claiming that they are losing advertising revenue by being "hijacked."

So far, the difficulty seems to be limited to situations in which one site that makes its money from advertising incorporates pages from another site that also has paid advertisements. The *framee* claims that the framer is capitalizing on the framee's work, displaying it within a frame that contains ads from the framer's sponsors. The result is that the framer is making money off somebody else's copyrighted publication.

What's funny to anyone who knows even a little bit about the Web is that any page can be set so that nobody can put it into a frame. (We show you how in Chapter 16.) Certainly, if anybody cares about preventing others from displaying their pages in unauthorized frames, it's a lot cheaper to pay a JavaScript programmer for a few minutes of work than to hire a battery of lawyers and go to court.

Whatever the merits to the claims of the parties, the legalities are following the usual run of absurdities—in Web-related law, it is often lawyers who don't understand the Web who argue their cases in front of judges who don't understand the Web. While this may all seem amusing to technical people, the stark reality of the situation is that lawyers and judges create the legal framework that affects us all, and the decisions handed down by judges can be used against you.

Because many of the framed sites on the Web do display material from other Web sites, such court cases can have far-reaching impact. If you want to make sure you're covered, consider asking the Webmasters whose sites you want to frame for permission. In most cases, they'll be more than happy to cooperate. If not, then you'd better leave them out of your plans.

Summary

Frames present an alternative method for displaying Web pages. They allow you to put multiple Web pages on screen at the same time. This chapter introduced you to the following concepts and topics:

- ✦ Frames are contained within the `FRAMESET` element.
- ✦ Frames can be set up to be horizontal or vertical. The two styles can also be mixed together.
- ✦ Frames are mainly used to keep one unchanging set of information visible while other parts of the screen are allowed to change.
- ✦ Each frame should have a unique name.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 13

Styling Web Pages with Cascading Style Sheets



In This Chapter

Testing your browser

Coping with browser support issues

Cascading priorities

Redefining HTML elements

Assigning classes

Using IDs

Defining nested elements with contextual selectors

Using style sheet–creation programs

Consulting the CSS reference



Cascading style sheets (CSS) have the potential to provide Web page designers with a fabulous degree of control over the appearance and functionality of their sites. Using CSS you can specify the styles used in HTML pages for a whole Web site, per page, or for each tag. This allows the designer to separate the content from the presentation. A good idea when it comes to making changes in a 1000-page Web site.



All of the formatting instructions in HTML have been deprecated in favor of their CSS equivalents.

Using CSS you can reduce the size of your HTML files, which in turn reduces the download time improving user satisfaction as the page downloads faster. It also reduces the disk space used on both server and client and lessens the load on the server and network. All this means that your Web site can support more users.

What if you don't use CSS? Several well-known office applications create HTML files that don't use CSS. However, the same file with CSS can be reduced by half or more of its original size and so will download two to three times faster. But there are drawbacks.

First, the CSS specifications are large and complex. Second, the browser doesn't give any error messages if there are errors in the style sheet. And third, not all browsers support CSS (or if they do, sometimes not according to the specifications).

This chapter gives a general overview of how CSS works within a Web page and across a Web site. Subsequent chapters explore specific implementations of these principles and how you can exploit them to your best advantage.

www.it-ebooks.info

Testing Your Browser

The World Wide Web Consortium has an online test suite through which you can see how well the browser you're using handles selected CSS properties. The test suite is located at `www.w3.org/Style/CSS/Test`. There are three different CSS specifications, CSS1, CSS2, and CSS2 revision 1, all of which are simultaneously in effect. CSS1 was issued way back in the dim mists of Web history (1996) and CSS2 came out in 1998. CSS1 was then revised in 1999. That's right—not CSS2, but CSS1, the year after CSS2 came out. And work is underway on CSS3. It's enough to make you wish you were back in the days when people used sticks to scratch messages on clay tablets, but this is the crazy world that Web designers have to cope with today. The official Web sites for the three CSS specifications can be found at:

- CSS1 (`www.w3.org/TR/REC-CSS1`)
- CSS2 (`www.w3.org/TR/REC-CSS2`)
- CSS2 revision 1 (`http://www.w3.org/TR/2004/CR-CSS21-20040225/`)

Coping with Browser Support Issues

New browser versions, both major and minor, come out at such odd and unpredictable intervals that it is impossible for any book to keep fully up-to-date with them. Although we will document the current state of browser support for the CSS properties covered in this chapter, you need to be aware that this could all change at any time. Either Microsoft or Netscape could decide tomorrow that it will fully implement every feature of CSS1 or CSS2 and develop a crash program to have the new browser finished in a month. It's just as possible that they might ignore CSS2 entirely and go straight into adding support for CSS3, as tentative as that proposal is right now. Or perhaps both of them will ignore the CSS support problem entirely and even introduce bugs in their latest versions that invalidate earlier CSS functionality. Stranger things have been known to happen.

So how do you, as a conscientious Web designer, know what's going on now? The answer, as you've probably guessed, is on the Web itself. There are several Web sites that keep track of browser version issues, including the state of their CSS support. Table 13-1 lists some sites where you can get the latest news.

Cascading Priorities

Cascading style sheets are called that for a good, solid reason. There are four different levels at which styles can be applied. An element can have its own style using the `style` attribute; a Web page can have a global style that is controlled by an embedded style sheet created with the `STYLE` element; the styles on the same page can also be controlled by an external style sheet that is linked to that page; and the user can specify his or her own stylesheet in the browser. With four

www.it-ebooks.info

Table 13-1
CSS Support Information Sites

Web Site	*Address*
QuirksMode	www.quirksmode.org/home.html
CSS Pointers Group	www.css.nu/pointers/bugs.html
Index DOT Css	www.blooberry.com/indexdot/css/supportkey/syntax.htm
Project Cool Developer Zone	www.projectcool.com/developer/reference/css_style.html
RichInStyle.com	www.richinstyle.com/bugs
Stephan Nedregård's Web Workshop	www.micropop.com/workshop/browsers.html

different ways of styling the same element, the possibilities for conflict are practically endless.

So, how could the designers of CSS manage to leave as much flexibility as possible for Web designers, yet still manage to handle the inevitable conflicts? The solution for this was to establish a set of priorities. The lowest level style setting—an individual element's `style` attribute—has the highest priority and will override any conflicting setting in both the `STYLE` element in its containing Web page and any linked style sheet. The `STYLE` element in a Web page similarly overrides any conflict created by an external style sheet that it's linked to. The external style sheet has the lowest priority of all, and any time it creates a conflict, it loses out. This prioritization system is seen as a sort of multilevel waterfall effect, and thus the term *cascading* came into use to describe it.

Going inline with the style attribute

The simplest way to set the style for an individual element is via its `style` attribute. For example, if you wanted to make a particular `H1` element turn magenta instead of its default black, you could use the following code:

```
<H1 style="color: magenta">Everything in this heading is
 magenta-colored.</H1>
```

This example sets the CSS `color` property to a value of `magenta`. If you want to set more than one CSS property, simply separate them with a semicolon (this is known as *grouping* the properties). The following example expands on the earlier one by also changing the `H1` element so that it also uses a Helvetica or Arial font:

```
<H1 style="font-family: helvetica; color: magenta">Everything
 in this heading is magenta-colored and uses a Helvetica
 font.</H1>
```

www.it-ebooks.info

Embedding with the <STYLE> tag

Setting a CSS property for one element is child's play compared with the kind of power that CSS was meant to supply. After all, there are already native capabilities in plain old HTML that enable you to do many of the things that CSS can do—for each individual element, that is. Creating an *embedded* style sheet—one that controls the whole Web page—however, gives you quite a bit more power.

The `<STYLE>` tag enables you to establish a broader impact. For example, you could set the CSS properties for *every* `H1` element at once instead of having to set each one, one at a time, and you can save yourself an awful lot of work by just making one global setting like this:

```
<HEAD>
<STYLE>
H1 { color: magenta }
</STYLE>
</HEAD>
```



The `<STYLE>` tag has to go within the `HEAD` of the Web page.

The syntax is very similar to that used in the `style` attribute, but there is a slightly different approach, and you have to be careful to get it just right. Between the `<STYLE>` and `</STYLE>` tags, the name of the element to be styled comes first (`H1`, in this case). In CSS, it is called the *selector*. Next comes the opening curly bracket (`{`), then the name of the CSS property (`color`, in this case). The property name is separated from the value of the property (`magenta`, here) by a colon (:). Finally, you must close the style declaration with a closing curly bracket (`}`). Leave out any one of those items and the whole thing won't work.



Never treat the selector as a tag instead of an element. In other words, you cannot put angle brackets `<>` around a selector. If you used `<H1>` instead of plain `H1` as the selector, no Web browser in the world would forgive the mistake.

You can set multiple global CSS properties simultaneously, just as you can with the `style` attribute for individual elements. With the `<STYLE>` tag, however, it's customary to put each property on a separate line. This is not a requirement, but it does make it a lot easier to read and understand the HTML code.

```
<HEAD>
<STYLE>
H1 {
font-family: helvetica;
color: magenta
}
</STYLE>
</HEAD>
```

www.it-ebooks.info

Linking external style sheets

External style sheets follow the same syntax as an embedded style sheet, with one exception. There is no <STYLE> tag involved because an external style sheet simply isn't a Web page, so it doesn't use any HTML. Instead, it's a CSS file, and it ends in a .css extension. The function of a CSS file is to provide information on how the various elements on a Web page should be styled. A portion of an external style sheet might look like this:

```
H1 {
font-size: 1in;
margin-left: .5in
}

H2 {
font-size: .75in;
margin-left: .5in
}

H3 {
font-size: .5in;
margin-left: .5in
}
```

This segment specifies the size of H1 elements as one inch high, H2 elements as $\frac{3}{4}$ of an inch high, and H3 elements as $\frac{1}{2}$ inch high; it also sets all of them $\frac{1}{2}$ inch away from the left border of the Web page.

If the CSS file that held these declarations were called site_x_sheet.css, it would be attached to a Web page by the following code:

```
<HEAD>
<LINK rel=stylesheet href="site_x_sheet.css" type="text/css">
</HEAD>
```



Just like the <STYLE> tag, the <LINK> tag has to go into the HEAD of a Web page.

The best and highest use of style sheets comes when the look of an entire Web site is established and controlled by a single external style sheet. We once had a site with hundreds of pages and got tired of the look after a while, so we began to experiment a bit. We finally settled on a new appearance that we liked better and, in those days before style sheets, we had to recode each one of the affected elements on every single page by hand. If we had been able to use a single style sheet linked to each Web page in the entire site, all we would have had to do would be to make a few changes in that one CSS file, and the look of hundreds of Web pages would have changed instantly.

You can also link more than one CSS file to a single Web page, just by stacking <LINK> tags like this:

```
<HEAD>
<LINK rel=stylesheet href="site_a_sheet.css">
```

www.it-ebooks.info

```
<LINK rel=stylesheet href="site_b_sheet.css">
<LINK rel=stylesheet href="site_c_sheet.css">
<LINK rel=stylesheet href="site_d_sheet.css">
</HEAD>
```

This is simple to do, although it does have the potential to tremendously complicate things. It's not a problem for a Web browser because the rules for cascading in this case are about as easy as they can get—whichever style sheet is linked first has priority over any that come after it. The problem is that, if you have many conflicting declarations in the multiple CSS files, *you* may not be able to figure out what the results will be until you see them onscreen. And they may not all be what you had in mind.

You can also import a style sheet, for example:

```
@import url(style.css);
body { background-image: url(sky.png);}
```

In this case the imported style sheet has less priority than the style sheet that includes it.

Redefining HTML Elements

The most potent way to use CSS is to redefine how an HTML element works—not an individual instance of that element, but all occurrences of it. You can, for example, make all `P` elements have purple text. We're not sure why you'd want to, but you can do it. Or you could make every `H3` centered. Or make all images invisible. CSS gives you the power, but, like all power, it's open to either constructive use or unfettered abuse.

You may want to assign a Helvetica-style font to all your heading elements, for example. Most Web designers have done this for eons, but they had to deal with it on an individual basis, heading by heading. This is a clearly constructive use of the power of CSS. Deliberately setting all `P` elements a certain distance from the edges of the screen would be another good example of constructive use.

You could, however, just as easily make all the tables on your pages use dingbat fonts—deliberately or otherwise. Take care when using redefinition of elements to make sure that you are indeed getting the exact effect you want because doing so causes a sweeping and possibly overpowering change all across every page that's affected by the style.

With these cautions in mind, here's an example of how element redefinition works in CSS:

```
<HTML>
<HEAD>
<STYLE>
```

www.it-ebooks.info

```
P {color: purple}
</STYLE>
</HEAD>
<BODY>
<H1>This text is black.</H1>
<P>This text is purple.</P>
</BODY>
</HTML>
```

Assigning Classes

Classes are a wonderful balancing act between the unfettered power of element selectors and the old style of doing things in plain HTML. They avoid the sweeping changes dictated by redefining entire tags and enable you to limit the impact of a CSS property declaration to specified elements on your Web page.

To assign a class and apply it to different elements on your Web pages, two things are necessary: the declaration of the class in the `STYLE` section in the `HEAD` of the Web page (or in an external style sheet) and the addition of a `class` attribute to the elements you want to alter. The following example code shows how it's done:

```
<HTML>
   <HEAD>
   <STYLE>
   .class1 {font-family: Arial, Helvetica, sans-serif; font-size:
    16px}
   .class2 {color: #FFCC00}
   </STYLE>
   </HEAD>
   <BODY>
   <p class="class1">This is an example of class 1.</p>
   <p class="class2">This is an example of class 2.</p>
   </BODY>
   </HTML>
```



The class declaration must be preceded by a period (.), but the period is not used in setting the `class` attribute.

The first class declaration (class1) sets the font to a Helvetica font, 16 pixels in height. The second class declaration (class2) sets the font to a mustard yellow color, but affects it in no other way.

Using IDs

While ID selectors aren't as useful or powerful, generally speaking, as redefining an HTML element or assigning a class, they do have a place. At first glance, this seems a pretty limited capability, however. Why? Because an ID selector is usable only with a

www.it-ebooks.info

particular, individual element—not all instances of an element, but an individual use of it. For example, the following code creates an ID selector and then applies it by assigning an `id` attribute to one item on the Web page:

```
<HTML>
<HEAD>
<STYLE>
#truegreen {color: #00FF00}
</STYLE>
</HEAD>
<BODY>
<p id="truegreen">This is an example of id assignment.</p>
</BODY>
</HTML>
```



The ID declaration must be preceded by a hash mark (#), but the hash mark is not used in setting the `id` attribute.

No other item on the entire page can have the same `id` value, which means that all the power of CSS is brought to bear upon one little spot on the Web page. There are times when it can be a handy tool. For example, if you have a header, footer, or menu and you want to specify how they should be formatted on every page, it's a good idea to give each an id and then use an external style sheet to change their appearance.

Defining Nested Elements with Contextual Selectors

If one element is nested inside another, and the parent element has a CSS property set, then in most cases the child element automatically has the same setting, even if it is not specified in the child element. This phenomenon is known as *inheritance*. To illustrate how it works, consider the following example:

```
<HTML>
<HEAD>
<STYLE>
H1 {color="red"}
</STYLE>
</HEAD>
<BODY>
<H1>All Heading 1 elements <I>must</I> be red in color.</H1>
</BODY>
</HTML>
```

The `I` element, as well as the `H1` element, will be red in color. However, there are many times when you may want to override inheritance and have, instead, a specific CSS property show up. In those cases, *contextual selectors* are the key. A contextual selector is one that specifies that one element, when contained within another, is to

www.it-ebooks.info

be shown in a certain manner. For example, you may want all italicized items within Heading 1 elements to be shown in a blue color. The following code would accomplish this handily, overriding the inheritance factor:

```
<HTML>
<HEAD>
<STYLE>
H1 {color="red"}
H1 I {color="blue"}
</STYLE>
</HEAD>
<BODY>
<H1>All Heading 1 elements <I>must</I> be red in color.</H1>
</BODY>
</HTML>
```

Cool effects with CSS

Many people use JavaScript to make menus, but few people realize that you can do the same thing with CSS. For example, the following code creates a menu that has images of stars instead of dots, and as the user hovers over each link it changes color and the text grows larger (see Figure 13-1).

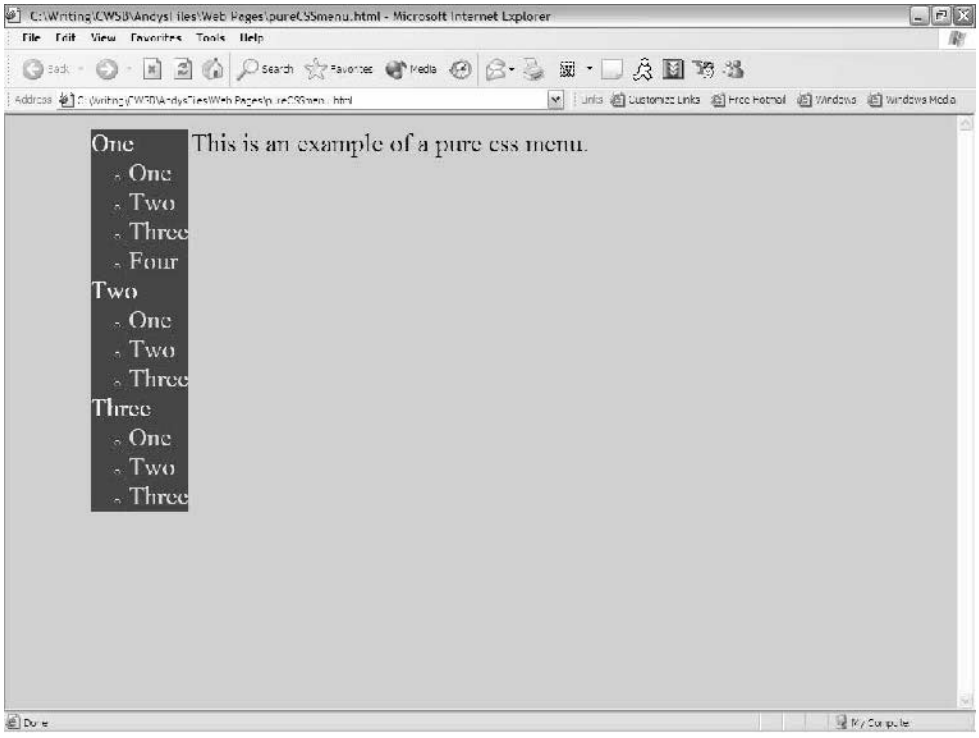


Figure 13-1: Menus can be built with CSS.

www.it-ebooks.info

```
<html>
<head>
  <link rel="stylesheet" href="hoverMenu.css" type="text/css" />
</head>
<body>
<ul id="menu">
  <li>One
         <ul>
         <li><a href="">One</a></li>
         <li><a href="">Two</a></li>
         <li><a href="">Three</a></li>
         <li><a href="">Four</a></li>
         </ul>
  </li>
  <li>Two
         <ul>
         <li><a href="">One</a></li>
         <li><a href="">Two</a></li>
         <li><a href="">Three</a></li>
         </ul>
  </li>
  <li>Three
         <ul>
         <li><a href="">One</a></li>
         <li><a href="">Two</a></li>
         <li><a href="">Three</a></li>
         </ul>
  </li>
</ul>
This is an example of a pure cssmenu.
</body>
</html>
```

The file hoverMenu.css:

```
body { background-color: lightblue;font-size: x-large; }
/* the background is lightblue with extra large fonts */
#menu { background-color:blue;float:left; }
/* the menu has a blue background and is floated to the left hand side */
#menu li { list-style-image: url(ylwstar.gif); font-size: x-large;
color:yellow;}
/* the list items in the menu have a star as the image and are in a
large yellow font */
#menu li a { color:lime; }
/* the links are lime colored */
#menu li a:hover { color:red; background-color: white; font-size:
larger; }
/* when the user hovers over a link it changes to red, the background
to white and the font grows larger */
```

Just by changing the CSS style sheet (as shown in the following code), we can make each submenu be hidden initially and appear as the user hovers over the menu items (see Figure 13-2):

www.it-ebooks.info

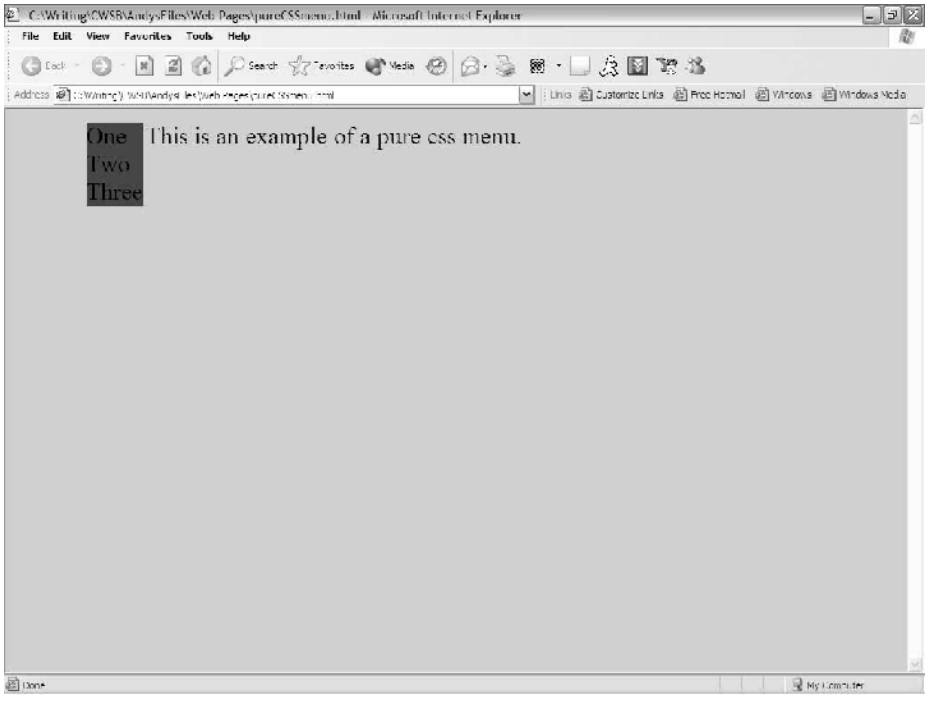


Figure 13-2: Submenus are hidden with a slight change in the style sheet.

```
body { background-color: lightblue;font-size: x-large; }
/* the background is lightblue with extra large fonts */
#menu { background-color:blue;float:left; }
/* the menu has a blue background and is floated to the left hand side */
#menu li { font-size: x-large; color; yellow; }
/* the list items in the menu are in a large yellow font */
#menu li:hover { color:red; }
/* list items change to red when the mouse passes over*/
#menu li a { color:lime; }
/* the links are lime colored */
#menu li a:hover { color:red; background-color: white; }
/* the links change to red with a white background when the mouse passes
over */
#menu li ul {display:none;position:absolute;background-color:blue; }
/* submenus are not displayed initially, absolutely positioned, and have a
blue background */
#menu li:hover ul { display:block; }
/* when the mouse passes over a list item its submenu is displayed */
```

Consulting the CSS Reference

The CSS1 specification is the most widely supported one among the major Web browsers. However, many CSS2 properties, such as absolute positioning, are also important.

www.it-ebooks.info

CSS1 properties

We have compiled the information in this section from contents in the World Wide Web Consortium's CSS1 specification, originally dated December 17, 1996, and revised January 11, 1999. This specification carries the status of a W3C Recommendation. The URL of the specification is `www.w3.org/TR/REC-CSS1`. Although the specification carries no specific copyright notice, we assume that the usual requirement of adding the following notice to information taken from W3C documents still applies:

> Copyright © World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved.

The properties in the following listing are specified by name (in alphabetical order) followed by this information:

- **Value:** Shows all possible values that can be used with this property
- **Initial:** Shows the default value if none is specified
- **Applies to:** Shows the elements the property can be applied to
- **Inherited:** Shows if the property can be inherited
- **Percentage values:** Shows how percentages are handled

Here is the list of CSS1 properties:

Background

Value: <background-color> || <background-image> || <background-repeat> || <background-attachment> || <background-position>

Initial: Not defined for shorthand properties

Applies to: All elements

Inherited: No

Percentage values: Allowed on <background-position>

background-attachment

Value: scroll | fixed

Initial: scroll

Applies to: All elements

Inherited: No

Percentage values: N/A

background-color

Value: <color> | transparent

www.it-ebooks.info

Initial: transparent

Applies to: All elements

Inherited: No

Percentage values: N/A

background-image

Value: <url> | none

Initial: None

Applies to: All elements

Inherited: No

Percentage values: N/A

background-position

Value: [<percentage> | <length>]{1,2} | [top | center | bottom] || [left | center | right]

Initial: 0% 0%

Applies to: Block-level and replaced elements

Inherited: No

Percentage values: Refer to the size of the element itself

background-repeat

Value: repeat | repeat-x | repeat-y | no-repeat

Initial: repeat

Applies to: All elements

Inherited: No

Percentage values: N/A

border

Value: <border-width> || <border-style> || <color>

Initial: Not defined for shorthand properties

Applies to: All elements

Inherited: No

Percentage values: N/A

border-color

Value: <color>{1,4}

Initial: The value of the 'color' property

Applies to: All elements

www.it-ebooks.info

Inherited: No

Percentage values: N/A

border-style

Value: None | dotted | dashed | solid | double | groove | ridge | inset | outset

Initial: None

Applies to: All elements

Inherited: No

Percentage values: N/A

border-top, border-right, border-bottom, border-left

Value: <border-top-width> || <border-style> || <color>

Initial: Not defined for shorthand properties

Applies to: All elements

Inherited: No

Percentage values: N/A

border-width

Value: [thin | medium | thick | <length>]{1,4}

Initial: Not defined for shorthand properties

Applies to: All elements

Inherited: No

Percentage values: N/A

border-top-width, border-right-width, border-bottom-width, border-left-width

Value: thin | medium | thick | <length>

Initial: 'medium'

Applies to: All elements

Inherited: No

Percentage values: N/A

clear

Value: none | left | right | both

Initial: None

Applies to: All elements

www.it-ebooks.info

Inherited: No

Percentage values: N/A

color

Value: <color>

Initial: UA specific

Applies to: All elements

Inherited: Yes

Percentage values: N/A

display

Value: block | inline | list-item | none

Initial: block

Applies to: All elements

Inherited: No

Percentage values: N/A

float

Value: left | right | none

Initial: None

Applies to: All elements

Inherited: No

Percentage values: N/A

font

Value: [<font-style> || <font-variant> || <font-weight>]? <font-size> [/ <line-height>]? <font-family>

Initial: Not defined for shorthand properties

Applies to: All elements

Inherited: Yes

Percentage values: Allowed on <font-size> and <line-height>

font-family

Value: [[<family-name> | <generic-family>],]* [<family-name> | <generic-family>]

Initial: UA specific

Applies to: All elements

Inherited: Yes

Percentage values: N/A

www.it-ebooks.info

font-size

Value: <absolute-size> | <relative-size> | <length> | <percentage>

Initial: medium

Applies to: All elements

Inherited: Yes

Percentage values: Relative to parent element's font size

font-style

Value: normal | italic | oblique

Initial: normal

Applies to: All elements

Inherited: Yes

Percentage values: N/A

font-variant

Value: normal | small-caps

Initial: normal

Applies to: All elements

Inherited: Yes

Percentage values: N/A

font-weight

Value: normal | bold | bolder | lighter | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900

Initial: normal

Applies to: All elements

Inherited: Yes

Percentage values: N/A

height

Value: <length> | auto

Initial: auto

Applies to: Block-level and replaced elements

Inherited: No

Percentage values: N/A

letter-spacing

Value: normal | <length>

www.it-ebooks.info

Initial: normal

Applies to: All elements

Inherited: Yes

Percentage values: N/A

line-height

Value: normal | <number> | <length> | <percentage>

Initial: Normal

Applies to: All elements

Inherited: Yes

Percentage values: Relative to the font size of the element itself

list-style

Value: [disc | circle | square | decimal | lower-roman | upper-roman | lower-alpha | upper-alpha | none] || [inside | outside] || [<url> | none]

Initial: Not defined for shorthand properties

Applies to: Elements with 'display' value 'list-item'

Inherited: Yes

Percentage values: N/A

list-style-image

Value: <url> | none

Initial: none

Applies to: Elements with 'display' value 'list-item'

Inherited: Yes

Percentage values: N/A

list-style-position

Value: inside | outside

Initial: outside

Applies to: Elements with 'display' value 'list-item'

Inherited: Yes

Percentage values: N/A

list-style-type

Value: disc | circle | square | decimal | lower-roman | upper-roman | lower-alpha | upper-alpha | none

Initial: disc

www.it-ebooks.info

Applies to: Elements with 'display' value 'list-item'

Inherited: Yes

Percentage values: N/A

margin

Value: [<length> | <percentage> | auto]{1,4}

Initial: Not defined for shorthand properties

Applies to: All elements

Inherited: No

Percentage values: Refer to width of closest block-level ancestor

margin-top, margin-right, margin-bottom, margin-left

Value: <length> | <percentage> | auto

Initial: 0

Applies to: All elements

Inherited: No

Percentage values: Refer to width of the closest block-level ancestor

padding

Value: [<length> | <percentage>]{1,4}

Initial: Not defined for shorthand properties

Applies to: All elements

Inherited: No

Percentage values: Refer to width of closest block-level ancestor

padding-top, padding-right, padding-bottom, padding-left

Value: <length> | <percentage>

Initial: 0

Applies to: All elements

Inherited: No

Percentage values: Refer to width of closest block-level ancestor

text-align

Value: left | right | center | justify

Initial: UA specific

Applies to: Block-level elements

www.it-ebooks.info

Inherited: Yes

Percentage values: N/A

text-decoration

Value: none | [underline || overline || line-through || blink]

Initial: none

Applies to: All elements

Inherited: No, but elements should match their parent

Percentage values: N/A

text-indent

Value: <length> | <percentage>

Initial: 0

Applies to: Block-level elements

Inherited: Yes

Percentage values: Refer to parent element's width

text-transform

Value: capitalize | uppercase | lowercase | none

Initial: none

Applies to: All elements

Inherited: Yes

Percentage values: N/A

vertical-align

Value: baseline | sub | super | top | text-top | middle | bottom | text-bottom | <percentage>

Initial: baseline

Applies to: Inline elements

Inherited: No

Percentage values: Refer to the 'line-height' of the element itself

white-space

Value: normal | pre | nowrap

Initial: normal

Applies to: Block-level elements

Inherited: Yes

Percentage values: N/A

www.it-ebooks.info

width

Value: <length> | <percentage> | auto

Initial: auto

Applies to: Block-level and replaced elements

Inherited: No

Percentage values: Refer to parent element's width

word-spacing

Value: normal | <length>

Initial: normal

Applies to: All elements

Inherited: Yes

Percentage values: N/A

CSS2 properties

This section contains the official list of all CSS properties from the World Wide Web Consortium's CSS2 specification, which carries the status of a W3C Recommendation. The URL of the specification is `www.w3.org/TR/REC-CSS2`. Although the specification carries no specific copyright notice, we assume that the usual requirement of adding the following notice to information taken from W3C documents still applies:

> Copyright © World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved.

The properties in the following listing are specified by name (in alphabetical order), and then the following information is presented:

- **Values:** Shows all possible values that can be used with this property
- **Initial value:** Shows the default value if none is specified
- **Applies to:** Shows the elements the property can be applied to
- **Inherited:** Shows if the property can be inherited
- **Percentages:** Shows how percentages are handled
- **Media groups:** Shows which media group (visual, aural, print, and so on) the property is intended for

Here is the list of CSS2 properties:

www.it-ebooks.info

Azimuth

Values: <angle> | [[left-side | far-left | left | center-left | center | center-right | right | far-right | right-side] || behind] | leftwards | rightwards | inherit

Initial value: center

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: aural

background

Values: ['background-color' || 'background-image' || 'background-repeat' || 'background-attachment' || 'background-position'] | inherit

Initial value: XX

Applies to: All

Inherited: No

Percentages: Allowed on 'background-position'

Media groups: visual

background-attachment

Values: scroll | fixed | inherit

Initial value: scroll

Applies to: All

Inherited: No

Percentages: N/A

Media groups: visual

background-color

Values: <color> | transparent | inherit

Initial value: transparent

Applies to: All

Inherited: No

Percentages: N/A

Media groups: visual

background-image

Values: <uri> | none | inherit

Initial value: none

www.it-ebooks.info

Applies to: All

Inherited: No

Percentages: N/A

Media groups: visual

background-position

Values: [[<percentage> | <length>]{1,2} | [[top | center | bottom] || [left | center | right]]] | inherit

Initial value: 0% 0%

Applies to: Block-level and replaced elements

Inherited: No

Percentages: Refer to the size of the box itself

Media groups: visual

background-repeat

Values: repeat | repeat-x | repeat-y | no-repeat | inherit

Initial value: repeat

Applies to: All

Inherited: No

Percentages: N/A

Media groups: visual

border

Values: ['border-width' || 'border-style' || <color>] | inherit

Initial value: See individual properties

Applies to: All

Inherited: No

Percentages: N/A

Media groups: visual

border-collapse

Values: collapse | separate | inherit

Initial value: collapse

Applies to: 'table' and 'inline-table' elements

Inherited: Yes

Percentages: N/A

Media groups: visual

www.it-ebooks.info

border-color

Values: <color>{1,4} | transparent | inherit

Initial value: See individual properties

Applies to: All

Inherited: No

Percentages: N/A

Media groups: visual

border-spacing

Values: <length> <length>? | inherit

Initial value: 0

Applies to: 'table' and 'inline-table' elements

Inherited: Yes

Percentages: N/A

Media groups: visual

border-style

Values: <border-style>{1,4} | inherit

Initial value: See individual properties

Applies to: All

Inherited: No

Percentages: N/A

Media groups: visual

border-top, border-right, border-bottom, border-left

Values: ['border-top-width' || 'border-style' || <color>] | inherit

Initial value: See individual properties

Applies to: All

Inherited: No

Percentages: N/A

Media groups: visual

border-top-color, border-right-color, border-bottom-color, border-left-color

Values: <color> | inherit

Initial value: The value of the 'color' property

Applies to: All

www.it-ebooks.info

Inherited: No

Percentages: N/A

Media groups: visual

border-top-style, border-right-style, border-bottom-style, border-left-style

Values: <border-style> | inherit

Initial value: None

Applies to: All

Inherited: No

Percentages: N/A

Media groups: visual

border-top-width, border-right-width, border-bottom-width, border-left-width

Values: <border-width> | inherit

Initial value: medium

Applies to: All

Inherited: No

Percentages: N/A

Media groups: visual

border-width

Values: <border-width>{1,4} | inherit

Initial value: See individual properties

Applies to: All

Inherited: No

Percentages: N/A

Media groups: visual

bottom

Values: <length> | <percentage> | auto | inherit

Initial value: auto

Applies to: Positioned elements

Inherited: No

Percentages: Refer to height of containing block

Media groups: visual

www.it-ebooks.info

caption-side

Values: top | bottom | left | right | inherit

Initial value: top

Applies to: 'table-caption' elements

Inherited: Yes

Percentages: N/A

Media groups: visual

clear

Values: none | left | right | both | inherit

Initial value: none

Applies to: Block-level elements

Inherited: No

Percentages: N/A

Media groups: visual

clip

Values: <shape> | auto | inherit

Initial value: auto

Applies to: Block-level and replaced elements

Inherited: No

Percentages: N/A

Media groups: visual

color

Values: <color> | inherit

Initial value: Depends on user agent

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: visual

content

Values: [<string> | <uri> | <counter> | attr(X) | open-quote | close-quote | no-open-quote | no-close-quote]+ | inherit

Initial value: Empty string

Applies to: :before and :after pseudo-elements

www.it-ebooks.info

Inherited: No

Percentages: N/A

Media groups: all

counter-increment

Values: [<identifier> <integer>?]+ | none | inherit

Initial value: none

Applies to: All

Inherited: No

Percentages: N/A

Media groups: all

counter-reset

Values: [<identifier> <integer>?]+ | none | inherit

Initial value: none

Applies to: All

Inherited: No

Percentages: N/A

Media groups: all

cue

Values: ['cue-before' || 'cue-after'] | inherit

Initial value: XX

Applies to: All

Inherited: No

Percentages: N/A

Media groups: aural

cue-after

Values: <uri> | none | inherit

Initial value: none

Applies to: All

Inherited: No

Percentages: N/A

Media groups: aural

www.it-ebooks.info

cue-before

Values: <uri> | none | inherit

Initial value: none

Applies to: All

Inherited: No

Percentages: N/A

Media groups: aural

cursor

Values: [[<uri> ,]* [auto | crosshair | default | pointer | move | e-resize | ne-resize | nw-resize | n-resize | se-resize | sw-resize | s-resize | w-resize| text | wait | help]] | inherit

Initial value: auto

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: visual, interactive

direction

Values: ltr | rtl | inherit

Initial value: ltr

Applies to: All elements, but see prose

Inherited: Yes

Percentages: N/A

Media groups: visual

display

Values: inline | block | list-item | run-in | compact | marker | table | inline-table | table-row-group | table-header-group | table-footer-group | table-row | table-column-group | table-column | table-cell | table-caption | none | inherit

Initial value: inline

Applies to: All

Inherited: No

Percentages: N/A

Media groups: all

elevation

Values: <angle> | below | level | above | higher | lower | inherit

www.it-ebooks.info

Initial value: level

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: aural

empty-cells

Values: show | hide | inherit

Initial value: show

Applies to: 'table-cell' elements

Inherited: Yes

Percentages: N/A

Media groups: visual

float

Values: left | right | none | inherit

Initial value: none

Applies to: All but positioned elements and generated content

Inherited: No

Percentages: N/A

Media groups: visual

font

Values: [['font-style' || 'font-variant' || 'font-weight']? 'font-size' [/ 'line-height']? 'font-family'] | caption | icon | menu | message-box | small-caption | status-bar | inherit

Initial value: See individual properties

Applies to: All

Inherited: Yes

Percentages: Allowed on 'font-size' and 'line-height'

Media groups: visual

font-family

Values: [[<family-name> | <generic-family>],]* [<family-name> | <generic-family>] | inherit

Initial value: Depends on user agent

Applies to: All

www.it-ebooks.info

Inherited: Yes

Percentages: N/A

Media groups: visual

font-size

Values: <absolute-size> | <relative-size> | <length> | <percentage> | inherit

Initial value: medium

Applies to: All

Inherited: Yes, the computed value is inherited

Percentages: Refer to parent element's font size

Media groups: visual

font-size-adjust

Values: <number> | none | inherit

Initial value: none

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: visual

font-stretch

Values: normal | wider | narrower | ultra-condensed | extra-condensed | condensed | semi-condensed | semi-expanded | expanded | extra-expanded | ultra-expanded | inherit

Initial value: normal

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: visual

font-style

Values: normal | italic | oblique | inherit

Initial value: normal

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: visual

www.it-ebooks.info

font-variant

Values: normal | small-caps | inherit

Initial value: normal

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: visual

font-weight

Values: normal | bold | bolder | lighter | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900 | inherit

Initial value: normal

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: visual

height

Values: <length> | <percentage> | auto | inherit

Initial value: auto

Applies to: All elements but non-replaced inline elements, table columns, and column groups

Inherited: No

Percentages: See prose

Media groups: visual

left

Values: <length> | <percentage> | auto | inherit

Initial value: auto

Applies to: Positioned elements

Inherited: No

Percentages: Refer to width of containing block

Media groups: visual

letter-spacing

Values: normal | <length> | inherit

Initial value: normal

Applies to: All

www.it-ebooks.info

Inherited: Yes

Percentages: N/A

Media groups: visual

line-height

Values: normal | <number> | <length> | <percentage> | inherit

Initial value: normal

Applies to: All

Inherited: Yes

Percentages: Refer to the font size of the element itself

Media groups: visual

list-style

Values: ['list-style-type' || 'list-style-position' || 'list-style-image'] | inherit

Initial value: XX

Applies to: Elements with 'display: list-item'

Inherited: Yes

Percentages: N/A

Media groups: visual

list-style-image

Values: <uri> | none | inherit

Initial value: none

Applies to: Elements with 'display: list-item'

Inherited: Yes

Percentages: N/A

Media groups: visual

list-style-position

Values: inside | outside | inherit

Initial value: outside

Applies to: Elements with 'display: list-item'

Inherited: Yes

Percentages: N/A

Media groups: visual

www.it-ebooks.info

list-style-type

Values: disc | circle | square | decimal | decimal-leading-zero | lower-roman | upper-roman | lower-greek | lower-alpha | lower-latin | upper-alpha | upper-latin | hebrew | armenian | georgian | cjk-ideographic | hiragana | katakana | hiragana-iroha | katakana-iroha | none | inherit

Initial value: disc

Applies to: Elements with 'display: list-item'

Inherited: Yes

Percentages: N/A

Media groups: visual

margin

Values: <margin-width>{1,4} | inherit

Initial value: XX

Applies to: All

Inherited: No

Percentages: Refer to width of containing block

Media groups: visual

margin-top, margin-right, margin-bottom, margin-left

Values: <margin-width> | inherit

Initial value: 0

Applies to: All

Inherited: No

Percentages: Refer to width of containing block

Media groups: visual

marker-offset

Values: <length> | auto | inherit

Initial value: auto

Applies to: Elements with 'display: marker'

Inherited: No

Percentages: N/A

Media groups: visual

marks

Values: [crop || cross] | none | inherit

Initial value: none

www.it-ebooks.info

Applies to: Page context

Inherited: N/A

Percentages: N/A

Media groups: visual, paged

max-height

Values: <length> | <percentage> | none | inherit

Initial value: none

Applies to: All elements except non-replaced inline elements and table elements

Inherited: No

Percentages: Refer to height of containing block

Media groups: visual

max-width

Values: <length> | <percentage> | none | inherit

Initial value: none

Applies to: All elements except non-replaced inline elements and table elements

Inherited: No

Percentages: Refer to width of containing block

Media groups: visual

min-height

Values: <length> | <percentage> | inherit

Initial value: 0

Applies to: All elements except non-replaced inline elements and table elements

Inherited: No

Percentages: Refer to height of containing block

Media groups: visual

min-width

Values: <length> | <percentage> | inherit

Initial value: Dependent on user agent

Applies to: All elements except non-replaced inline elements and table elements

Inherited: No

Percentages: Refer to width of containing block

Media groups: visual

www.it-ebooks.info

orphans

Values: <integer> | inherit

Initial value: 2

Applies to: Block-level elements

Inherited: Yes

Percentages: N/A

Media groups: visual, paged

outline

Values: ['outline-color' || 'outline-style' || 'outline-width'] | inherit

Initial value: See individual properties

Applies to: All

Inherited: No

Percentages: N/A

Media groups: visual, interactive

outline-color

Values: <color> | invert | inherit

Initial value: invert

Applies to: All

Inherited: No

Percentages: N/A

Media groups: visual, interactive

outline-style

Values: <border-style> | inherit

Initial value: none

Applies to: All

Inherited: No

Percentages: N/A

Media groups: visual, interactive

outline-width

Values: <border-width> | inherit

Initial value: medium

Applies to: All

Inherited: No

www.it-ebooks.info

Percentages: N/A

Media groups: visual, interactive

overflow

Values: visible | hidden | scroll | auto | inherit

Initial value: visible

Applies to: Block-level and replaced elements

Inherited: No

Percentages: N/A

Media groups: visual

padding

Values: <padding-width>{1,4} | inherit

Initial value: XX

Applies to: All

Inherited: No

Percentages: Refer to width of containing block

Media groups: visual

padding-top, padding-right, padding-bottom, padding-left

Values: <padding-width> | inherit

Initial value: 0

Applies to: All

Inherited: No

Percentages: Refer to width of containing block

Media groups: visual

page

Values: <identifier> | auto

Initial value: auto

Applies to: Block-level elements

Inherited: Yes

Percentages: N/A

Media groups: visual, paged

page-break-after

Values: auto | always | avoid | left | right | inherit

Initial value: auto

www.it-ebooks.info

Applies to: Block-level elements

Inherited: No

Percentages: N/A

Media groups: visual, paged

page-break-before

Values: auto | always | avoid |left | right | inherit

Initial value: auto

Applies to: Block-level elements

Inherited: No

Percentages: N/A

Media groups: visual, paged

page-break-inside

Values: avoid | auto | inherit

Initial value: auto

Applies to: Block-level elements

Inherited: Yes

Percentages: N/A

Media groups: visual, paged

pause

Values: [[<time> | <percentage>]{1,2}] | inherit

Initial value: Depends on user agent

Applies to: All

Inherited: No

Percentages: See descriptions of 'pause-before' and 'pause-after'

Media groups: aural

pause-after

Values: <time> | <percentage> | inherit

Initial value: Depends on user agent

Applies to: All

Inherited: No

Percentages: See prose

Media groups: aural

www.it-ebooks.info

pause-before

Values: <time> | <percentage> | inherit

Initial value: Depends on user agent

Applies to: All

Inherited: No

Percentages: See prose

Media groups: aural

pitch

Values: <frequency> | x-low | low | medium | high | x-high | inherit

Initial value: medium

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: aural

pitch-range

Values: <number> | inherit

Initial value: 50

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: aural

play-during

Values: <uri> mix? repeat? | auto | none | inherit

Initial value: Auto

Applies to: All

Inherited: No

Percentages: N/A

Media groups: aural

position

Values: static | relative | absolute | fixed | inherit

Initial value: static

Applies to: All elements, but not to generated content

Inherited: No

www.it-ebooks.info

Percentages: N/A

Media groups: visual

quotes

Values: [<string> <string>]+ | none | inherit

Initial value: Depends on user agent

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: visual

richness

Values: <number> | inherit

Initial value: 50

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: aural

right

Values: <length> | <percentage> | auto | inherit

Initial value: auto

Applies to: Positioned elements

Inherited: No

Percentages: Refer to width of containing block

Media groups: visual

size

Values: <length>{1,2} | auto | portrait | landscape | inherit

Initial value: auto

Applies to: The page context

Inherited: N/A

Percentages: N/A

Media groups: visual, paged

speak

Values: normal | none | spell-out | inherit

www.it-ebooks.info

Initial value: normal

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: aural

speak-header

Values: once | always | inherit

Initial value: once

Applies to: Elements that have table header information

Inherited: Yes

Percentages: N/A

Media groups: aural

speak-numeral

Values: digits | continuous | inherit

Initial value: continuous

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: aural

speak-punctuation

Values: code | none | inherit

Initial value: none

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: aural

speech-rate

Values: <number> | x-slow | slow | medium | fast | x-fast | faster | slower | inherit

Initial value: medium

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: aural

www.it-ebooks.info

stress

Values: <number> | inherit

Initial value: 50

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: aural

table-layout

Values: auto | fixed | inherit

Initial value: auto

Applies to: 'table' and 'inline-table' elements

Inherited: No

Percentages: N/A

Media groups: visual

text-align

Values: left | right | center | justify | <string> | inherit

Initial value: Depends on user agent and writing direction

Applies to: Block-level elements

Inherited: Yes

Percentages: N/A

Media groups: visual

text-decoration

Values: none | [underline || overline || line-through || blink] | inherit

Initial value: none

Applies to: All

Inherited: No (see prose)

Percentages: N/A

Media groups: visual

text-indent

Values: <length> | <percentage> | inherit

Initial value: 0

Applies to: Block-level elements

Inherited: Yes

www.it-ebooks.info

Percentages: Refer to width of containing block

Media groups: visual

text-shadow

Values: none | [<color> || <length> <length> <length>? ,]* [<color> || <length> <length> <length>?] | inherit

Initial value: none

Applies to: All

Inherited: No (see prose)

Percentages: N/A

Media groups: visual

text-transform

Values: capitalize | uppercase | lowercase | none | inherit

Initial value: none

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: visual

top

Values: <length> | <percentage> | auto | inherit

Initial value: auto

Applies to: Positioned elements

Inherited: No

Percentages: Refer to height of containing block

Media groups: visual

unicode-bidi

Values: normal | embed | bidi-override | inherit

Initial value: normal

Applies to: All elements, but see prose

Inherited: No

Percentages: N/A

Media groups: visual

www.it-ebooks.info

vertical-align

Values: baseline | sub | super | top | text-top | middle | bottom | text-bottom | <percentage> | <length> | inherit

Initial value: baseline

Applies to: Inline-level and 'table-cell' elements

Inherited: No

Percentages: Refer to the 'line-height' of the element itself

Media groups: visual

visibility

Values: visible | hidden | collapse | inherit

Initial value: inherit

Applies to: All

Inherited: No

Percentages: N/A

Media groups: visual

voice-family

Values: [[<specific-voice> | <generic-voice>],]* [<specific-voice> | <generic-voice>] | inherit

Initial value: Depends on user agent

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: aural

volume

Values: <number> | <percentage> | silent | x-soft | soft | medium | loud | x-loud | inherit

Initial value: medium

Applies to: All

Inherited: Yes

Percentages: Refer to inherited value

Media groups: aural

white-space

Values: normal | pre | nowrap | inherit

Initial value: normal

www.it-ebooks.info

Applies to: Block-level elements

Inherited: Yes

Percentages: N/A

Media groups: visual

widows

Values: <integer> | inherit

Initial value: 2

Applies to: Block-level elements

Inherited: Yes

Percentages: N/A

Media groups: visual, paged

width

Values: <length> | <percentage> | auto | inherit

Initial value: auto

Applies to: All elements but non-replaced inline elements, table rows, and row groups

Inherited: No

Percentages: Refer to width of containing block

Media groups: visual

word-spacing

Values: normal | <length> | inherit

Initial value: normal

Applies to: All

Inherited: Yes

Percentages: N/A

Media groups: visual

z-index

Values: auto | <integer> | inherit

Initial value: auto

Applies to: Positioned elements

Inherited: No

Percentages: N/A

Media groups: visual

www.it-ebooks.info

Using Style Sheet–Creation Programs

No matter what approach you take to Web page creation—hardnosed hand-coding, WYSIWYG prototyping, or a combination of the two—you can always make life and work a little bit easier. Top-of-the-line Web page–creation programs such as Dreamweaver and GoLive have built-in capabilities for handling the creation of cascading style sheets—both internal and external—as well as Web pages (see Figure 13-3).

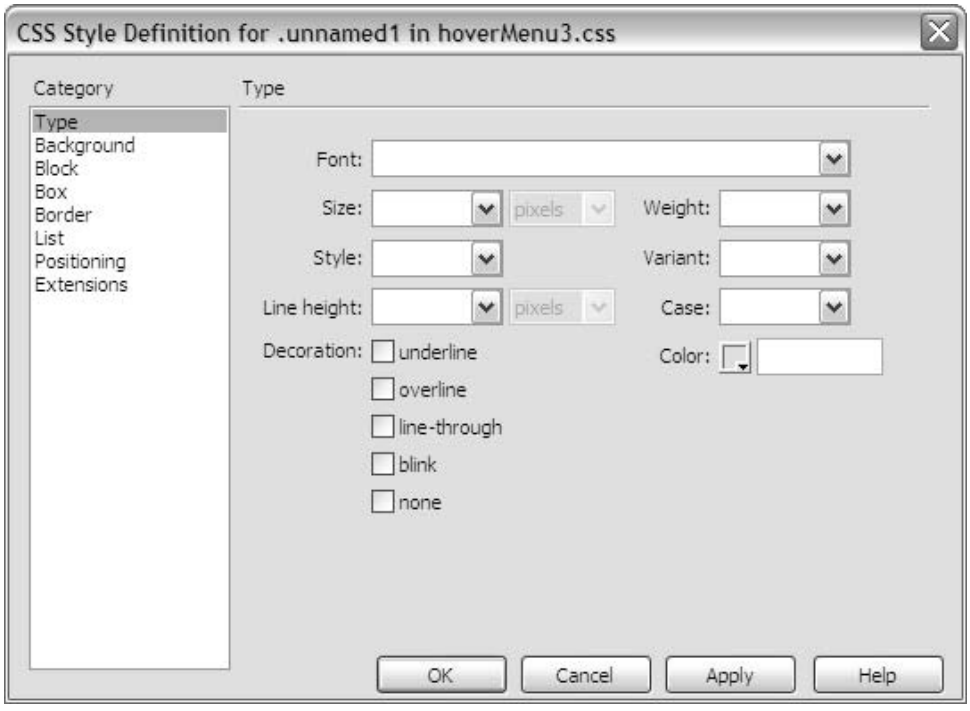


Figure 13-3: Dreamweaver has a built-in capability to generate style sheets.

However, you can also find a number of specialized programs for making cascading style sheets. Table 13-2 lists two of them you may want to try out.

Table 13-2
Style Sheet Creation Programs

Name	*Platform*	*Web Address*
Style Master	Windows & Macintosh	`www.westciv.com/style_master`
StylePro	Windows	`www.leovdb.dircon.co.uk`

www.it-ebooks.info

Interesting CSS sites

If you want to find out more about CSS there are many good sites on Internet.

Table 13-3
List of useful CSS sites

Website	*URL*
Complete CSS Guide	`www.westciv.com/style_master/academy/css_tutorial/`
CSS Discuss	`css-discuss.incutio.com`
CSS Pointers Group	`css.nu`
CSS Tutorial	`www.w3schools.com/css/`
CSS Vault	`cssvault.com`
CSS Zen Garden	`csszengarden.com`
Web Design Group	`www.htmlhelp.com/reference/css/`

CSS Zen Garden (`www.csszengarden.com/`) is a great site that uses different CSS style sheets to transform the same HTML page in artistic ways. It shows the true potential of CSS to transform the same html into completely different visual formats. Also there are some good links to CSS tricks and techniques.

Summary

CSS (Cascading Style Sheets) is one of the most promising Web technologies of all.

This chapter introduced you to the following concepts and topics:

- Style specifications "cascade" in importance, with the most important being the style closest to the element and the external style sheet having the lowest priority.
- External style sheets may be linked to several different Web pages simultaneously, thus creating overall stylistic control of an entire site.
- The appearance of HTML elements can be redefined via CSS.
- Any element may become part of a CSS class, thus redefining specified attributes.
- Any element relationship may be specified via contextual selectors.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 14

Positioning Elements with DIVs

In This Chapter

Adding layers

Overlapping layers

Making layers invisible

Clipping layer content

Handling large elements with overflow attributes

The capability to position layers on Web pages gives Web designers a fabulous amount of control over the appearance of their pages. The kind of layout control that previously existed only in print publications is now available for Web sites.

Adding Layers

In theory, any element on a Web page can be positioned on its own. In practice, however, you'll find that implementation of Cascading Style Sheet (CSS) properties is spotty at best, and that most Web browsers in use today do not fully support positioning. To play it safe, simply encapsulate whatever you want to position within a layer defined by a DIV element.



Netscape introduced a layer tag as an extension to HTML to position content. However, the tag is only supported by Netscape Navigator 4.x. The layer tag has been replaced by the DIV tag in all modern browsers, but the original terminology stuck, so you will often hear DIVs referred to as layers.

Using CSS positioning properties

The following HTML code positions a layer defined by a DIV element 20 pixels from the top of the Web page and 80 pixels from its left side (see Figure 14-1):

```
<HTML>
<HEAD>
<TITLE>
Positioning a Layer
```

www.it-ebooks.info

```
</TITLE>
</HEAD>

<BODY>

<DIV
style="position:absolute;top:20;left:80;width:200;height:200;background-
color:blue;">
</DIV>

</BODY>

</HTML>
```

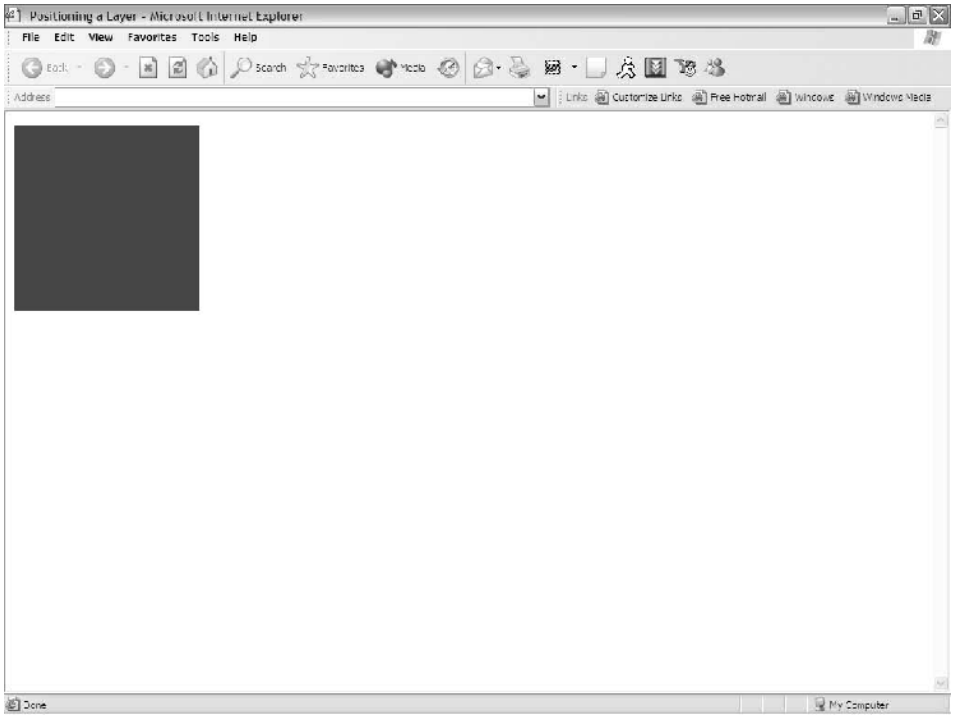


Figure 14-1: Absolute positioning allows full control over the placement of layers.



Giving layers negative values for the top and left attributes can move it off the screen. Of course, the values have to be at least as large as the height or width of the layer, or a portion of it will stick out from the edge into the visible area of the Web page.

Going with the flow

To really understand the positions in HTML, you have to understand that everything is formatted as if it were surrounded by a box. These boxes are then stacked side by

www.it-ebooks.info

side (inline formatting) or one beneath the other (block formatting). Elements are classed as inline or block. The elements "span, u,b,i" are inline and are placed inline while "div,p,img" are block elements and are block formatted.



See Chapter 3 for details of the formatting of HTML elements.

For example, normal text is formatted inline, **bold text** as well, and superscripts too. However a paragraph is a block element and is placed below the preceding block. Notice that the elements flow across or down the page.

Normal flow is the position where the next element would be positioned by default. However, this can be changed by the use of positioning instructions. To complicate matters, a block can be nested inside a block—for example, an image inside a paragraph.

Choosing absolute or relative positioning

Absolute positioning is based on the upper-left corner of a parent block. Relative positioning, on the other hand, is based on the place where the element would have been placed by default, according to the normal flow. Also absolutely positioned elements don't affect the normal flow. Consider the following HTML code:

```
<HTML>
<HEAD>
<TITLE>
Absolute versus Relative Positioning
</TITLE>
</HEAD>

<BODY style="margin: 0 0 0 0">

<DIV
style="position:absolute;top:50;left:30;width:200;height:200;background-
color:green;">
</DIV>

<DIV
style="position:relative;top:50;left:30;width:200;height:200;background-
color:blue;">
</DIV>

</BODY>

</HTML>
```

Notice in Figure 14-2 that the last block obscures the previous one. If the order is changed then you will only see the green block.

Note that the margin for the body has been set to zero. If the statement `style="margin: 0 0 0 0"` is removed then we see something like Figure 14-3.

www.it-ebooks.info

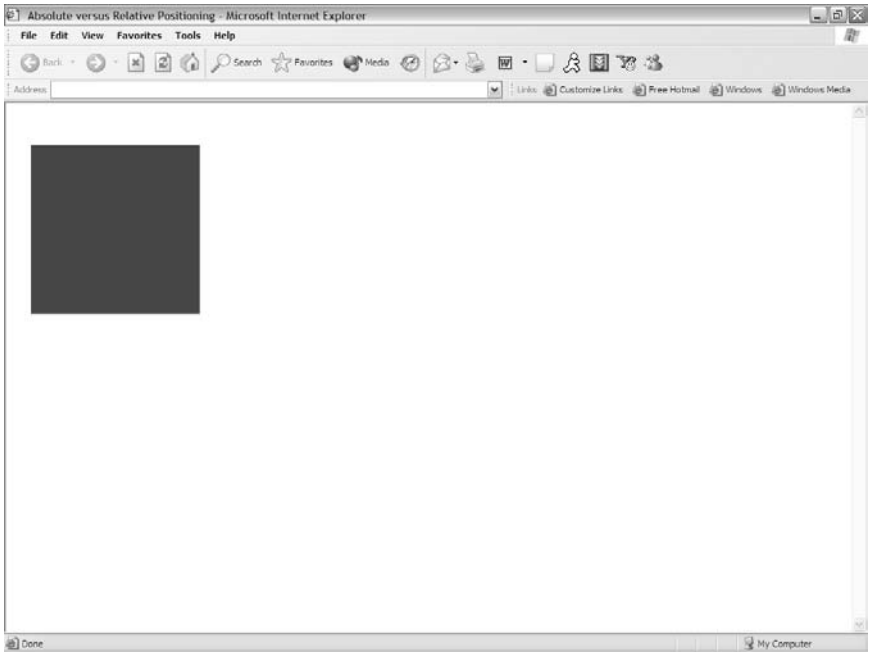


Figure 14-2: Relative positioning is based on the place where the element would have been placed by default.

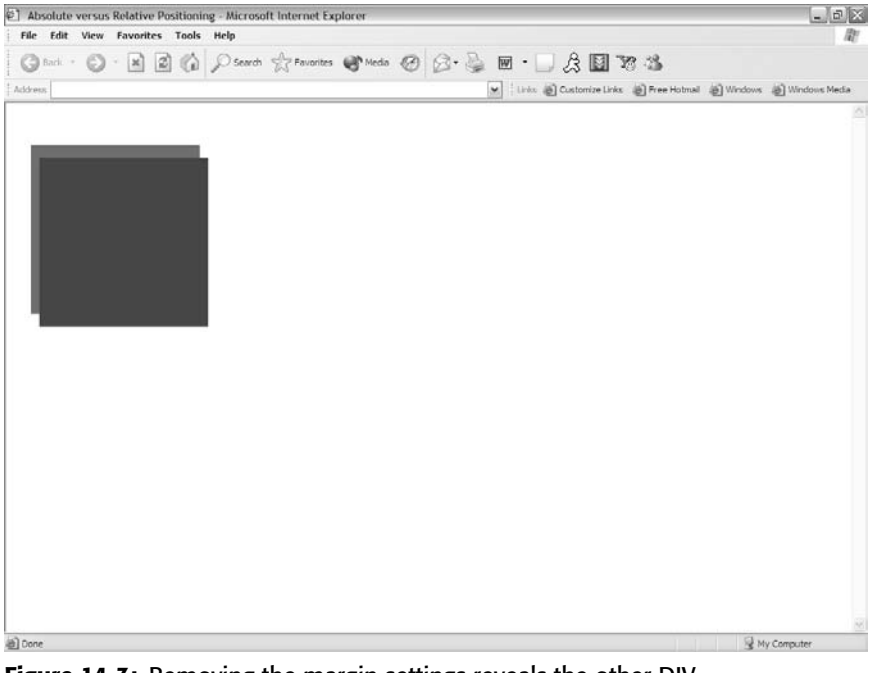


Figure 14-3: Removing the margin settings reveals the other DIV.

www.it-ebooks.info

By default the body element has a margin, so the normal flow of its first element, the blue square, will be offset. However, the position of the absolutely positioned green square is relative to the top left of the parent element; in this case the parent is the body of the web page, as shown in Figure 14-4.

It might be a bit confusing that even absolutes are relative to something. It just goes to show that Einstein was right, and everything is relative.

To explain further let's see what happens when some text is inserted:

```
<html>
<head>
<title>Absolute and Relative Positioning with Text</title>
</head>
<body style="margin:0 0 0 0">
Some text to move the relatively positioned block
<div style="position:absolute; top:50; left:30;
  width:200; height:200; background-color:blue;">
This is Absolutely amazing
</div>
<div style="position:relative; top:50; left:30; width:200;
   height:200; background-color:yellow;">
This is Relatively cool
</div>
</body>
</html>
```

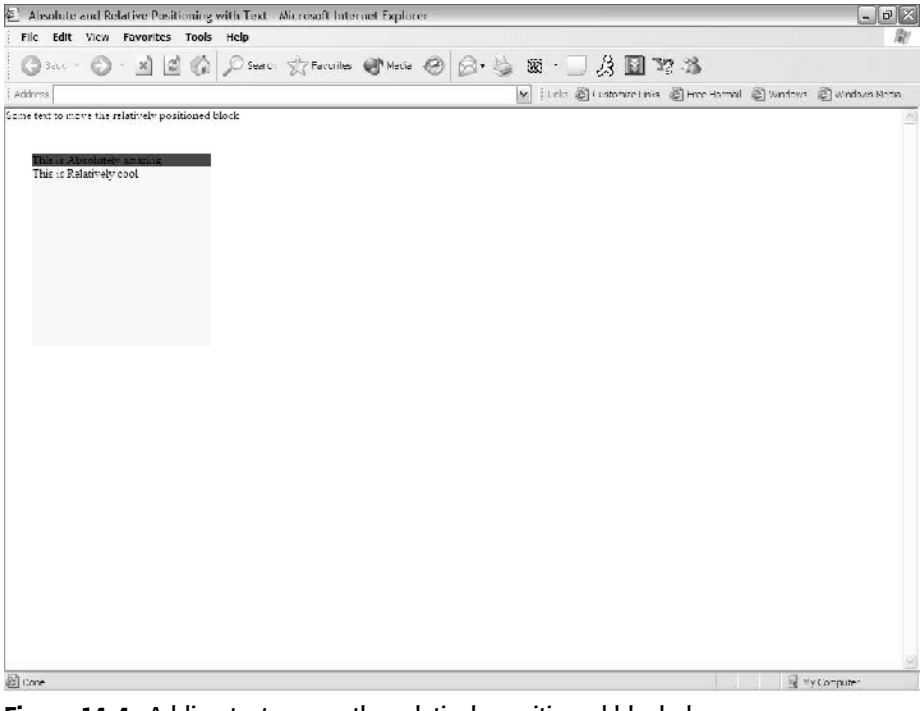


Figure 14-4: Adding text moves the relatively positioned block down.

www.it-ebooks.info

Nested elements

What happens if there is a box inside a box? Take a look at the following:

```
<html>
<head>
<title>Nested boxes Absolute and Relative Positioning</title>
</head>
<body style="margin:0 0 0 0">
<div style="position:absolute;top:50;left:30;width:200;height:200;
  background-color:blue;">
<span style="font-size: large; color: lime ">
This is Absolutely Relatively amazing</span>
<div style="position:relative;top:50;left:30;width:30;height:30;
  background-color:red;">
</div>
<div style="position:absolute;top:50;left:30;width:30;height:30
  background-color:yellow;">
</div>
</div>
</body>
</html>
```

You can see in Figure 14-5 that the first large blue box has two boxes inside a red relatively positioned box and a yellow absolute box. You can see that the yellow box

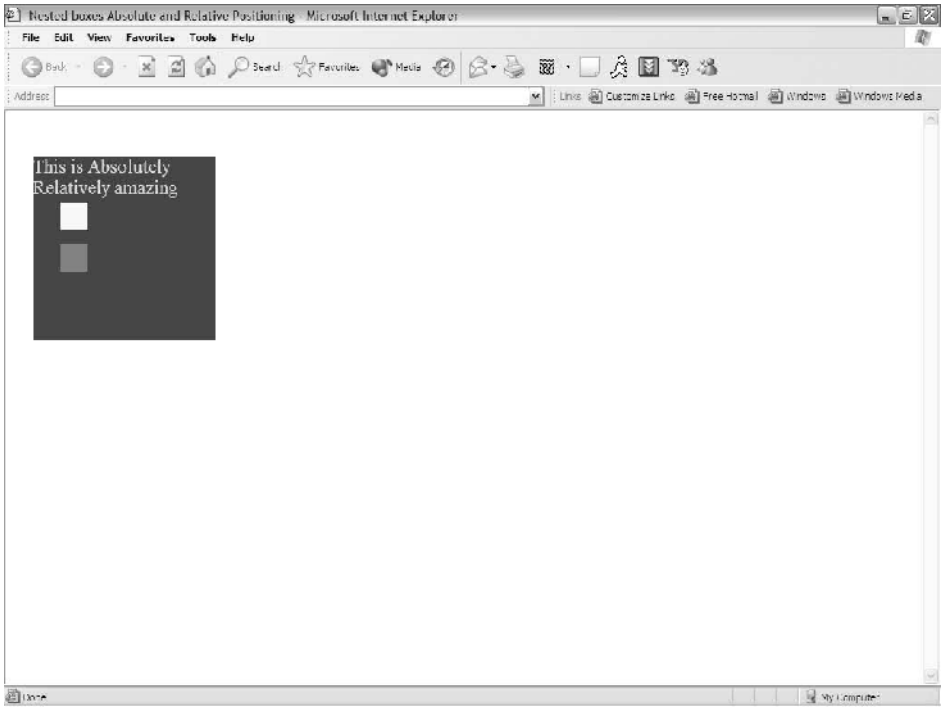


Figure 14-5: The yellow box is placed relative to the top left of the blue box, and the red box is shifted downwards.

www.it-ebooks.info

is placed relative to the top left of the blue box, and the red box is shifted downwards due to the text.

Overlapping Layers

Because layers can be positioned without regard to the location of other layers on the same Web page, as we have seen they can occupy the same location on the page. This, obviously, can present problems seeing all the elements because one of them can obscure another. However, it also opens up some intriguing possibilities in Web page design.

Stacking layers with z-order

The basic solution for handling overlapping layers is called *z-order* because there are three axes involved in positioning. If you think of the Web page as existing only on the screen of your monitor, the X axis runs across the screen from left to right. The Y axis runs from the top to the bottom of the screen. The Z axis, however, requires a little more imagination. It runs from the back of your monitor to the front of it.

The order of layers on the Z axis is specified via the `z-index` attribute. Any layer with a lower z-index value will be *underneath* another layer with a higher z-index value. Thus, in the following code sample, the first layer is on top of the second layer.



If no z-index value is specified, it defaults to zero:

```
<DIV style="position:absolute; top:0; left:0; width:200;
          height:200; z-index:5">
</DIV>
<DIV style="position:absolute; top:0; left:0; width:200;
          height:200; z-index:4">
</DIV>
```

If multiple layers have the same z-index value, the one that comes first in the code is the one that is beneath all the others, and the one that comes last is the one that is above all the others.

Using transparency and background color

While z-order is often used to make one layer cover another to hide it from view, the background color of the DIV element can be manipulated to create varying effects. If no background color is specified, then the layer is transparent and any underlying elements can show through.

In the following HTML code, the first layer completely covers the second one, and the background color of the second layer is still fully visible, as shown in Figure 14-6.

```
<HTML>
<HEAD>
<TITLE>
```

www.it-ebooks.info

```
Using Background Color
</TITLE>
</HEAD>

<BODY>

<DIV style="position:absolute; top:0;left:0; width:200; height:200;
  z-index:1; color:yellow; font-size:large;">Can you see me?
</DIV>

<DIV style="position:absolute; top:0; left:0; width:200; height:200;
  background-color:blue">
</DIV>

</BODY>

</HTML>
```

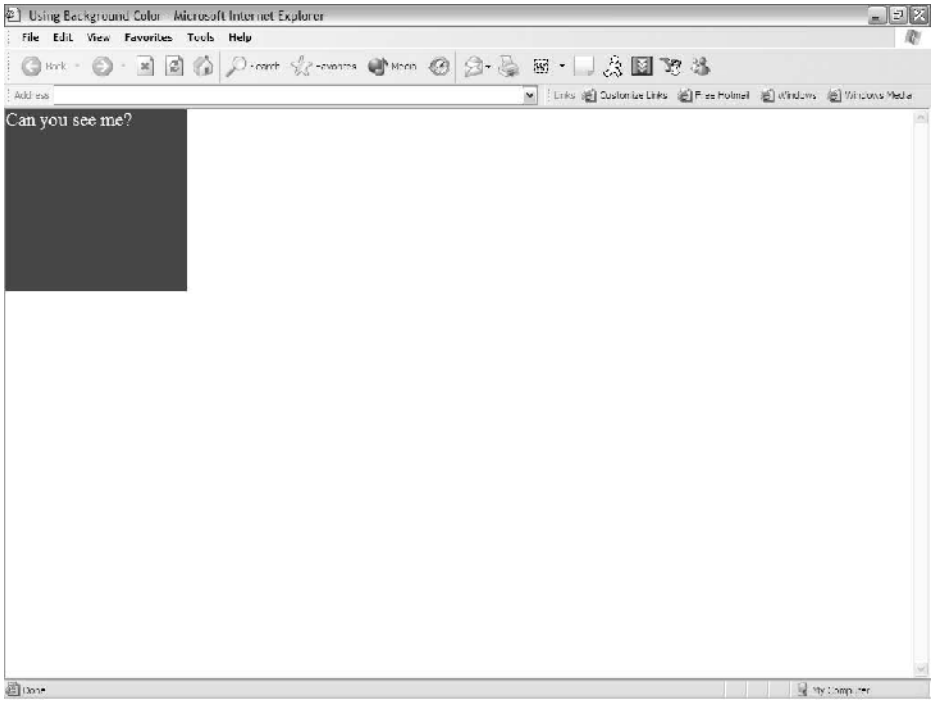
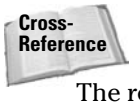


Figure 14-6: Backgrounds show through transparent layers.

Clipping Layer Content

Clipping a layer is pretty much the same thing as cropping an image, with one exception—the remainder of the layer is still there, but not visible. As with cropping, you are specifying a rectangular portion of the layer within which the visible portion of the element will appear.

www.it-ebooks.info



See Chapter 9 for information on cropping images.

The rectangle is set up with the clip property, and the values for it run as follows: `top`, `right`, `bottom`, and `left`. The following HTML code demonstrates how to set up a clipping region with a rectangle specified as `rect(20,80,100,10)` that has its top-left corner 20 pixels from the top, the right clip 80 pixels from the left side of the box, the bottom clip is 100 pixels from the top of the box and 10 pixels in from the left side. So the clipped region is 80 – 10 = 70 pixels wide and 100 – 20 = 80 pixels high. The results are shown in Figure 14-7.

```
<HTML>
<HEAD>
<TITLE>
Clipping a Region
</TITLE>
</HEAD>

<BODY>
<DIV style="position:absolute; top:0; left:0; width:200; height:200;
  background-color:yellow;">
<DIV style="position:absolute; top:0; left:0; width:200; height:200;
  clip:rect(20,80,100,10); background-color:blue;">
</DIV>

</BODY>

</HTML>
```

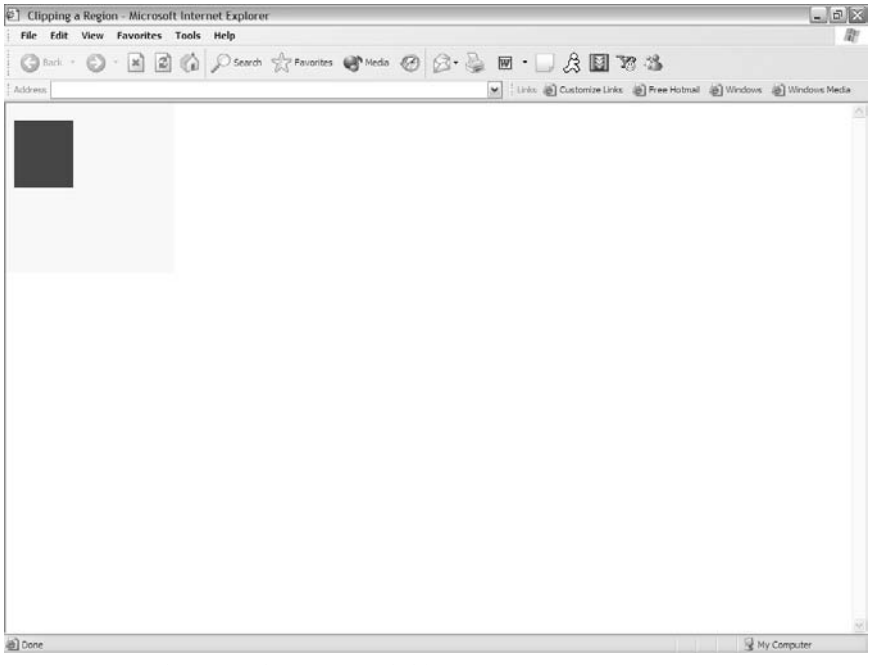


Figure 14-7: Clipping a layer is much like cropping an image.

www.it-ebooks.info



At the moment, most browsers (including Internet Explorer 6.0 and Mozilla 1.2.1) don't follow the CSS2 specification published by the W3C consortium. This states that the clip regions work in the same way as margins do—that is the right side is measured from the right and the bottom is measured from the bottom.

Visibility of layers

Not only can layers be positioned, but, like any HTML element, they can be set to be either visible or invisible. Obviously, an invisible layer isn't much use because nobody can see it. The trick is to make it visible at some point to achieve a particular effect.

You can make a layer invisible by including the attribute `style="visibility: hidden"`. See more details in Chapter 18.

Handling Large Elements with Overflow Attributes

When an element is larger than the size of the layer that contains it, it is said to *overflow* the layer. There are various methods for handling this situation, ranging from letting the element's size override the layer size to adding scroll bars so the whole element can be viewed from within the layer's fixed size.

Visible overflow

When the overflow is set to visible, then the entire content becomes—you guessed it—visible, even if the layer that contains it is smaller than it is. The following HTML code illustrates how to accomplish this feat, and Figure 14-8 shows the results:

```
<HTML>
<HEAD>
<TITLE>
Showing Overflowing Content
</TITLE>
</HEAD>

<BODY>
<DIV style="position:absolute; top:0; left:0; width:200; height:200;
  overflow:visible; background-color:blue">
<IMG src="breakfast.gif">
</DIV>

</BODY>

</HTML>
```

www.it-ebooks.info

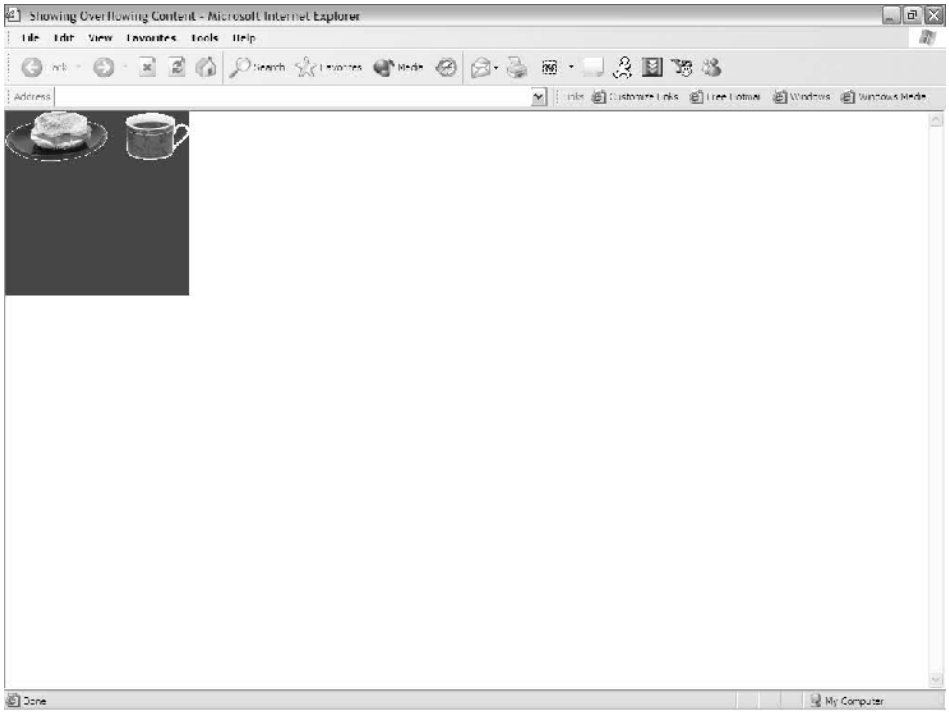


Figure 14-8: Overflowing content can be fully displayed despite the layer size.

Hidden overflow

Hiding the overflowing content of a layer may seem similar to clipping a region, but what you're really doing is simply not showing anything that falls outside the layer. The following HTML code demonstrates how to set a layer so that only the portions of elements that are actually within its confines are displayed, and Figure 14-9 shows the results:

```
<HTML>
<HEAD>
<TITLE>
Hiding Overflowing Content
</TITLE>
</HEAD>

<BODY>

<DIV style="position:absolute; top:0; left:0; width:200; height:200;
  overflow:hidden">
<IMG src="breakfast.gif">
</DIV>

</BODY>

</HTML>
```

www.it-ebooks.info

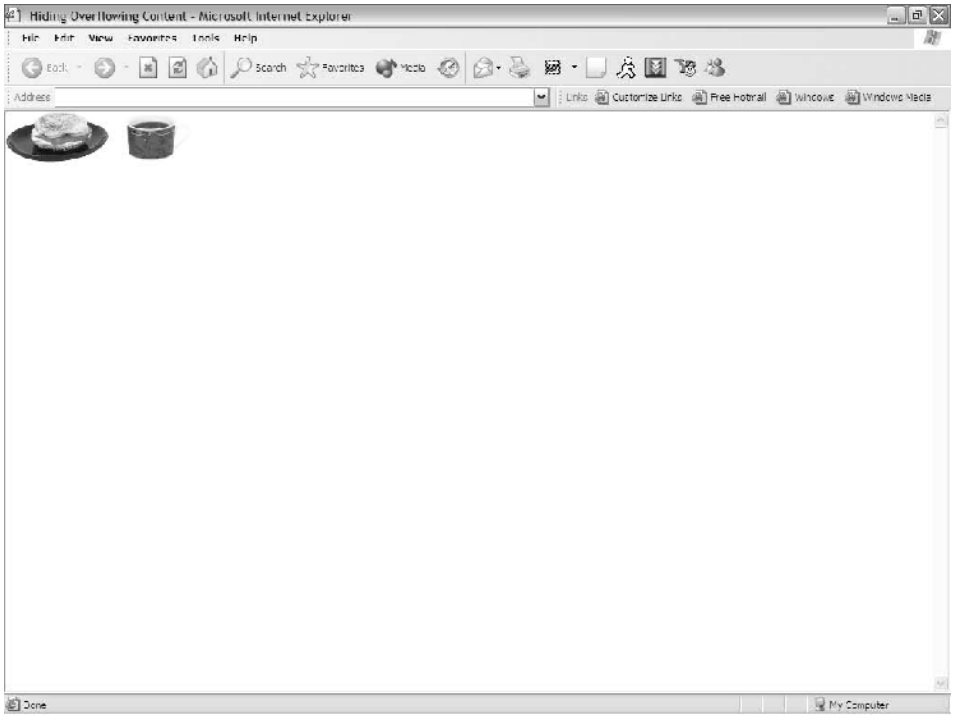


Figure 14-9: Content that overflows but is hidden is not visible outside of the layer.

Scroll bars

As a compromise between full visibility and chopping off any overflowing content, CSS offers scroll bars. The following HTML code shows how to add scroll bars to your layers, and Figure 14-10 shows how they look:

```
<HTML>
<HEAD>
<TITLE>
</TITLE>
</HEAD>

<BODY>

<DIV style="position:absolute; top:0; left:0; width:200; height:200;
  overflow:scroll">
<IMG src="breakfast.gif">
</DIV>

</BODY>

</HTML>
```



You can also set overflow to auto, but there's not much point to that. The results are the same as not bothering to specify the overflow property at all.

www.it-ebooks.info

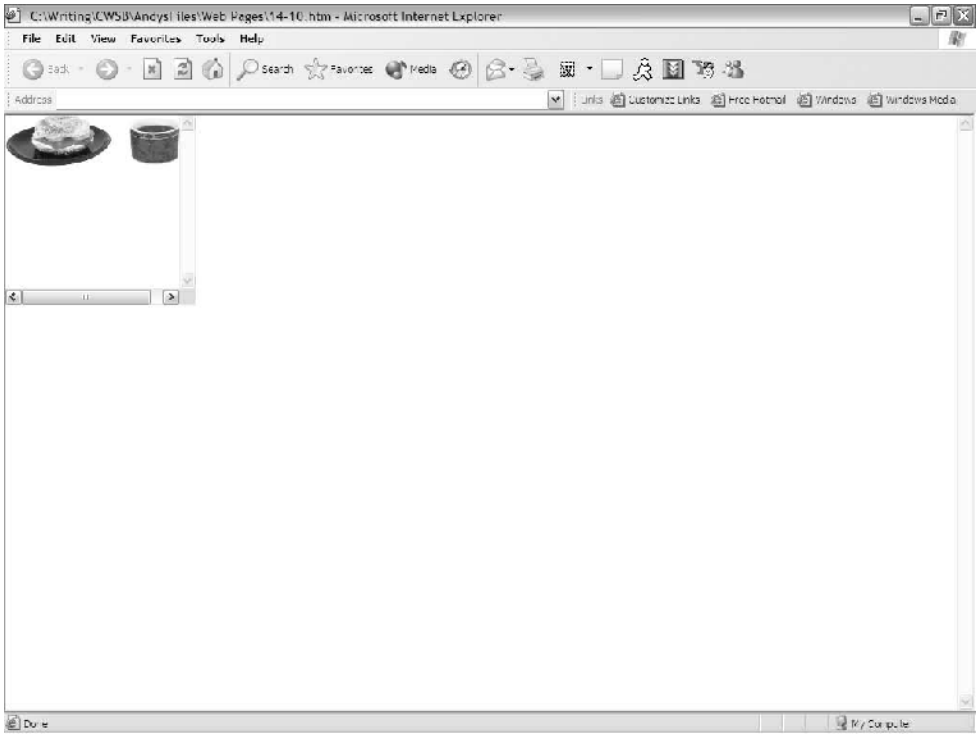


Figure 14-10: Make overflowing content fully visible within the confines of the layer by adding scroll bars.

Summary

Layers provide a good way to position elements on your Web pages. This chapter introduced you to the following concepts and topics:

- ✦ Absolute positioning is relative to the top left of the parent element.
- ✦ Relative positioning is relative to where the element would normally be displayed.
- ✦ The z-index value of an element determines which one is "on top" of another that occupies the same position on the Web page. The one with the lowest value is on the bottom. If two elements have the same z-index the last one is displayed on top of the first.
- ✦ Layers can be transparent or opaque.
- ✦ The content of a layer can be clipped or allowed to overflow the boundaries of the layer.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

PART

IV

Adding Interactivity With Flash, Forms, and Other Tools

In This Part

Chapter 15
Getting Input with Forms

Chapter 16
Making Dynamic Pages with JavaScript

Chapter 17
Navigating Your Web Site

Chapter 18
Adding Dynamic Page Elements with DHTML

Chapter 19
Animating with Macromedia Flash

Chapter 20
Adding Multimedia and Other Objects

Chapter 21
Blogging

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 15

Getting Input with Forms

In This Chapter

Adding forms

Getting short data with text boxes

Gathering information with text areas

Making choices with check boxes and options buttons

Saving space with the SELECT and OPTION elements

Using INPUT buttons

Using the BUTTON element

Adding hidden fields

Adding labels

Setting tab order

Specifying access keys

Submitting the form

Forms provide a way for your site visitors to give you information. Using a variety of different input techniques, forms enable you to gather all sorts of data ranging from simple yes or no answers to complex written responses.

Forms are used for several different purposes. Most often, they are used to gather contact and delivery information for situations ranging from e-mail newsletters to shipment of physical products. As HTML's native method for acquiring user input, forms are a powerful part of your Web design repertoire, enabling limitless opportunities for gathering information.

When it comes to visual presentation, however, forms leave a little bit to be desired. In fact, if you stick with the plain method of adding form controls and their accompanying text to your pages, you end up with a pretty sorry-looking design. To avoid this, embed your forms within tables, as the majority of Web designers do to keep their forms neat and attractive. This chapter is dedicated only to the use of forms. The technique for meshing them with tables is covered in detail in Chapter 11.

Adding Forms

Forms in HTML are made up of elements such as text boxes, check boxes, and option buttons, which are also known as radio buttons see Figure 15-1). Collectively, the elements in a form are known as *controls*. A form begins with the `<FORM>` tag and ends with the `</FORM>` tag. All of the form's controls are placed between them.



Form controls can actually be used outside a `FORM` element, but to keep things working properly, put them inside a `FORM` element.

www.it-ebooks.info

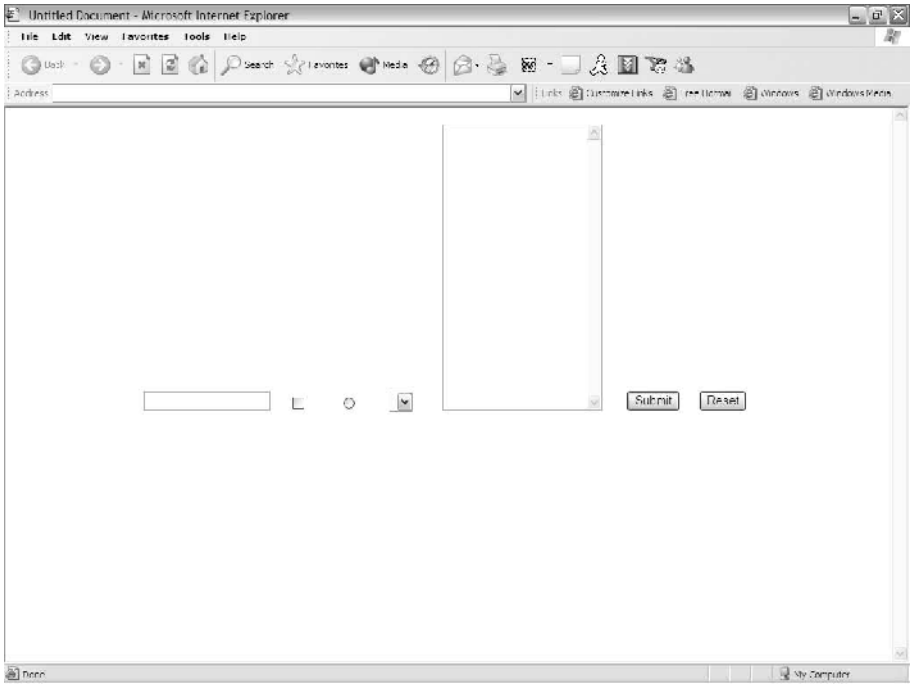


Figure 15-1: This figure shows various form controls.

The INPUT element

The majority of form controls are variations on a single element called `INPUT`. This is a unique situation in HTML in which items that have a totally different appearance and method of operation are represented by distinct elements. The `TABLE` element can't be used for anything but to create a table, the `IMG` element can only be used to add an image, and so forth. Yet, the `INPUT` element has 10 different aspects, some of which are very similar to one another, whereas others are totally different. Table 15-1 summarizes the many faces of the `INPUT` element.

Table 15-1
INPUT Types

Type	*Description*
Button	A custom button without a defined function in HTML. May be used to trigger JavaScript actions.
Check box	A small square that, when checked (filled), means "yes" and when empty, means "no." Any number of check boxes can be selected simultaneously.
File	A specialized text box into which a file name is entered. The file may also be selected by clicking a Browse button that accompanies the file text box. The specified file is uploaded along with other form information.

www.it-ebooks.info

Type	*Description*
Hidden	Information that is not displayed to the user, but is still transmitted along with the other form information.
Image	An image that functions as a Submit button.
Password	Identical to `text` except for the fact that it disguises the input.
Radio	A small circle that, when checked (filled), means "yes" and when empty, means "no." These buttons are distinguished from check boxes by the fact that you cannot select more than one button in a group.
Reset	A button that resets all entries a user has made in the form to their initial values.
Submit	A button that causes the form information to be transmitted.
Text	A text box that enables users to enter short information.

Each variation of the `INPUT` element is specified by its `type` attribute. For example, the following code creates a text box and a Submit button:

```
<FORM>
<INPUT type="text">
<INPUT type="submit">
</FORM>
```

Naming elements

Each element should also be given a distinctive name that, as you might guess, falls in the province of the `name` attribute. The reason for this is that the name of the element is paired with the value of its entry when the form data is submitted. If the element has no name, there is no way to tell what the entry itself means.

To illustrate this, let's say you have a text box that asks visitors to state their age. You give it the name "age" so that it's clear what the entered number means when it shows up in a report based on the form submission. This is accomplished by the following code:

```
<INPUT type="text" name="age">
```

If a user enters the number 37 into that text box when the form is submitted, the data that is transmitted is `age=37`, which clearly tells you—or the data processing program that receives the form input—what is meant.

If you want to use JavaScript with your controls then you should also give them an `id` attribute. To keep things simple make the `id` attribute identical to the value of the `name` attribute. This is because the `name` attribute is sent to the server and the `id`

www.it-ebooks.info

attribute is used locally to refer to the control. Ideally, there should be only one definition of the `name` and `id` attributes. The HTML standard is moving towards just using the `id` attribute, but as you can see there is still some way to go.



The one exception to the rule of giving each element a separate name is with option buttons. Option buttons that are grouped together all share one name.

Of course, each type has a more complex series of settings than what we've covered so far. The following sections explore the details of the `INPUT` element and the other three elements that are used inside forms.

Getting Short Data with Text Boxes

Text boxes are used to collect short strings of information such as first and last name, street address, and so on. They are the single most often used type of form control. Some forms, in fact, consist of nothing but a single text box and a button to submit the text box's contents. This is the case with the most popular search engines (see Figure 15-2), although they may also have a second, more complex, form available.

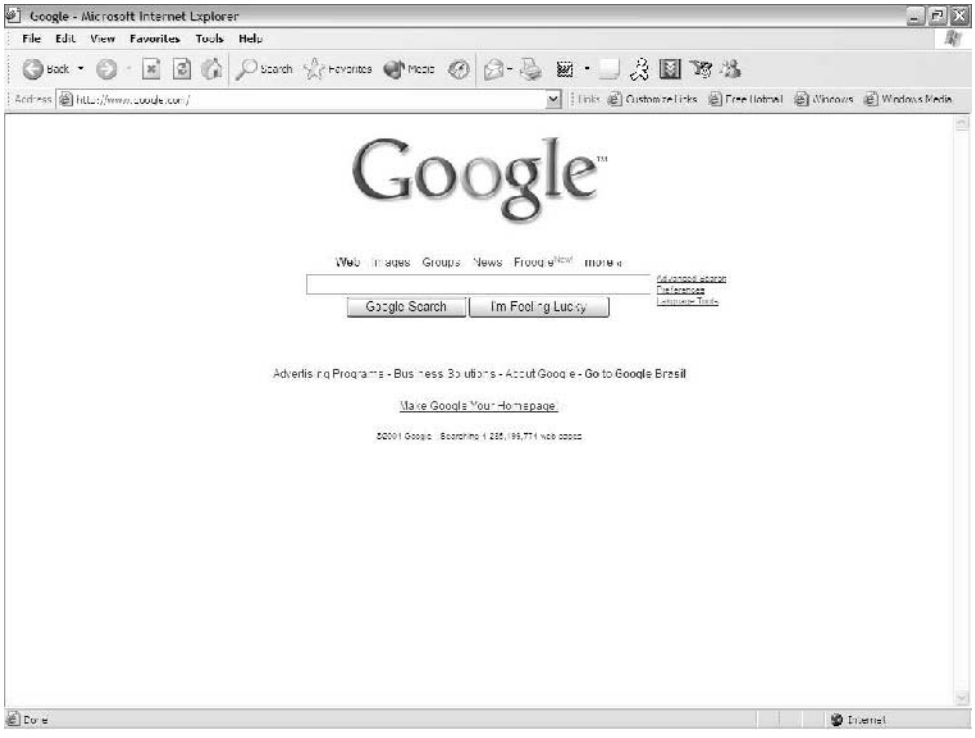


Figure 15-2: Google, one of the best search engines, uses a simple form interface.

www.it-ebooks.info



If there is only one text box in a form and the user presses Enter, the form is automatically submitted, as you may have noticed in Google.

Password fields are similar to text boxes. However, when someone enters a password, the password field displays a string of asterisks instead of showing the actual password. This is a security precaution. Although the password itself is correctly passed through the form, it is impossible for anyone to identify the password by simply looking at the screen. Figure 15-3 shows a typical login screen that uses both a text box and a password field.

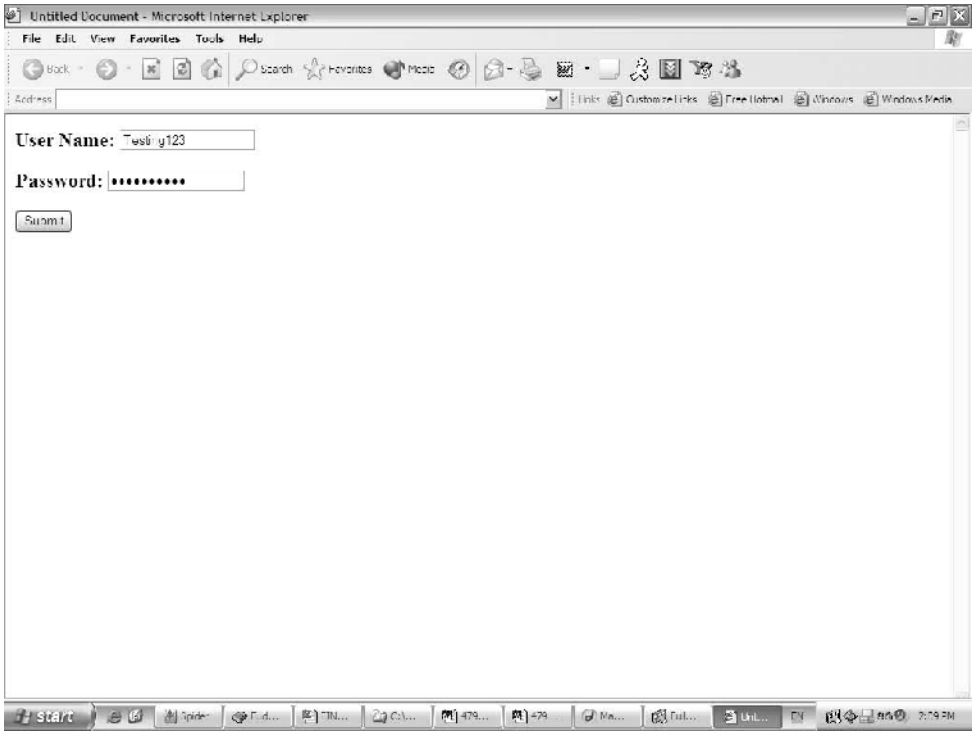


Figure 15-3: The only difference between password fields and text boxes is the display of the user's entry.

Remember that if you're using http, the password is not encrypted when it's sent over the Internet; in the case of https it is. To check that the connection is secure, look for a locked padlock or key icon at the bottom of the browser window or ensure that the URL in the address bar starts with https.

Setting box size

The physical width of a text box is set with the `size` attribute. Unlike width and height settings, this setting does not use pixels as the default unit for measurement.

www.it-ebooks.info

Instead, it uses characters. Thus, the following code creates a text box that is the same width as 20 letters:

```
<INPUT type="text" size="20">
```



The exact number of how many characters fit in a box vary when using proportional width fonts, because each character takes up a different amount of space. It's a good idea to test your Web pages on several browsers before you go into production.

Figure 15-4 shows several text boxes, ranging in size from 10 to 50 characters wide.

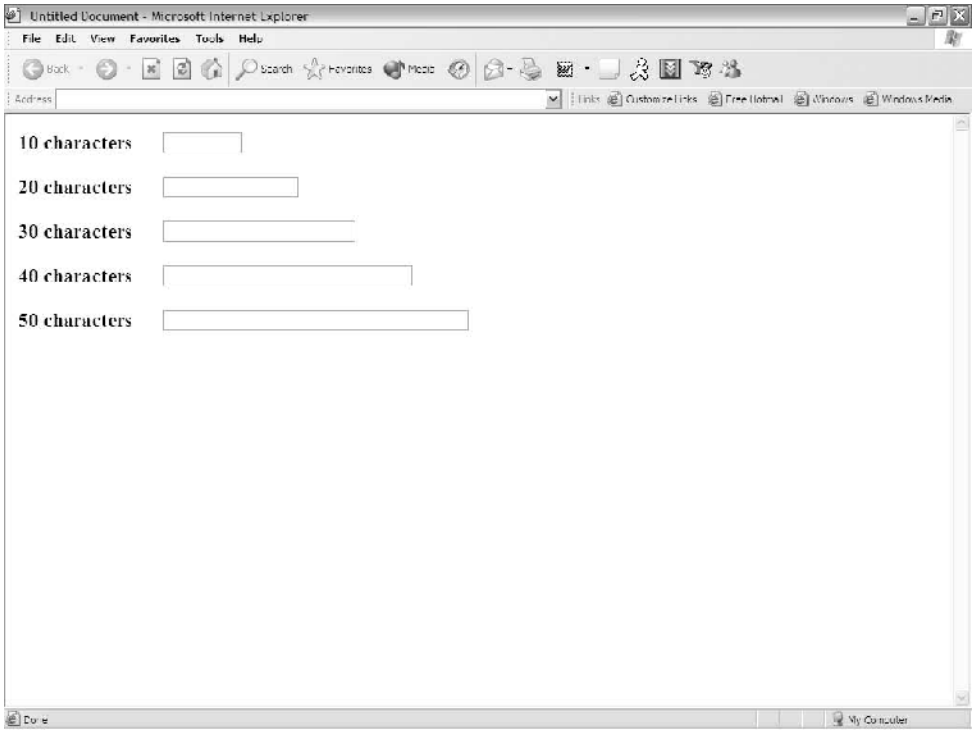


Figure 15-4: Text boxes can be set to any width in characters.

Defining the maximum length of input

The physical width has nothing to do with how much text can fit into a text box. If you have a text box that's 30 characters wide and a visitor to your site enters 80 characters, the text box still takes the input. The text simply scrolls as it's entered (see Figure 15-5). The amount of text that can be input is unlimited.

However, the processing program receiving the form's information may have an input size limitation. That's where the `maxlength` attribute comes into play.

www.it-ebooks.info

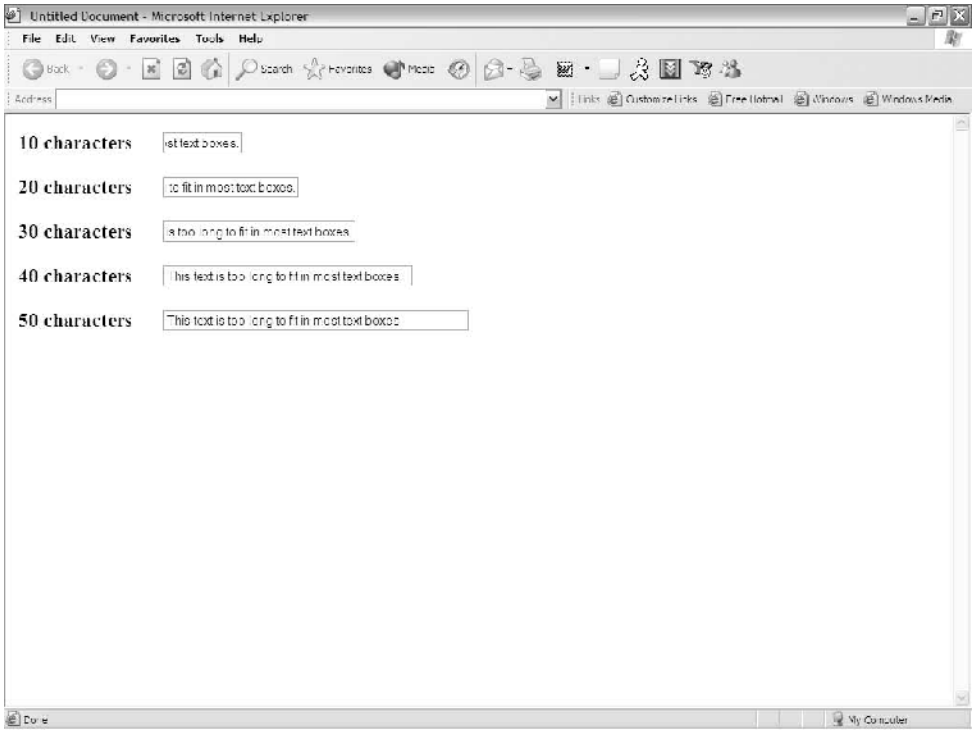


Figure 15-5: Text entries that are larger than the text box scroll across it.

Say, for example, that the data processing program can only accept text input that's 30 characters or less, yet your design constraints mean that you still have to use a 20-character-wide text box. The following code solves both problems in one swoop:

```
<INPUT type="text" size="20" maxlength="30">
```

Of course, there is no requirement that the `maxlength` value be larger than the `size` value. They could be set to the same number, which would mean that there would be no scrolling at all, but the text box could still be filled completely. The following code accomplishes this:

```
<INPUT type="text" size="20" maxlength="20">
```



You can actually set the `maxlength` value to be less than the `size` value. Even though this is possible, it could create a counter-intuitive form that would only confuse users. It's better to keep `maxlength` at least equal to size.

Prefilling a text box

There are times when you can presume to know what value a visitor will enter in a text box. For example, if you're selling a magazine subscription, the odds are pretty

www.it-ebooks.info

good that one person will order only one copy—very few, if any, people want multiple copies of the same magazine sent to them every month.

In that instance, it would be safe to specify "1" as the value for the number of subscriptions ordered. However, if someone does have a need for multiple copies, it's wise to not lock them in to ordering just one. To accomplish this, specify content for the text box that can be altered by anyone who fills out the form. Thus, the majority of subscribers are saved the trouble of filling in the number, whereas the minority can still do what they want.

This is accomplished with the `value` attribute, as in the following code:

```
<INPUT type="text" size="3" name="numberofsubs" value="1">
```

The results are shown in Figure 15-6.

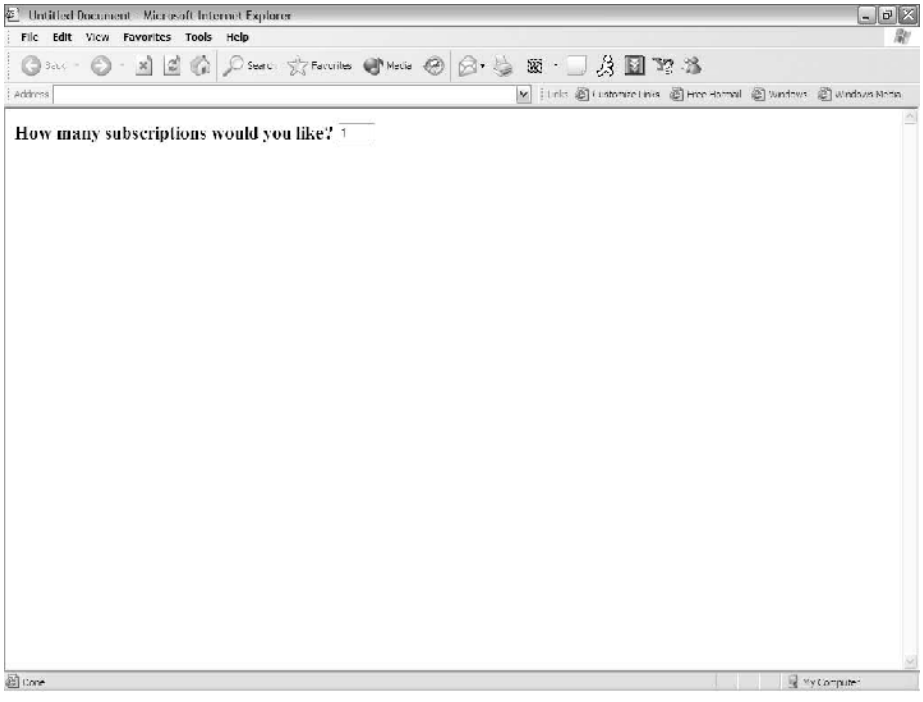


Figure 15-6: For this simple form we set a default value.



Setting the size to 3 allows a potential subscriber to order up to 999 subscriptions.

Locking content with readonly and disabled

If you're absolutely, totally certain that you want to lock into place a particular value for a text box's content, use the `readonly` attribute. It's simply a matter of specifying the content and adding that attribute to the code, as in the following example:

www.it-ebooks.info

```
<INPUT type="text" size="1" name="numberofsubs" value="1" readonly>
```

However, there is a drawback to this approach—users filling out the form are shown a text box, an element they are used to entering information into, but one that refuses to cooperate with them. The potential for user frustration—and, therefore, for refusal to complete the form—is high with this approach.

An alternative would be to disable the text box using the `disabled` attribute. This prevents the user from even placing the cursor in the text box. Disabled controls are normally displayed as grayed out.



The content of a control is sent to the server if it is marked `readonly` but not if it is marked `disabled`.

```
<INPUT type="text" size="1" name="numberofsubs" value="1" disabled>
```

If you just want to send some data to the server, a far better technique is to use the `hidden` type, which is discussed in the section "Adding Hidden Fields" later in this chapter.

Gathering Information with Text Areas

Text boxes are good for information that can be entered in one line, such as a first name, last name, or zip code. But what about situations in which you need a multiline response, such as directions to a farm or comments on the usefulness of a product? The `TEXTAREA` element enables you to accept multiline responses. Figure 15-7 shows how a text box compares to a text area.

Sizing text areas

The number of characters you want to be displayed sets the width of text areas, as is the case with text boxes. However, because text areas have more than one line, you can also set the height of the text area. The width is set by the `cols` attribute and the height by the `rows` attribute. Each column equals the width of one character, and each row is the height of one line of text. The following example shows how to set a text area that is 50 characters wide and 10 lines high:

```
<TEXTAREA cols="50" rows="10">
</TEXTAREA>
```

Figure 15-8 shows several text areas of various sizes.

Setting word wrap

There are a few different ways in HTML for the input of text areas to be wrapped, but only two of them have any effect on how the text is actually displayed on the screen of a Web browser. Word wrap is controlled by the `wrap` attribute, as in the following example code:

```
<TEXTAREA wrap="virtual">
</TEXTAREA>
```

www.it-ebooks.info

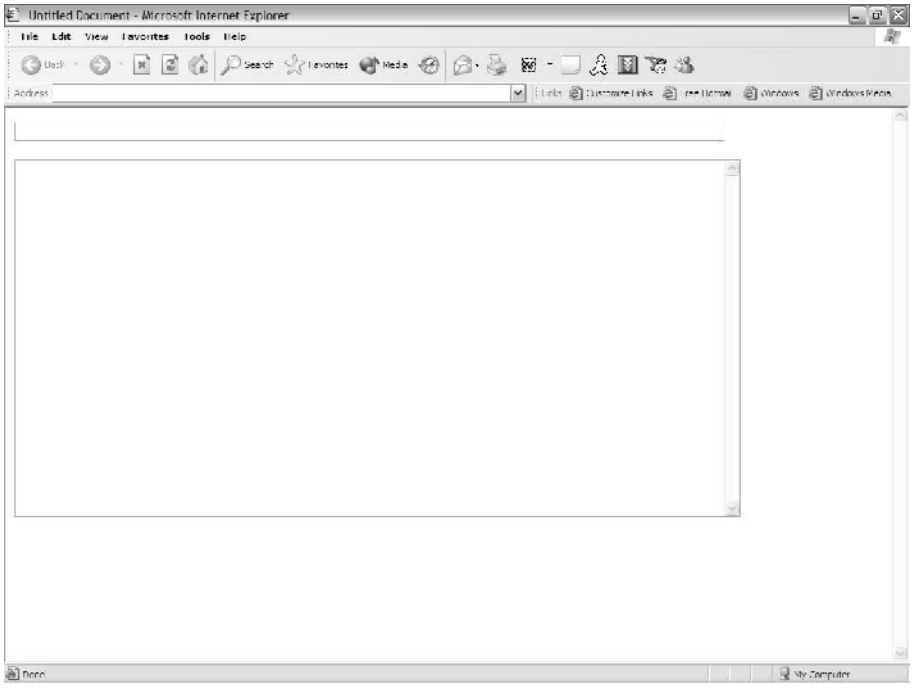


Figure 15-7: Text areas are capable of holding multiple lines of text.

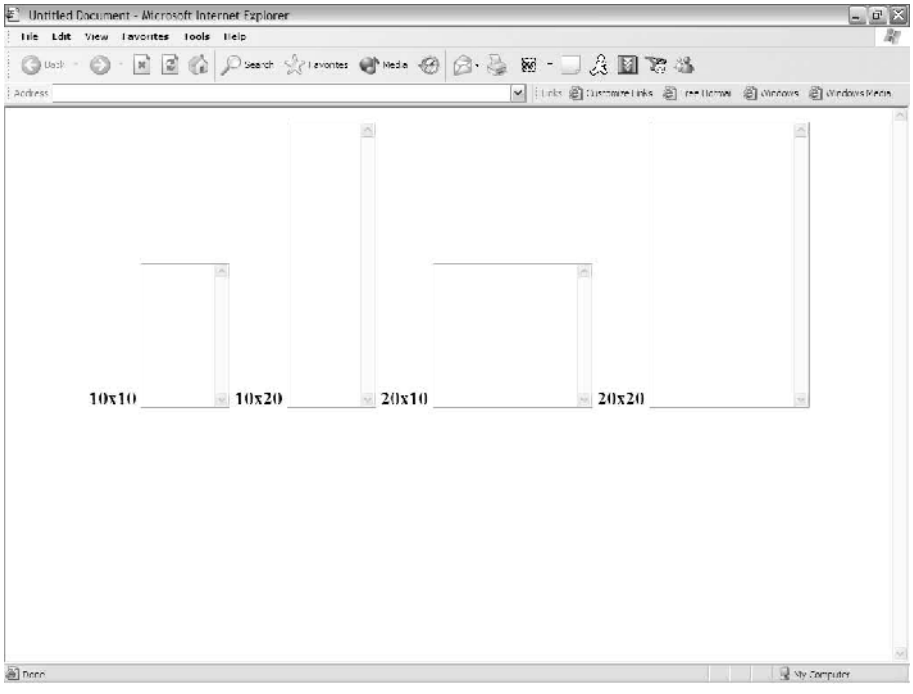


Figure 15-8: Text areas can be set to any character width or number of lines.

www.it-ebooks.info

Table 15-2 shows what each value of the `wrap` attribute means.

Table 15-2
Word Wrap Values

Value	*Description*
off	The text is displayed on one line until the user presses the Enter key, at which point a new line begins. If the line is longer than the text area is wide, the text scrolls.
virtual	The text wraps to the next line when it reaches the edge of the text area. However, it is sent as one long line when the form is submitted.
physical	The text wraps to the next line when it reaches the edge of the text area and is sent as displayed when the form is submitted.



You can use the values `soft` and `hard` to replace `virtual` and `physical`, but they are only recognized by Netscape Navigator and Internet Explorer versions 4 and later.

If no value is specified, the word wrap relies on the default behavior of the particular Web browser that's used to view the form. In Netscape Navigator, this is the same as `off`, whereas in Internet Explorer and Mozilla, it is the same as `virtual`.

Figure 15-9 shows both scrolling and nonscrolling text areas.

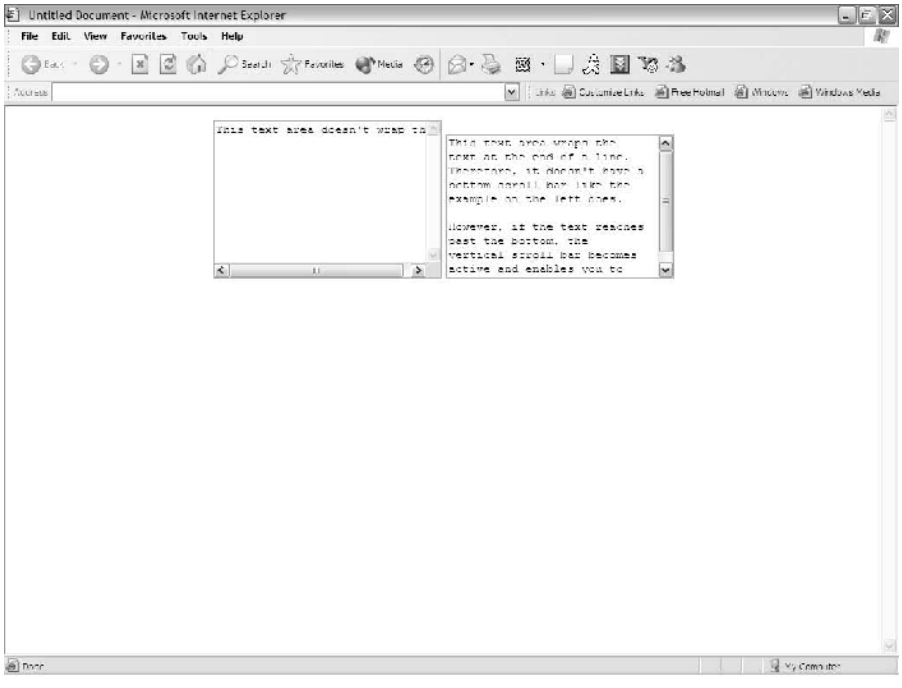


Figure 15-9: Text areas can be made to wrap or scroll.

www.it-ebooks.info

Setting default contents

As with text boxes, text areas can hold default content; however, there is no attribute to be set. Instead, any text that is placed between the <TEXTAREA> start tag and the </TEXTAREA> end tag shows up onscreen as the default content.

For example, the following code displays a helpful message in a text area:

```
<TEXTAREA name="comments" cols="50" rows="10">
Please tell us what you think.
</TEXTAREA>
```

The results of this code are shown in Figure 15-10.

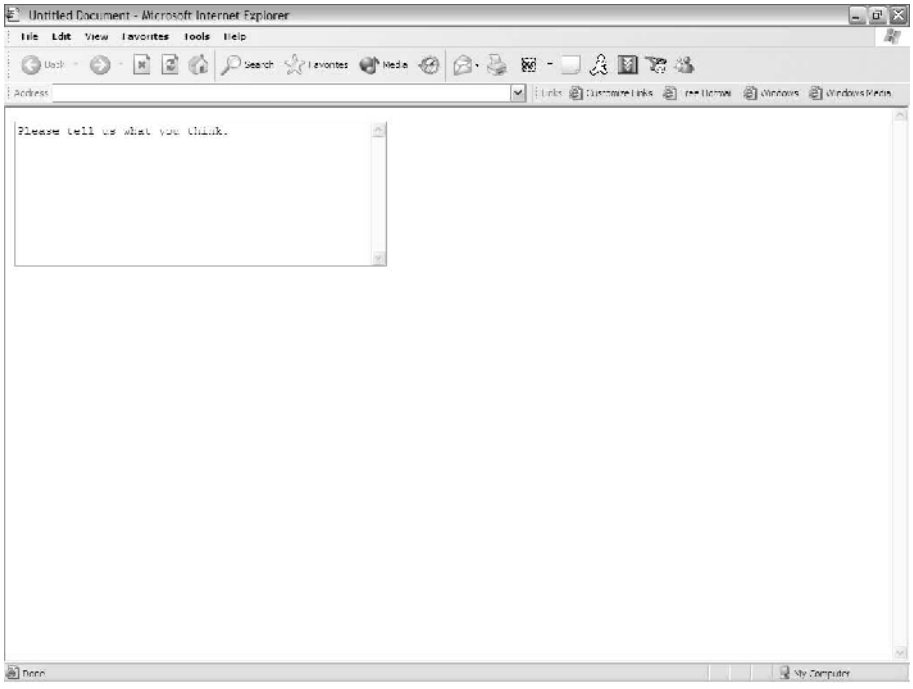


Figure 15-10: Text areas can have default content.

Making Choices with Check Boxes and Option Buttons

While text input is important in forms, it is often easiest—for both you and your visitors—if you present a set of choices instead. Check boxes and option buttons solve this problem handily. Both are types of the INPUT element and are very similar in function and appearance. Check boxes are, as the name implies, square, whereas

www.it-ebooks.info

option buttons are round (see Figure 15-11). Beyond that, however, they are pretty much the same thing—a user clicks them to indicate a "yes" or "no" choice.

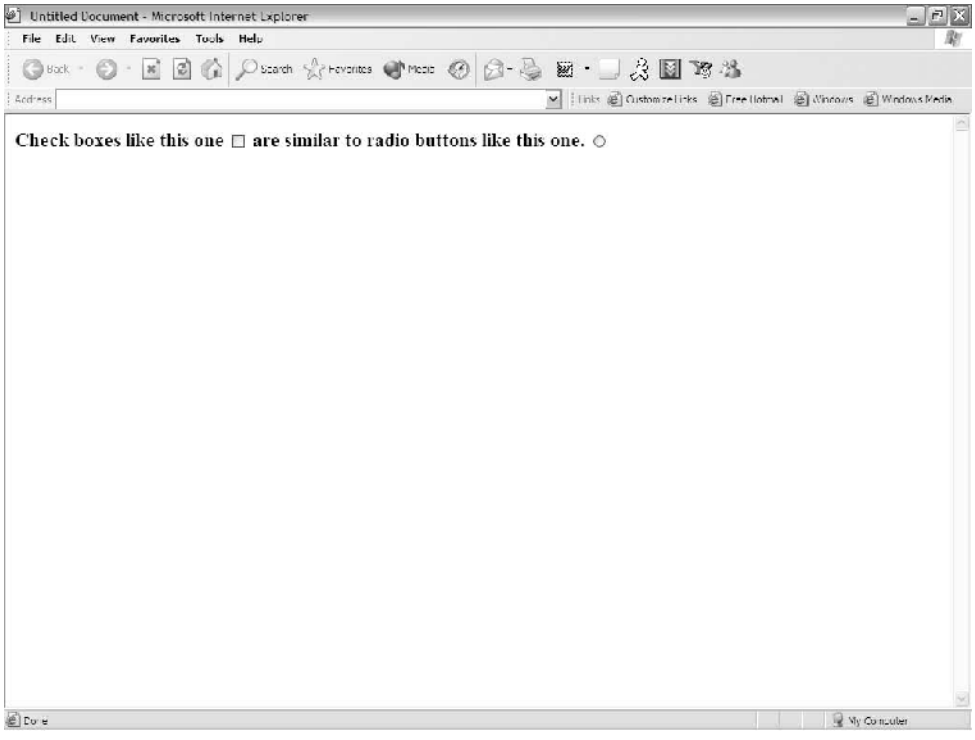


Figure 15-11: Check boxes and option buttons are similar, but have important differences.

There is, however, one critical difference in the way they can be used, as discussed in the following sections.

Adding check boxes

Check boxes are used when a choice to be made does not conflict with another possible choice. For example, you might use check boxes if you are offering information on several different products. You certainly wouldn't want to limit your visitors to asking for only one brochure! The form in Figure 15-12 shows a set of check boxes that offer such a set of choices.

Even a single check box, offering one simple "yes" or "no" possibility, makes sense. For instance, you often see, at the bottom of a form asking for personal information a single check box that asks whether or not you want to receive mailings from third parties. Such a check box may be coded as follows:

```
<INPUT type="checkbox" name="askmailing" value="permissiongranted">
```

www.it-ebooks.info

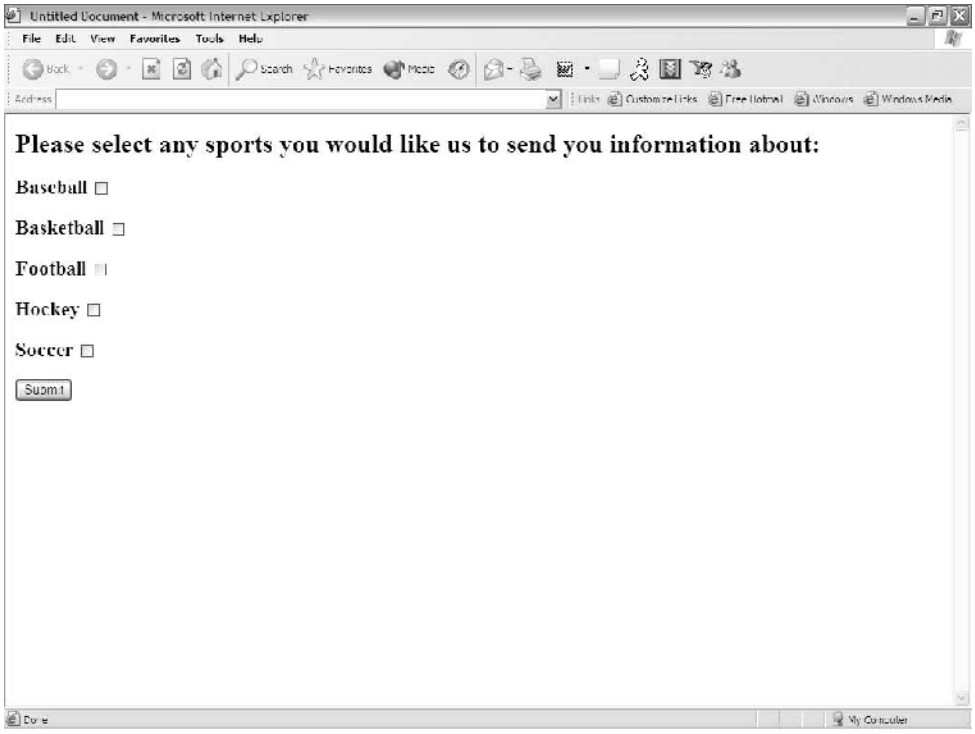


Figure 15-12: Check boxes offer noncompetitive choices.

If a visitor does click the check box to select it, when the form is submitted, the name/value pair `askmailing=permissiongranted` is sent. If the check box remains unselected, it is ignored when the form is submitted, and no input from it exists.

Grouping option buttons

As noted earlier, option buttons are different from all the other elements that make up forms. Although you can use an individual option button in the exact same way as an individual check box, the true power of option buttons comes when they are used in groups. Because option button choices are mutually exclusive, when they are grouped together as a unit and you choose one option button in that group, you are prevented from making a conflicting choice.

For example, say you are selling hats. A visitor can order a small, medium, or large hat, but one hat cannot be two or more sizes. If you were to use three check boxes, it would be possible for someone to place an order for a single hat that was small, medium, and large. When you received the form data, you would not be able to tell if the person filling out the form wanted three different hats or was simply confused. The input would be totally useless.

www.it-ebooks.info

Three option buttons, however, would be perfect for the job. If your customer used the form shown in Figure 15-13, and they chose the radio button for small, but also clicked the medium button, the small would be automatically deselected. If they continued on to click the large option, the medium option button would, in turn, be deselected. As you can see, this process totally eliminates any possibility of confusion among a set of conflicting choices.

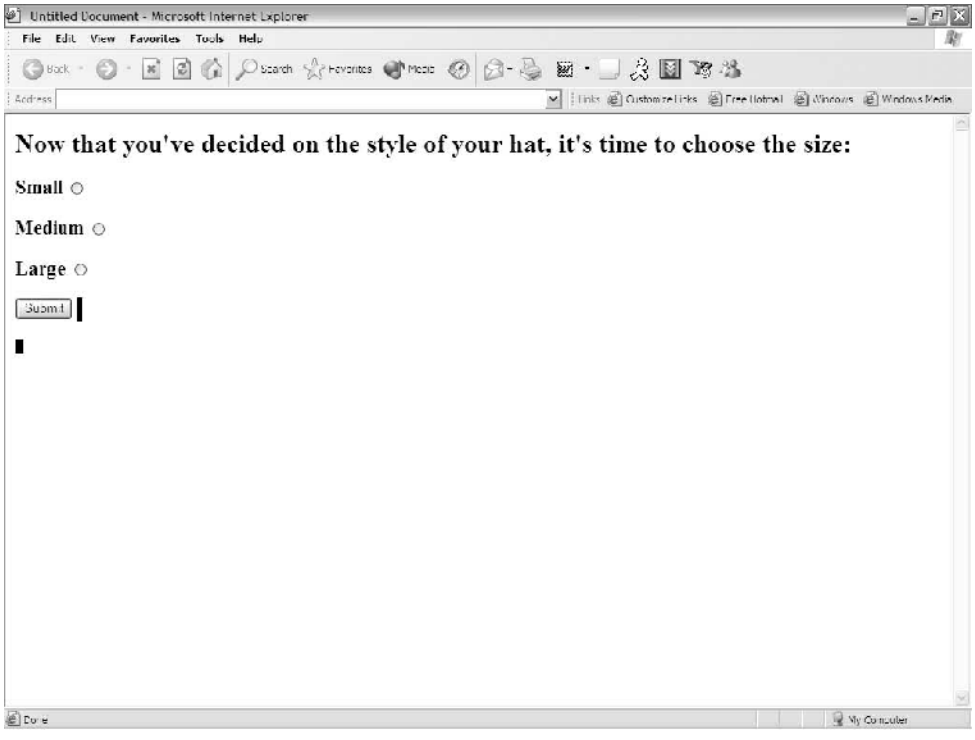


Figure 15-13: Option buttons offer mutually exclusive choices.

Although all other elements in a form must have separate names, option buttons are a different story entirely. An option button group is created by giving the same name to more than one option button. The `value` attribute of each option button is what distinguishes one from the other. The hat size option button group, for instance, would be created by code such as this example:

```
<INPUT type="radio" name="greenhat" value="small">
<INPUT type="radio" name="greenhat" value="medium">
<INPUT type="radio" name="greenhat" value="large">
```



Care should be taken to ensure that all option buttons in a group are actually placed in the same area of the form. There is nothing in HTML to prevent you from creating a group that is scattered about the form in different locations.

www.it-ebooks.info

Setting a default choice

Both option buttons and check boxes can be set as the default choices in a form. Returning to our example, where a visitor is asked permission to share private information with third parties, some Web sites force the issue by setting the check box so that it is selected from the start without any action on the part of the visitor. To do so, they simply add the `checked` attribute as in the following example:

```
<INPUT type="checkbox" name="askmailing" value="permissiongranted" checked>
```

Now, a visitor must click the check box to deselect it. If not, the permission has been given.

This technique, in addition to being commonly used to trap unwary people into giving permission for things they wouldn't want, has perfectly valid uses. Anytime you have an option that you believe most people would select, it makes sense to have it preselected to save your visitors the bother of filling out every detail—and to save yourself the frustration of an incomplete form being submitted. It's similar to adding default content to a text box or text area.

For option buttons, you should set only one in the group. For instance, if you know that almost every order you get is for a medium-sized hat, the following code would create a form that assisted most of your customers:

```
<INPUT type="radio" name="greenhat" value="small">
<INPUT type="radio" name="greenhat" value="medium" checked>
<INPUT type="radio" name="greenhat" value="large">
```



Although it is possible to add the `checked` attribute to more than one option button in a group, you need to ensure that this doesn't happen. The result in Internet Explorer is that only the last one with the `checked` attribute is shown as checked. In Netscape Navigator, all option buttons with the `checked` attribute are checked. If all of the option buttons are set this way, then none of them work.

Saving Space with the SELECT and OPTION Elements

If you only have to add a few options, such as small, medium, or large, to your form, check boxes and option buttons are definitely the way to go. But what if you need to do something a lot more complex than that? What if you have to add choices for all 50 U.S. states or every country in the continent of Europe? As Figure 15-14 shows, you end up with so many boxes or buttons that your form is an unusable mess.

That's where pop-up menus and list boxes come into the picture. As you can see in Figure 15-15, the use of a pop-up menu creates a much more pleasing form, and still enables all the necessary options to be included.

www.it-ebooks.info

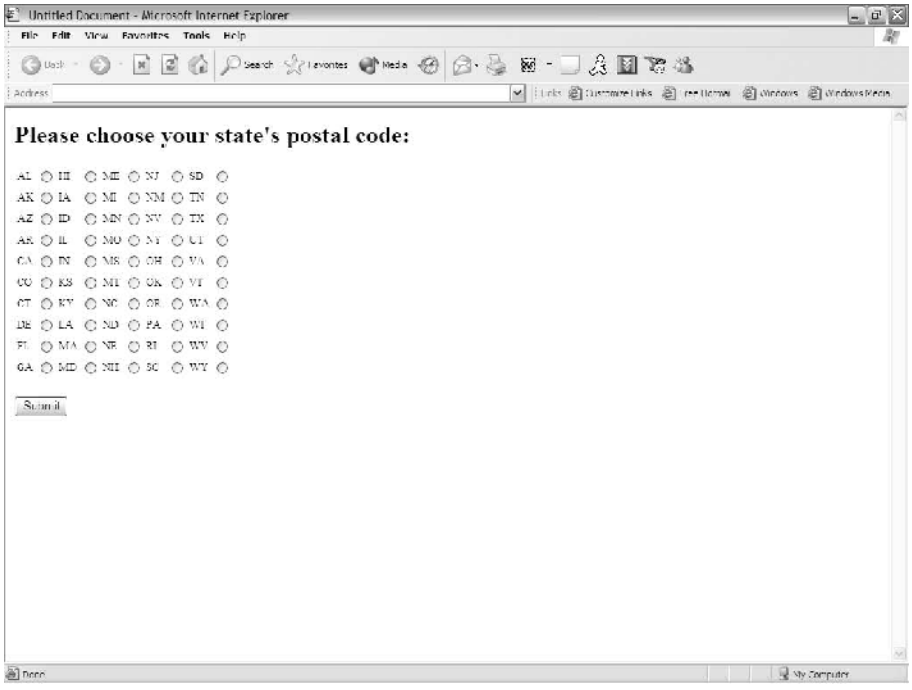


Figure 15-14: Too many check boxes or option buttons make a form look cluttered.

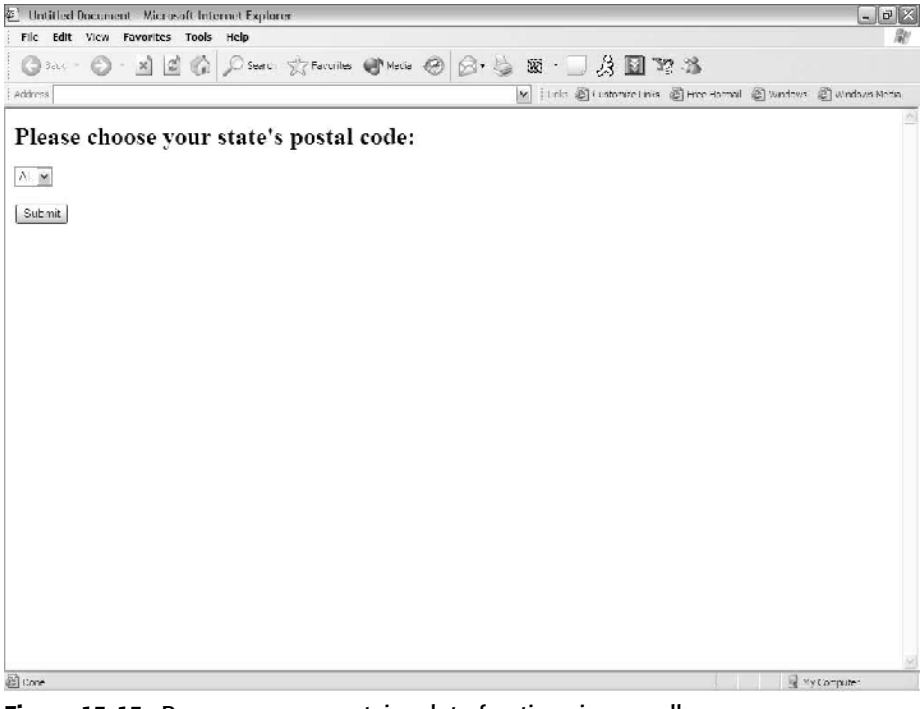


Figure 15-15: Pop-up menus contain a lot of options in a small area.

www.it-ebooks.info

Both pop-up menus and list boxes are created using a combination of the SELECT and OPTION elements. The major difference between them is their appearance onscreen. A pop-up menu is simply a list of options showing only one choice, whereas other choices are accessed by clicking an arrow that opens the list for selection. A list box is a list of options showing more than one of the options onscreen.

The following code creates a pop-up menu enabling users to select one of four geographical regions:

```
<SELECT name="region">
<OPTION>South</OPTION>
<OPTION>North</OPTION>
<OPTION>East</OPTION>
<OPTION>West</OPTION>
</SELECT>
```

The SELECT element defines where the pop-up menu begins and ends, whereas the OPTION elements specify the choices included in the list.



The OPTION element does not take a name.

Figure 15-16 shows the pop-up menu when a visitor clicks its arrow.

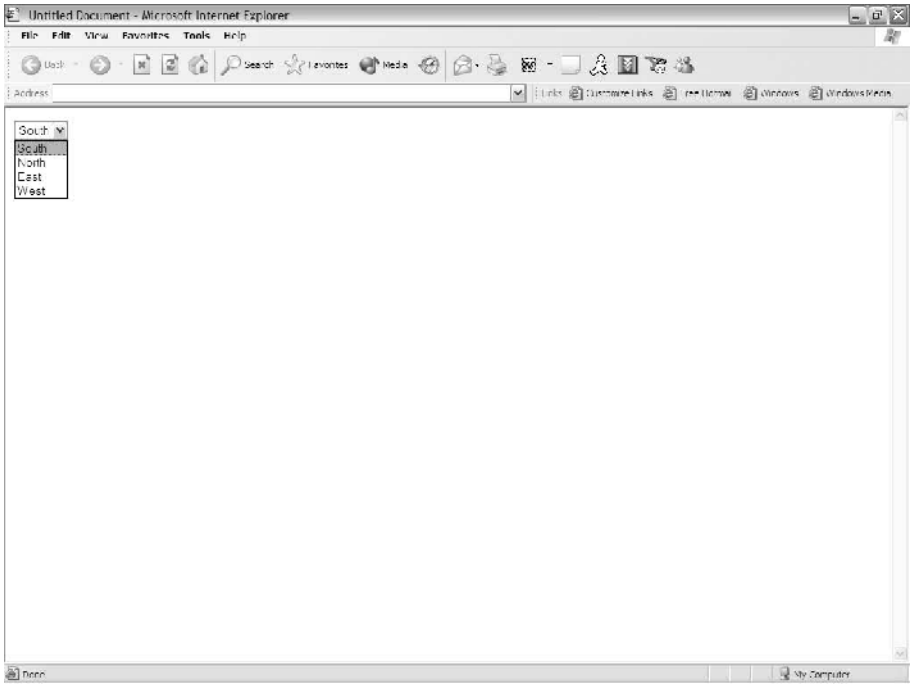


Figure 15-16: Pop-up menus display all the choices when activated.

www.it-ebooks.info

Specifying values

The name/value pair, which is sent when the form is submitted, would be `region=South` if the first option were chosen, `region=North` for the second option, and so forth. However, there are times when a value needs to be submitted that doesn't make a good, readable option for people to choose. What, for example, do you do if company regulations specify that the southern region is to be referred to by some internal code such as `SoReg`, but the form is to be used by customers who aren't familiar with such abbreviations?

The optional `value` attribute for the `OPTION` element can solve this problem easily. If it is specified, it overrides the content of the option. The preceding example can be modified as follows:

```
<SELECT name="region">
<OPTION value="SoReg">South</OPTION>
<OPTION value="NoReg">North</OPTION>
<OPTION value="EaReg">East</OPTION>
<OPTION value="WeReg">West</OPTION>
</SELECT>
```

Thus, the users of the form see easily comprehensible choices, but the internal code is submitted instead of the plain language.

Setting menu types

The same options could be presented as a list box simply by specifying how many of the options should be visible. This is done with the `size` attribute. If you wanted all four of the options to show at once, you would change the code as follows:

```
<SELECT name="region" size="4">
<OPTION value="SoReg">South</OPTION>
<OPTION value="NoReg">North</OPTION>
<OPTION value="EaReg">East</OPTION>
<OPTION value="WeReg">West</OPTION>
</SELECT>
```

To make three of the options show, the following code would do the trick:

```
<SELECT name="region" size="3">
<OPTION value="SoReg">South</OPTION>
<OPTION value="NoReg">North</OPTION>
<OPTION value="EaReg">East</OPTION>
<OPTION value="WeReg">West</OPTION>
</SELECT>
```

If the number of options showing is less than the total, scroll bars appear, enabling users to scroll up and down through the list. Figure 15-17 shows two samples of this list box, one set to show three of the options and the other set to show all of them.

www.it-ebooks.info

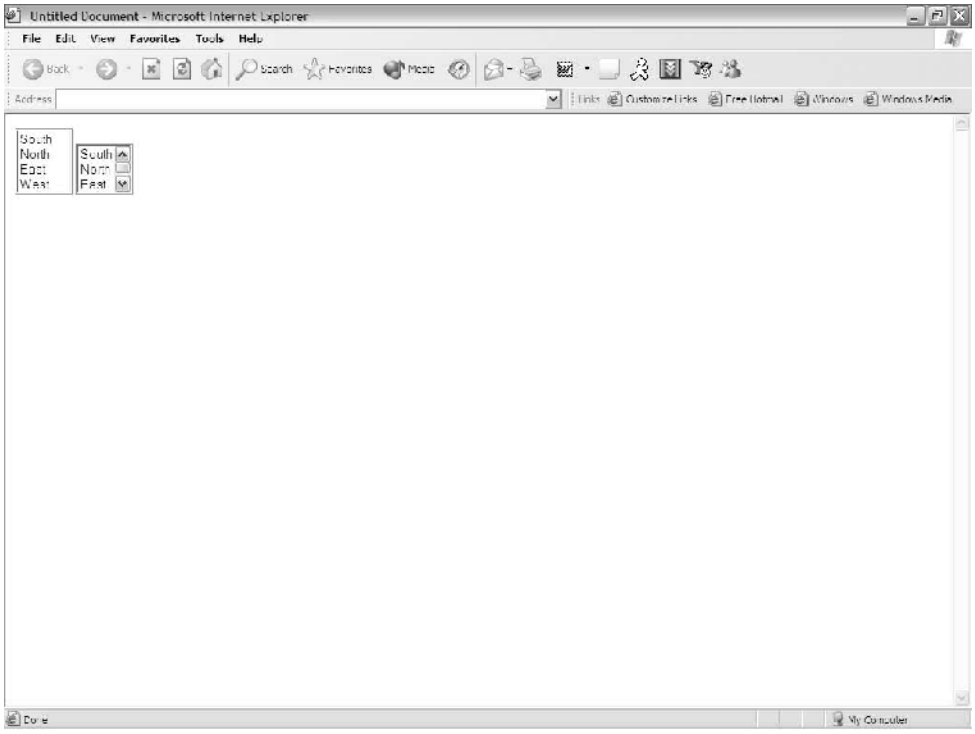


Figure 15-17: List boxes can be set to show different numbers of options.

Allowing multiple selections

Aside from the difference in appearance, one significant difference between pop-up menus and list boxes exists. There are times when you want someone to be able to make more than one choice from among the options, and list boxes enable you to do so using the `multiple` attribute. The following code illustrates its use:

```
<SELECT name="region" size="4" multiple>
<OPTION value="SoReg">South</OPTION>
<OPTION value="NoReg">North</OPTION>
<OPTION value="EaReg">East</OPTION>
<OPTION value="WeReg">West</OPTION>
</SELECT>
```



In fact, you do not even need to specify the size of a list box that has the `multiple` attribute, unless you want the list to show less than the full amount of options. The presence of that attribute automatically changes a pop-up menu into a list box, because pop-up menus don't have multiple choices as a possibility.

www.it-ebooks.info

Figure 15-18 shows a list box with multiple options selected.

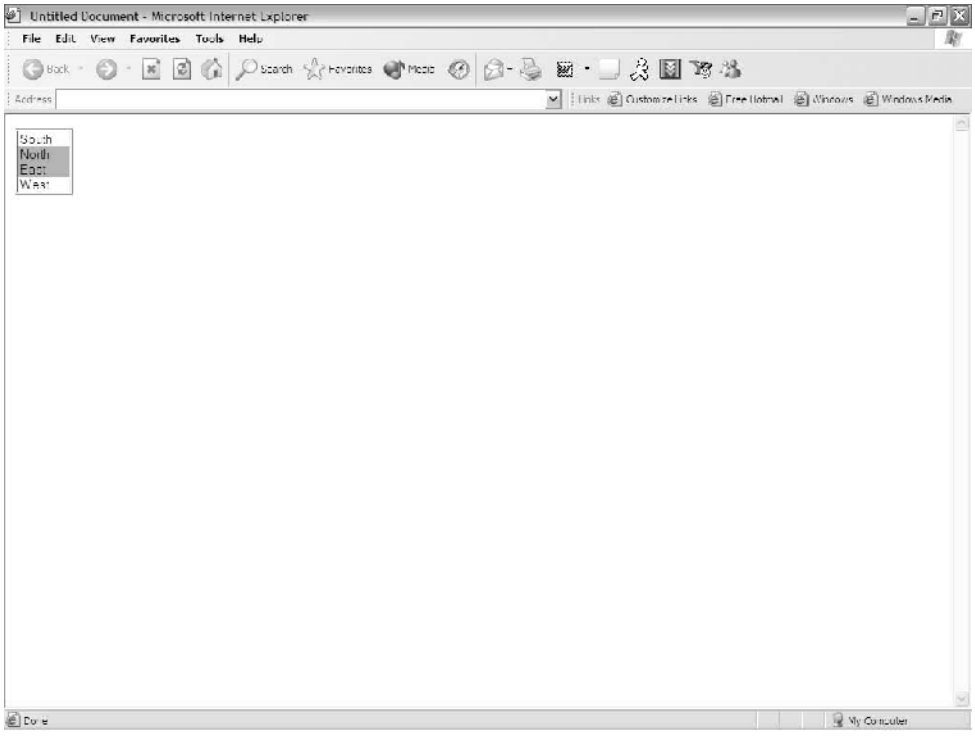


Figure 15-18: List boxes can enable multiple choices.

Setting a default choice

As with check boxes and option buttons, default choices can be set for pop-up menus and list boxes. However, the attribute is different—in this case, the `selected` attribute is used. If you want the first attribute to be selected, you don't have to do nothing anything, because that's the default selection. However, if you wanted, for instance, the third option to be the default choice, you would change the code as follows:

```
<SELECT name="region" size="4" multiple>
<OPTION value="SoReg">South</OPTION>
<OPTION value="NoReg">North</OPTION>
<OPTION value="EaReg" selected>East</OPTION>
<OPTION value="WeReg">West</OPTION>
</SELECT>
```

You can set more than one option to be selected, at least with list boxes. Pop-up menus do not enable multiple selections.

www.it-ebooks.info

Using INPUT Buttons

All forms have buttons that trigger various actions. The most common are the two basic buttons, Submit and Reset. Beyond those two, you can also create custom buttons that do whatever you decide, or you can replace the plain Submit button with a graphical one of your own design. All four of these are various types of the `INPUT` element.

Adding the Submit button

The Submit button does just what it says—it causes the information in the form to be sent to the data processing program. It has no other function, and no form can do anything without this button unless its creator has worked a custom JavaScript program into the equation. The Submit button is required on most forms.



The HTML standard specifies that you can have more than one Submit button in a form. This is for cases when the program that receives the form should do something different depending on the button pressed. For example, the "Next" and "Previous" submit buttons for a series of forms could be used to submit the data regardless of which direction the user chooses and then go to the next or previous form.

The simplest code for inserting a Submit button is this:

```
<INPUT type="submit">
```

There are a number of possible options you can play with. If you don't specify the text to be used on the Submit button, it simply says "Submit Query," which is accurate, but dull. To change the caption on the button to something more interesting such as the one shown in Figure 15-19, you can use the following code:

```
<INPUT type="submit" value="Place My Order!">
```

The button expands, if necessary, to be large enough to display the entire caption.

You can also change the size of the button by specifying values for its `width` and `height` attributes. The following code specifies a Submit button that is 100 pixels wide and 50 pixels high:

```
<INPUT type="submit" width="100" height="50">
```

This is perfectly valid, just as for any other type of the `INPUT` element. However, Internet Explorer does not recognize the values. It always displays a Submit button at the same size unless a custom caption forces it to expand.

Adding a Reset button

The Reset button has only one function—to clear all information that a visitor has entered into the form. It returns all the form controls to their default settings and

www.it-ebooks.info

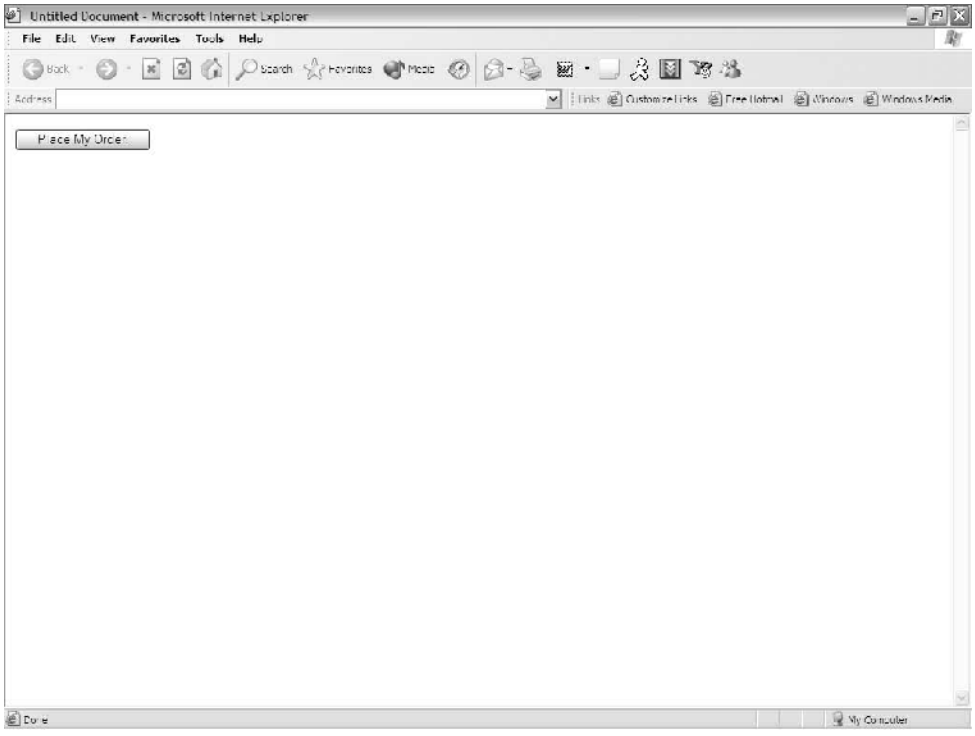


Figure 15-19: A Submit button can say anything you want it to.

restores any initial values you may have specified, such as text area content or selected options in list boxes.

```
.jpg<INPUT type="reset">
```

While the Reset button is present in nearly every form you see on the World Wide Web, its usefulness is open to question. After all, can you ever imagine a circumstance in which somebody would want to start completely from scratch after filling out a form? Granted, a visitor may want to change a thing or two, but we're talking about *everything*. It's a bit unlikely that anyone would complain too much if there were no Reset button to be found.



You can alter the size and caption of the Reset button just as with a Submit button.

Creating custom buttons

Custom buttons are not for the casual Web page designer, because they require the use of JavaScript in order to be effective. Custom buttons, called *push buttons* in the HTML standard, enable you to do just about anything you want, but require a lot more knowledge—and work—on your part than using regular buttons.

www.it-ebooks.info

The code needed to create a custom push button is as follows:

```
<INPUT type="button" name="dosomethingspecial" value="showthistext"
onClick="runthisScript">
```

The value of the `value` attribute is displayed in a Web browser as the text on the custom push button. The onClick value launches a JavaScript program that you have designed to exploit the existence of the custom push button.



See Chapter 16 for information on using JavaScript.



You can alter the size of a push button in the same way as a Submit button.

Going pictorial with graphical buttons

When the basic Submit button that HTML supplies just won't do the job, it's time to create your own. A graphical Submit button won't do anything that the plain Submit button won't do, but it can look a great deal nicer. If you're into plain utilitarianism when it comes to forms, this won't interest you, but if you'd like to toss in a little bit of flair, this is pretty much the easiest way to do it.



See Chapter 9 for information on creating images.

Figure 15-20 shows a graphical Submit button instead of the plain one.



Because this button is designed to replace the functionality of the typical Submit button, it should display some type of text that fits in with that function, such as "Click Me" or "Place My Order."

As with so many other form controls, this is a variation of the `INPUT` element. HTML code, as found in the following example, is required in order to add a graphical Submit button:

```
<INPUT type="image" name="pictureSubmit" src="graphicbutton.jpg">
```



See Chapter 4 for a discussion of URLs and file addresses.

The size of the button is the same as the size of the image. Graphical Submit buttons, because they are images by their very nature, also use the `align` attribute. This affects the display of any other element found in the same line—or, effectively, the same block-level element—with the graphical Submit button.



See Chapter 8 for information on aligning images.

www.it-ebooks.info

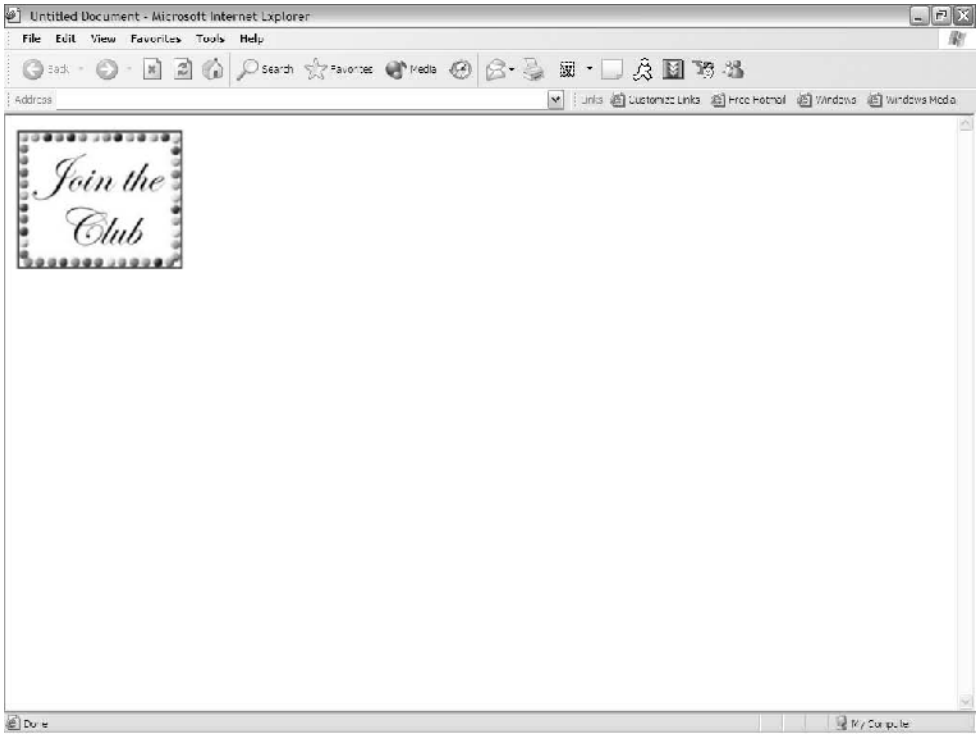


Figure 15-20: You can beautify your forms with graphical Submit buttons.

One of the most fascinating things about graphical buttons is that the HTML standard intended for them to be able to do much, much more than simply replace the usual Submit button with a prettier alternative. A graphical Submit button is, in fact, a miniature image map. When the form information is sent to the data processing program, the location of the pointer when the graphical Submit button was clicked is included along with all the other information.



See Chapter 8 for more information on image maps.

The location of the mouse pointer measuring from left to right (the X axis) and from top to bottom (the Y axis) is sent at the time of submission. Each of these coordinates is appended to the name of the graphical Submit button by adding .x and .y to the button's name. Using the preceding code example as a base, the values of the mouse pointer coordinates would be sent as `pictureSubmit.x` and `pictureSubmit.y`.

This provides one way to create a server-side image map. An example of a good use of this is that instead of presenting all the countries of the world in a drop-down list box on your Web page, you could show a map of the world and prompt the user to click on an area that marks the location where they live. This would add a much more dynamic feel to your Web site.

www.it-ebooks.info

Using the BUTTON Element

Often, there's more than one way to do the same thing in HTML, and each approach has its own ups and downs. A classic example is the `BUTTON` element in forms. We've just covered four different ways to create buttons in forms, all of which use the `INPUT` element and specify a particular kind of button via the `type` attribute. Well, you can do the same thing with the `BUTTON` element's `type` attribute, too. But there's a major difference here—the `BUTTON` element has both a start tag and an end tag, which means that it can hold content. And the content is displayed on the button.

There are three possible values for the `type` attribute for this element: `button`, `submit`, and `reset`. They create, in order, push buttons, Submit buttons, and Reset buttons. The following code, for example, creates a Reset button that is identical in both function and appearance to the standard one:

```
<BUTTON type="reset">
Reset
</BUTTON>
```

The possibilities, however, are many. Think of this kind of button as a tiny Web page. The content can include character styles such as bold and italic lettering. It can be an image or just about anything else. We once added a table to a form button in this way (see Figure 15-21). Not that there's a practical use for doing that, but it does illustrate how much latitude you have.

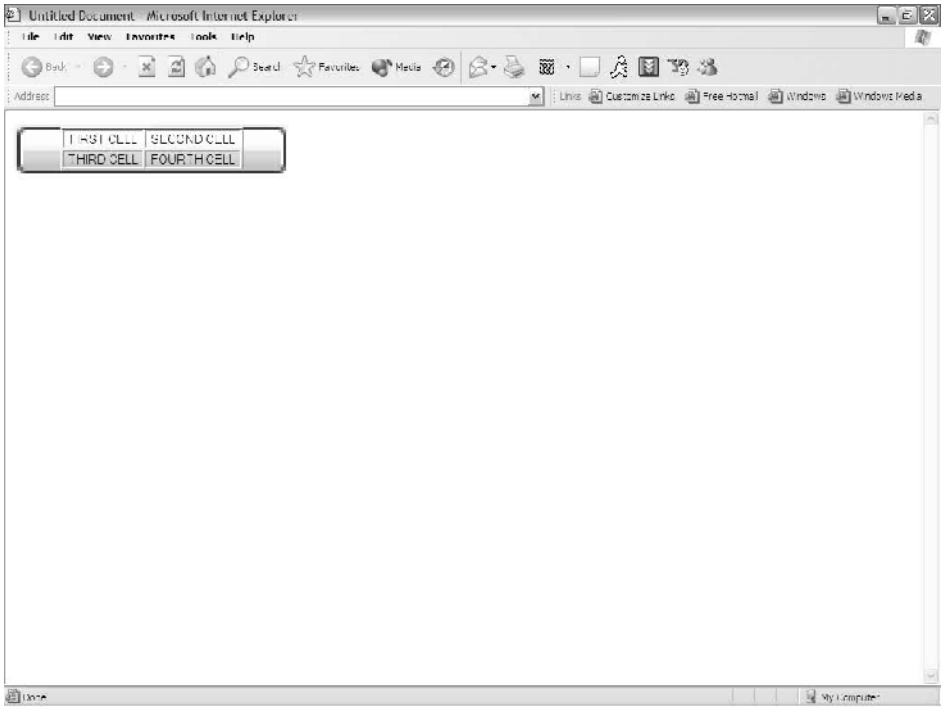


Figure 15-21: You can do just about anything with the BUTTON element.

www.it-ebooks.info

The only thing that is explicitly forbidden to use in a BUTTON element is an image map. However, as with the table-in-a-button example, the fact that you can pull something off in HTML doesn't necessarily mean that it has real value. Because the INPUT element already has the capability to display an image on a button, and the three values of the type attribute for the BUTTON element are duplicates of the ones in the INPUT element, only one true, practical difference exists between INPUT-based buttons and BUTTON-based buttons. The best use of this element is to include formatted text as the button caption. The following code creates a Submit button that uses various text styles to make the final word red in color and two sizes larger than the preceding text:

```
<BUTTON type="submit" name="colorsubmit" value="submit">
SEND IT RIGHT <I><FONT color="#FF0000" size="+2">NOW!</FONT></I>
</BUTTON>
```

Figure 15-22 shows the difference between this custom button and a normal Submit button.

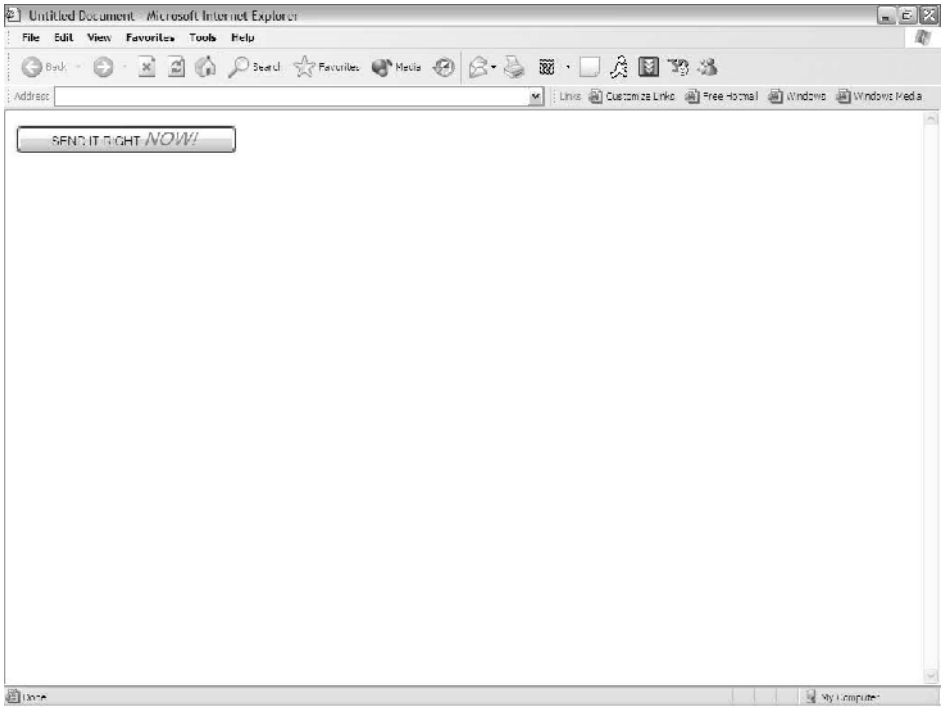


Figure 15-22: Character styles can be used to beautify buttons.

Adding Hidden Fields

For all the information that you want to have, but don't want your visitors to see or alter, hidden fields are the answer. We have already talked about the possibility of setting some form fields so that they could not be altered (see "Locking content with

www.it-ebooks.info

readonly and disabled" earlier in this chapter), but that practically ensures that some of your visitors will become frustrated. After all, the whole purpose of a form is to get user input. If they can't provide that input—or at least make changes to your default entries—there's no point to the display existing at all.

Hidden fields give you the best of both worlds. They enable you to include information that may be important to you, but should not be altered by visitors, and at the same time, shields visitors from the fact of the information's existence. You may, for example, have an identical form on different Web pages and want to know which page the submission came from. This isn't the kind of information that needs to go onto a form to be shown to your visitors, but it's perfect for the `hidden` type.

You can achieve this by using the following code in your forms, changing "`thispage`" to show the name or URL of the page the form is on:

```
<INPUT type="hidden" name="whichpage" value="thispage">
```

You should bear in mind that it is possible for someone to change your page so that it sends a different value. Don't put the price you will charge for some goods in a hidden field and then trust this value when it gets to the server. There is no way around this you just have to take it into account when you design your Web application.

Adding Labels

There isn't, in our opinion, much real use for form labels, and browser support for them is spotty at best. However, they do exist, so we'll briefly talk about them. Almost every form control has some kind of text relating to it. A check box for instance, may have some text that says, "Click in the box if you want to get our monthly newsletter."

Generally, this is accomplished by the simple expedient of entering text into the form along with the form controls, such as this:

```
<FORM>
<P>
First Name:
 <INPUT type="text" name="firstname" size="20" maxlength="50">
</P>
<P>
Last Name:
 <INPUT type="text" name="lastname" size="20" maxlength="50">
</P>
<INPUT type="submit">
<INPUT type="reset">
</FORM>
```

This tends to satisfy the needs of 99.9 percent of form makers. However, there is one small factor to take into consideration when adding labels instead of normal text—and it's the only difference there is—labels are directly associated with the

www.it-ebooks.info

form control they're attached to. This means that, if you click the label, it's the same as if you select the form control. This is hardly a great thing to get excited about, in our opinion, but if it happens to be what you need or want, here's how to go about it.

First, you need to define the text as a LABEL element. Then you need to use the for attribute to explain to the Web browser which form control the label text is attached to. The following variation on the preceding code shows the relationship:

```
<FORM>
<P>
<LABEL for="firstname">First Name:
</LABEL>
<INPUT type="text" name="firstname" size="20" maxlength="50">
</P>
<P>
<LABEL for="lastname">Last Name:
</LABEL>
<INPUT type="text" name="lastname" size="20" maxlength="50">
</P>
<INPUT type="submit">
<INPUT type="reset">
</FORM>
```

You can also go for what is technically known as an *implied relationship* using the LABEL element, where the associated form control is contained within the <LABEL> and </LABEL> tags:

```
<FORM>
<P>
<LABEL>First Name:
 <INPUT type="text" name="firstname" size="20" maxlength="50">
</LABEL>
</P>
<P>
<LABEL>Last Name:
 <INPUT type="text" name="lastname" size="20" maxlength="50">
</LABEL>
</P>
<INPUT type="submit">
<INPUT type="reset">
</FORM>
```

Setting Tab Order

By default, the *tab order*—the order in which the controls in a form become active (or *receive focus*, if you want to use the technical phrase) as a user presses the Tab button on a keyboard—starts from the first form element and ends at the last form element. At that point, instead of wrapping around and beginning the whole process over again at the first form element, the tab order moves to the Address bar of the Web browser. One more press on the Tab button and you're back in the first form control.

www.it-ebooks.info

Generally speaking, this is the way that most Web designers want their forms to work (with the possible exception of the detour through the Address box). However, if you have some unusual situation where you want to change the tab order, it can be done. There is, unfortunately, no way to keep the tab order from including the Address box—this technique is strictly for specifying which form controls are activated in which order.

The `tabindex` attribute works with all the elements used to create form controls, with one exception—the `OPTION` element, where it isn't needed or useful. `OPTION` elements are always contained within `SELECT` elements, and the tab order of the `SELECT` element would automatically encompass its contents.

The form control with the lowest `tabindex` value is the first one in the tab order, whereas the one with the highest `tabindex` value is the final one in the tab order. In the following example, the Submit button is first, whereas the `textarea` is second:

```
<FORM>
<TEXTAREA>
First Name:
 <INPUT type="text" name="firstname" size="20" maxlength="50">
</TEXTAREA>
<P>
<INPUT type="submit" name="lastname">
</P>
<INPUT type="submit">
<INPUT type="reset">
</FORM>
```

Specifying Access Keys

Access keys—having nothing to do with handicapped access—are common features in all programs. For example, nearly every program already defines Ctrl+S as the way to save a file and Alt+F as the way to open the File menu. So you might confuse users if you try to use access keys with forms.

The access key you assign for your form overrides the same one in the Web browser. Internet Explorer uses already F, E, V, A, T, and H for its own menu access keys. Even if your needs can be taken care of by working around those letters, you have to consider the possibility that future versions of Internet Explorer may have different menu access keys.

With these caveats in mind, if you still want to assign access keys to your form, it's a simple enough matter. Just add the `accesskey` attribute and specify which key you want to use as the value:

```
<INPUT type="text" name="initial" accesskey="I">
```



It doesn't matter if you use a capital or lowercase letter for the value.

You can even use access keys to submit the form by specifying the `accesskey` attribute for the Submit button.

www.it-ebooks.info

When an access key has been assigned, Windows and Linux users can access the control by using the access key in combination with the Alt key; Mac users use the Command key instead.

Because there is no indication to the user that the access key exists, you'll have to take care of that yourself. If you want to use the standard approach of showing the access key by underlining the appropriate letters, you could do it as follows:

```
<P><U>I</U>initial: </P>
<INPUT type="text" name="initial" accesskey="I">
```

Figure 15-23 shows the results.

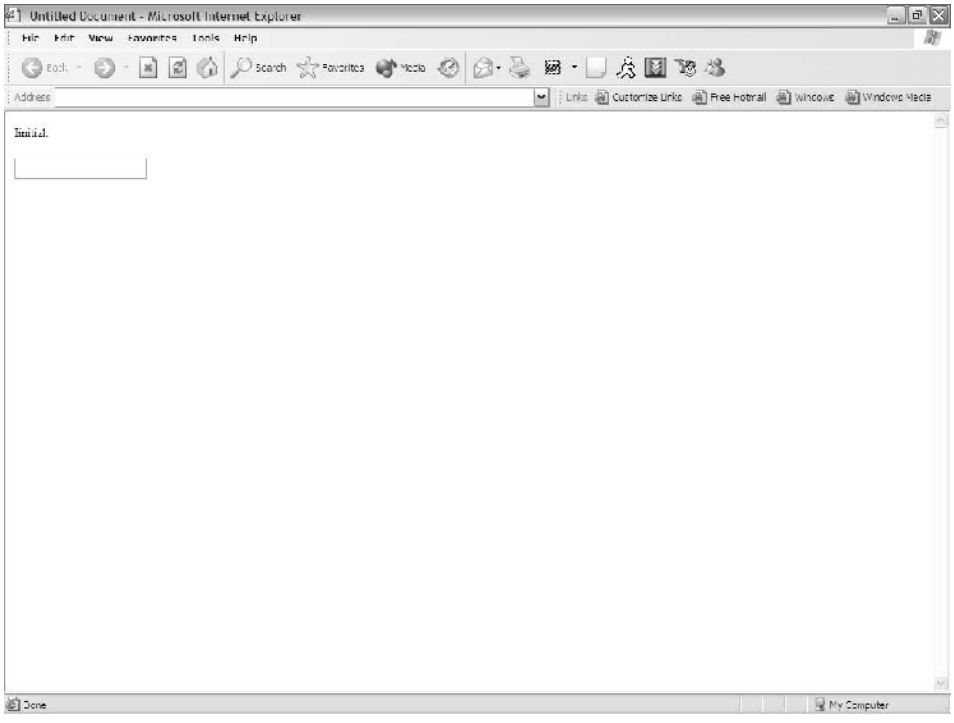


Figure 15-23: This is one example of how to use access keys in your form.

Submitting the Form

There are two attributes of the `FORM` element that need to be specified for a form to be submitted properly when a user clicks the Submit button: `action` and `method`. The action is the URL of the program that handles the submitted data. The method specifies the manner in which the data is transmitted.

Two different possibilities exist for the value of the `method` attribute: `get` and `post`. Of the two, `post` is generally preferred, because it is the most flexible and efficient. The get method works by appending the name/value pairs from the form to the URL

www.it-ebooks.info

specified in the `action` attribute, which limits the amount of data that can be sent using that approach.

Since the information that the `get` method sends is visible in the navigation bar, it's not a good option to use if you want to keep your password secret. Also, since the browser remembers the URLs for the pages you visited, if you visit the page again by clicking on the back arrow the browser may send the form information again (not too good of an idea if the form was to transfer money from your bank account). However, the `get` method is great for search engines, particularly where you want people to be able to bookmark a search and it doesn't matter how many times it is executed. The `post` method, on the other hand, sends the form data separately from the URL, thus allowing it to send an unlimited amount of information. If you want to use a file input box, you have to use the `post` method and specify the content encoding of the form using the attribute `enctype="multipart/form-data"`.

There are various ways to handle the form data. Some of the most common are Java Servlets, Java Server Pages (JSP), PHP programs, Active Server Pages (ASP), and CGI programs.

Java runs on any platform—including Windows, Mac, Linux and Unix. Also, the software to create the pages and programs as well as the application server to run them is free of charge. If you have worked with JavaScript, you'll find the Java language is quite similar.

PHP, which started up as a scripting language to make personal Web sites, but is used now in large systems, is also free and works on any platform.

Microsoft Active Server Pages (ASP) are similar to Basic and run in the Internet Information Server (IIS). They have the disadvantage that the server has to be running a Microsoft operating system, and currently more than half the servers on the Internet don't.

CGI (Common Gateway Interface) programs are a means of getting information from Web pages into external programs for processing. Thousands of CGI programs are available either at no cost or for a very reasonable fee. However, they have the disadvantage of starting a separate process on the server for every form that is sent. This may not seem like a problem, but when you suddenly find your site on the Internet news site Slashdot (`http://slashdot.org`) you might have performance issues. There are several technologies designed to be more efficient. Most of these combine a programming language with HTML in a hybrid format.

To illustrate how form submission works, if you had a page that was designed to enable registered users to log in to a private section on your site, it would have to accept as its input the user's name and password. You could pass this data to a program called, let's say, `checkident.pl` that's kept in your `cgi-bin` folder. The following form would handle all the necessary parts:

```
<FORM action="cgi-bin/checkident.pl" method="post">
<INPUT type="text" name="username">
<INPUT type="password" name="pw">
```

www.it-ebooks.info

```
<INPUT type="submit">
</FORM>
```

In addition to programs that you house on your own Web server, a growing number of programs are hosted by other company's Web servers. You can, nonetheless, use a remotely hosted program as the action for your own form.

For example there is a form mailer at `www.hazlorealidad.com` which is free. To use it you specify the form action as `www.hazlorealidad.com/mailer`, which sends the request to a Java servlet. This in turn sends the form contents by e-mail to a specified address and redirects the browser to the next page in your site. Note that you need to specify the mail address and the next page to be shown as hidden fields in the form. Full instructions are at `www.hazlorealidad.com/formmailer.html`.

```
<FORM action="http://www.hazlorealidad.com/mailer" method="post">
<INPUT type="hidden" name="mailto" value="myemail@myhost.com">
<INPUT type="hidden" name="redirectto"
value="http://www.myhost.com/next.html">
<INPUT type="text" name="username">
<INPUT type="text" name="comment">
<INPUT type="submit">
</FORM>
```

Summary

Forms are the one major way for gathering user input on Web pages. Although you may devise other methods using sophisticated programming techniques, this is the way that most of your visitors are used to. This chapter introduced you to the following concepts and topics:

- The INPUT element is the only one you have to use in order to create workable forms.
- Each element within the form should be given a distinctive name so that it will be clear what kind of information is presented when the form is submitted.
- When writing JavaScript programs, each element should also have a unique `id` attribute so that it can be referenced.
- Text boxes—single line, rectangular areas—are useful for the gathering of information like names and addresses.
- Text areas—multiline, rectangular areas—are used for getting more complex information like shipping instructions.
- Check boxes are used for making choices among several nonexclusive alternatives.
- Options buttons are used for making choices between absolutes.
- The SELECT element is used to present multiple options in a list.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 16

Making Dynamic Pages with JavaScript

In This Chapter

Trying out JavaScript

Understanding JavaScript

Using events to trigger scripts

Seeing elements as objects

Examining browser compatibility

JavaScript reference

Many people confuse JavaScript with Java. It's understandable, given the similarity in their names, but the two are not the same. Although some of JavaScript is somewhat similar to Java, Java is closer to C++—all programming languages bear some degree of resemblance to one another. Java is a major language that is used for developing standalone programs, many of which are huge and extremely complex. JavaScript, on the other hand, is used to write short and simple programs that can run only in Web browsers.

Of course, you can use Java to write short and simple programs; many of the Java applets available to use on Web pages certainly fit this description. And you can also use JavaScript to create some incredibly intricate interactive programs on your Web pages. As with anything else on the Web, the limits are mainly up to you.



See Chapter 20 for information on adding Java applets to your Web pages.

Trying Out JavaScript

Before we get into any of the details of how JavaScript works, let's see just how simple it can be. Type the following code into your HTML editor:

```
<HTML>
<HEAD>
<TITLE>JavaScript Is Great</TITLE>
</HEAD>
<BODY onLoad="alert('Hi!')" onUnload="alert('So long -
hurry back!')">
```

www.it-ebooks.info

```
<P>Visit <A href="http://www.wiley.com/"> </A>.</P>
</BODY>
</HTML>
```

Save the file, and open it in a Web browser. The first thing you'll see is the pop-up message window shown in Figure 16-1. Click OK to make the pop-up go away. Then click the link. You'll get another pop-up message with the "So long" message. Click its OK button and the link activates.



Figure 16-1: A JavaScript pop-up message window.

Simple, isn't it? Yet in this short example, you've already trapped two events, used that fact to invoke the appropriate event handlers, and called on a built-in method of the window object, feeding it two different parameters to display two different pop-up messages. In plain English, you told the Web browser to display one message when the Web page is opened and another message when it went to another location.

Understanding JavaScript

JavaScript *isn't* Java, and it *isn't* HTML, either. JavaScript was built from the bottom up, however, to work together with HTML so that Web page designers could create programs to read and alter the values of HTML elements.

From time to time, people still ask if scripts are "really" programs. The answer is yes. A program is a set of instructions to be carried out by a computer, and that's just what JavaScript creates. JavaScript, while a simpler language than many, is indeed a viable programming language in its own right.

We can't possibly cover everything that you can do with JavaScript in a single chapter. However, we can give you everything you need to know to begin using the language. In subsequent chapters, we show you how JavaScript can be applied to specific situations to help you make your Web pages work better.

Working with variables and literals

Variables are the heart of programming. Without them, life would be much more difficult. A variable is an item that holds a value that is expected to change over time. The assignment operator is an old friend you've known since elementary

www.it-ebooks.info

school—the equal sign (=). If you want to assign the value `42` to the variable `answer`, for example, you simply write `answer=42`.



The variable must always be on the left side of the assignment operator. The value to be assigned to it must always be on the right side of the assignment operator.

Declaring variables

Before it is actually used in a script, a variable should be *declared*, which means that you're giving the Web browser notice that it eventually has to deal with the variable. Variable declaration is done with the `var` keyword:

```
var answer
```

You can kill two birds with one stone by assigning an initial value to the variable at the time of declaration:

```
var answer=42
```

You do not, however, have to assign a value at declaration, nor do you *have* to use the `var` keyword. In fact, you often see JavaScript programs that have a line such as this in them:

```
textEntry = window.document.orderNow.visitorName.value;
```

In this case, the variable `textEntry` is both created and fed a value in one statement. It is, however, a good idea to explicitly declare the variable.

Naming variables

Although you have wide latitude in naming variables, the names should be descriptive of the kind of information they hold. The current temperature, for instance, might be expressed by the variables `currentTemperature`, `currTemp`, or even `cT`. If you wanted to specify that it was in Fahrenheit, you might invent the variables `degreesFahrenheit`, `degF`, or `dF` instead.

It is customary, but not required, to start variables that consist of more than one word with a lowercase letter, and then to uppercase the first letter of each subsequent word. However, bear in mind that JavaScript is case sensitive, so whatever capitalization you use has to be consistent. If you call a variable `shirtSize` in one place and `ShirtSize` in another, don't expect your program to work properly.

There are a few restrictions on how you can name your variables. For example, you cannot give a variable the same name as any of the JavaScript reserved words (see the JavaScript Reference in Appendix D for a list of reserved words). Your variable names cannot have any spaces in them, either. Finally, keep in mind that you cannot begin a variable name with anything but a letter or the underscore character (_). Table 16-1 shows examples of valid and invalid variable names.

www.it-ebooks.info

Table 16-1
Valid and Invalid Variable Names

Valid Example	*Invalid Example*	*Explanation*
number1Choice	#1Choice	The invalid example begins with something other than a letter or underscore.
moonPhase	moon Phase	The invalid example has a space between the words.
_switch	switch	The invalid example is a reserved word. The valid one, even though technically legal, isn't a good idea, either, because it's just too close to the reserved word for comfort. A simple typographical error could easily change `_switch` into `switch`.

As long as the name is legal and you know what the variable name means, that's fine. But it's a good idea to bear in mind that you may have to come back to the script one day and revise it. And that day may be months or even years after you wrote it—months or years during which you've written many other scripts that all tend to blur together in your memory. It's always tempting to go for a very short and pithy variable name, but the more clearly it expresses its meaning, the easier it is for you to work with it over time.

Assigning different values

Once you've decided on a name for your variable, you can continue to assign different values to it forever. The `currentTemperature` variable may hold a value of `16` on a winter morning, `37` that same afternoon, or `105` in the depths of August. No matter what value is assigned to it, the variable itself remains the same, only its value changes. A *literal*, on the other hand, is a fixed value that never changes. You could set up a variable called `yardstick` that you assigned a value of `36` to, but there's no point in that because a yard is always 36 inches long. The value never changes, and the whole idea of variables is that they are a way to cope easily with changing values.

Value types

A value, whether literal or held in a variable, can be one of several types. The three most commonly used are numeric, string, and Boolean:

- **Numeric:** Just as it sounds—numbers such as 8 or 37.
- **String:** Alphabetical characters or, to be more accurate, alphanumeric characters. Both "John ate a hot dog" and "78" are strings because the number 78 is enclosed in quotation marks. If you intend to use numbers for calculation instead of using them as words in their own right, they should be numeric, not string.
- **Boolean:** Logical values; the only two possibilities are true and false.

www.it-ebooks.info

Affecting values with operators

You've already met one JavaScript operator, the assignment operator (=), earlier in this chapter in the section "Working with variables and literals." There are, however, many more, each of which performs a different task.

Comparison operators

The equality operator, which tests to see if two expressions are equal, is two equal signs (= =). If you need to see if A is equal to B and take different actions depending on the answer, you could use code such as the following example:

```
if (A==B) {
actionOne;
}
else {
actionTwo;
}
```

See the section "Making choices with If and If . . . Else" later in this chapter for details on how the if statement works.



It's really, really easy to confuse the assignment operator (=) with the equality operator (==), even for professional programmers. After all, unless you learned programming in kindergarten, you were using "=" to mean "equals" long before you ever tried programming, and it's pure reflex to type it that way instead of "==". This is probably the single most common programming bug. If your script isn't working right, this is one of the first things to look for.

The following example shows the right way to use both the equality operator and the assignment operator:

```
if (A==B) {
B=C;
}
else {
B=D;
}
```

The meaning of this example, step by step, is that the script looks first to see if variable A has the same value as variable B. If it does, B gets the value of C assigned to it. If not, B gets the value of D assigned to it. Whichever result you get for the equality comparison, the value of B changes. The values in the other two variables—C and D—remain unchanged because assignment is strictly a one-way street.



The assignment of a value always goes from right to left. Thus, the value in the variable on the left ends up the same as that in the variable on the right.

Other commonly used comparison operators include the following:

- ✦ !=: Not equal to
- ✦ >: Greater than

www.it-ebooks.info

- <: Less than
- >=: Greater than or equal to
- <=: Less than or equal to

Arithmetic operators

The arithmetic operators used most often reflect, of course, the most common mathematical operations: addition, subtraction, multiplication, and division. Also, the unary negation operator and the increment and decrement operators come into play quite often.

- +: Addition. Adds two values together. When used with string values, it concatenates the strings—in other words, makes them into one string. In that case it is known as the concatenation operator.
- -: Subtraction. Subtracts one value from another.
- *: Multiplication. Multiplies one value by another.
- /: Division. Divides one value by another.
- -: Unary Negation. Although this uses the exact same sign as the subtraction operator, there is a subtle shade of difference. It is placed before a variable (literal numerical value) to reverse its current status. If the value is negative, the result becomes positive; if it is positive, the result becomes negative.
- ++: Increment. Increases the value by 1. If placed before a variable (as in `++variable`), the value of that variable is returned, and then the increment takes place. If placed after a variable (as in `variable++`), the value of the variable after the increment takes place is returned.
- --: Decrement. Decreases the value by 1. If placed before a variable (as in `--variable`), the value of that variable is returned, and then the decrement takes place. If placed after a variable (as in `variable--`), the value of the variable after the decrement takes place is returned.



The + operator, which is used either to add two numerical values or concatenate two string values, has a built-in solution for situations in which a JavaScript program tries to add a numerical value and a string value. It assumes that you mean the number to be interpreted as a string—as if it were within quotation marks—and converts it for you. Thus, if you try something such as `A = 78 + "modem"` you get "78modem" as the result. Also, `Y = "37" + 10` gets you "3710" instead of 47.

There is a special kind of shorthand you can perform with the basic arithmetic operators to handle common situations such as the following one:

```
tax = tax - deductions
```

Alternatively, this can be expressed in JavaScript shorthand like this:

```
tax -= deductions
```

www.it-ebooks.info

The addition, multiplication, and division operators can also be handled this way. `A += B` is the same as `A = A + B`, just as `A *= B` is the same as `A = A * B`, and `A /= B` is the same as `A = A / B`.

Logical operators

The three logical operators common to all programming languages are AND, OR, and NOT. When two values are compared with an AND operator, both of them must be true or the answer to the AND comparison is false.

When two values are compared with the OR operator, either one of them may be true and still return a value of true. Only if both of them evaluate to false is a false return possible.

The logical NOT, on the other hand, is the logical equivalent of the arithmetic unary negation operator. If the value being evaluated is true, it returns false; if the value is false, it returns true.

These are expressed in JavaScript as follows:

- &&: AND
- ||: OR
- !: NOT

Adding statements

Any time you use JavaScript to tell a browser to do one thing, you're making a *statement*. Each line in the following code is a separate statement:

```
document.write("This is being added to the page.");
alert("This is in a popup message window.");
globe = round;
```

The three ways to put statements into your Web pages are: as attributes of an HTML element, as plain statements in the contents of a `<SCRIPT>` tag, and as part of a function in the contents of a `<SCRIPT>` tag.

HTML elements, of course, have various attributes. The `BODY` element, for instance, has a background color attribute, `bgcolor`. If you want to specify a value for that attribute, use this code:

```
<BODY bgcolor="blue">
```

In addition to the simple attributes you're used to, the elements also have some special attributes called *event handlers*. Event handler attributes are assigned values just as with any other attribute. You put the name of the attribute in, follow it with an equal sign, and then put the value within quotation marks. The following line

www.it-ebooks.info

from the opening example in this chapter shows how the process is identical to the way a background color is assigned. The only difference is the name of the attribute and the value that is assigned to it. The value, in this case, is a JavaScript statement.

```
<BODY onLoad="alert('Hi!')" onUnload="alert('So long - hurry back!')">
```

See the section "Using Events to Trigger Scripts" later in this chapter for more information on event handlers.

With event handler attributes, you can even specify multiple actions to be taken for a single attribute. Just separate each statement with a semicolon:

```
<BODY onLoad="alert('Hi!');alert('Welcome!')">
```

This causes the first statement—`alert('Hi!')`—to take place first, launching a pop-up window that contains the text `Hi!`. When you've finished reading that statement and click OK, the next statement executes, launching the second pop-up message window as shown in Figure 16-2.



Statements always execute in the order in which they appear.



Figure 16-2: The second statement executes.

The eagle-eyed will have already noticed the use of two different kinds of quotation marks—double quotation marks (") to contain the attribute's value and single quotation marks (') within the statements themselves. This is a critical factor. If you tried the following line instead, you'd be creating trouble:

```
<BODY onLoad="alert("Hi!");alert("Welcome!")">
```

The problem here is that the Web browser knows that the first double quotation mark shows where the attribute's value begins, and the next one is where it ends. Thus, a Web browser reads this value as `alert(` and comes to a screeching halt because it's a meaningless value. You can, if you want to, use single quotation marks

www.it-ebooks.info

to delimit the attribute's value and double quotation marks inside the statements. The following code is perfectly valid:

```
<BODY onLoad='alert("Hi!");alert("Welcome!")'>
```

As long as you use a different kind of quotation mark to set off the attribute's value from the contents of the statements, you'll be fine.

Using the <SCRIPT> tag

Another way to add statements is within the <SCRIPT> tag:

```
<SCRIPT language="JavaScript">
document.write("This is the first line.<BR>");
document.write("This is the second line.<BR>");
document.write("This is the third line.<BR>");
document.write("This is the fourth line.");
</SCRIPT>
```



It is technically not necessary to end a statement inside a <SCRIPT> tag with a semicolon (;), as long as each statement is on its own separate line. It is simply customary, and it's a good idea to do so because it keeps you in the habit of separating statements with semicolons, which *is* required when you use multiple statements within event handler attributes.

The first line opens with the <SCRIPT> tag to tell a Web browser that it's about to encounter a script, and it specifies JavaScript as the scripting language using the language attribute. If you do not specify a language, JavaScript is assumed to be the language used. However, nobody can guarantee that this will always be the case, so it's best to go ahead and specify it.

The four statements simply add four separate sentences to a Web page, each one placed on a different line because there is a line break (BR element) included at the end of each of the first three sentences (see Figure 16-3). Candidly, there are much better ways to achieve the same end, but it illustrates exactly how multiple statements have to be included.

The </SCRIPT> end tag specifies that there is no more script to follow.

HTML comments are normally used to add notes on the meaning of the HTML code on a Web page. This helps both the original author and anyone else who works on the Web page later to grasp the meaning of any complex or unusual code. They have a special use when it comes to JavaScript, however. Because several Web browsers in use still do not support JavaScript, it is customary to wrap the material inside <SCRIPT> tags within HTML comments. These comments have a start tag of <!-- and an end tag of -->. The following shows how they're used:

```
<SCRIPT language="JavaScript">
<!-- Make this script invisible to old browsers.

document.write("This is the first line.<BR>");
```

www.it-ebooks.info

```
document.write("This is the second line.<BR>");
document.write("This is the third line.<BR>");
document.write("This is the fourth line.");

// Back to normal. -->
</SCRIPT>
```

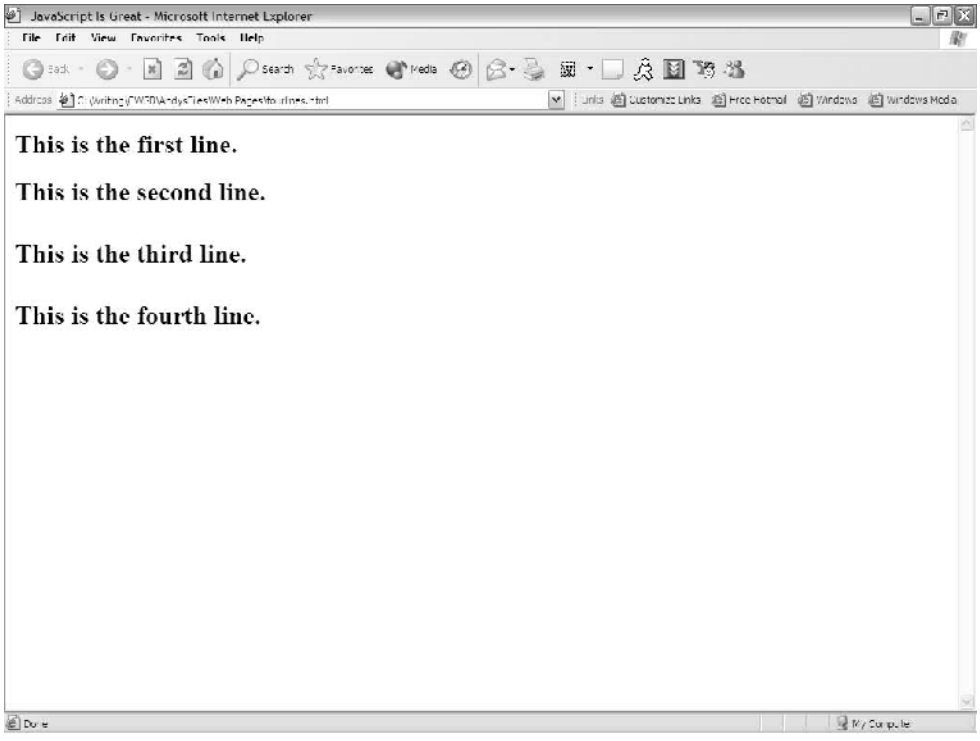


Figure 16-3: Four statements are executed.

The reason this works is that a Web browser ignores anything within HTML comment tags. It also ignores any tags it encounters that it doesn't understand, which is why the <SCRIPT> and </SCRIPT> tags don't have to be inside the HTML comments. If a non-JavaScript browser encountered a script that wasn't inside comments, however, it would treat it as text to be displayed on the Web page.

Understanding the double slash (//) comment marker

But what about the double slash (//) at the beginning of the last line before the end tag? That's a comment marker, too, but it's a JavaScript comment this time, not an HTML comment. Take a moment to relax; it's okay if you're a bit confused at this point. Just remember that JavaScript isn't HTML, and HTML isn't JavaScript. If that last line just read "Back to normal" and ended in an HTML comment end tag, then the JavaScript interpreter wouldn't much like it. Although the interpreter would ignore

www.it-ebooks.info

the HTML comment end tag (just as it ignores the HTML comment start tag), it would see the "Back to normal" part as just plain text, neither HTML nor JavaScript.

Including comments

JavaScript comments are used for the same purpose as HTML comments—to add commentary to the Web page that won't be interpreted as any kind of actionable code. The following bit of JavaScript, for instance, includes a notation on its meaning:

```
alert("You have entered an invalid e-mail address.");
// Notifies the visitor that they need to go back and change the entry in
the e-mail address text box.
```

The // method is one of two ways to add comments to JavaScript, and it has a simple limitation—it hides only one line. It doesn't matter how long that comment is, as long as you don't press your Enter key before you're done with it. If you press your Enter key to add a line break, however, you begin typing on a new line. In the following example, everything after that is outside the comment:

```
alert("You have entered an invalid e-mail address.");
// Notifies the visitor that
they need to go back and change the entry in the e-mail address text box.
```

If you need to make long comments, there is an alternative method, which uses both a start and end tag. The start tag for JavaScript comments is /* and the end tag is */. The following code shows how they're used:

```
alert("You have entered an invalid e-mail address.");
/* This is the alert message that corresponds
to one of the new form entry checking algorithms
developed by XYZ Corporation. They were added to
our site in February. This one notifies the visitor
that they need to go back and change the entry in
the e-mail address text box.*/
```

Processing statements

The <SCRIPT> tag can be placed anywhere on a Web page, either in the HEAD or the BODY elements. However, you must remember that statements are processed in the order they're found. This means that if you place a statement before the things it acts on, then you're asking for trouble. Consider the following bad example:

```
<HTML>
<HEAD>
<TITLE>Do Not Attempt This At Home</TITLE>
</HEAD>
<BODY>

<SCRIPT language="JavaScript">
<!-- Make this script invisible to old browsers.
document.write("The initial value was John.<BR>");
```

www.it-ebooks.info

```
document.write("The current value is " +
window.document.formOne.firstName.value);
// Back to normal. -->
</SCRIPT>
<FORM id="formOne" name="formOne">
<P>Enter your first name: </P>
<INPUT type="text" id="firstName" name="firstName" value="John">
</FORM>

</BODY>
</HTML>
```

When you load this example into a Web browser, only the first `document.write` statement takes effect because the second one requires a value that hasn't been defined yet—the value in the text box named `firstName`, which is contained in the form named `formOne`. If the code is modified to place the `<SCRIPT>` tag and its contents after that form and text box are defined, however, it's a different story:

```
<HTML>
<HEAD>
<TITLE>This Time, It Works</TITLE>
</HEAD>
<BODY>

<FORM name="formOne" method="post">
<P>Enter your first name: </P>
<INPUT type="text" name="firstName" value="John">
</FORM>

<SCRIPT language="JavaScript">
<!-- Make this script invisible to old browsers.
document.write("The initial value was John.<BR>");
document.write("The current value is " +
window.document.formOne.firstName.value);
// Back to normal. -->
</SCRIPT>

<P>Enter a new name, then click on the "Refresh" or "Reload" button at the
top of your browser.</P>

</BODY>
</HTML>
```

Now, when the page opens, both `document.write` statements work. Because the initial value is still the same as the current value, both say the value is John. Type a different entry into the text box, and then click your browser's Refresh or Reload button, however, and the current value shown by the second `document.write` statement changes, as shown in Figure 16-4.

Encountering unexpected behavior

This is probably as good a place as any to point out that Web browsers and JavaScript don't always do what you expect them to do. Generally, when you reload

www.it-ebooks.info

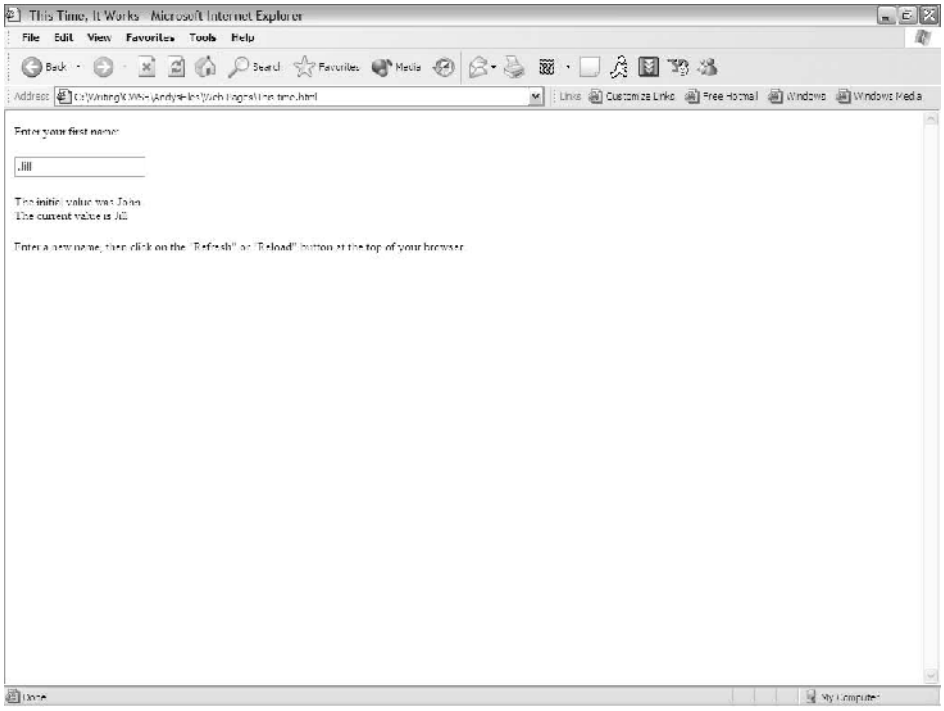


Figure 16-4: This figure illustrates displaying a changed value.

a Web page in a browser, everything is reset to its initial values. However, this doesn't work when the value in question is held in memory by the browser during the page refresh, which is why the second `document.write` statement is able to display the changed value even though the page has been reloaded. To contrast this with the expected behavior, here's a version that forces a total page refresh, including resetting the values in the browser's memory, using JavaScript:

```
<HTML>
<HEAD>
<TITLE>Click the Button</TITLE>
</HEAD>
<BODY>

<FORM name="formOne">
<P>Enter your first name: </P>
<INPUT type="text" name="firstName" value="John">
<INPUT TYPE="button" VALUE="Click Here"
onClick="self.location.reload()">
</FORM>

<SCRIPT language="JavaScript">
<!-- Make this script invisible to old browsers.
document.write("The initial value was John.<BR>");
document.write("The current value is " +
window.document.formOne.firstName.value);
```

www.it-ebooks.info

```
// Back to normal. -->
</SCRIPT>

</BODY>
</HTML>
```

The resulting Web page is shown in Figure 16-5.

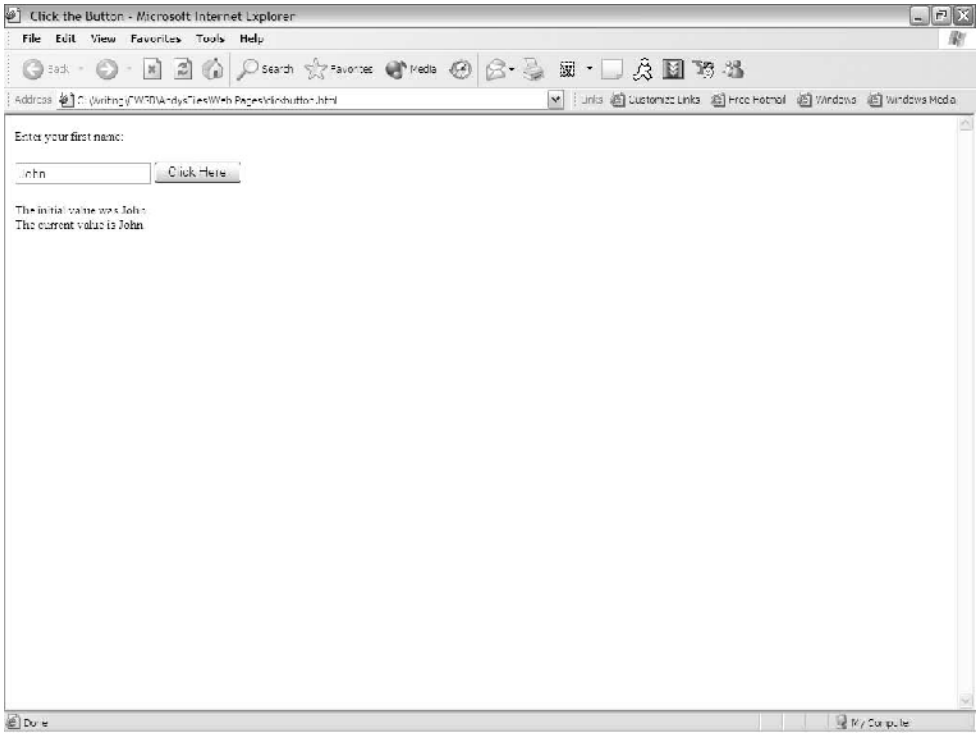


Figure 16-5: Refreshing a Web page.

The key line here is the addition of the button to the form:

```
<INPUT TYPE="button" VALUE="Click Here"
onClick="self.location.reload()">
```

The `self` portion works just as the reserved name `_self` does with frames—it opens the new page in the currently active window. The `location` portion specifies that the current Web page is the one to be used, and the `reload()` portion clears the memory and refreshes the page.



See Chapter 12 for information on working with frames.

www.it-ebooks.info

Combining statements into functions

A *function* is a grouped series of statements that execute one after another. They are used to create more complex programming constructs than can be accomplished with simple statements alone. The basic layout isn't much different from any series of statements:

```
<SCRIPT>
function functionName(){
firstStatement;
secondStatement;
thirdStatement;
}
</SCRIPT>
```

The only differences are that a function is declared with the `function` keyword, which is followed by the name of the function, a closed pair of parentheses, and then an opening curly bracket. The statements are then listed as usual, and a closing curly bracket ends the function.

Calling a statement

A standalone statement or series of statements, however, runs automatically. Statements that are inside of functions run only when *called*. There's nothing mysterious about calling a statement. Just as you might call a friend's name to attract her attention, you call a function by using the function's name.

The following example illustrates how this works:

```
<HTML>
<HEAD>
<TITLE>Reading and Displaying a Value</TITLE>

<SCRIPT>
function getTheName(){
var textEntry;
textEntry = window.document.orderNow.visitorName.value;
alert(textEntry);
}
</SCRIPT>

</HEAD>
<BODY>

<P>Enter your name, then click on the button.</P>

<FORM name="orderNow">
<INPUT type="text" name="visitorName">
<INPUT type="button" value="Click Here" onClick="getTheName()">
</FORM>

</BODY>
</HTML>
```

www.it-ebooks.info

Here's what happens in the code, step by step:

1. The function called `getTheName()` is declared. The Web browser reads it and stores all the information in it in memory, including the existence of the variable, `textEntry`. Although the function is ready to go whenever it is invoked, it does nothing at all at this time.
2. The visitor follows the instructions, makes an entry into the text box, and then clicks the button specified in the form.
3. The button's `onClick` event handler calls the `getTheName` function.
4. The current value of the text inside the text box is assigned to the variable `textEntry`.
5. The `alert()` method is triggered, with the `textEntry` value as its parameter. As a result, the value of the text inside the text box is displayed in a pop-up message box, as shown in Figure 16-6.



Figure 16-6: This figure shows the displays a value.



The `alert()` method is a built-in method of the window object and can also be written as `window.alert()`.

Understanding the scope of variables

The *scope* of a variable is the part of a Web page that it exists in. A variable that is declared before any functions are defined can be used in any function or by any statement that follows. A variable that is declared within a function, on the other hand, has scope only within that function.

This means that you can declare many different variables, all of which have the same name, without creating a conflict—*as long as those variables are declared within separate functions*. This is one of those capabilities that it's good to be leery of, however, because you might cause confusion.



A common example of this type of variable name duplication is the `count` variable that is normally used in loops. See the section "Going in loops" later in this chapter.

www.it-ebooks.info

Making choices with If and If . . . Else

A program, as mentioned earlier, is a series of instructions that are carried out in sequence. The sequence, however, doesn't have to be the same every time. The `if` statement and the slightly more complex `if...else` statement give a program the power to make decisions based on current conditions.

In plain English, an `if` statement says, "If the condition is true, then take this action. If it isn't true, then don't take any action." The JavaScript code version is as follows:

```
if (thisIsTrue) {
doThis;
}
```

The condition represented by `thisIsTrue` can be any kind of comparison test. The part that reads `doThis` can be replaced with any JavaScript statement or function. In the following example, we test to see if the value of the variable `x` is less than 3. If it is, then the variable `y` gets assigned a value of 12 and the variable `z` gets assigned a value of 53:

```
if (x < 3) {
y = 12;
z = 53;
}
```

If `x` is equal to or greater than 3, however, nothing happens. The variables `y` and `z` are not changed, and the program flow simply continues on down to the next line of the code.

The `if...else` statement allows for a similar, but more complex, branching decision. This one might be expressed in English as "If the condition is true, then take the first action. If it isn't true, then take the second action." The JavaScript version is as follows:

```
if (thisIsTrue) {
doThis;
}
else {
doThat;
}
```

The following example tests to see whether the value of the `party` variable is "`democrat`" and invokes different functions to display one message if so and another message if not:

```
if (party = "democrat") {
dem_message();
}
```

www.it-ebooks.info

```
else {
rep_message();
}
```

If you need to test for multiple conditions before deciding what action to take, `if` statements can be nested so that one condition is tested after another one is passed. This says, "If the first condition is true, then—and only then—check to see if the second condition is true. If the second one is true, too, then take an action." The following is the JavaScript version:

```
if (thisIsTrue) {

if (thatIsTrue) {
doThis;
}
}
```

The following example tests first to see if a credit card number is correct, and then checks to see if the balance of available credit is enough to pay for the product. The credit limit is not checked unless the card number is a valid one. Only if both of these comparisons are true will the purchase be completed.

```
if (cardNumber == valid) {

if (limit == highEnough) {
completePurchase();
}
}
```

You can also nest an `if` statement within an `if...else` statement, an `if...else` statement within an `if` statement, or an `if...else` statement within another `if...else` statement. The levels of complexity you can reach are truly mind-blowing, allowing for a practically infinite number of conditional decisions.



As with any other kind of power, it's best to exercise this in moderation. If you have 37 different `if...else` statements nested within 24 different `if` statements that are themselves nested within another dozen `if...else` statements, you'll never be able to untangle things if you need to modify the code later.

Going in loops

Most of us spend at least some time running around in circles. In JavaScript, you sometimes want to do this deliberately—a *loop*, in programming terms. If you've ever cooked a turkey on Thanksgiving, you already know all about loops. You open the oven and look at the meat thermometer to see if it's up to the "poultry" mark yet. If it isn't, you wait a little while, and then check again. You keep doing this until the thermometer shows the right reading, and then you break out of the loop and eat dinner.

www.it-ebooks.info

The most commonly used loop is called a `for` loop, and in plain English, it could be articulated as, "For as long as a condition remains the same, I'll keep taking action. When it changes, I'll stop." Here's what it is in JavaScript:

```
for (initial expression ; condition ; increment expression) {
statement or statements
}
```

The initial expression is simply where you declare a variable and assign a value to it. The variable name that is traditionally used here is `count` or `i` because the `for` loop counts over and over again (or you could say it *iterates*, if you want to get fancy about it).

The condition is anything using the comparison operators. You might, for example, establish a condition of `count == 36`, a condition of `count <= 100`, or just about any other comparison you want.

The increment expression is used to change the value of the variable established in the initial expression by adding 1 to it. It is poorly named because you can decrement the value as well, subtracting 1 from it. To add 1, you use `count++`; to subtract one, you use `count--`.

Let's say that you wanted to run through a loop that counted up from 1 to 10. Here's the JavaScript code you'd use:

```
for (var count = 1; count <= 10; count++) {
statement or statements
}
```

We're being a little picky by including the `var` keyword. This is one of those situations in which you really don't need it, but it never hurts to specify that you're declaring the variable at this point. The variable `count` is assigned a value of 1. Next, the looping condition is set so that the loop continues for as long as the value of `count` is less than or equal to 10. Finally, the increment operator ensures that the value of `count` increases by one every time the loop comes around again.

When the loop begins, `count` is 1. The statement or statements are executed. When they're all done, the loop starts over again and checks to see if the value of `count` is still less than or equal to 10. It isn't, so 1 is added to 1 to make it 2, and then the loop runs again. This goes on until the tenth time the loop completes. At that time, the condition is met because `count` finally holds a value of 10. The loop is finished at that point.

The following example shows this kind of count-up in action. It displays the numbers 1 through 5, each as an `H1` header:

```
<HTML>
<HEAD>
<TITLE>Looping Up</TITLE>
</HEAD>
```

www.it-ebooks.info

```
<BODY>

<SCRIPT language="JavaScript">
<!-- Make this script invisible to old browsers.
for (var count = 1; count <= 5; count++) {
document.write("<H1>" + count + "</H1>");
}
// Back to normal. -->
</SCRIPT>

</BODY>
</HTML>
```

Figure 16-7 shows the results.

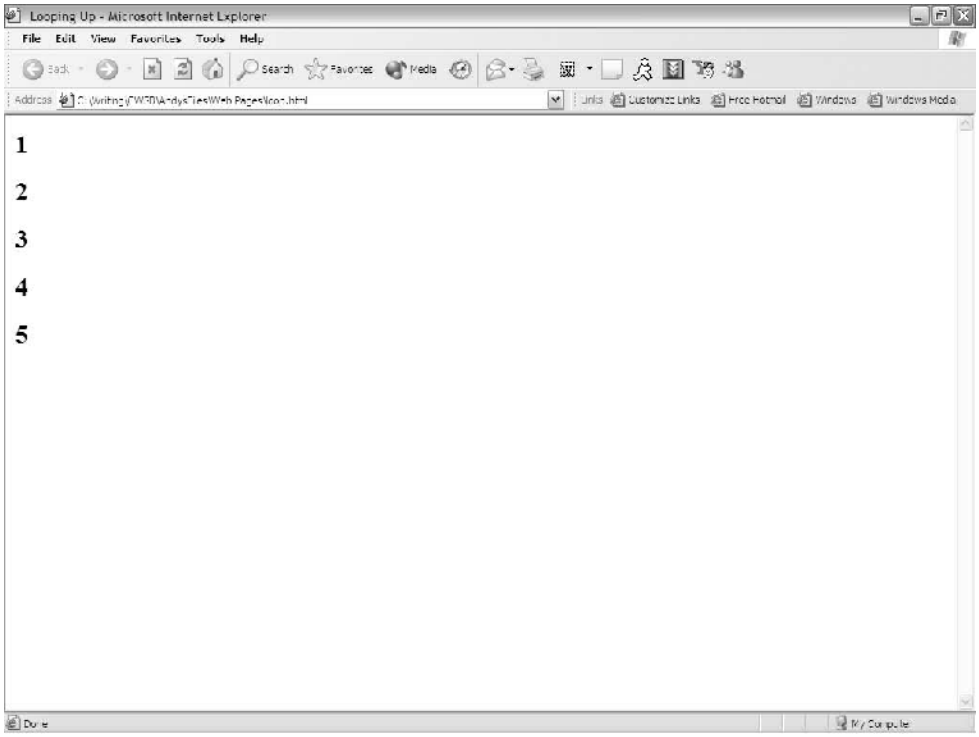


Figure 16-7: Here is the result of printing a looping value.

You can accomplish the reverse of this—counting down from 5 to 1—by a slight variation in the code:

```
<HTML>
<HEAD>
```

www.it-ebooks.info

```
<TITLE>Looping Down</TITLE>
</HEAD>
<BODY>

<SCRIPT language="JavaScript">
<!-- Make this script invisible to old browsers.
for (var count = 5; count >= 1; count--) {
document.write("<H1>" + count + "</H1>");
}
// Back to normal. -->
</SCRIPT>

</BODY>
</HTML>
```

Here, the value of `count` is set to 5 to begin with and decremented by 1 with each pass through the loop. The condition that is tested for is whether or not the value in `count` is 1 or higher. Figure 16-8 shows the countdown page.

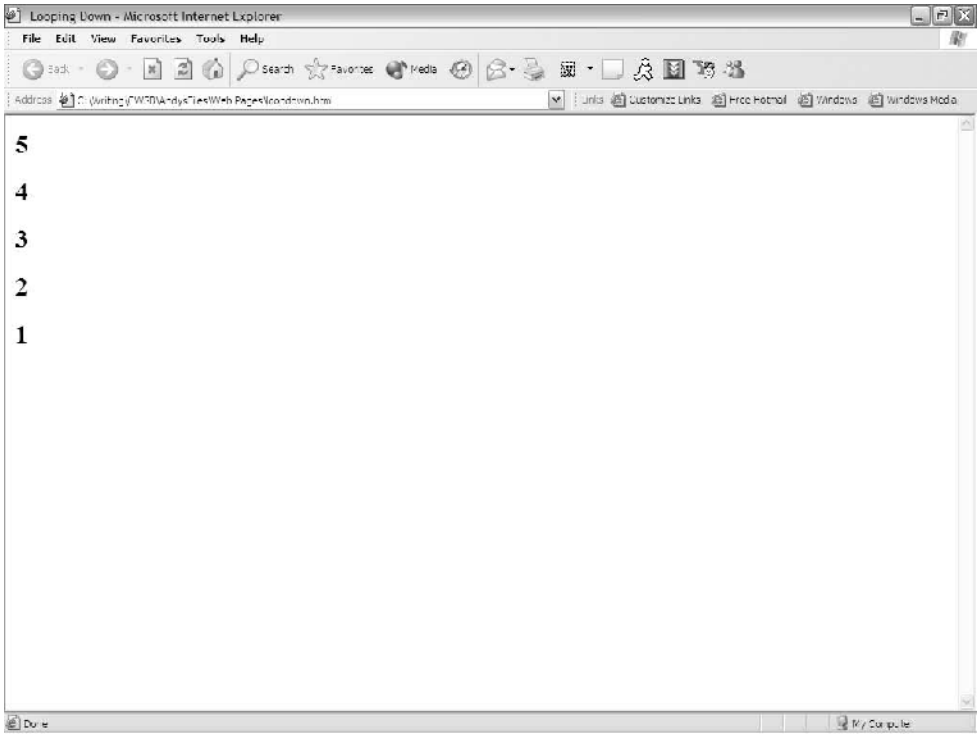


Figure 16-8: This figure shows the result of counting down with a decrement.

See "Arithmetic operators," earlier in this chapter, for an explanation of the `--(decrement)` operator.

www.it-ebooks.info

Which JavaScript?

JavaScript was invented by Netscape and was originally called LiveScript. LiveScript was a wonderful idea that got a lot of Web designers interested in the possibilities, but it was pretty limited in what it could do. JavaScript has changed a lot since those early days and is currently in its fifth version. With each new revision, new capabilities are added. This means, of course, that older Web browsers can't interpret the latest changes because they don't even know they exist and you can't teach an old browser new tricks.

Microsoft is certainly not known for its love of Netscape and, as part of the browser wars, introduced a JavaScript variant called JScript. Although it is essentially identical to JavaScript, there are just enough differences to create problems for Web designers. The solution here is simple compared with other cross-browser compatibility issues, however. Internet Explorer supports only JScript variants, but JavaScript works on both Netscape Navigator and Internet Explorer. If you just stick with normal JavaScript, you're better off.

In an attempt to develop a standard version of JavaScript, the European Computer Manufacturers Association, a European standards body generally known by the short form ECMA, came up with ECMAScript. As you can guess if you have any experience with Web standards, this standard is way behind the times. As Netscape politely puts it in its own JavaScript documentation, "JavaScript will always include features that are not part of the ECMA specification; JavaScript is compatible with ECMA, while providing additional features."

Using Events to Trigger Scripts

An *event* in JavaScript is an action that is taken by a visitor. A visitor may, for example, move his mouse pointer over an image, enter data into a form, or click a link. An *event handler* is a script that you write to react to an event. Event handlers for particular events always begin with `on`, followed by the name of the event. A click event, for example, is processed by an `onClick` event handler. Table 16-2 shows the event handlers that are common to both Netscape Navigator and Internet Explorer.

Table 16-2
Alphabetical List of Event Handlers

Event	*Meaning*
`onAbort`	A user clicks the Stop button while an image is loading.
`onBlur`	An element loses focus.
`onChange`	An element changes.
`onClick`	The mouse button is pressed and released once.
`onDblClick`	The mouse button is pressed and released twice in rapid succession.

www.it-ebooks.info

Event	*Meaning*
onError	An error occurs.
onFocus	An element receives focus.
onKeyDown	A key is pressed.
onKeyPress	A key is pressed and released.
onKeyUp	A pressed key is released.
onLoad	The Web page is loaded into the browser.
onMouseDown	The mouse button is pressed.
onMouseMove	The mouse pointer moves.
onMouseOut	The mouse pointer moves off the object.
onMouseOver	The mouse pointer moves over the object.
onMouseUp	The mouse button is released.
onMove	A window is moved.
onReset	A form is reset
onResize	A window is resized.
onSelect	Text is selected in a text box or text area
onSubmit	A form is submitted
onUnload	The browser moves on to another page.



If you look at other JavaScript documentation, you find a great deal of confusion about events and event handlers. When a page loads into a Web browser, for instance, this is a `Load` event. The event handler that reacts to that always begins with `onLoad=` and then some JavaScript code. You often, however, find a `Load` event referred to as an `onLoad` event, even though `onLoad` is technically the event handler, not the event itself. The HTML standard even falls into this trap. It's best to just shrug this sort of thing off—as long as you know what's meant, that's all that's important.

In addition to the common event handlers, specialized ones are particular to specific elements. The `FORM` element, for example, has `onSubmit` and `onReset` form handlers. Respectively, these mean that someone has clicked the Submit and Reset buttons.

Triggering on page loads and unloads

When you go to a Web page, you cause a `Load` event. When you leave it, you cause an `Unload` event. A Web page is *loaded* when all the code for it has been transmitted

www.it-ebooks.info

from the Web server to the Web browser. A Web page is *unloaded* when a different Web page (or other file) is loaded into the browser. Unloading can occur in many ways—when a visitor clicks a link, uses the Back or Forward buttons in the browser, manually enters an address in the address bar, or selects a page from his or her bookmarks (or favorites) listing or the History listing. Shutting down the browser doesn't cause an `Unload` event in in Internet Explorer, but not in Netscape Navigator.

One of the handiest things to do on a page load is to break your page out if it is contained within another Web site's frames. The following code shows one way to do it:

```
<HTML>
<HEAD>
<TITLE>Breaking Out Of Frames</TITLE>
<SCRIPT language="JavaScript">
<!-- Make this script invisible to old browsers.

function frameBuster() {
if (parent.location != window.location) {
parent.location = window.location;
}
}
// Back to normal. -->
</SCRIPT>

</HEAD>

<BODY onLoad="frameBuster()">

<h1>Hahahahahahahahahaha! You can't keep me in your frame!</h1>
</BODY>
</HTML>
```

Figure 16-9 shows the Web page this code creates.

The function `frameBuster()` is called when the page is finished loading. It checks to see if the parent frame (`parent.location`) has the same URL as the Web page itself (`window.location`). If the values of the two are not equal, they are made equal. The effect of this is to automatically detect the presence of a frame around your Web page and automatically ditch the frame.



See Chapter 12 for information on frames and framesets.

Reacting to mouse movements

One of the most popular of all JavaScript techniques is the `onMouseOver` event handler (commonly called a *rollover*), which detects when the mouse pointer is over an element. Two related event handlers, `onMouseOut` and `onMouseMove`, are not as common, but are still a valuable part of your JavaScript armory. The three event handlers are distinguished from one another by when they are detected. An

www.it-ebooks.info

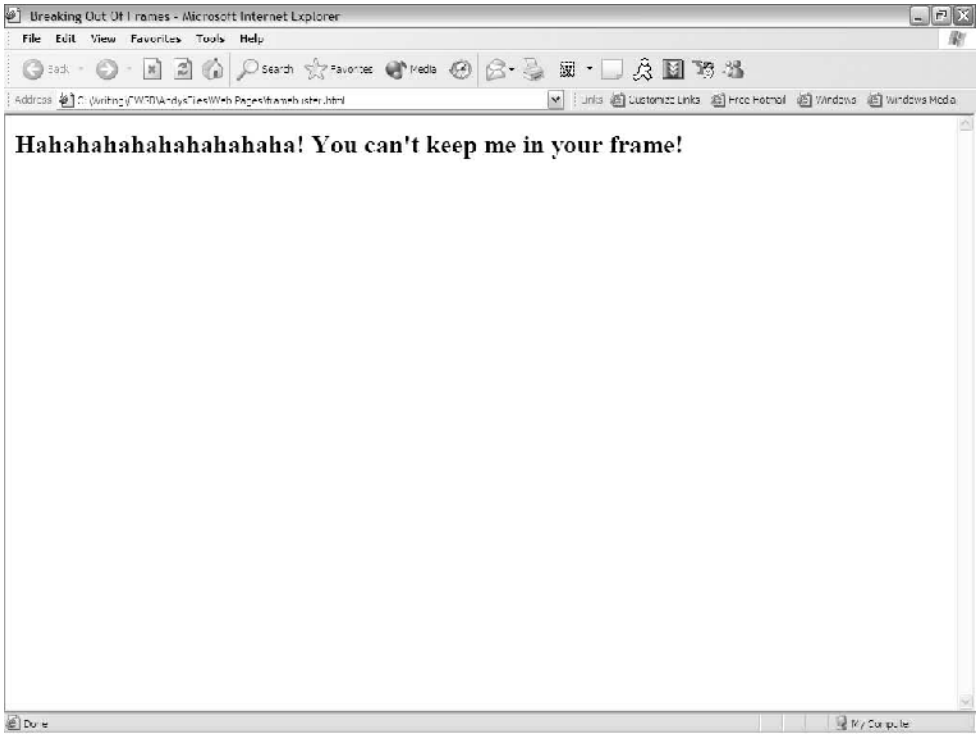


Figure 16-9: This figure shows the result of breaking out of frames.

`onMouseOver` event handler is triggered when the mouse pointer first crosses the edge of an element (usually an image, such as a navigation button), and remains active for as long as the pointer isn't moved off the element. An `onMouseOut` event handler is triggered when the mouse pointer is moved off the element. An `onMouseMove` event handler is triggered when the mouse pointer is over the object and is moved.

The following code demonstrates how these events work:

```
<HTML>
<HEAD>
<TITLE>Testing Mouseovers</TITLE>

<SCRIPT language="JavaScript">
<!-- Make this script invisible to old browsers.
function showOver() {
document.outputForm.readItHere.value = "Over";
}

function showOut() {
document.outputForm.readItHere.value = "Out";
}
```

www.it-ebooks.info

```
// Back to normal. -->
</SCRIPT>

</HEAD>

<BODY>

>
<IMG src="female student relaxing_200.gif" width="200" height="177"
border="0" onMouseOver="showOver()" onMouseOut="showOut()">

<FORM id="outputForm" name="outputForm">
<INPUT type="text" id="readItHere" name="readItHere" value="Nada">
</FORM>

</BODY>
</HTML>
```

Figure 16-10 shows the page.

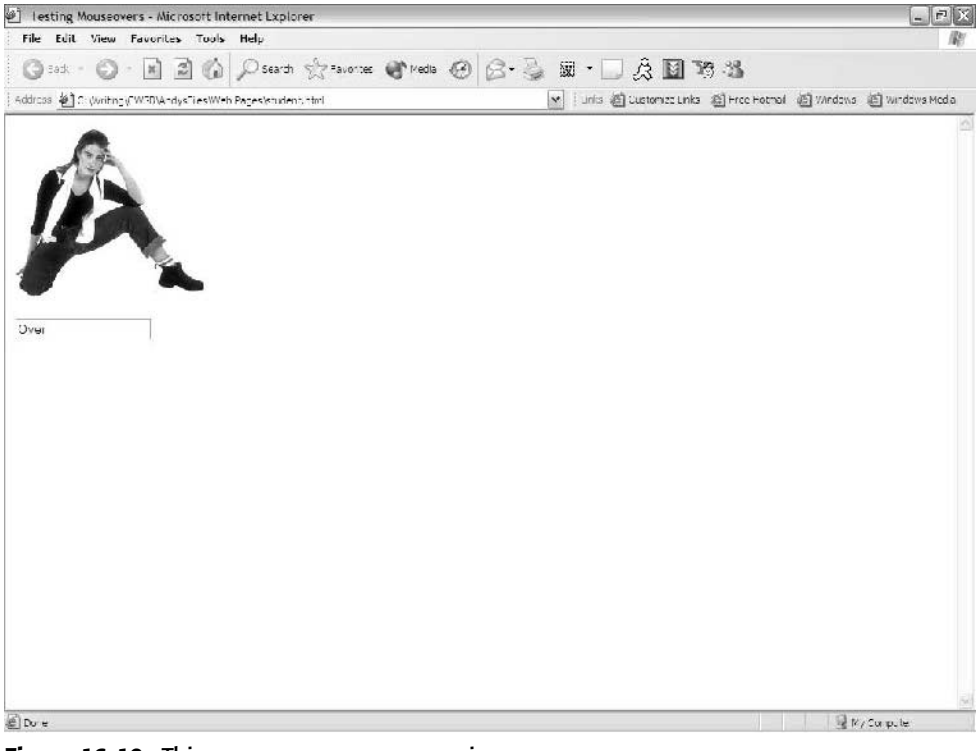


Figure 16-10: This page uses a mouseover image.

The two functions each write a word (Over or Out) into a text box depending on whether the `MouseOver` or `MouseOut` event is occurring.

www.it-ebooks.info



It is strongly recommended that you use both the id tag and a name tag with identical values. The DOM uses the `id` attribute to identify elements, for example when using `document.getElementById()`. However, browsers use the name value to identify elements when using expressions like `document.outputForm.readItHere.value`. If you are going to send a form, you need to specify the `name` attribute. If you don't the information won't even be sent to the server.

Clicking and double-clicking

Every time you click with your mouse, you are causing three different events. Pressing the mouse button down is a `MouseDown` event, and releasing it causes a `MouseUp` event. The combination of these two is the third event, which is called an `OnClick` event.



Although you can assign all three events to a single element, it is not recommended to assign both a `MouseUp` and an `OnClick` event to the same element because both are triggered—when the mouse button is released.

The following example creates a graphic demonstration in which the background color of the page changes in response to `MouseDown` and `MouseUp` events:

```
<HTML>
<HEAD>
<TITLE>Changing Background Colors</TITLE>

<SCRIPT language="JavaScript">
<!-- Make this script invisible to old browsers.

function makeBlue() {
document.bgColor="blue";
}
function makeWhite() {
document.bgColor="white";
}
// Back to normal. -->
</SCRIPT>

</HEAD>

<BODY>

<A href="#" onMouseDown="makeBlue()" onMouseUp="makeWhite()">Click
this to change the background color back and forth.</A>
</BODY>
</HTML>
```

Two functions are defined—one to set the document's background color to blue, and the other to return it to white. The first is triggered when the mouse button is pressed over the link, and the second when it is released. Figure 16-11 shows the Web page when the mouse button is down.

www.it-ebooks.info

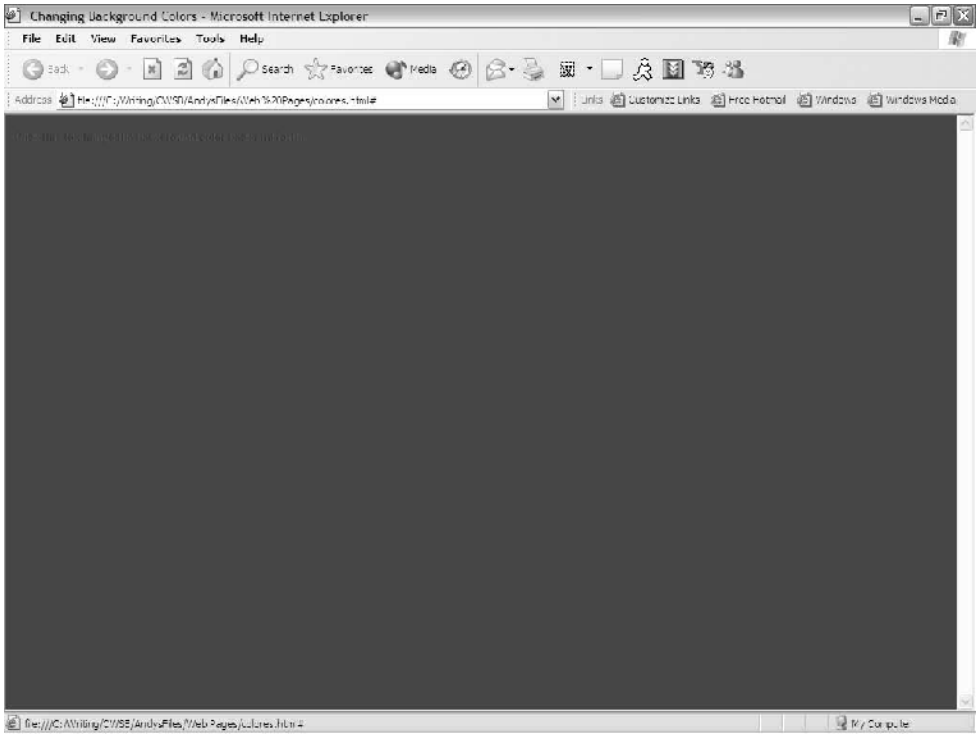


Figure 16-11: Changing background colors with MouseDown and MouseUp events.

Pressing and releasing keys

Using your keyboard is pretty much the same thing as clicking a mouse button as far as JavaScript is concerned. Once again, what we normally think of as a single action—pressing a key—is broken down into three different events. When the key is pressed, but before it is released, it is causing a `KeyDown` event. Releasing the key causes a `KeyUp` event, and the combination of the two actions is a `KeyPress` event.

This works exactly as the mouse buttons, and the preceding example could be rewritten as follows:

```
<HTML>
<HEAD>
<TITLE>Changing Background Colors With Keys</TITLE>

<SCRIPT language="JavaScript">
<!-- Make this script invisible to old browsers.

function makeBlue() {
document.bgColor="blue";
}
function makeWhite() {
document.bgColor="white";
}
```

www.it-ebooks.info

```
// Back to normal. -->
</SCRIPT>

</HEAD>

<BODY>
<FORM>
<INPUT type="text" onKeyDown="makeBlue()" onKeyUp="makeWhite()">
</FORM>

</BODY>
</HTML>
```

In this version, the background colors change in reaction to typing that takes place within a text box in a form. Each time a key is pressed, the background color changes to blue. Each time it is released, the background color reverts to white. This causes the screen to rapidly flash back and forth between the colors as you type.

Seeing Elements as Objects

Object-oriented programming sounds ominous at first, but it's simply a way of looking at things. Even if you've never done any programming at all, you're already familiar with the way it works, because it's simply an expression of the way the real world works. The book in your hands is an object. It contains other objects such as chapters, illustrations, example code, and so forth. Similarly, a Web browser is an object that contains other objects that are represented by HTML elements.

When you're using JavaScript, a Web browser is called a *window object*. The Web page it contains is known as a *document object*. If there is an image in the Web page, it's an *image object*. You get the idea—it's all just another way of stating what's showing up in the browser and what's on the page. Objects are referenced from the top down, starting with the window object, and each object is separated from the next by a period. Thus, the Web page is referenced as `window.document`. If it contains an element named `sidePanel`, that form can be referenced as `window.document.sidePanel`. It's just as if you're going down a stairway one step at a time.



It's common practice among Web designers to skip the stage specifying the window object—because one Web browser is involved, you don't usually have to state that fact. Thus, many Web designers would simply write `document.sidePanel` instead of `window.document.sidePanel`. However, this approach can be hazardous, occasionally creating unforeseen results. Forms that are specified by name instead of position in the forms array, for example, can return a false error message if the entire object hierarchy leading to them is not used.

The Document Object Model (DOM for short) is a way of looking at a Web page that sees every element as an object and all of the element's attributes as properties of that object. The first DOM that Netscape came up with was pretty limited. Microsoft, however, took a quantum leap in Web programming when they expanded the DOM to

www.it-ebooks.info

include every element on the entire Web page. The W3C DOM Level 2 standard, issued in November 2000, establishes this as the officially correct way to do things, and most modern browsers provide full support for it.

Every object on a Web page is also a property of some object that is higher in the hierarchy than it is.

Understanding properties

Every object has properties. For instance, as an object a coat has several properties, and some of these are the same for all coats—the number of sleeves, for example. Other coats have properties specific to that particular coat. The coat as an object has a size property. It has a color property. It has a lapel property, a pocket property, a lining property, and so on.

In JavaScript, properties are referenced by appending them to the object name. The two—object and property—are separated by a period, just as objects in the object tree are. The size property, for instance, would be expressed as `coat.size`.

To put it all together, if you want to get the text that a visitor typed into a text box named `myText` in the form named `formSeven`, you would express it as `document.formSeven.myText.value`. This flows down from the document object to the form object to the text box object, and finally to the value of that last object.

In order to get the value in a form that you can actually work with, you assign it to a variable, as in the following example:

```
wordsTyped = document.formSeven.myText.value;
```

From that point on, all you have to do is to work with the variable. The following example shows how this can be done in a form that allows a visitor to change the background color of the page:

```
<HTML>
<HEAD>
<TITLE>Working With Properties</TITLE>

<SCRIPT language="JavaScript">
<!-- Make this script invisible to old browsers.

function getColor(){
colorChoice =
document.pickAColor.visitorColor.options[document.pickAColor.visitorColor
.selectedIndex].value;
document.bgColor = colorChoice;
}

// Back to normal. -->
</SCRIPT>

</HEAD>
```

www.it-ebooks.info

```
<BODY bgcolor="">

<H1>Pick a color, any color.</H1>

<FORM id="pickAColor" name="pickAColor">
<SELECT id="visitorColor" name="visitorColor" onChange="getColor()">
<OPTION value="Aqua">Aqua</OPTION>
<OPTION value="Black">Black</OPTION>
<OPTION value="Blue">Blue</OPTION>
<OPTION value="Fuschia">Fuschia</OPTION>
<OPTION value="Gray">Gray</OPTION>
<OPTION value="Green">Green</OPTION>
<OPTION value="Lime">Lime</OPTION>
<OPTION value="Maroon">Maroon</OPTION>
<OPTION value="Navy">Navy</OPTION>
<OPTION value="Olive">Olive</OPTION>
<OPTION value="Purple">Purple</OPTION>
<OPTION value="Red">Red</OPTION>
<OPTION value="Silver">Silver</OPTION>
<OPTION value="Teal">Teal</OPTION>
<OPTION value="White">White</OPTION>
<OPTION value="Yellow">Yellow</OPTION>
</SELECT>
</FORM>

</BODY>
</HTML>
```

The results, as shown in Figure 16-12, are that the color that is chosen from the list sets the background color of the page.

Here's how it works, step by step:

1. The visitor chooses a color from the list.
2. The `onChange` event handler is triggered, which calls the function `getColor()`.
3. The function reads the value of the choice—the `selectedIndex` of the list—and assigns that value to a variable called `colorChoice`.
4. The value of that variable is assigned to the `document.bgColor` property, thus changing the color of the page.

Working with methods

A *method* is a capability of an object rather than a property of it. Think of a method as a built-in function that you don't have to bother with writing. For example, a refrigerator with an automatic ice-maker has the capability to make ice, which is a method of the refrigerator object, and it is expressed as `refrigerator.makeIce()`.

The document object, for example, has a `write` method, which enables you to programmatically add new text to the Web page. It is expressed as `document`

www.it-ebooks.info

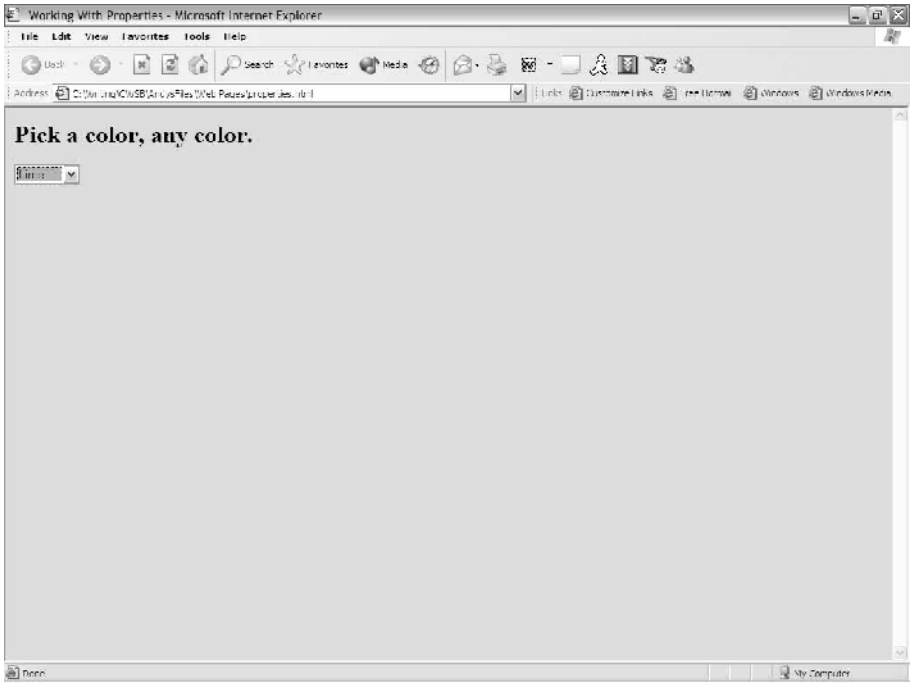


Figure 16-12: You can changing colors using list properties.

`.write()`. It's easy to confuse properties and methods at first, because both follow the object's name. Just remember that, as with functions, methods end in a closed set of parentheses.

The parentheses are there for a reason—the same reason they are there in functions. They are meant to hold parameters for the method to process. The parameters for the `document.write()` method are the text or HTML code to be placed onto the Web page. To add the sentence, "Hey, I'm changing the page!" to the page, you would write the following:

```
document.write(Hey, I'm changing the page!);
```

Remember, you're not limited to adding only plain text—you can use HTML code as well. The following variation is just as valid as the original example:

```
document.write(Hey, I'm <I>changing</I> the page!);
```

Form Validation

One of the most common uses of JavaScript is to validate forms before they are submitted to the Web server.

The following code shows how to verify that the data in a form contains a name and an e-mail address. It uses regular expressions to perform the validation.

www.it-ebooks.info

```
<html>
<head>
<title>Validating Forms</title>

<script language="JavaScript">
<!-- Make this script invisible to old browsers.

function check()
{
  if (document.form.name.value.length==0 )
  {
         alert ("You must enter your name");
         return false;
  };
  var nameRe=new RegExp("^[A-Za-z]+$");
  if (!nameRe.test(document.form.name.value))
  {
         alert ("You must enter your name only alphabetic characters
are allowed no spaces");
         return false;
  }
  var emailRe=new RegExp("^[A-Za-z0-9]+@[A-Za-z0-9.]+$");
  if (!emailRe.test(document.form.email.value))
  {
         alert ("You must enter a valid email address eg:
user@host.com");
         return false;
  }
  return true;
};
// Back to normal. -->
</script>

</head>

<body>

<form id="form" name="form" onSubmit="return check();">
First Name <INPUT type="text" id="name" name="name">
Email <INPUT type="text" id="email" name="email">
<input type="submit" value="Send">
</form>

</body>
</html>
```

In the form we specify the `onSubmit` handler to call the function check before the form is submitted. If the event handler returns `false` then the form is not submitted, so we say `return check()`. The `check` function validates the form and returns `true` if it is valid.

The following expression returns the number of characters in the string entered in the "name" text box. If it is zero then we ask the user to enter something.

```
document.form.name.value.length
```

www.it-ebooks.info

Now for the complicated part—regular expressions. RegularExpressions, or REs for short, are used to search for matches of a particular character sequence in text. They are part of many programming languages—the UNIX operating system uses them extensively in tools from the command line, and they have been included in JavaScript since version 1.1.

The following is brief (and simplified) explanation of REs.

You are probably familiar with wild card characters such as ? (representing one character) and * (representing any number of characters) in the DOS and Unix command line. The regular expression for ? would be `.` any character or alternatively `[A-Za-z0-9]` which means one character in the ranges of A to Z, a to z, and 0 to 9. If it is not in the given ranges then it doesn't match. So for * it would be:

```
.*
```

or alternatively:

```
[A-Za-z0-9]*
```

The * at the end means zero or more of the preceding character or range of characters. It's a bit confusing that the RE for * has a * in it, but you have to bear in mind that they are different syntaxes. If we wanted one or more, we could write:

```
[A-Za-z0-9][A-Za-z0-9]*
```

But there is a better way:

```
[A-Za-z0-9]+
```

+ means one or more of the previous expression. In the preceding code we checked the name with:

```
RegExp("^[A-Za-z]+$")
```

This is because when the test method of the RE is called it returns `true` if the character sequence is found anywhere in the text. However, we want it to only return `true` if the whole text given corresponds to the RE. The `^` character specifies the start of the text and the `$` specifies the end of the text.

So `^[A-Za-z]+$` translates into English as a match for text that from start to end is composed of a sequence of one or more alphabetic characters. Now for the email:

```
^[A-Za-z0-9]+@[A-Za-z0-9.]+$
```

It starts with a sequence of one or more alphanumeric characters, has a @ in the middle, and then ends with a sequence of one or more alphanumeric characters. However, this isn't quite specific enough for a real email address. If you really want to graduate in REs, take a look at the following:

```
^[A-Za-z0-9]+[-A-Za-z0-9._]*@([-A-Za-z0-9]+\.)+[A-Za-z]{2,5}$
```

www.it-ebooks.info

Table 16-3 takes you through this complex example of REs step by step.

Table 16-3
Complex Example of Regular Expressions

Part of the Regular Expression	*Explanation*
^	Starts with
[A-Za-z0-9]+	A sequence of one or more alphanumeric characters
[-A-Za-z0-9._]*	A sequence of zero or more alphanumeric characters or `-`, `.` or _ The `-` is specified as the first character because in this position it doesn't signify a range just the character itself
@	`@` itself
([-A-Za-z0-9]+\.)+	The brackets turn a list of expressions into one group expression so this really means one or more of the sequences alphanumeric character or – followed by a `.`, so this matched `abc.xyz`.

You can also check to see if check boxes, option buttons, or menu options are being selected. The following example shows how.

```
<html>
<head>
<title>Validating Forms</title>

<script language="JavaScript">
<!-- Make this script invisible to old browsers.

function check()
{
  if(! (document.form.football.checked &&
document.form.swimming.checked))
  {
         alert("We only accept members who play football and swim so
you might as well select them both");
         return false;
  }
  if (document.form.gender[0].checked)
  {
         alert("You have to select your gender");
         return false;
  }
  if (document.form.language.selectedIndex==0)
  {
         alert("Please select a language");
         return false;
  }
```

www.it-ebooks.info

```
  return true;
};
// Back to normal. -->
</script>
</head>

<body>
<h2>Club Membership</h2>
<form id="form" name="form" onSubmit="return check();">
First Name <INPUT type="text" id="name" name="name">
<br>
Gender
<input type="radio" name="gender" value="none" checked="checked">
<input type="radio" name="gender" value="male">
<input type="radio" name="gender" value="female">
<br>
Hobbies
<input type="checkbox" name="football">
<input type="checkbox" name="swimming">
<br>
Language <select name="language">
<option value="none">Please Select</option>
<option value="en">English</option>
<option value="de">German</option>
<option value="es">Spanish</option>
</select>
<br>
<input type="submit" value="Send">
</form>

</body>
</html>
```

Examining Browser Compatibility

That old bugaboo of Web designers, browser incompatibility, is nowhere more evident than when working with JavaScript. For example, the following code is perfectly valid JavaScript, yet it works only in Internet Explorer:

```
<HTML>
<HEAD>
<TITLE>Testing Mouseovers</TITLE>

<SCRIPT language="JavaScript">
<!-- Make this script invisible to old browsers.

function showOver() {
outputForm.readItHere.value = "Over";
}
function showOut() {
outputForm.readItHere.value = "Out";
}
```

www.it-ebooks.info

```
// Back to normal. -->
</SCRIPT>

</HEAD>

<BODY>
<IMG src="female student relaxing_200.gif" width="200" height="177"
onMouseOver="showOver()" onMouseOut="showOut()">
<FORM id="outputForm" name="outputForm">
<INPUT type="text" id="readItHere" name="readItHere" value="Nada">
</FORM>
</BODY>
</HTML>
```

In order to make the same code work with Netscape Navigator 4, the reference to the text box's value must be changed from `outputForm.readItHere.value` to `document.outputForm.readItHere.value`. This is necessary even though a form can only be a property of the document object. Specifying the hierarchy in this detail isn't actually required by the DOM, but you have to do what works.

In addition, the image cannot be used as the source of the rollover in Netscape Navigator 4, so it needs to be wrapped in a link via the `A` element. Netscape Navigator, for reasons unknown, doesn't recognize an image `MouseOver` event but does recognize the same event for a link. This means that the `onMouseOver` and `onMouseOut` event handlers need to be moved from the `IMG` element to the `A` element.

The revised version, which works with Internet Explorer, Mozilla, Opera, and Netscape 4 and later, is as follows:

```
<HTML>
<HEAD>
<TITLE>Testing Mouseovers</TITLE>

<SCRIPT language="JavaScript">
<!-- Make this script invisible to old browsers.

function showOver() {
document.outputForm.readItHere.value = "Over";
}
function showOut() {
document.outputForm.readItHere.value = "Out";
}
// Back to normal. -->
</SCRIPT>

</HEAD>

<BODY>
<A href="#" onMouseOver="showOver()" onMouseOut="showOut()">
<IMG src="female student relaxing_200.gif" width="200" height="177"
border="0">
</A>
```

www.it-ebooks.info

```
<FORM id="outputForm" name="outputForm">
<INPUT type="text" id="readItHere" name="readItHere" value="Nada">
</FORM>
</BODY>
</HTML>
```

All modern browsers support mouse over and mouse out events for all elements.

You could become the world's greatest expert on JavaScript and designing complex code that works with both browsers. Believe us, some of the things you need to do to make the same function work on both browsers are incredibly complex. We've only scratched the surface. Other than that unpalatable alternative, though, what can you do about the differences? Many Web designers have adopted the technique of creating one Web site for each browser, making an entry page that determines the browser type and version. This page automatically redirects a visitor to the appropriate Web page.

Browser detection and redirection is done with the navigator object. Despite the name, it's present in Internet Explorer also. The browser name is found with the `navigator.appName` property and the browser version is detected with the `navigator.appVersion` property. The following code creates a Web page that displays these values for you:

```
<HTML>
<HEAD>
<TITLE>Reading and Displaying Browser Information</TITLE>
</HEAD>
<BODY>

<SCRIPT language="JavaScript">
<!-- Make this script invisible to old browsers.

document.writeln("Your browser is: " + navigator.appName + "<BR>");
document.writeln("The version is: " + navigator.appVersion + "<BR>");

// Back to normal. -->
</SCRIPT>

</BODY>
</HTML>
```

Figure 16-13 shows the results. This means that the browser used is Netscape version 4.75, the language is English, the operating system is Windows 98, and the encryption is U.S. standard.



The actual output, of course, depends upon which browser you are using.

To apply this capability of sending a visitor to an appropriate Web page for their browser, you would use an `if` statement. Basically, you need to say, "If the browser is Netscape Navigator, then go to the Netscape-enabled page. If it's Internet Explorer, then go to the Microsoft-enabled page instead." Of course, there are all sorts of

www.it-ebooks.info

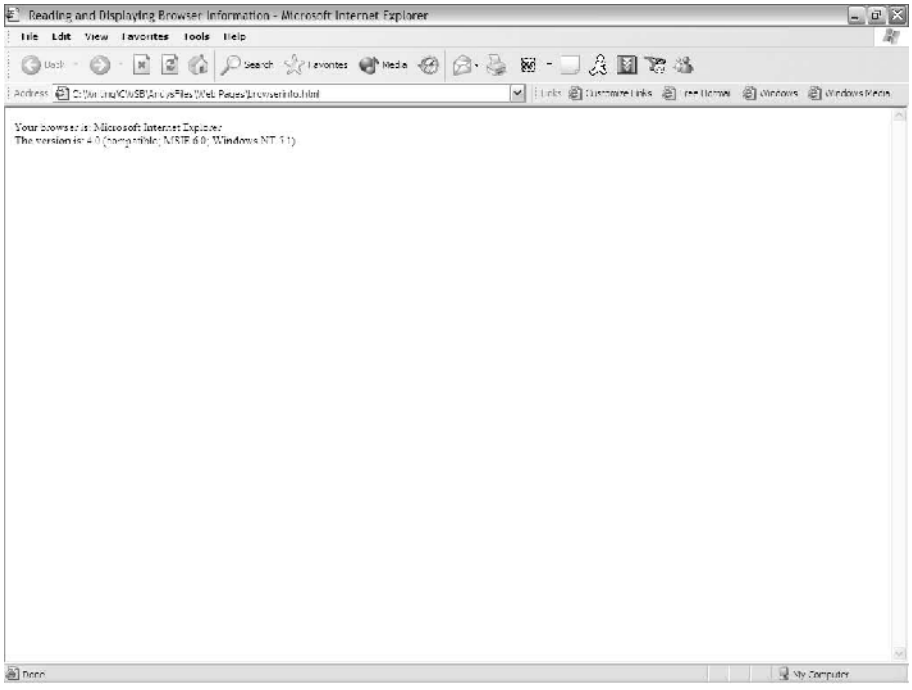


Figure 16-13: This figure shows the result of reading and displaying browser information.

variations depending on your needs. You might want to take action only for Internet Explorer users, for example. The principles, however, are the same.

The following code tests to see if the browser is Netscape Navigator. If not, it tests to see if it's Internet Explorer. If it's either one, it sends the browser to the appropriate Web page. If it's neither, this Web page displays a message stating that the visitor is using a third kind of browser:

```
<HTML>
<HEAD>
<TITLE>Redirecting Browsers</TITLE>

<SCRIPT language="JavaScript">
<!-- Make this script invisible to old browsers.

if (navigator.appName == "Netscape"){
window.location = "netscapeindex.html";
}

else {
if (navigator.appName == "Microsoft Internet Explorer") {
window.location = "explorerindex.html";
}
}
```

www.it-ebooks.info

```
// Back to normal. -->
</SCRIPT>

</HEAD>
<BODY>

<H1>You're not using either Netscape or Internet Explorer.</H1>

</BODY>
</HTML>
```

Although this technique of browser sniffing is very common on Web sites, there are several major problems. The first is that there is no real standard as to how browsers identify themselves in the app name so you might not identify it correctly. Also, there are more than 30 browsers currently in use on the Web—Internet Explorer, Mozilla, Opera, and Netscape Navigator are among the most popular. Imagine the complex `if` statement required to support all of these. And what happens if a new super cool browser suddenly appears on the market and everybody starts using it? You would have to change all of your Web pages to support it.

If you want to use specific methods the best way to handle browser detection is to test for capabilities. For example, you can test to see if the browser supports the DOM by checking for `getElementById`, whether it is Internet Explorer by checking for `document.all`, and whether it is Netscape Navigator 4 by checking `document.layers`. The code is as follows:

```
if (document.getElementById)
{
  //code that requires DOM support goes here
}
else if (document.all)
{
  //code specific to Internet Explorer goes here
}
else if (document.layers)
{
  //code specific to Netscape Navigator 4 goes here
}
else
{
  //Its none of the above should we kindly
  //ask the user to upgrade???
  //In fact it could be a braille browser
  //or some other custom browser
}
```

Summary

JavaScript is a programming language that works within a Web browser to give you greater control over the appearance and behavior of your Web pages. This chapter introduced you to the following concepts and topics:

www.it-ebooks.info

- JavaScript is used to create programs that read and alter the values of HTML elements.
- A variable is an item that holds a value that is expected to change over time.
- Any time you use JavaScript to tell a browser to do one thing, you're making a *statement*.
- A *function* is a grouped series of statements that execute one after another.
- Each element on the Web page is a separate object.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 17

Navigating Your Web Site

In This Chapter

Setting navigation rules

Adding navigation bars

Displaying links with lists and menus

Unless you are only putting up a single Web page, your visitors have to be able to get from one page to another within your Web site. It's easy to just throw a bunch of Web pages together, think of them as a site, and rely on the fact that somewhere there are links that can be followed if they are found. A coherent, well-designed Web site, however, is more than the sum of its pages. While individual pages can come and go, the site itself remains. It is the one overall, overriding factor, and the navigation scheme you choose helps to define it in your visitors' minds.

Setting Navigation Rules

The navigational setup of your site is even more important than its appearance. If a site cannot be easily and intuitively explored, then it matters little how pretty it is. In fact, you'll probably find that some of your best work doesn't even get seen because it cannot be found.

Let's qualify that last expression. Anything on the Internet can be found if someone puts enough time and effort into it. But people are, of course, only human, and unless you have the single most fantastic and useful site on the entire World Wide Web, few people are likely to go to a lot of trouble to find their way around it. It's up to you to ensure that finding the way through your site is a pleasure instead of a burdensome task.

Avoiding home page blunders

The home page of a well-designed site serves one of two functions. It is either the main crossroads for everything in the site, or it is the point at which a journey through the site begins. In the first case, the home page provides a set of links that, in a small site, lead to every page and, in a large site, lead to sub-home pages. The structure of a site with both a main

www.it-ebooks.info

home page and sectional sub-home pages, as shown in Figure 17-1, consists of a series of links that fan out, each one spreading to its own small subset of related Web pages.

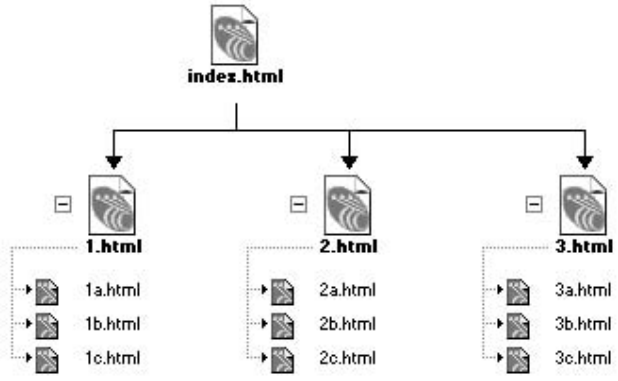


Figure 17-1: This site has a set of three sub-home pages.

In the second case, the journey can be strictly linear, as in a tutorial where a strict pattern is followed to form a clear understanding, or the journey can allow visitors to jump from one section to another, bypassing intervening Web pages. Figure 17-2 shows a diagram of a tutorial-type site.

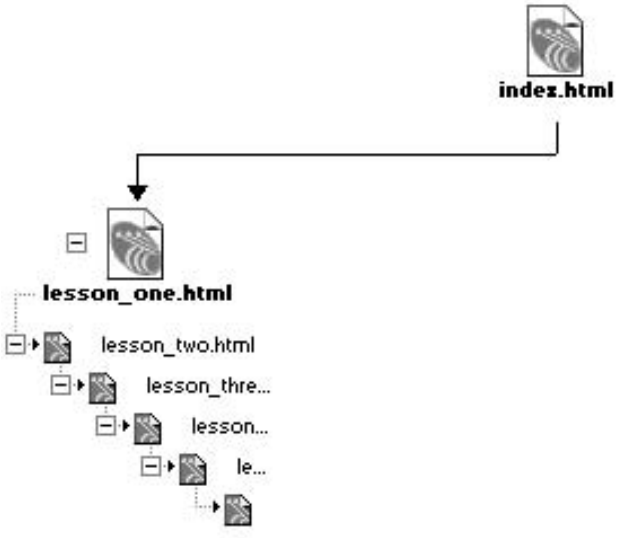


Figure 17-2: This is an example of a linear tutorial site.

www.it-ebooks.info

Many home pages suffer from what we call the *barrier syndrome*, where the journey doesn't lead a visitor anywhere except to another Web page that is, functionally speaking, the real home page for the site. These pages typically show some sort of fancy graphic along with a text link with words such as, "Click here to enter the site." Figure 17-3 shows such a home page. This is a major design blunder because there is no function or purpose to this page—at least from the visitors' point of view. The page is just an extra, unnecessary step that they have to put up with.

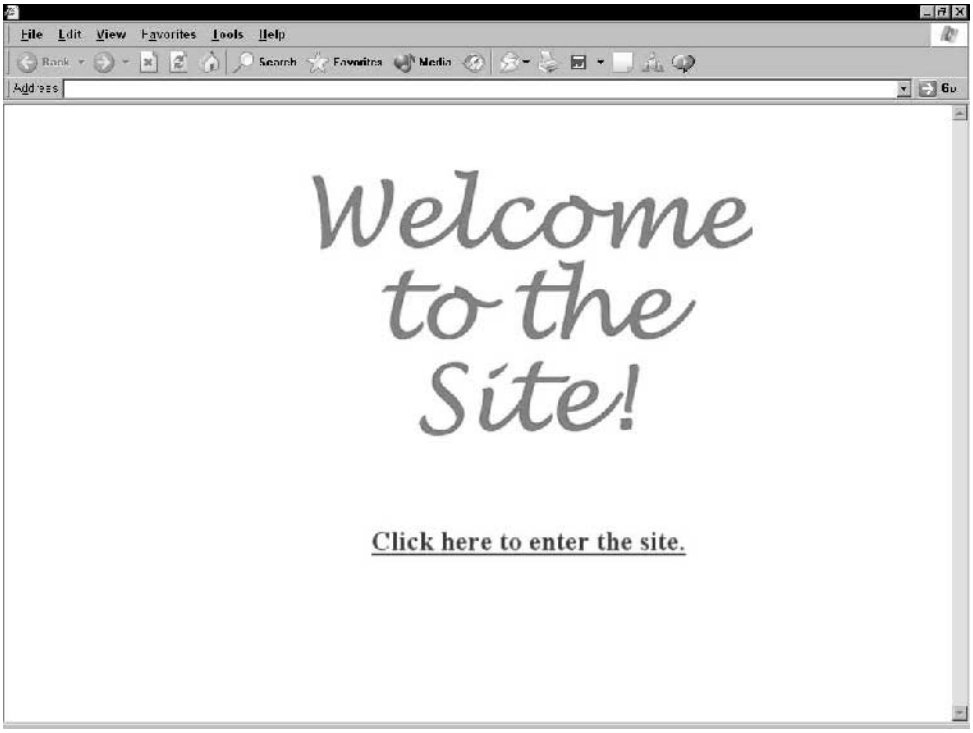


Figure 17-3: Some home pages present unnecessary barriers to navigation.

Navigation blunders are even more common among Web sites that use Macromedia Flash animations. For reasons we've never been able to figure out, the vast majority of Flash-enabled sites fail to integrate the animations into their page designs. Not only do they separate their Flash animations from the rest of the site, but they also tend to put the Flash animations on their home pages. As a result, not only is there an extra, unnecessary step involved in entering the site, but download time is also increased. On top of this, the animation itself is often unnecessary, which makes the whole experience of entering the site frustrating. We're not knocking Flash—it's one of the greatest tools and it can add a fabulous degree of interactivity, sophistication, and beauty to your site. Just bear in mind that it shouldn't be used to create a roadblock.

www.it-ebooks.info



See Chapter 19 for details on using Macromedia Flash.

Avoiding orphan pages

The most common navigation blunder on the World Wide Web is the *orphan page*—the Web page from which you can't get to the rest of the site. This situation arises when a Web designer relies on the Web browser's Back and Forward buttons to provide navigation. Well, "provide" isn't really the right word here. It's actually a kind of dereliction of the Webmaster's duty. Orphan pages are almost always found on amateur sites, but you can still, surprisingly, find them on some professionally designed sites.

As a designer, you can get away with this if a visitor starts at your home page, follows links in the text, and moves back and forth in an orderly manner. Even so, the burden is on the visitor to keep track of where they've been and where they need to go next. However, this lackadaisical approach fails to take into account the fact that not everybody gets into the site via the home page. Visitors may be entering a page on a site because of a match in a search engine, a link from someone else's site, or even a link sent from a friend in an e-mail or in an instant message.



It's true that a savvy visitor can get to the home page by monkeying with the URL in their Web browser's address bar, but the basic concept of good Web design is to make site functionality easy and transparent to your visitors.

Every page on your entire site should have, at the absolute minimum, a link to the home page. It's even better if there is a navigation bar or some other kind of link set that enables your visitors to move around your entire site at will from anywhere within it. The next section shows you how to add navigation bars to your site.

Adding Navigation Bars

The simplest kind of navigation bar is nothing more than multiple links in a single-row table such as the one shown in Figure 17-4. While this approach lacks a bit of style and flair compared to some of the more complex navigation bars, it is easy to implement and still offers all the functionality of the more complicated techniques.

The following is the HTML code used to create the navigation bar in Figure 17-4:

```
<HTML>
<HEAD>
<TITLE>Simple Single Row Navbar</TITLE>
</HEAD>
<BODY bgcolor="#FFFFFF">

<TABLE border=2 width="100%" >

<TR align="center" bgcolor="#FFCC33">
```

www.it-ebooks.info

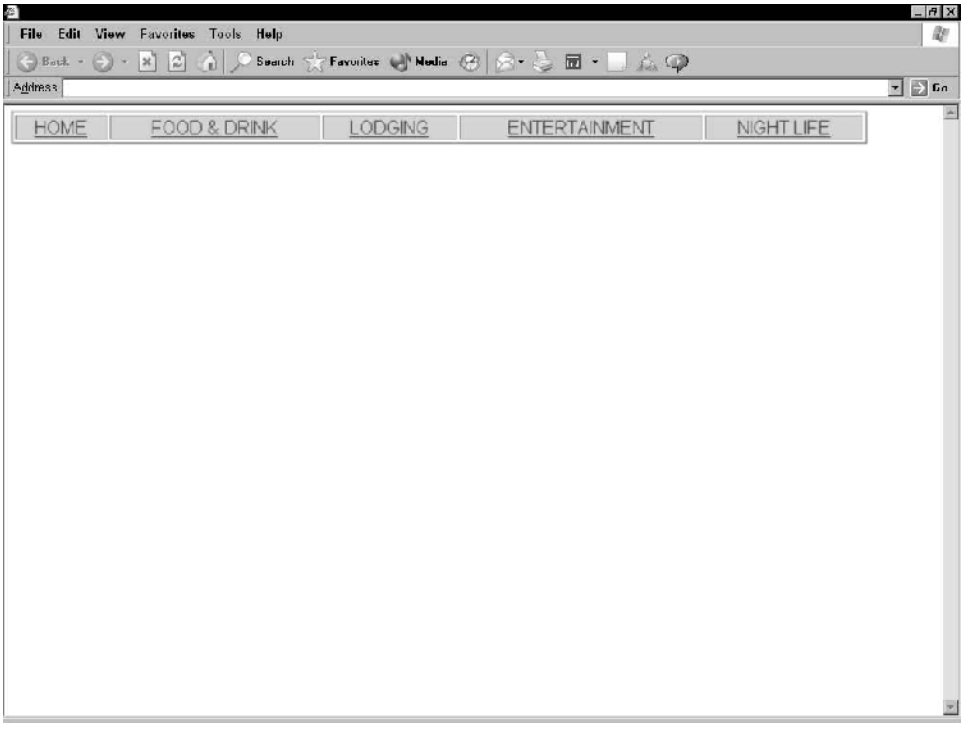


Figure 17-4: A simple, but attractive, navigation bar can be created with a table and text.

```
<TD><FONT face="Arial, Helvetica, sans-serif"><A
href="index.html">HOME</A></FONT></TD>

<TD><FONT face="Arial, Helvetica, sans-serif"><A
href="dining.html">FOOD &amp; DRINK</A></FONT></TD>

<TD><FONT face="Arial, Helvetica, sans-serif"><A
href="sleeping.html">LODGING</A></FONT></TD>

<TD><FONT face="Arial, Helvetica, sans-serif"><A
href="attractions.html">ENTERTAINMENT</A></FONT></TD>

<TD><FONT face="Arial, Helvetica, sans-serif"><A
href="party.html">NIGHT LIFE</A></FONT></TD>

</TR>

</TABLE>

</BODY>
</HTML>
```

The table is set to 100 percent of the width of the Web page and given a border width of 2 pixels. The single row (`TR` element) is set so that the cells' contents are centered

www.it-ebooks.info

within the cells, and it has a background color that contrasts well with the default text link color. Each cell (TD element) is set to use a font that looks different from the normal text font, and a link is added to the cell.

Using graphics to enhance bars

While simple text can do the job, a little bit of graphical work can go a long way in making your navigation bars look great. The navigation bar shown in Figure 17-5 is a variation on Figure 17-4, but it uses images instead of plain text to fill the table.

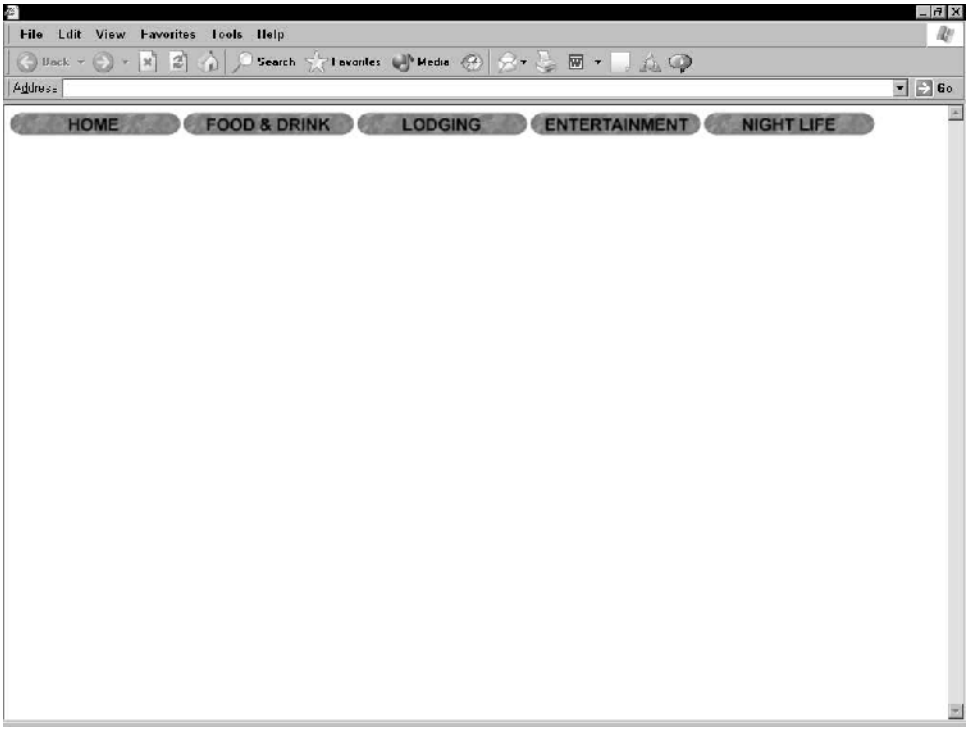


Figure 17-5: Graphics can make a navigation bar much more attractive.

The HTML code is much simpler in this case because, since no text is used, there is no need to specify a font:

```
<HTML>
<HEAD>
<TITLE>Graphical Navbar</TITLE>
</HEAD>
<BODY bgcolor="#FFFFFF">

<TABLE width="100%" border="0" cellpadding="0" align="center">
```

www.it-ebooks.info

```
<TR align="center">

<TD><A href="index.html"><IMG src="HOME.gif" width="152"
height="20" border="0"></A></TD>

<TD><A href="dining.html"><IMG src="FOOD&DRINK.gif" width="152"
height="20" border="0"></A></TD>

<TD><A href="sleeping.html"><IMG src="LODGING.gif" width="152"
height="20" border="0"></A></TD>

<TD><A href="attractions.html"><IMG src="ENTERTAINMENT.gif"
width="152" height="20" border="0"></A></TD>

<TD><A href="party.html"><IMG src="NIGHTLIFE.gif" width="152"
height="20" border="0"></A></TD>

</TR>

</TABLE>

</BODY>
</HTML>
```



Chapters 7, 8, and 9 cover Web graphics.

The appearance of a graphical navigation bar is usually better than one created in HTML, and the code is simpler. However, the graphical navigation bar will load more slowly, because the graphics must be requested and sent over the Internet to the user's computer before the navigation bar can be displayed.

Including pointers

It's a fact of Web life—not all visitors to your Web site are always fully aware of what page they're viewing. As an aid to prevent confusion, and to avoid frustration when a visitor clicks a button that represents the page they're viewing and nothing appears to happen, some designers add a pointer to indicate the current page. Figure 17-6 illustrates one such design.

This approach requires a two-row table, in which a cell in the top row holds the pointer:

```
<HTML>
<HEAD>
<TITLE>Navbar Pointer</TITLE>
</HEAD>
<BODY bgcolor="#FFFFFF">

<TABLE width="100%" border="0">

<TR align="center">
<TD height="36"> </TD>
```

www.it-ebooks.info

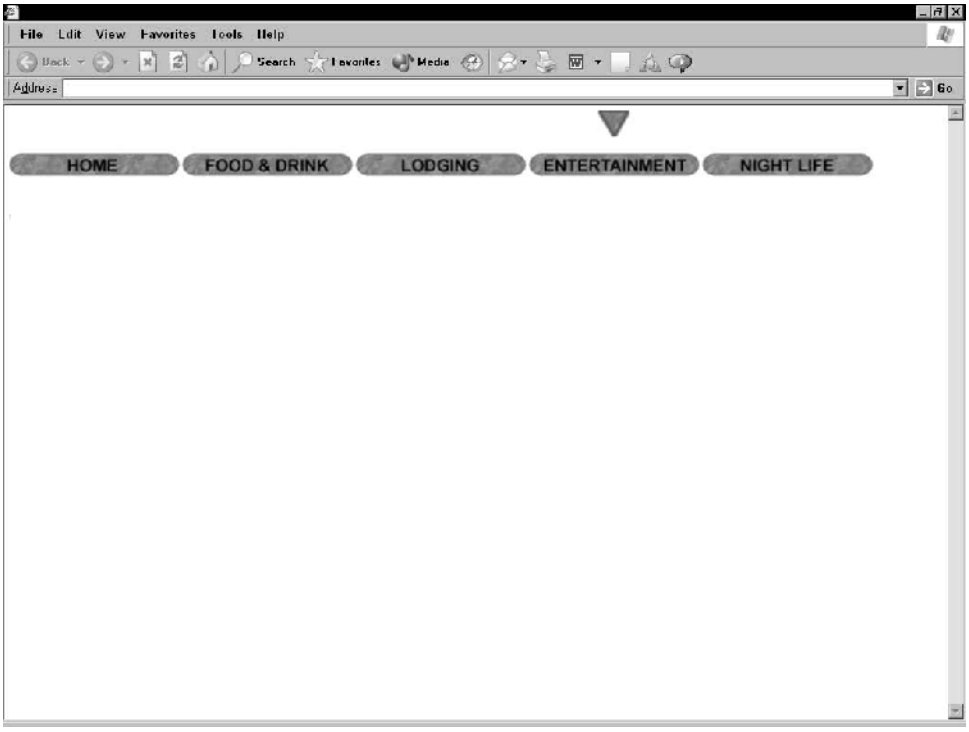


Figure 17-6: A pointer or set of pointers keeps visitors oriented.

```
<TD height="36"> </TD>
<TD height="36"> </TD>
<TD height="36"><IMG src="Dark_Blue_Point_Small_Down.gif"
width="27" height="23"></TD>
<TD height="36"> </TD>
</TR>

<TR align="center">

<TD height="36"><A href="index.html"><IMG src="HOME.gif"
width="152" height="20" border="0"></A></TD>

<TD height="36"><A href="dining.html"><IMG src="FOOD&DRINK.gif"
width="152" height="20" border="0"></A></TD>

<TD height="36"><A href="sleeping.html"><IMG src="LODGING.gif"
width="152" height="20" border="0"></A></TD>

<TD height="36"><A href="attractions.html"><IMG
src="ENTERTAINMENT.gif" width="152" height="20"
border="0"></A></TD>

<TD height="36"><A href="party.html"><IMG src="NIGHTLIFE.gif"
width="152" height="20" border="0"></A></TD>
```

www.it-ebooks.info

```
</TR>

</TABLE>

</BODY>
</HTML>
```



The drawback is that you have to create a slightly different table for each Web page—one in which the pointer is above the appropriate button.

A million different variations on this theme can be found on various Web sites. Some use arrows, others use stylized pointers like the one in the preceding example. Some use a pointing hand such as the ones you've seen in old-fashioned advertisements. Some simply use a different color combination for the button that represents the current page.

As long as it is obvious, practically any pointing-type graphic or change to the button will do. Some sites, however, use indicators that are not instantly comprehensible as pointers. While you may know that a colored dot indicates the current page, it may not be obvious to a first-time visitor to your site.

Using JavaScript rollovers

When you bring JavaScript into the picture, you can have some really fancy navigation bars. With a few simple graphics and some JavaScript, there's really no limit to what you can do. Using several different methods, you can simply set a rollover for navigation that swaps one image for another. Some designers use two button images that are identical except for the colors. The rollover image usually has lighter text and often a lighter background color; thus, when the mouse pointer is over a button, it lights up, making it stand out from the surrounding buttons. Figure 17-7 shows an example of this kind of rollover.

In this example, `onMouseOver` and `onMouseOut` events are added to the `A` element surrounding each button:

```
<HTML>
<HEAD>
<TITLE>Navbar Rollover</TITLE>
</HEAD>

<BODY bgcolor="#FFFFFF">

<TABLE width="100%" border="0" align="center">

<TR align="center">

<TD><A href="index.html"
onMouseOver="document.homebutton.src='HOME_ROLLOVER.gif'"
onMouseOut="document.homebutton.src='HOME.gif'"><IMG
src="HOME.gif" width="152" height="20" border="0"
name="homebutton"></A></TD>
```

www.it-ebooks.info

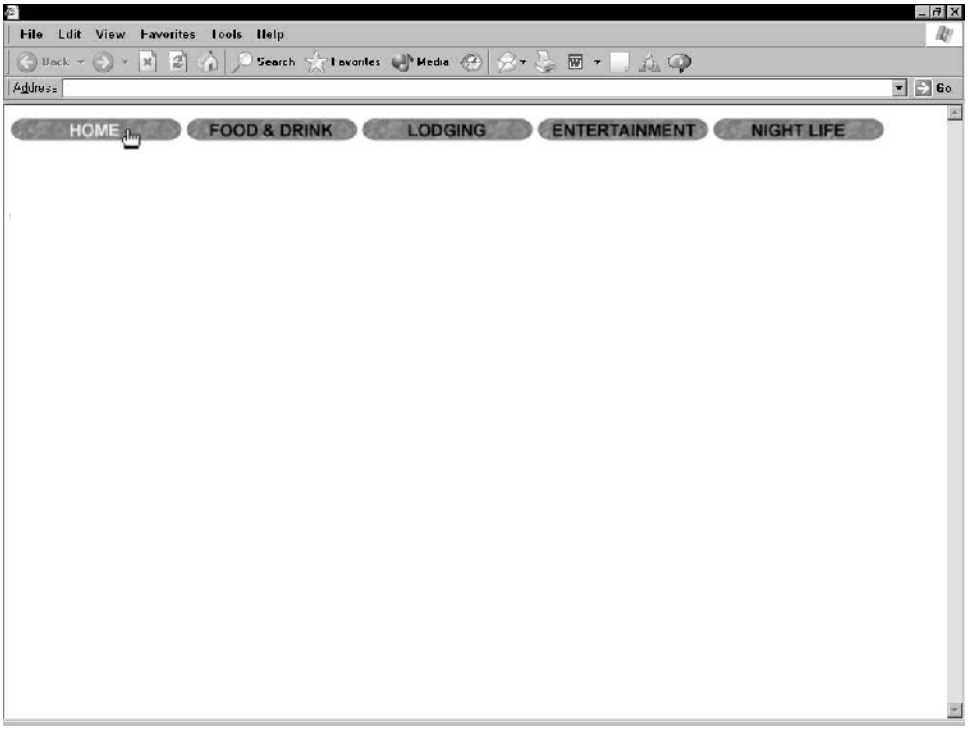


Figure 17-7: JavaScript rollovers change the appearance of a button when the mouse pointer is over it.

```
<TD><A href="dining.html"
onMouseOver="document.homebutton.src='FOOD&DRINK_ROLLOVER.gif'"
onMouseOut="document.homebutton.src='FOOD&DRINK.gif'"><IMG
src="FOOD&DRINK.gif" width="152" height="20"
border="0"></A></TD>

<TD><A href="sleeping.html"
onMouseOver="document.homebutton.src='LODGING_ROLLOVER.gif'"
onMouseOut="document.homebutton.src='LODGING.gif'"><IMG
src="LODGING.gif" width="152" height="20" border="0"></A></TD>

<TD><A href="attractions.html"
onMouseOver="document.homebutton.
src='ENTERTAINMENT_ROLLOVER.gif'"
onMouseOut="document.homebutton.src='ENTERTAINMENT.gif'"><IMG
src="ENTERTAINMENT.gif" width="152" height="20"
border="0"></A></TD>

<TD><A href="party.html"
onMouseOver="document.homebutton.src='NIGHTLIFE_ROLLOVER.gif'"
onMouseOut="document.homebutton.src='NIGHTLIFE.gif'"><IMG
src="NIGHTLIFE.gif" width="152" height="20"
```

www.it-ebooks.info

```
border="0"></A></TD>

</TR>

</TABLE>

</BODY>
</HTML>
```



See Chapter 18 for more information on creating rollovers.

The `onMouseOver` event triggers a change in the source of the image and, therefore, which image is displayed. As a result, the rollover version shows instead of the original image. The `onMouseOut` event changes the image source back to the original button image.

Another common variation is to use a rollover image that is set a few pixels to the right and down from the position of the original button image but is otherwise absolutely identical. The result is a button that appears to be pushed in as the mouse pointer rolls over it. This approach, however, is probably best used with an `onClick` event rather than a `onMouseOver` event, simply because people are already used to programs in which buttons behave in this way when they are clicked.

You can also get into some really complicated scripting using a set of arrows or pointers such as the ones covered earlier in this chapter in the section "Including pointers." Instead of creating a separate set of graphics for each page, you can use one set for all your pages, altering the visibility property of the pointers depending on the page's URL.



See Chapter 14 for information on layer visibility, Chapter 16 for information on creating JavaScript programs, and Chapter 18 for information on rollovers.

Choosing bar orientation

Choosing whether to have a horizontal or vertical navigation bar is simply a question of personal taste, including how well each style fits into your overall page design. Both types of bars are shown in Figure 17-8. Neither design is inherently superior. Horizontal navigation bars have one limitation that is not shared by vertical ones—they can only contain a relatively few number of navigation items, since they are limited by the width of the screen (unless you want to make users scroll horizontally, which is a design no-no). Vertical navigation bars can take advantage of the fact that more buttons fit on a typical Web page vertically without scrolling than horizontally, as well as the fact that user's don't much mind scrolling vertically.

The number of links you want to have in a single navigation bar is another question entirely. In our opinion, horizontal navigation bars should be limited to around half a dozen links, and vertical ones to maybe a dozen. If you do have zillions of links that you need to make available, consider using plain, textual link lists within a table (see "Displaying Links with Lists and Menus").

www.it-ebooks.info

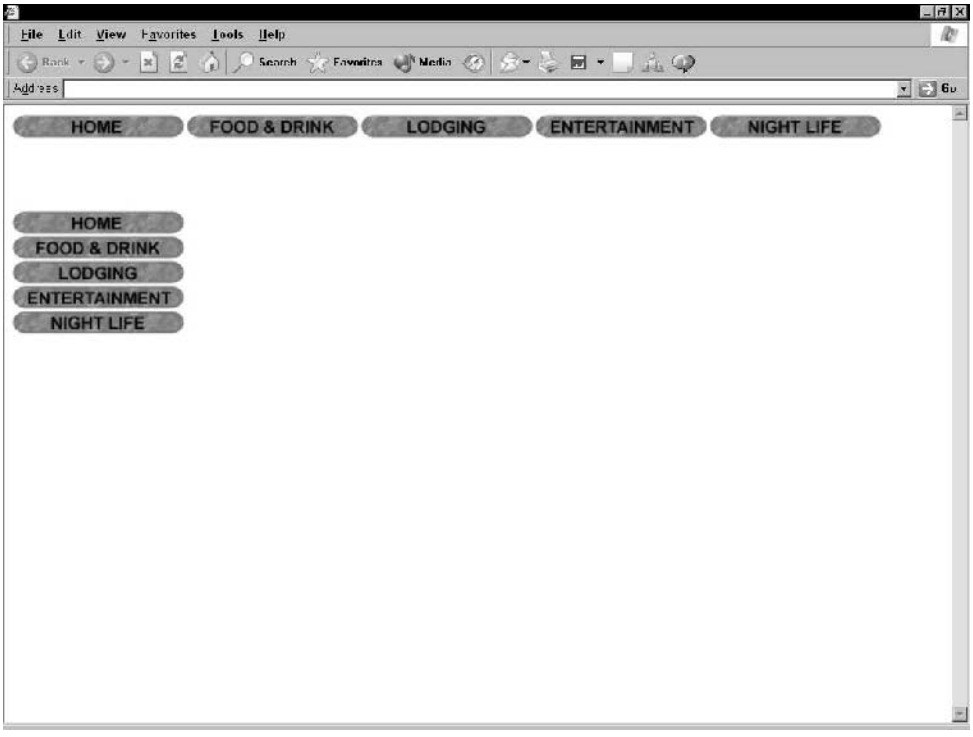


Figure 17-8: You can find use vertical or horizontal navigation bars.

Displaying Links with Lists and Menus

Links can be displayed rather simply by just using plain text, HTML lists, and drop-down menus. The first alternative is the single easiest way to manage link listings, and it works in every Web browser. HTML lists, either ordered or unordered, provide a way to quickly lay out links and also work in all Web browsers. Drop-down menus have a distinct advantage over other approaches because they can contain a large listing in a small space on the Web page.

Using plain text link listings

The quickest and simplest way to lay out plain text link listings is to set them up one link to a line, using the `BR` (line break) element to separate them:

```
<HTML>
<HEAD>
<TITLE>Vertical Link Listing</TITLE>
</HEAD>

<BODY bgcolor="#FFFFFF">
```

www.it-ebooks.info

```
<A href="index.html">Home</A><BR>
<A href="dining.html">Food & Drink</A><BR>
<A href="sleeping.html">Lodging</A><BR>
<A href="attractions.html">Entertainment</A><BR>
<A href="party.html">Night Life</A><BR>

</BODY>
</HTML>
```

A slightly more sophisticated look can be achieved by putting the links into a horizontal setup similar to a navigation bar. In the following HTML code, notice that the first link has a vertical pipe character (|) at both the beginning and end, but the remaining links have one only at the end. This puts a solid line at the start of the link list, between each link in it, and at the end of it (see Figure 17-9). The nonbreaking spaces set a slight distance between the textual content and the vertical lines.

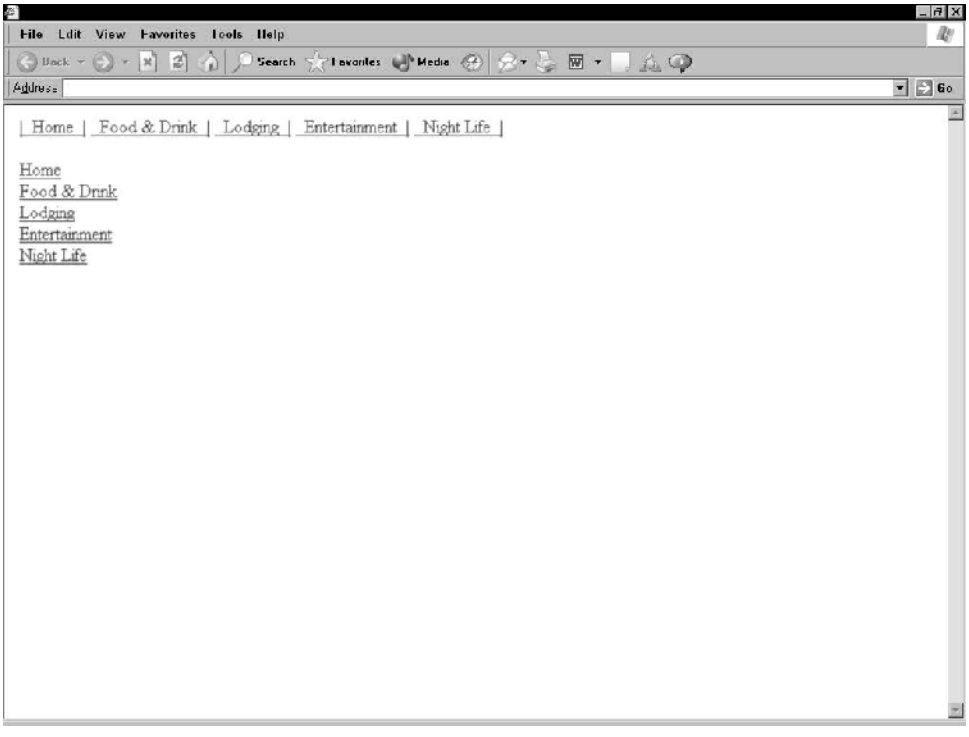


Figure 17-9: Plain link lists can be either horizontal or vertical.

```
<HTML>
<HEAD>
<TITLE>Horizontal Link Listing</TITLE>
</HEAD>

<BODY bgcolor="#FFFFFF">
```

www.it-ebooks.info

```
<A href="index.html">|  Home  |</A>
<A href="dining.html">  Food & Drink  |</A>
<A href="sleeping.html">  Lodging  |</A>
<A href="attractions.html">  Entertainment  |</A>
<A href="party.html">  Night Life  |</A>

</BODY>
</HTML>
```

If you have a very large number of links to show, you can use a table to display, as shown in Figure 17-10. This table can be either on a page of its own or tucked away at the bottom of a Web page. If the table is on a separate Web page, make sure that a link to that page can be easily found on all the other pages in the site.

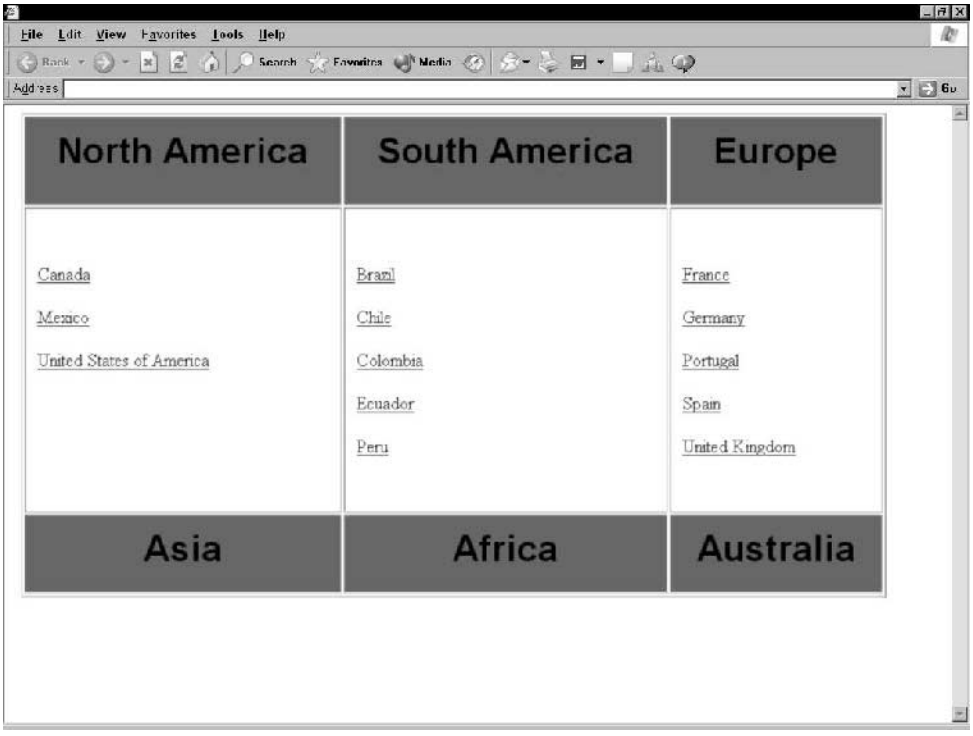


Figure 17-10: Large plain link lists can be organized within a table.

Creating ordered and unordered link lists

Unordered lists, created by the `UL` element in HTML, are commonly called *bulleted lists*. Figure 17-11 shows an example of an unordered list. Ordered lists, created by the `OL` element, are better known as *numbered lists*.

Although HTML lists are almost always only used for plain text, they are much more flexible than that. Each list item, denoted by the `LI` element, can hold nearly any

www.it-ebooks.info

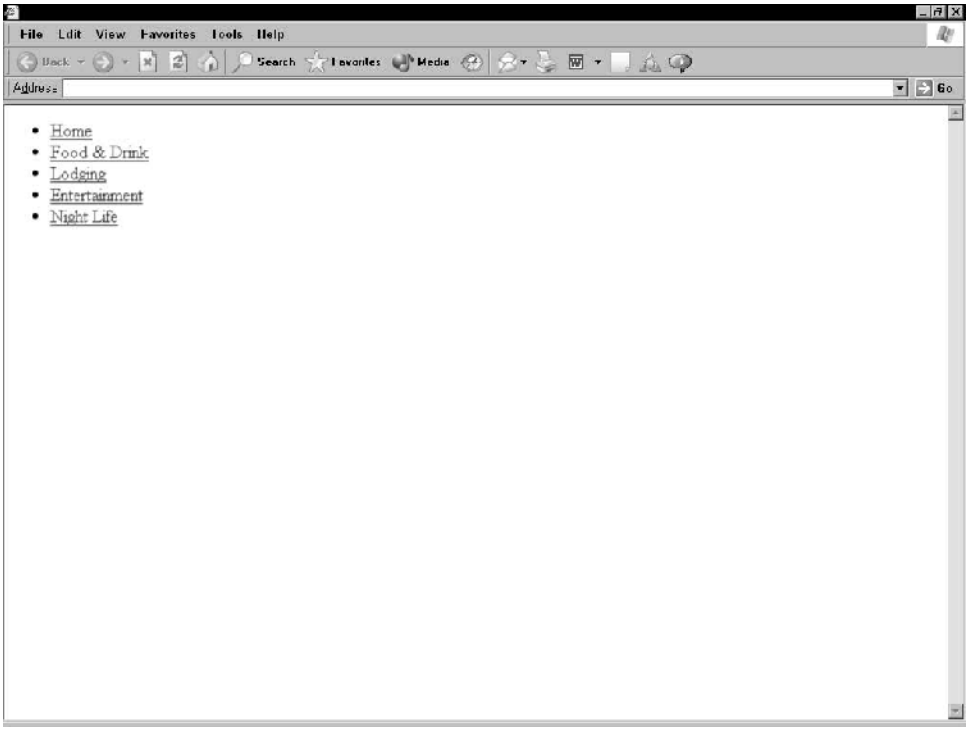


Figure 17-11: This is an unordered (bulleted) list.

HTML element, including tables and images. Here, however, we're taking advantage of using a link (the A element) as a list item.

The following code establishes a bulleted list that provides site visitors with an easily accessed method of linking to different pages:

```
<HTML>
<HEAD>
<TITLE>Unordered Link Lists</TITLE>
</HEAD>

<BODY bgcolor="#FFFFFF">

<UL>
<LI><A href="index.html">Home</A></LI>
<LI><A href="dining.html">Food & Drink</A></LI>
<LI><A href="sleeping.html">Lodging</A></LI>
<LI><A href="attractions.html">Entertainment</A></LI>
<LI><A href="party.html">Night Life</A></LI>
</UL>

</BODY>
</HTML>
```

www.it-ebooks.info

Numbered lists are very similar to bulleted lists, and the code for them is virtually identical to that for bulleted lists. (See Figure 17-12 for an example.) However, their intent and function are totally different. Bulleted lists are used for items whose order doesn't matter, such as a list of parts included in a kit. Numbered lists, however, are used for items whose order is critical, such as a bestseller list or the month's top-ranked salespeople. The following code shows how to create a numbered list:

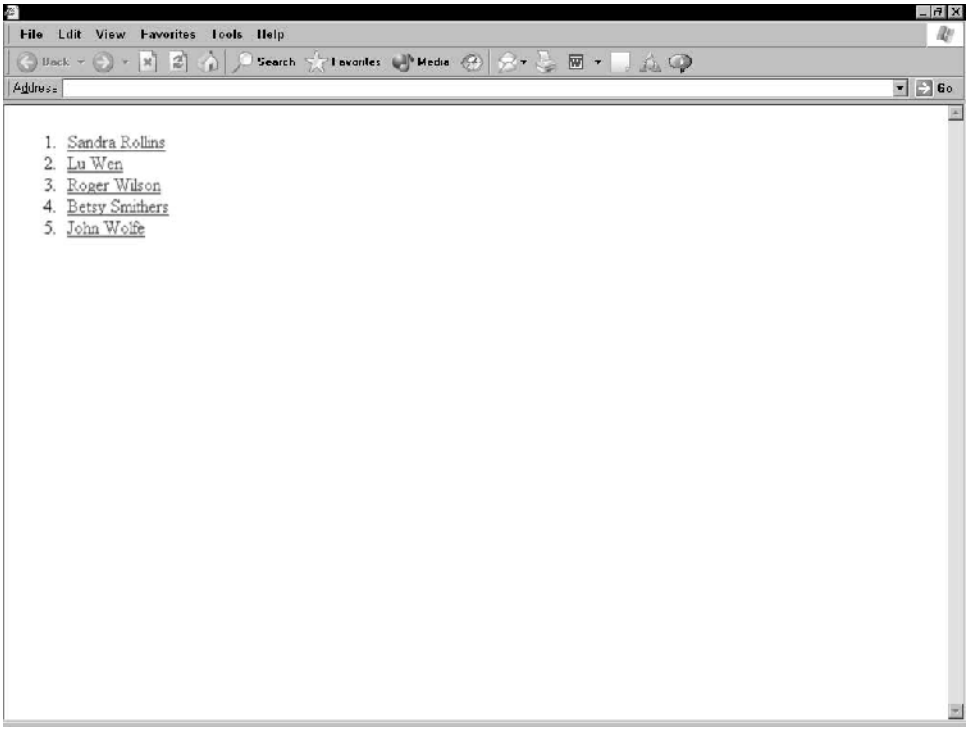


Figure 17-12: This is an ordered (numbered) list.

```
<HTML>
<HEAD>
<TITLE>Ordered Link Lists</TITLE>
</HEAD>

<BODY bgcolor="#FFFFFF">

<OL>
<LI><A href="sales01.html">Sandra Rollins</A></LI>
<LI><A href="sales02.html">Lu Wen</A></LI>
<LI><A href="sales03.html">Roger Wilson</A></LI>
<LI><A href="sales04.html">Betsy Smithers</A></LI>
<LI><A href="sales05.html">John Wolfe</A></LI>
</OL>

</BODY>
</HTML>
```

www.it-ebooks.info

A Java applet called Outline, from programmer David Griffiths, offers a marvelous way to use lists to create a really sophisticated, interactive experience for your visitors. Don't worry if you don't know how to use Java applets. This one is special, and it works without any of the technical stuff you normally need to get a Java applet to work—through a clever use of backward compatibility, it even works in a non-Java compatible browser. You can get Outline from Griffiths's Web site at `www.larchive.com/java/david/popmenu/`. You can't beat the price, because it doesn't cost anything. Mr. Griffiths does have a favorite charity, however, so if you like, you can make a donation to Ty Hafan Children's Hospice (`www.tyhafan.org`).



See Chapter 20 for more information on Java applets.

This applet turns a set of HTML lists into an interactive, collapsible outline—the same sort that Windows Explorer uses to display the folders and files on a computer's hard drive. You need to unzip the file Outline.zip and place the file Outline.class in the same folder that houses your Web page.

Next, you create an unordered list with the following code. Each list item within it becomes a "folder," containing other unordered lists. The list items on the lists contained within the folder become expandable links.

```
<HTML>
<HEAD>
<TITLE>Using the Outline Applet</TITLE>
</HEAD>
<BODY bgcolor="#FFFFFF">

<APPLET code="Outline" width=200 height=400>

Click the listings below to expand them.

<UL>

<LI><A href="index.html">Home</A></LI>

<LI><A href="dining.html">Food & Drink</A></LI>
     <UL>
     <LI><A href="besteats.html">Best Restaurants</A></LI>
     <LI><A href="bestdrinks.html">Best Lounges</A></LI>
     <LI><A href="fastfood.html">Quick & Cheap</A></LI>
     <LI><A href="latest.html">New Spots</A></LI>
     </UL>

<LI><A href="sleeping.html">Lodging</A></LI>
     <UL>
     <LI><A href="cheapest.html">Under $100 a Night</A></LI>
     <LI><A href="midlevel.html">Under $150 a Night</A></LI>
     <LI><A href="good.html">Under $200 a Night</A></LI>
     <LI><A href="tops.html">Price is No Object</A></LI>
     </UL>

<LI><A href="attractions.html">Entertainment</A></LI>
```

www.it-ebooks.info

```
    <UL>
    <LI><A href="close.html">Within the City</A></LI>
    <LI><A href="notfar.html">Within 10 Miles</A></LI>
    <LI><A href="far.html">Afternoon Excursions</A></LI>
    <LI><A href="farthest.html">Overnight Journeys</A></LI>
    </UL>

<LI><A href="party.html">Night Life</A></LI>
     <UL>
     <LI><A href="clubs.html">Clubs</A></LI>
     <LI><A href="movies.html">Movies</A></LI>
     <LI><A href="plays.html">Plays</A></LI>
     </UL>

</UL>

</APPLET>

</BODY>
</HTML>
```



You do not have to indent or space your HTML code as shown in the example. We have done so to show clearly where each outline's level begins and ends.

Figures 17-13 and 17-14 show how Outline works using the Web page created with the preceding HTML code.

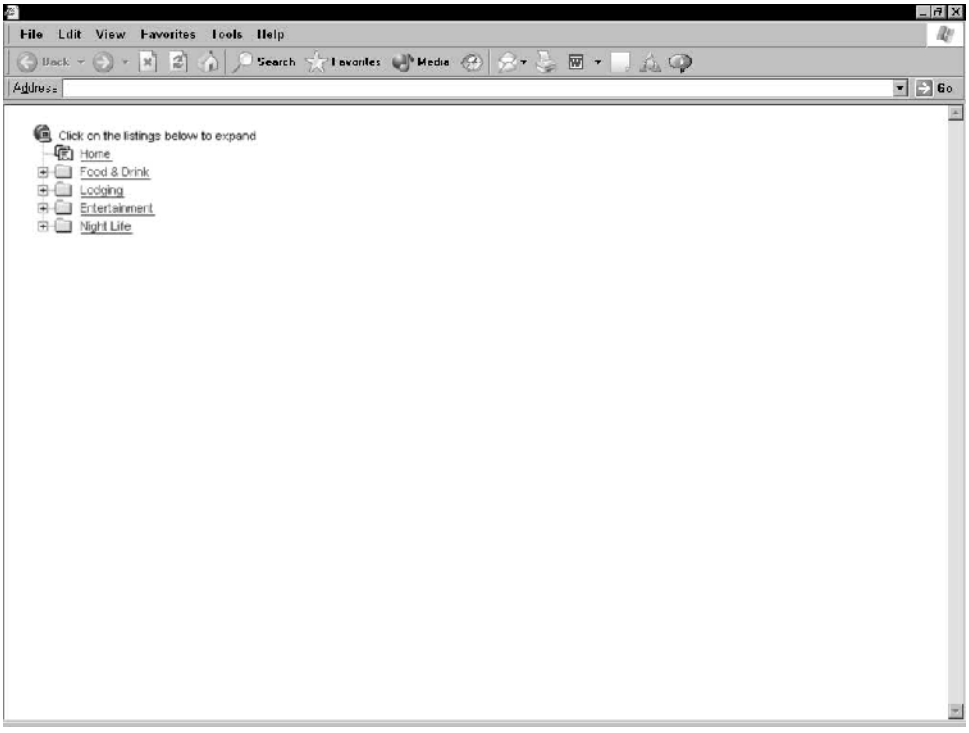


Figure 17-13: Outline makes your lists into a collapsible directory.

www.it-ebooks.info

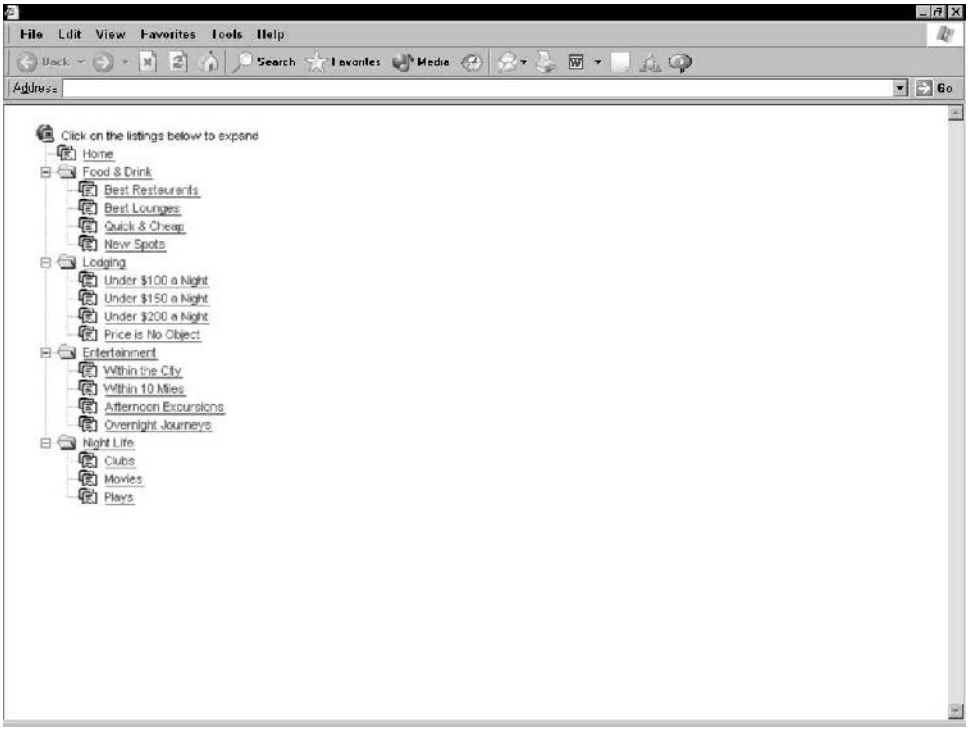


Figure 17-14: The directory expands when the + (plus) marks are clicked.



Because any Web browser that doesn't support Java ignores the `APPLET` element, the contents of the `UL` and `LI` elements are displayed normally in older browsers. The collapsible directory, of course, does not show up, but the basic functionality remains intact regardless of what browser is used.

Making menus with the SELECT element

Link menus created with the `SELECT` element can either resemble a simple list or take the form of a more sophisticated drop-down menu. Because there are many ways to create simple lists, we are going to focus on the creation of drop-down menus, which can be used to store an unlimited number of links that take up only a single line on the Web page.

The `SELECT` element is used in forms. We'll be setting the form action to trigger a simple JavaScript program that displays the page linked to a visitor's selection from the drop-down menu. Figure 17-15 shows the menu created by the following HTML code:

```
<HTML>
<HEAD>
<TITLE>Linking via Forms</TITLE>
</HEAD>
```

www.it-ebooks.info

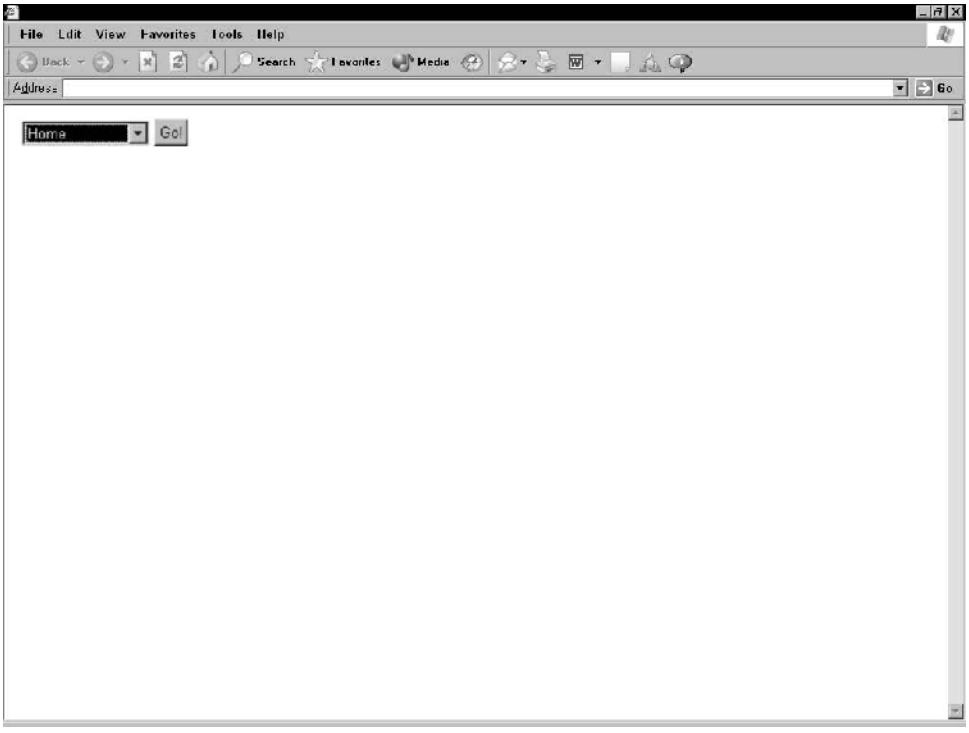


Figure 17-15: Drop-down menus can be combined with JavaScript to create a compact set of links.

```
<BODY bgcolor="#FFFFFF">

<FORM name="goThere">

<SELECT name="pickOne">
<OPTION value="index.html">Home</OPTION>
<OPTION value="dining.html">Food & Drink</OPTION>
<OPTION value="sleeping.html">Lodging</OPTION>
<OPTION value="attractions.html">Entertainment</OPTION>
<OPTION value="party.html">Night Life</OPTION>
</SELECT>

<INPUT type=button
onClick="location=document.goThere.pickOne.options
[document.goThere.pickOne.selectedIndex].value" value="Go!">

</FORM>

</BODY>
</HTML>
```

The `onClick` event handler assigned to the button changes the URL of the displayed Web page to equal the value of the selected option. The result is that the Web browser loads the selected page.

www.it-ebooks.info



See Chapter 15 for information on forms and form actions.

Summary

Navigating your Web site should never be a burden to your visitors. There are many ways in which you can make the experience easier on them.

This chapter introduced you to the following concepts and topics:

- ✦ Navigation rules should be planned early in the site development process.
- ✦ Navigation bars are a useful approach to providing easy access to your pages.
- ✦ Graphics can enhance both the appearance and functionality of navigation bars.
- ✦ JavaScript can be used in combination with lists to save space on the Web page.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 18

Adding Dynamic Page Elements with DHTML

In This Chapter

Understanding the hazards of animation

Getting dynamic with rollovers

Putting elements in motion

Color fades

Dynamic page elements add a bit of action and interaction to your Web pages. In the early days of the Web, the only interactivity available to Web users was to click a link and go to a new Web page. With the advent and increasing sophistication of both JavaScript and Web page content, however, a whole new range of possibilities has opened up.

Understanding the Hazards of Animation

While it's true that people have come up with some nice ways to use animation for adding functionality to Web pages, the main appeal of animation is just plain fun. You can achieve the same functional goals through more pedestrian methods.

Even a bit of color and movement is more appealing than anything static. The human eye, unique among the eyes of all creatures, is geared toward detecting both texture and sudden changes. Your visitors, as advertisers have long been aware, cannot help but have their eye caught by flash and motion.

However, it is easy to get carried away with animation, and that's detrimental to your site for a couple of reasons. First, it's like getting carried away with spices—eventually, you can't taste anything else and the flavor of the original food is lost altogether. If you put in too many animations, none of them will have much appeal. Second, most animation represents a drain on someone's system resources—yours or those of your visitors. Good programming practice says to have only the minimum necessary amount of drain.

www.it-ebooks.info



See Chapter 19 for information on creating animations with Macromedia Flash.

Getting Dynamic with Rollovers

Rollovers are user interactions that are based on JavaScript `onMouseOver` and `onMouseOut` events. Thus, they are triggered when a visitor moves the mouse pointer over a graphic or when he or she moves it off the graphic. Although many Web designers use rollovers only for setting up navigation bars, you can do much more with them.



See Chapter 16 for details on how JavaScript works and Chapter 17 for more information on site navigation.

Swapping images

The single most common use of rollovers is to swap one image for another so that when a visitor moves his or her mouse pointer over an image, it changes into another image. The following code example shows how it's done:

```
<html>
<head>
<title>Simple Image Rollover</title>
</head>
<body style="background:blue">
<img src="cloud1.png" width="200" height="100"
onMouseOver="this.src='cloud2.png'"
onMouseOut="this.src='cloud1.png'">
</body>
</html>
```

You want the original image and the image shown after a `onMouseOut` event to be the same. The image shown during a `onMouseOver` event (while the mouse pointer is over the image) is the one that should be different. Figure 18-1 shows the second image while it is activated.

However, if you're going to set up an image swap rollover, you need to deal with one little detail in order to make the whole thing work smoothly—you need to preload all the images. Generally, when a Web page downloads to a Web browser, the only images that are downloaded along with it are the ones that are actually displayed at the time the Web page is loaded. Any other images are left behind on the Web browser and are only downloaded when called upon. What this means for rollovers, in practical terms, is that if someone mouses over the first image, nothing happens until the second image is downloaded from the Web server; only then can it be displayed.



You won't see this effect when working with rollovers on your local computer system because the images are already there on your own drive.

www.it-ebooks.info

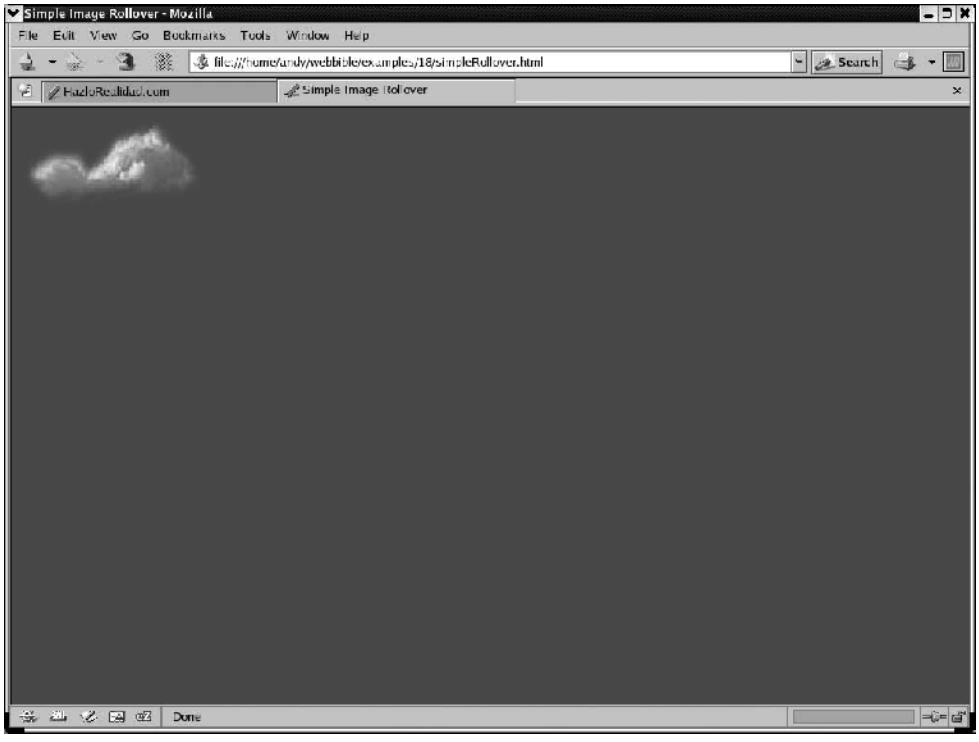


Figure 18-1: The second image appears during a rollover.

The following example code demonstrates how to preload images so that there will be no delay in displaying them on your Web pages:

```
<html>
<head>
<title>Preloaded Image Rollover</title>
<script language="javascript">
<!-- start hide
var firstPicture = new Image();
var secondPicture = new Image();
firstPicture.src="cloud1.png";
secondPicture.src="cloud2.png";
// end hide -->
</script>
</head>
<body>
<img src="cloud1.png" width="200" height="100"
onMouseOver="this.src='cloud2.png'"
onMouseOut="this.src='cloud1.png'">
</body>
</html>
```

The two variables, `firstPicture` and `secondPicture`, are assigned values that create two empty image objects. The files that belong to these image objects are

www.it-ebooks.info

then specified with the `firstPicture.src` and `secondPicture.src` properties. This forces the image files to be downloaded along with the Web page.

Triggering changes in other elements

While the vast majority of rollovers on the World Wide Web are used to change the element affected by the `onMouseOver` event, there is actually no requirement that this should be so. It is a common practice to use a form button to trigger changes in other elements, but few Web designers seem to realize that any event can be used to launch practically anything else on a Web page, and rollovers are no exception.

Changing colors

The colors of other elements can be altered by an `onMouseOver` event as well. The following example code shows how to change the background color of a Web page to different colors depending upon which link is moused over:

```
<HTML>
<HEAD>
<TITLE>Changing Background Color</TITLE>
</HEAD>
<BODY bgcolor="#FFFFFF">

<A href="" onMouseOver="document.bgColor='#FF0000';"
onMouseOut="document.bgColor='#FFFFFF'">Red</A>

<A href="" onMouseOver="document.bgColor='#00FF00';"
onMouseOut="document.bgColor='#FFFFF'">Green</A>

<A href="" onMouseOver="document.bgColor='#0000FF';"
onMouseOut="document.bgColor='#FFFFFF'">Blue</A>

</BODY>
</HTML>
```

Figure 18-2 shows the background color changing in relation to which link is moused over.

The W3C recommends that Web designers change to using CSS—according to them the last example is deprecated.

The following shows the last example modified to use CSS:

```
<html>
<head>
<title>Changing Background Color Using CSS</title>
</head>
<body id="body" style="background:#FFFFFF;">

<a href="" onMouseOver="document.body.style.background='#FF0000';"
onMouseOut="document.body.style.background='#FFFFFF'">Red</a>
```

www.it-ebooks.info

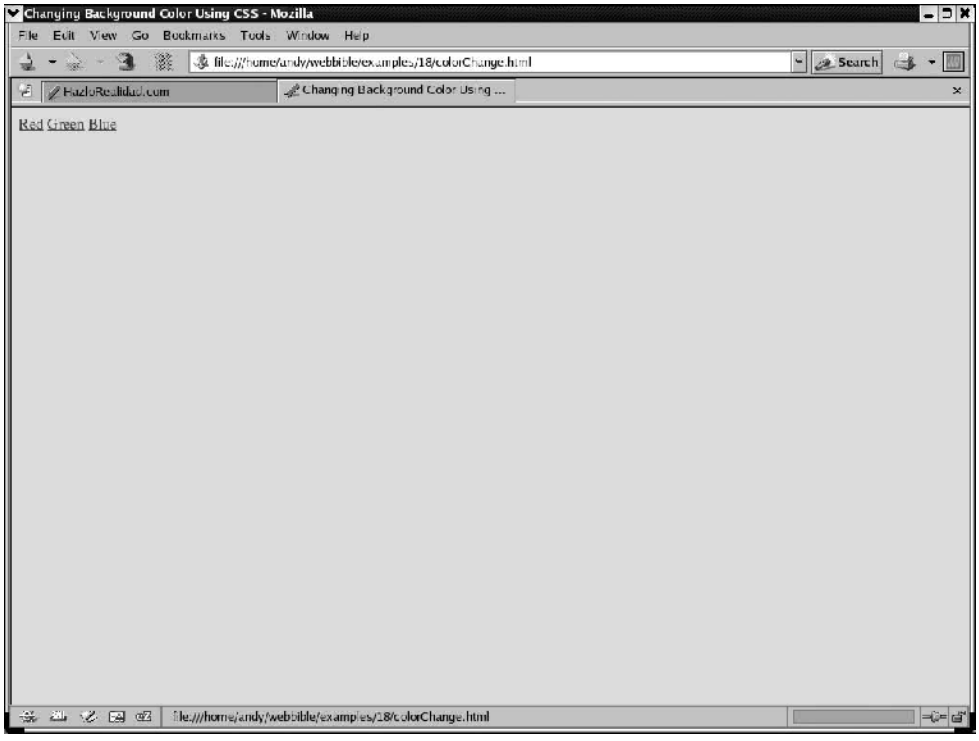


Figure 18-2: MouseOver events can be used to change background color.

```
<a href="" onMouseOver="document.body.style.background='#00FF00';"
onMouseOut="document.body.style.background='#FFFFFF'">Green</a>

<a href="" onMouseOver="document.body.style.background='#0000FF';"
onMouseOut="document.body.style.background='#FFFFFF'">Blue</a>

</body>
</html>
```

Setting the status bar text

It's easy to write your own text into a Web browser's status bar—the message section on the very bottom—but there is some question among Web designers about exactly when it should and shouldn't be done. The status bar contains useful information about the current state of affairs as you surf the Web—that's what it's there for. When you write your own messages for it, you override anything else that's already there, and thus, you should exercise caution when making such a change. The following code example shows how to do it:

```
<HTML>
<HEAD>
<TITLE>Sending a Message to the Status Bar</TITLE>
<SCRIPT language="JavaScript">
```

www.it-ebooks.info

```
<!-- Make this script invisible to old browsers.

function barWrite(barMessage) {
window.status = barMessage;
}

// Back to normal. -->
</SCRIPT>

</HEAD>
<BODY bgcolor="#FFFFFF">

<A href="" onMouseOver="barWrite('This Space Available'); return true;"
onMouseOut="barWrite(''); return true;">Mouse Over These Words and Look at
the status bar</A>

</BODY>
</HTML>
```

Figure 18-3 shows the message in the status bar.

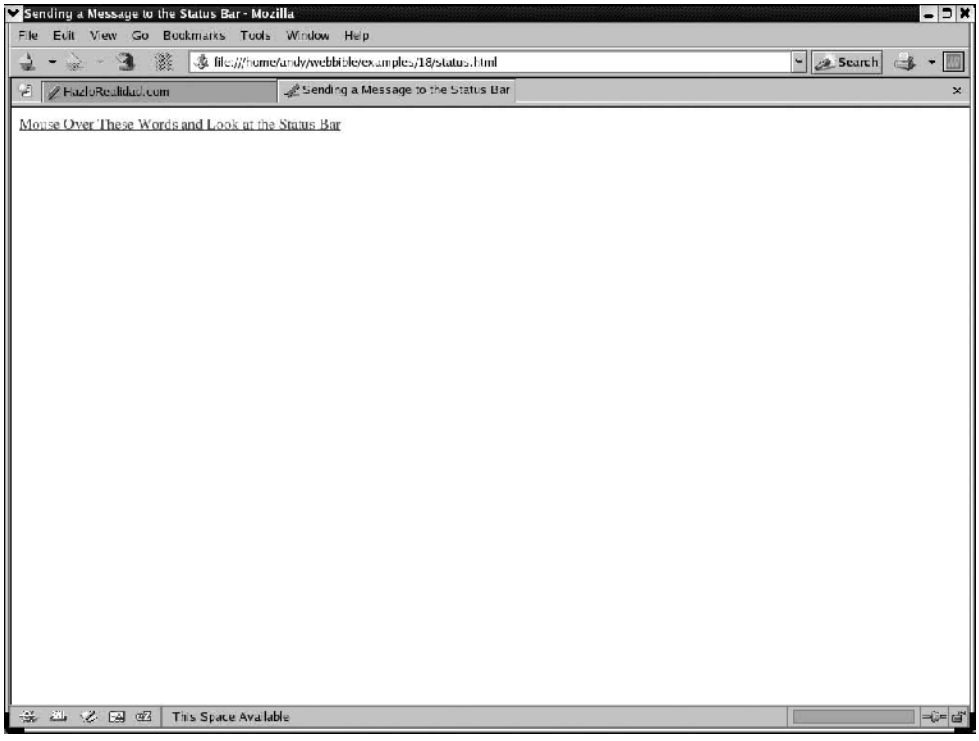


Figure 18-3: You can custom-write status bar messages.

The function `barWrite()` accepts the input from the `onMouseOver` event and assigns it to the `window.status` method, which writes it into the status bar. The

www.it-ebooks.info

same function handles the `onMouseOut` event, writing a blank message to the status bar to clear it.

Putting Elements in Motion

In theory, under the Document Object Model (DOM), every element on a page can be absolutely positioned. While this may be true in the latest model Web browser, it's also true that most people on the Web aren't using the latest model Web browser, at least not yet. For now, it's safest to stick with animating layers.



See Chapter 14 for more information on using layers.

Calculating absolute position

It's an easy thing to move a layer around the screen, and the level of mathematical skill required is minimal. In most cases, it's simply a matter of figuring out how many pixels you want the layer to move, and the only complexity is deciding whether or not you want to move vertically as well as horizontally.

You could, of course, use higher mathematics to work up some incredibly complex layer animation pathway, but it's doubtful whether this would actually add to either the functionality of your Web page or the enjoyment of your visitors' experience in most cases.

If you want to reveal your page when the user enters it like a slide show, you could do this in the following way:

```
<html>
<head>
<title>Sliding Left Transitions DOM Browser only</title>
<script language="JavaScript">
<!-- Make this script invisible to old browsers.
//The interval timer we will use
//(we need this so that we can cancel it)
var proc;
//The element that we will move
var element;

function start()
{
  //obtain the element we will move
  element=document.getElementById("panel");
  //start the task to move it every 50 milliseconds
  proc=setInterval("change()",50);
};

function getLeft()
{
  valueAsString=element.style.left;
```

www.it-ebooks.info

```
  //parseInt needed to stip off trailing "px" from
  //the position
  value=parseInt(valueAsString);
  if (isNaN(value))
  {
        value=0;
  }
  return value;
};

function getWidth()
{
  valueAsString=element.style.width;
  value=parseInt(valueAsString);
  if (isNaN(value))
  {
        value=0;
  }
  return value;
};

function setLeft(value)
{
  element.style.left=value+"px";

};

function change()
{
  //get the current x coordinate and width
  left=getLeft();
  width=getWidth();
  //move it left 5 pixels
  left=left-5;
  setLeft(left);
  //when we have moved it completely off the left hand side
  //cancel the task
  if (left+width<0)
  {
         clearInterval(proc);
  }
};

// Back to normal. -->
</script>

</head>
<body onload="start();" bgcolor="#FFFFFF">

<div id="panel" style="position:absolute;width:600;
height:400;background-color:yellow;font-size:36pt;color:red;
text-align:left;z-Index:5;">
Is this a cool transition?
</div>
```

www.it-ebooks.info

```
<div id="text" style="width:600; height:100; font-size:36pt;color:blue">How
do you like the effect?</div>
</body>
</html>
```

This is designed to work in any modern browser (technically speaking, one that supports the DOM). We'll discuss how to make it work in older browsers later on. But first a few explanations on how the code breaks down.

The two `div` elements in the body of the HTML code could contain anything, including an image or your logo. For the first `div`, the position is specified in absolute coordinates and is set to be on top of the second `div`, using the `z-index` attribute. (See Chapter 14 for details.)

When the page loads the `onload` attribute of the body element tells it to call the function `start()`.

The method `start()` saves a reference to the element that we will be moving and sets up a timer to call the method `change()` that will actually do the moving.

The `setInterval()` method creates this timer which will cause another function to repeat until it is canceled. In this case, it repeats the function `change()`. The 50 in the `setInterval` invocation sets the speed of the repetition in milliseconds, 50ms will be 1000/50 = 20 frames a second, a good number as this is what cine films use, the highest speed is 1—once every millisecond (thousandth of a second). To make things change more slowly, use a higher number; the number 100 would cause the called function to repeat every 100 milliseconds (or every tenth of a second), the number 1000 every second, and so on. Of course, you don't have to use multiples of 10; you can use any integer.

The `change()` function does the actual moving of the layer. It determines if the right side of the layer is still on the screen. If it is, then the left side is moved one more pixel to the left. When the layer is invisible, the `clearInterval()` method stops the movement, as shown in Figure 18-4.

Support for old browsers

The previous example uses the DOM, but this is a more recent development. Today there are still a small percentage of people using browsers that don't support the DOM. The reason the DOM was created was to standardize Web development, because each of the old browsers has its own standards.

However, if you want your site to be seen by every person on the Internet then you need to take these differences into account. The best practice is to factor out every part that varies into a separate JavaScript function so that the part where you put the actual movement is the same for every browser. This helps you or anyone else reading the code to understand it better, and you can also reuse the functions in all your programs.

www.it-ebooks.info

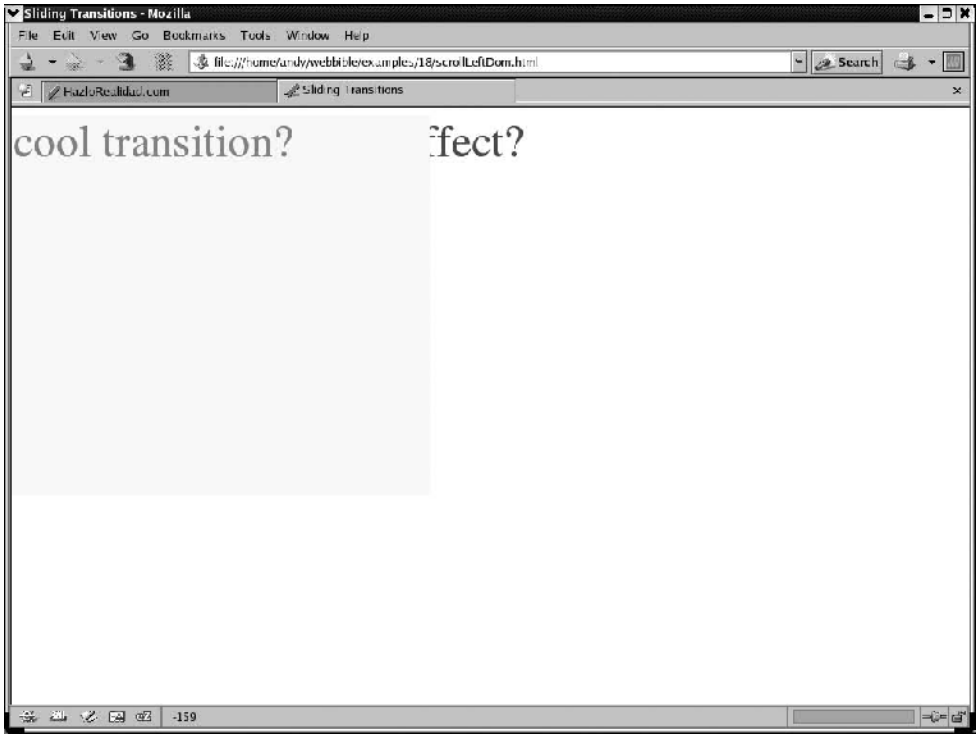


Figure 18-4: The `clearInterval()` method halts the repetitious calling of the `slideThis()` function.

You'd be surprised how many programmers don't do this. The following code shows how to add support for early versions of Internet Explorer and Netscape Navigator:

```
<html>
<head>
<title>Sliding Transitions</title>
<script language="JavaScript">
<!-- Make this script invisible to old browsers.
var proc;
var element;

function start()
{
  // if we are a DOM browser
  if (document.getElementById)
  {
        element=document.getElementById("panel");
  }
  // if we are an early version of Internet Explorer
  else if (document.all)
  {
        element=document.all.panel;
  }
```

www.it-ebooks.info

```
  // if we are early version of Netscape Navigator
  else if (document.layers)
  {
         element=document.panel;
  }
  // if we are some other browser
  else
  {
         alert("To see the animation you need to upgrade your browser");
         return;
  }
  // set up timer for the animation
  proc=setInterval("change()",50);
};

function getLeft()
{
  if (document.getElementById)
  {
         // DOM way to get the left coordinate
         valueAsString=element.style.left;
  }
  else if( document.all)
  {
         // Explorer's way
         valueAsString=element.style.pixelLeft;
  }
  else if (document.layers)
  {
         // Navigator's way
         valueAsString=element.left;
  }
  // No else as we never started the timer if we are not
  // one of the above and so should never get here

  // convert a string in a number
  value=parseInt(valueAsString);
  // if the value is Not A Number then set it to 0
  if (isNaN(value))
  {
         value=0;
  }
  return value;
};

function getWidth()
{
  if (document.getElementById)
  {
         // DOM width
         valueAsString=element.style.width;
  }
  else if( document.all)
  {
         // Explorer Width
```

www.it-ebooks.info

```
        valueAsString=element.style.width;
  }
  else if (document.layers)
  {
        //Navigator Width
        valueAsString=element.width;
  }
  // Same as in getLeft - convert to number
  value=parseInt(valueAsString);
  if (isNaN(value))
  {
       value=0;
  }
  return value;
};

function setLeft(value)
{
  if (document.getElementById)
  {
        // DOM way to set left value
        element.style.left=value+"px";
  }
  else if(document.all)
  {
        //Explorer's way to set the left value
        element.style.pixelLeft=value;
  }
  else if (document.layers)
  {
        // Navigator's way to set left coordinate
        element.left=value;
  }
};

// Now that all of the browser dependent functions have been
// set up the actual movement code is really simple
function change()
{
  left=getLeft();
  width=getWidth();
  left=left-5;
  setLeft(left);
  if (left+width<0)
  {
        clearInterval(proc);
  }
};


// Back to normal. -->
</script>

</head>
<body onload="start();" bgcolor="#FFFFFF">
```

www.it-ebooks.info

```
<div id="panel" style="position:absolute;width:600;
height:400;background-color:yellow;overflow:hidden;
font-size:36pt;color:red;text-align:left;vertical-align:middle;
white-space:nowrap;z-Index:5;">
Is this a cool transition?
</div>
<div id="text" style="width:600; height:100; font-size:36pt;
color:blue">How do you like the effect?</div>

</body>
</html>
```

Determining screen size

There are, however, a few important things to keep in mind when figuring out where your layers are going to travel and where they're going to land. First and foremost, keep in mind the anticipated screen size the typical site visitor is working with. This factor has always had to be taken into account with anything that might show up differently on different screen resolutions, such as the size of background images or the use of preformatted text.

Many sites state that "the pages are best viewed with a screen resolution of 800 × 600". However, statistics show that most people on the Web today have the screen resolution of 1024 × 786. The Web page should take into account the screen size. It is absurd to try to enforce a standard screen size. Anyone visiting a site that doesn't show up well in his or her screen resolution is most probably just going to visit another site.

When it comes to absolute positioning of layers, however, screen size becomes especially important because it's not just a matter of accidentally creating a less-than-pleasing background. If you want to move the layer off to the right, you could just decide to change the line `left=left-5` to `left=left+5`. That would move it to the right, but it would never stop moving. What you really want to do is stop when the left side of the layer is more than the width of the browser window.

But how far away is the right side of the Web page? It's not going to be the same for everyone. If someone comes in with a screen resolution other than 800 × 600 (for example, 1024 × 768 or 1600 × 1200), you have a whole different situation on your hands. If you have your JavaScript code fixed to move the layer 600 pixels to the right, then it's going to end up hanging somewhere near the right border or even near the middle of the screen.

Luckily, JavaScript has a property for the screen width (called `screen.width`). However, it doesn't tell you the width of the window. This is given by the property `window.innerWidth`.

The modified function change becomes:

```
function change()
{
```

www.it-ebooks.info

```
    left=getLeft();
    width=getWidth();
    left=left+5;
    setLeft(left);

    if (left+width<0 || left>window.innerWidth)
    {
           clearInterval(proc);
    }
};
```

However, there is a problem. In most browsers the moving layer creates a horizontal scroll bar as it moves past the right side of the screen, as shown in Figure 18-5.

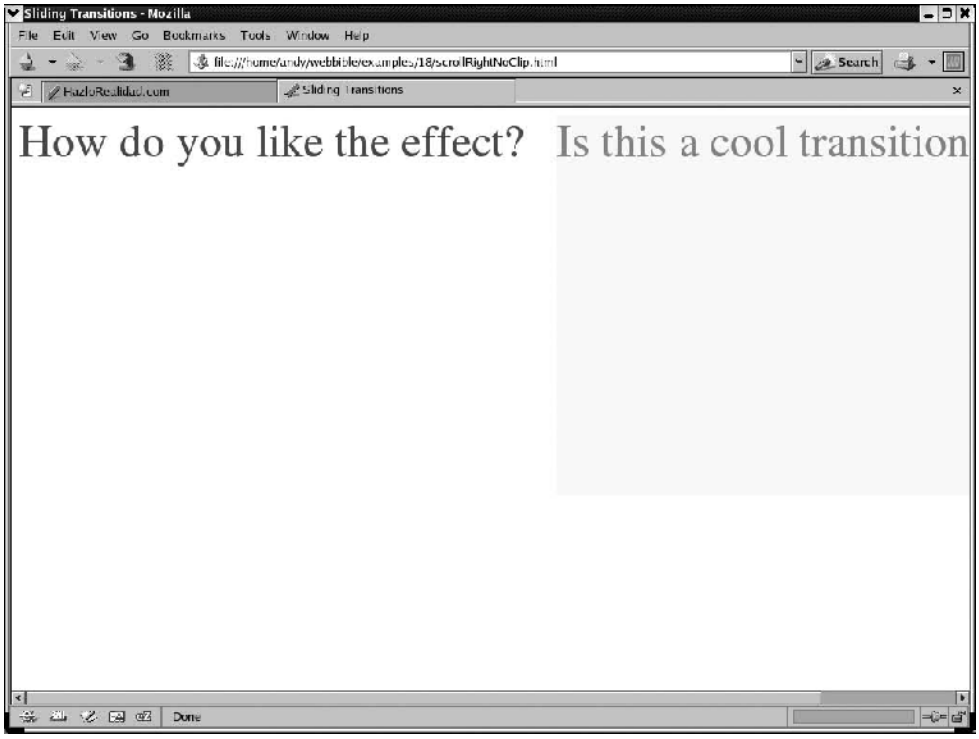


Figure 18-5: Most browsers create a horizontal scrollbar when a layer moves off the right side of the screen.



In both Web browsers, however, you can move a layer off the left side or above the top without creating a scroll bar.

This is a bit of a problem because the user can use the scrollbar to see the layer that you have just tried to hide, in addition to being aesthetically unpleasing. But how

www.it-ebooks.info

can you reduce the width of the layer as it is moving so that it never actually goes off the screen? The answer is to write a `setWidth` function:

```
function setWidth(value)
{
  if (value<=0)
  {
        value=0;
  }
  if (document.getElementById)
  {
        element.style.width=value+“px”;
  }
  else if(document.all)
  {
        element.style.width=value;
  }
  else if (document.layers)
  {
        element.width=value;
  }
};
```

Then modify the change function:

```
function change()
{
  left=getLeft();
  width=getWidth();
  left=left+5;
  // the available width on the window
  dif=window.innerWidth-left;
  // if the actual width is more than what’s available
  if (width>dif)
  {
        // change width to be what’s available
        setWidth(dif);
  }
  // change left position after changing width to
  // avoid the scrollbar appearing
  setLeft(left);
  // if we moved off screen to the left or right then stop
  if (left+width<0 || left>window.innerWidth)
  {
         clearInterval(proc);
  }
};
```

Problem solved? Not quite. Have you noticed that as the text moves off to the right, the browser decides that the text doesn’t fit in the width of the layer and so helpfully inserts a few line breaks? Then when the text doesn’t fit in the box it displays it anyway (see Figure 18-6). This is not what we want.

www.it-ebooks.info

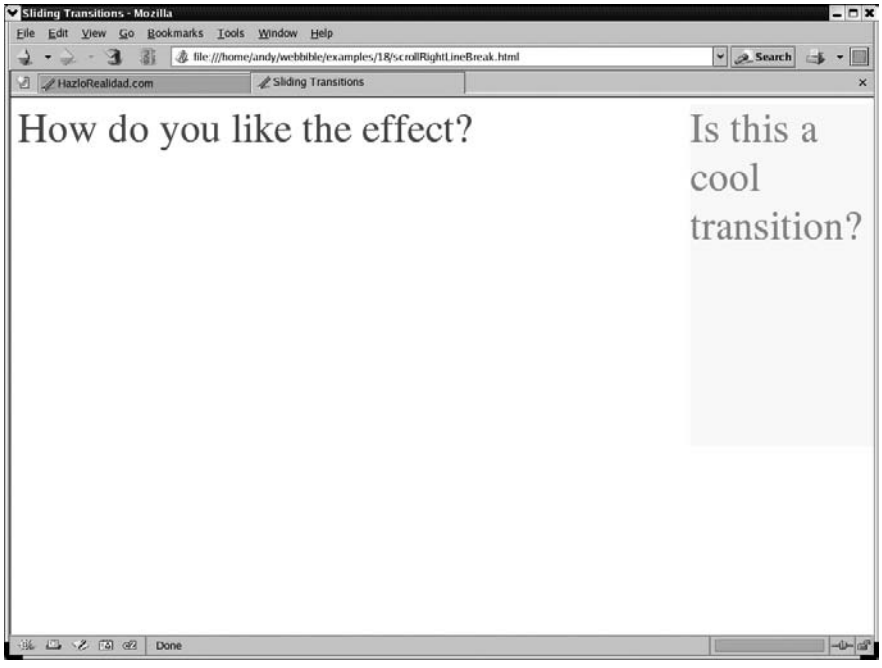


Figure 18-6: Note that the browser inserts line breaks into the text as it moves off the screen.

To fix this change the style for the panel:

```
<div id="panel" style="position:absolute; width:600; height:400;
background-color:yellow; overflow:hidden; font-size:36pt; color:red;
text-align:left; white-space:nowrap; z-Index:5;">
Is this a cool transition?
</div>
```

Here's what is defined in the `div`: `overflow` is what happens when there is not enough physical space to fit all of the contents of a layer inside itself; `hidden` means don't write outside the layer hide what doesn't fit; `white-space:nowrap` means don't insert extra line breaks in order to try to make the contents fit. Once you understand and implement this, you should have no more problems. Figure 18-7 shows that the problem is now solved.



Chapter 13 for more details on CSS.

Negative position values move layers out of the visible area of the screen.



You could also use the visibility characteristics of a layer (see Chapter 14) to render it invisible at the beginning and end of this run across the screen. However, this code handles the situation more elegantly, both by making the layer's appearance and disappearance smoother and by using the same single stretch of code to achieve the appearance, the transit, and the disappearance.

www.it-ebooks.info

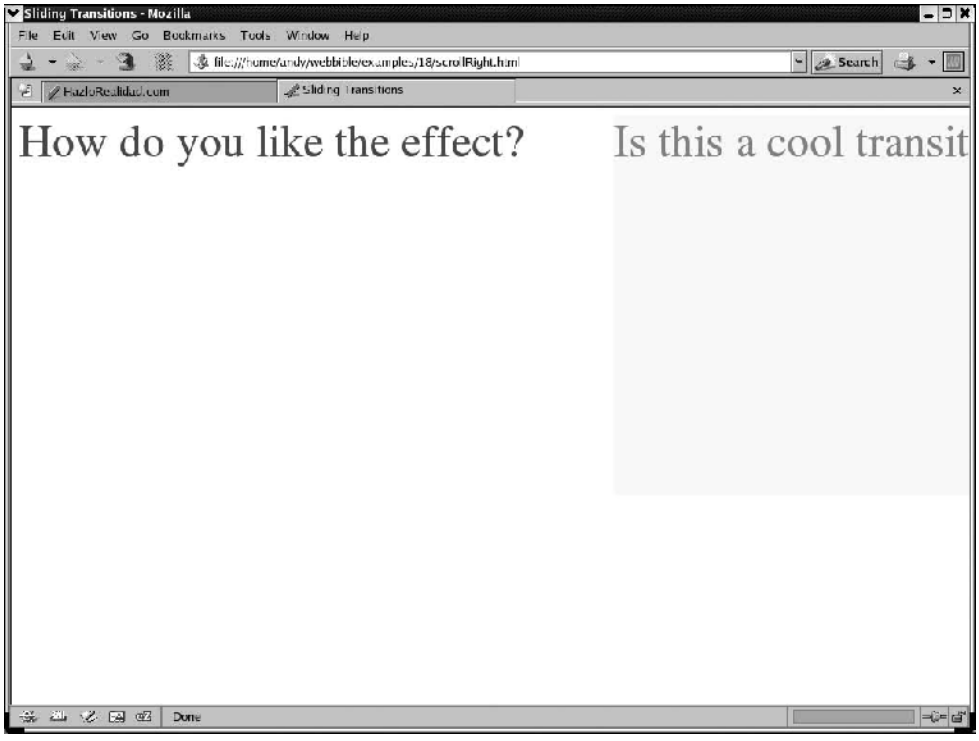


Figure 18-7: Now it works as it should with no line breaks and no text shown outside the div.

Color Fades

You may have guessed by now that you can animate almost any CSS property. Let's have a look at animating color—a really nice effect that you can add to your Web pages.

The following example is very similar to the last, but this time the line that does all the work is this:

```
document.getElementById("panel").style.color
="rgb("+curRed+","+curGreen+","+curBlue+")";
```

All of the rest of the script is just to determine what is the current color. We define the starting color, the end color, the number of steps, and the interval between each frame. Then we use a bit of high school mathematics to interpolate the color according to which frame or step we are displaying. Remember all those boring classes drawing graphs of y=mx+c? Now we are going to find a good use for it. In this case y is the color to show, x is the step number, c the start color (at step 0) and m the color change per step. Plugging the values into the equation gives a smooth color fade from the start color to the end color. Note that we use `Math.round` to round up or down the color to the nearest whole number. To verify that the code is

www.it-ebooks.info

working how we expect, we have used the status bar to display the red, green, and blue components of the color.

We won't go into too much detail on the code. Here it is:

```
<html>
<head>
<title>Color Fades</title>
<script language="JavaScript">
<!-- Make this script invisible to old browsers.
var proc;

var curRed;
var curGreen;
var curBlue;
var startRed;
var startGreen;
var startBlue;
var endRed;
var endGreen;
var endBlue;
var step;
var steps;
var interval;

function start()
{
  step=0;
  steps=100;
  time=5000;
  interval=time/steps;
  setStartColor(0,0,255);
  setEndColor(255,0,0);

  proc=setInterval("change()",interval);

};

function setStartColor(red,green,blue)
{
  startRed=red;
  startGreen=green;
  startBlue=blue;
};

function setEndColor(red,green,blue)
{
  endRed=red;
  endGreen=green;
  endBlue=blue;
};

function change()
{
  curRed=Math.round((endRed-startRed)*step/steps+startRed);
  curGreen=Math.round((endGreen-startGreen)*step/steps+startGreen);
```

www.it-ebooks.info

```
  curBlue=Math.round((endBlue-startBlue)*step/steps+startBlue);
  window.status=curRed+" "+curGreen+" "+curBlue;
  document.getElementById("panel").style.color="rgb("+curRed+","+curGreen+",
  "+curBlue+")";
  if (++step>steps)
  {
        clearInterval(proc);
  }
};

// end hide -->
</script>

</head>
<body onload="start();">

<div id="panel" style="width:600; height:400;background-color:yellow;color:
red;font-size:36pt;text-align:center;">
What a great color fade!
</div>

</body>
</html>
```

Figure 18-8 shows the code in action. Note the debugging information in the status bar.

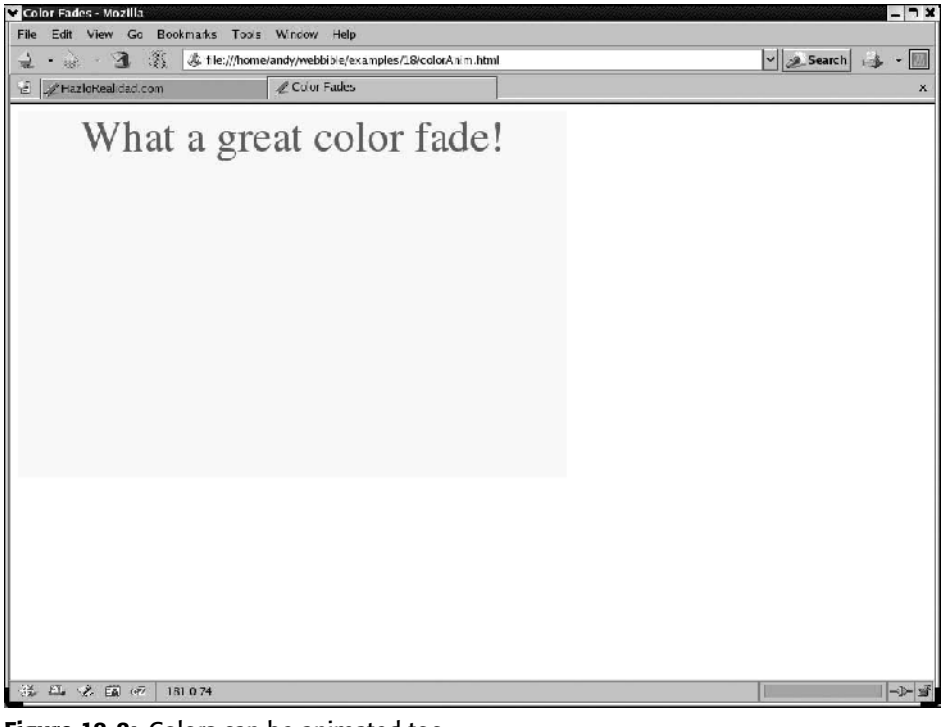


Figure 18-8: Colors can be animated too.

www.it-ebooks.info

Summary

Adding dynamic elements to your pages using either rollovers or positioning can improve your site. This chapter introduced you to the following concepts and topics:

- ✦ Rollovers are used to replace one image with another.
- ✦ Rollovers usually take place when the mouse pointer is placed over an image.
- ✦ All images should be preloaded to improve interactive response.
- ✦ Rollovers can also be used to trigger changes in other elements.
- ✦ The absolute position of a layer can be used to provide movement for all its contained elements. Almost all CSS properties can be animated such as position and color.

✦ ✦ ✦

www.it-ebooks.info

CHAPTER 19

Animating with Macromedia Flash

In This Chapter

Understanding the Flash layout

Creating objects

Modifying objects

Working with text

Working with animation timelines

Using advanced techniques

Exporting and publishing

Macromedia Flash is a vector animation application that creates sequences of action and sound, which are referred to as movie clips or movies. Flash movies can be played on Web sites. The application's strong points are that it greatly automates creating the movies and uses vector graphics that create relatively small files. If you ever created an animated GIF frame by frame using a photo-editing program, you will definitely appreciate the work you can save by using Flash.

Flash organizes space and time in a format that enables you to control the action as you work. The Stage is the space for your movie's action, and the Timeline enables you to set up the sequence for that action. Although bitmaps can be imported into Flash movies, they add to the file size and take longer to load into the user's browser than if you were to create all your animations in Flash.

Understanding the Flash Layout

Although you can create still graphics in Flash, you are most likely to use Flash to animate your Web pages. A Flash movie could consist of a menu with menu items presented so that they change in some way to create the effect of motion. For example, text can change color when the mouse is passed over it or is selected with a mouse click.

Launching Flash for the first time reveals a screen with title bars showing Movie1, Scene1, and Layer1. The Flash layout consists of the following components, which are discussed in this section:

- Stage
- Scenes
- Layers
- Timeline
- Toolbox and tool panels

www.it-ebooks.info

The Stage

When you launch Flash, you see a large, white space in the middle of your screen (see Figure 19-1)—the *Stage.* This is where you place the objects and set the action for your movie.

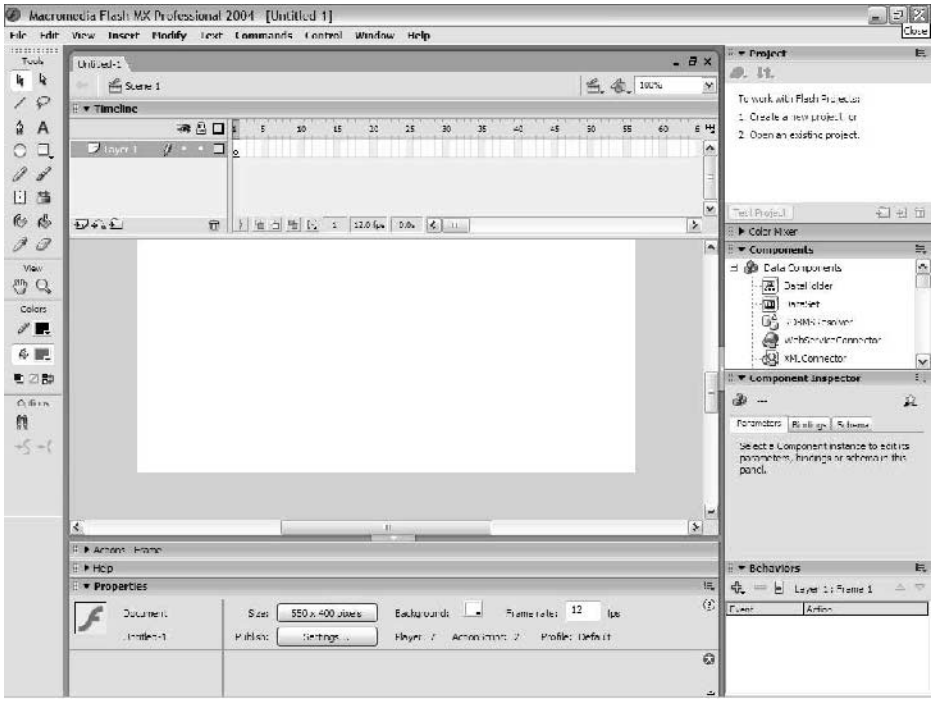


Figure 19-1: The Stage is the main part of the Flash layout.

You can control the size, shape, and color of the Stage, depending on what type of movie you are making. For example, if your movie were a Web page banner, you might want to shape it as a wide rectangle to span the top of the Web page. To modify the size and color, select Document from the Modify menu (see Figure 19-2). The window that pops up enables you to change the Stage dimensions, the movie frame rate, background color, and the measurement units (pixels, inches, and so on). In addition, you can choose to have the Stage mimic a printer page (Printer) or the actual objects (Contents) already placed on the Stage.

Scenes

A movie can have multiple *scenes.* Each scene consists of its own Stage, layers, and Timelines. You simply set the Stage for each scene, just as if you were setting the stage for a play with different scenes. When the movie is played, the scenes follow each other in the sequence they were created. You can select each scene (if there is

www.it-ebooks.info

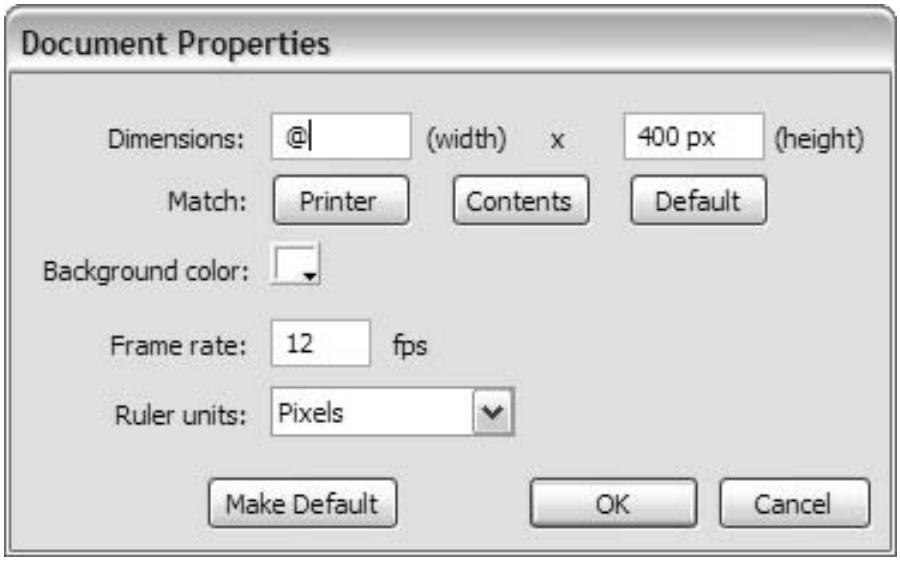


Figure 19-2: View the Document Properties window on the Modify menu.

more than one) by clicking the Edit Scene icon at the right side of the Scene title bar. If you need to rearrange the sequence, you can do so by clicking and dragging a scene up or down in the list.

Layers

Under the Scene title bar is the Layer area. Layers are important because they separate graphics within the same scene. A scene can have many layers. You should name each layer differently to indicate what it does in the scene. For example, you could name a layer *Background* if it contains the movie backdrop. You create a new layer by clicking the plus sign in the lower left-hand corner of the Layer area. The new layer appears above the original layer (see Figure 19-3), so that the oldest layer is on the bottom and the newest layer is on the top, near the Scene title bar. Double-click the layer name to rename it according to the new layer's theme.



Usually, you work from the bottom up, but you can drag layers up and down to change locations. For example, you might want to change a layer's location, if you created a foreground layer before the background layer.

The Timeline

In the right pane of the Layer area, a ruler is displayed containing numbers incremented at regular intervals. This is the *Timeline*, which represents the time span of the movie. It enables you to direct the action. Using the Timeline, you can determine the sequence of how your objects move. A Timeline can be broken into a series of points that are called frames in which some activity or change from previous activity takes place. Each layer you create has its own Timeline.

www.it-ebooks.info

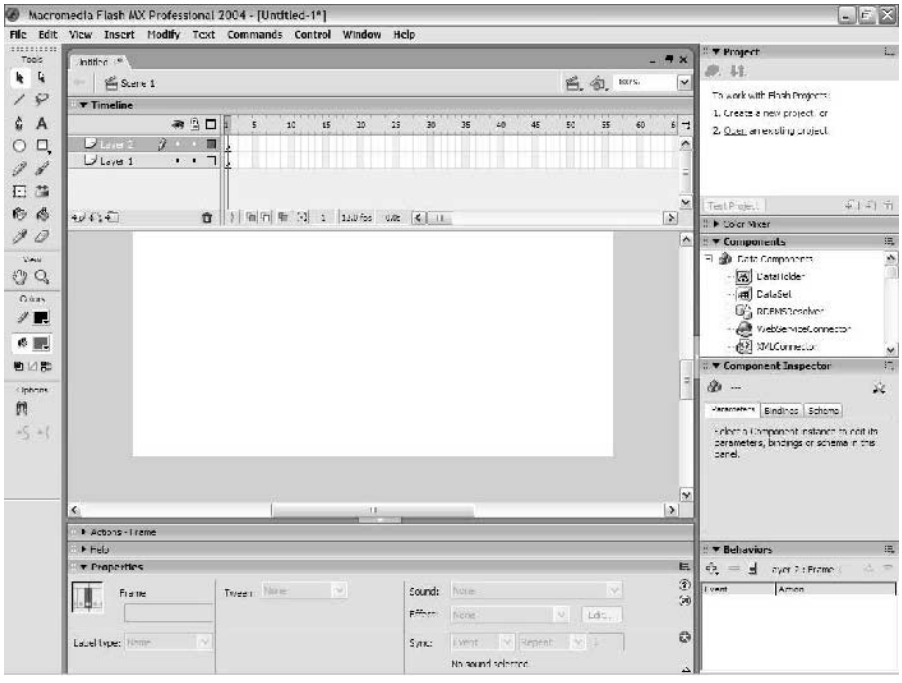


Figure 19-3: View two layers.



To add a frame to a Timeline, you place the cursor on one of the Timeline points, which resemble rectangular boxes, and right-click to select the Insert Frame option from the shortcut menu.

The toolbox

The Flash Tools panel provides the set of tools that enables you to select and manipulate objects on the Stage. The Tools panel is usually positioned on the left side of the work area. Sometimes called the Drawing toolbar, the Tools panel items enable you to draw, paint, and write. Unlike the toolbox, most of the Toolbar tools are generic and work as those in other programs.



To float the Tools panel, click and drag it with the mouse. Grab it by the gray area above the icons. Move the Tools panel to a part of the screen you find more convenient. The Tools panel is now floating (see Figure 19-4). You can replace (dock) the Tools panel by returning it to the area on the left side of the screen under the File menu.

Most of the Tools panel items have selectable modifiers to enable you to vary the action. For example, you can select the Radius modifier for the Rectangle tool to add curved corners to the rectangles you draw. You can, of course, select line or fill color for all of the tools, with the exception of the Eraser, Lasso, Arrow, and Subselect tools. Table 19-1 shows the major functions for each Tools panel tool.

www.it-ebooks.info

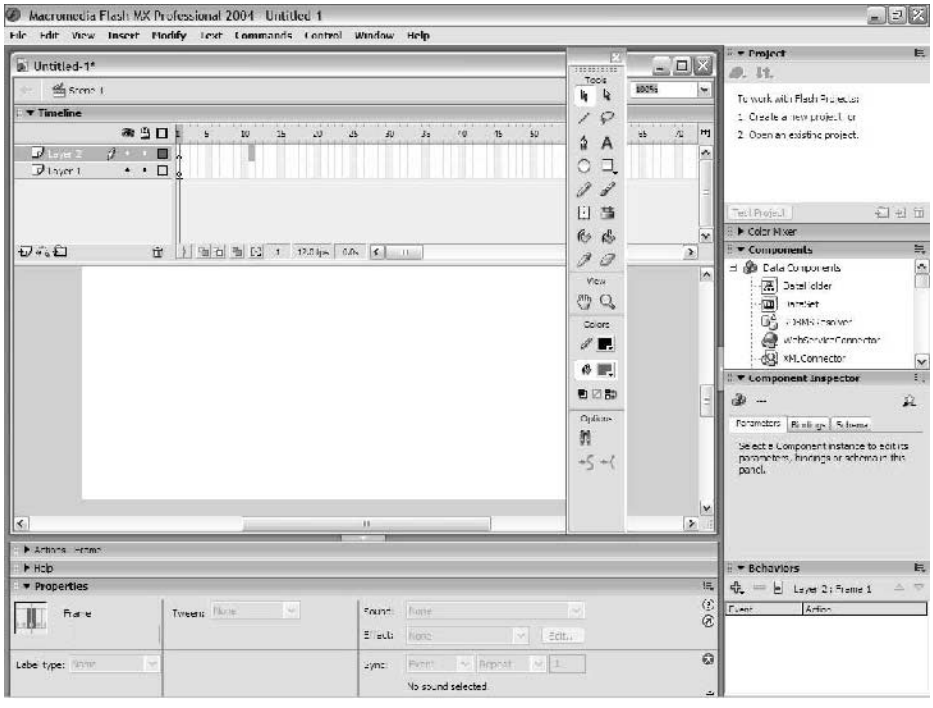


Figure 19-4: The Flash Tools panel can be floated into any position.

Table 19-1
Tools panel Tools and Functions

Tool	*Function*
Selection	Enables you to select objects.
Subselection	Creates handles, which enable you to modify the selected object.
Line	Enables you to draw lines. You can select line color.
Lasso	Creates an irregular selection area of objects, based on the line drawn. Polygon and Magic Wand modifiers change the selection area.
Pen	Enables you to draw Bezier curves with nodes.
Text	Enables you to create text. You can modify font size, color, style, line spacing, margins, and set the paragraph alignment.
Oval	Enables you to draw oval and circular shapes. You can select line and fill colors.
Rectangle	Enables you to draw rectangular and square shapes. You can select line and fill colors and change the radius.
Pencil	Enables you to draw freeform lines. You can select tool modifiers to straighten and smooth lines. You can also select line color.

Continued

www.it-ebooks.info

Table 19-1 *(continued)*

Tool	*Function*
Brush	Enables you to paint using different brush sizes and shapes. You can also select fill color as well as modifiers for Normal, Fills, Paint Behind, Paint Selection, Paint Inside modes.
Free Transform	Enables you to rotate, scale, distort, envelope, and snap to objects.
Fill Transform	Enables you to transform gradient or bitmap fills and snap to objects.
Ink Bottle	Changes color, thickness, and style of lines based on the currently selected color.
Paint Bucket	Enables you to fill objects with color. Modifiers set the gap size to be closed by the Paint Bucket.
Eyedropper	Copies line, fill, and text styles from the drawing.
Eraser	Erases lines and fills. Modifiers set the mode to Erase Normal, Fills, Lines, Selected Fills, or Inside and set the size and shape of the Eraser.
Hand	Moves the drawing's viewpoint.
Zoom	Enlarges or shrinks the drawing's view.

Tool panels

Panels are windows containing various tool options and can be accessed from the Window menu. Each of the tools in the Tools panel has a set of options, which are displayed immediately under the toolbox and are incorporated in the Tool panels. You also can select the Default Layout from the Panel Sets item on the Window menu (see Figure 19-5). Once you select the default layout, you can reposition the panels to suit your preference and save the layout with a name. Then you can recall it to view your preferred panel layout.



Although the top of the Tools panel remains the same, the options at the bottom of the Tools panel change, depending on the currently selected tool.

Flash classifies most of its panels into three categories: Design Panels, Development Panels, and Other Panels, which can be accessed through the Windows menu. Tables 19-2 through 19-4 list the various panels by category.

Creating Objects

An *object* is anything you create or import into Flash. When you create an object on the Stage, you can direct how it appears and functions. You can change its color, size, and shape as well as specify which frames it appears in and what it does.

www.it-ebooks.info

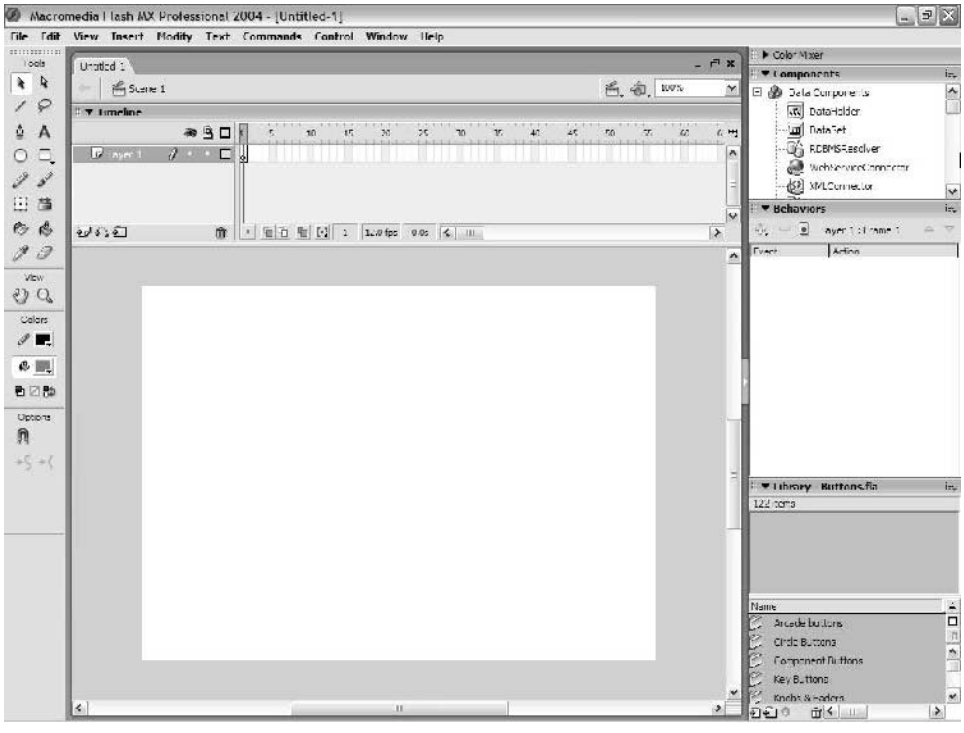


Figure 19-5: The Default Panel layout can be customized.

Table 19-2
Design Panels

Panel	***Function***
Align	Enables you to change the alignment of elements with respect to one another and to the stage.
Color Mixer	Enables you to mix the colors Red, Green, Blue, and Alpha transparency by percentage values. Also displays Line and Fill palettes, Black and White, No color, and Swap color options.
Color Swatches	Displays a palette of colors and gradients.
Info	Shows the coordinates X and Y, Width and Height dimensions, and Red, Green, Blue, and Alpha (transparency) values.
Scene	Enables you to duplicate, add, or delete scenes.
Transform	Enables you to resize, rotate, or skew objects.

www.it-ebooks.info

Table 19-3
Development Panels

Panel	*Function*
Actions	Enables you to add scripted actions to your files.
Behaviors	Enables you to add scripted behaviors to your files.
Components	Provides a centralized source for dragging common components such as buttons onto the stage.
Component inspector	Provides information on the selected component.
Debugger	Traces problems in your `.swf` files.
Output	Keeps track of variable values and such as your file runs.
Web Services	Lists Web service URLs.

Table 19-4
Other Panels

Panel	*Function*
Accessibility	Sets name, description, shortcut and tab index for objects.
History	Keeps track of all changes.
Movie Explorer	Displays all current elements.
Strings	Provides an avenue for language translation.
Common Libraries	Enables you to add objects from either the default libraries or your own custom ones.

Drawing lines with the Line tools

Use the Line tool when you want to draw completely straight lines (see Figure 19-6). The Pencil tool is for freehand drawing; although you can use the tool modifiers to smooth the lines it draws. Select the Line tool and draw a line on the Stage by clicking at one point and dragging the mouse to another point. Then release the button, and a line forms between the two points.

Setting stroke characteristics

Select the Line tool by clicking it with the Arrow selection tool. Click the line color palette in the Colors area in the Tools panel and select the color red from the palette. Use the mouse to draw a line on the Stage. You can thicken the line using the Property inspector. Use the slider on the arrowed list box to increase the line thickness or enter the desired number in the box.

www.it-ebooks.info

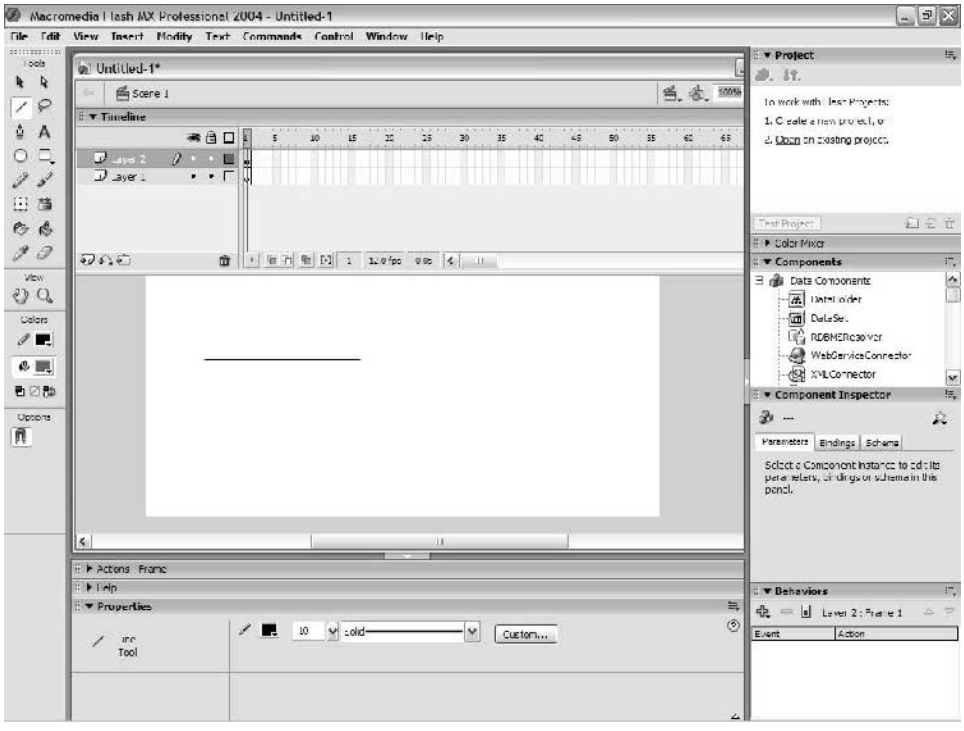


Figure 19-6: You can draw lines with the Line tool.

You can use the modifier tools to change the way the Pencil tool draws lines. Select the Pencil tool. Notice that a stair step icon opens below the Options area at the bottom of the Tools panel and the Colors box. Select the icon and then select the Smooth tool modifier. Now, draw a heart shape on the Stage (see Figure 19-7). Do not worry about drawing a perfect heart. The Smooth modifier makes the curves smoother than you draw them. Release the button when you complete the heart shape. The shape probably looks much better than it was drawn. You can experiment with the Straighten and Ink modifiers. They greatly affect the way shapes look when drawn with the Pencil tool.

You can use the Ink Bottle tool to change the characteristics of lines after they have been drawn. Select the Ink Bottle tool. Select the dotted line from the list box on the Stroke panel. Change the line width to 8 points. Now use the Ink Bottle tool to click anywhere on the outline of the heart shape you just drew. Presto! The line changes style and thickness.

Selecting and deleting objects

By now, your drawing area is somewhat cluttered. You can remove unwanted artwork several different ways. You can select a line by double-clicking it and pressing the Delete key. When an object is selected, it appears crosshatched or grayed out. Alternatively, you can use the Selection tool to draw a selection box

www.it-ebooks.info

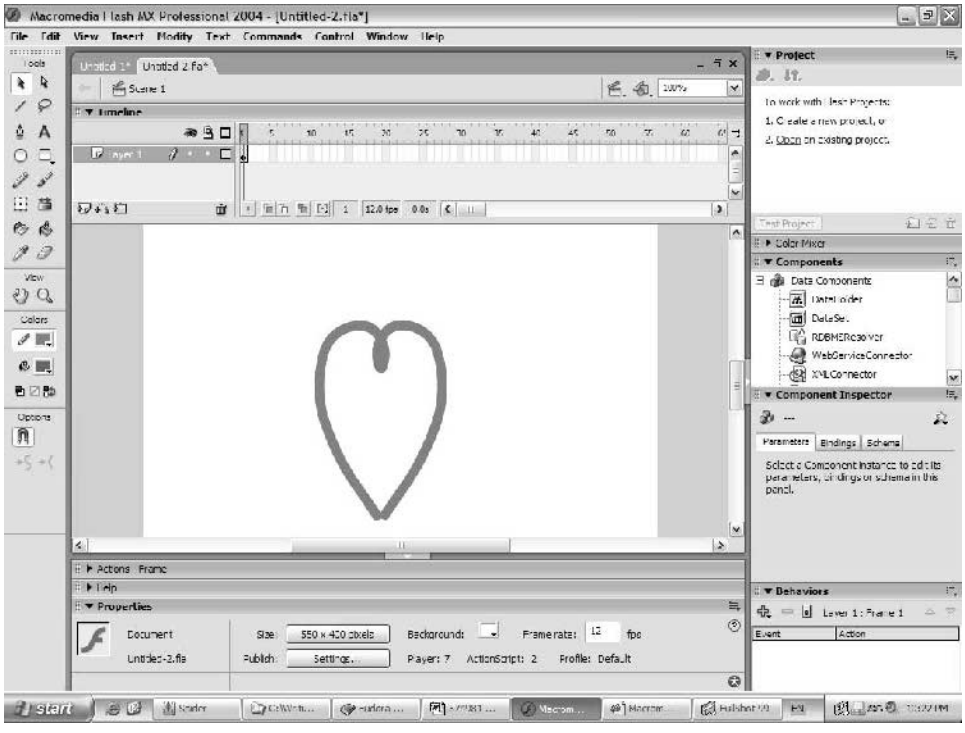


Figure 19-7: Draw a heart with the Pencil tool.

around the object you want to delete and then use the Delete key to erase it. You also can use the Eraser tool to erase unwanted objects from the Stage. Yet another way to remove objects is to use Edit ➪ Select All, then select Edit ➪ Clear to clear the entire Stage. Use your preferred method to remove the lines and heart.

Making shapes with tools

You can make different shapes just as easily as you can draw lines on the Stage. Select the Oval tool and draw a circle on the Stage. You will notice that the circle has the same line thickness, style, and color as the heart. Change the line style by selecting the wavy pattern from the Style list box in the Property inspector (see Figure 19-8).

Select the Rectangle tool and draw a square or rectangular shape. This time, change the fill color rather than the line color. Select a fill color that contrasts with the line color. Select the Paint Bucket tool and click inside the rectangle. The color fills the inside perfectly. Now use the Selection tool to select the fill by clicking only on the fill (don't double-click or click the line). Press the Delete key to remove the color.

Select the Brush tool and paint inside the rectangle. If you aren't careful, the fill color will "splash" outside the lines. You can control this by using the Paint modifier. With the Brush tool selected, click the Brush Mode icon in the top of the Options area. Now select Paint Inside from the drop-down menu. Paint the rectangle. This time, the

www.it-ebooks.info

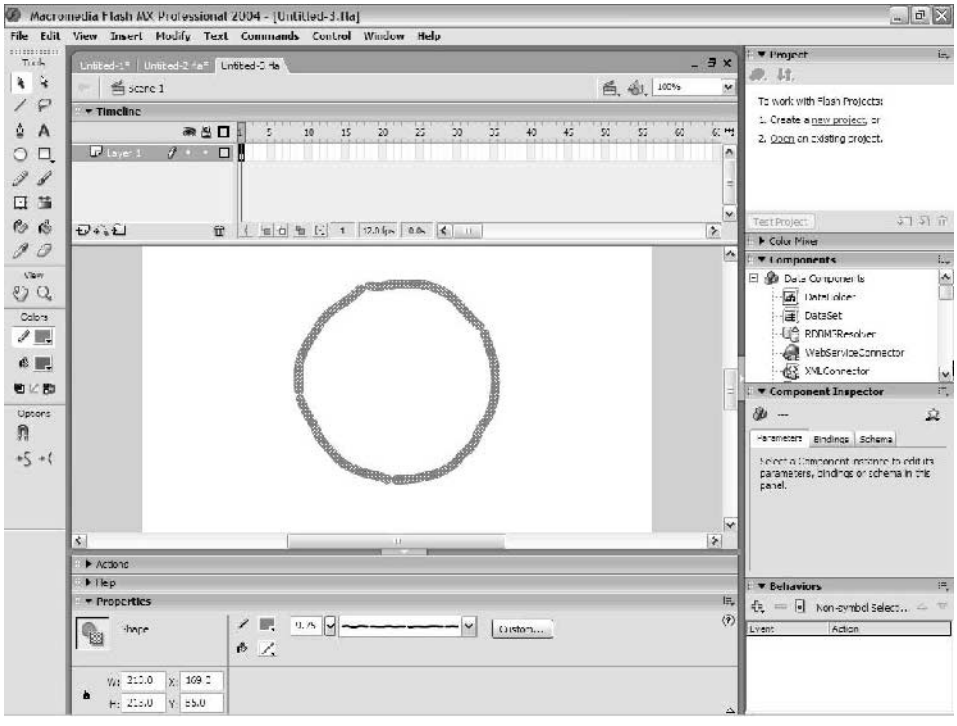


Figure 19-8: Change the line style.

paint stays inside the lines. Of course, you need to make several strokes to completely fill the rectangle. You can make it easier by selecting a larger brush from the Brush Size modifier in the Options area.

Saving your work

If you create any movie or part of a movie that you may want to build upon or reuse, you should save it. Remember all of the steps you went through to create even a simple shape with a gradient fill. To save your work, select File ➪ Save. Your file is saved in the default location as Movie1, Movie2, or whatever the current movie name is. If you did not rename your movie, it is named Movie with a number appended. If you want to give your movie a specific name, use File ➪ Save As and type the name you want in the File name box. The Save box shows the current default folder where the file is located. You can click the icon showing a folder with an up arrow to move to a parent folder. Save your current file as test. Flash appends the `.fla` extension to the file name, signifying the Flash Movie file type.

Importing art

Importing art created in other applications is as easy as selecting File ➪ Import. Most image types can be imported into Flash. Flash supports the familiar TIFF (`.tif`), JPEG (`.jpg`), GIF (`.gif`), Windows Bitmap (`.bmp`), Adobe Illustrator (`.ai`),

www.it-ebooks.info

Enhanced Metafile (.emf), Windows Metafile (.wmf), PhotoShop (.psd), AutoCAD (.dxf), and Macintosh PICT (.pct) formats. In addition, Flash supports several animated file types, including QuickTime Movie (.mov), Animated GIF (.gif), and Flash Player (.swf). Flash also imports sound formats such as Windows Wave (.wav), MP3 Sound (.mp3), and AIFF (.aif).

You can use imported bitmaps as patterned fill for shapes you draw. Other uses for imported art include animating images by changing the size, shape, and orientation to simulate animation. You can use this technique with still photos to create the illusion of movement. Also, bitmaps and vector art often make good backgrounds for movies.

Imported files, particularly some movie types and PostScript files, are amazingly large. They can easily climb into the double-digit megabyte range. Bitmaps also increase the file size of the authoring file (.fla) and the compiled Flash Player version (.swf). The file size and the browser download time are important to keep in mind when creating Flash movies because large files force your visitors to wait forever until they see the movie. Obviously, waiting for many minutes to see your movie is not going to create a favorable disposition toward your Web site.

Using the Library panel

When you import images, they appear in the Library panel. You can access this panel by selecting Window ➪ Library. The Library panel displays the names of all imported objects. Clicking a name in the list displays the object in the Library panel. Items listed in the Library panel can be reused in other scenes and even other movies.

Modifying Objects

You have seen how easily you can create objects on the Stage. You have also experimented a little with modifying them. But you are not limited to the oval and rectangular shapes in Flash. You can easily change the nature of the shape by using a click-and-drag routine.

Stretching and distorting objects

Clear the Stage and select the Oval tool to draw a new oval. Make sure that the object's line or fill is not selected. Choose the Selection tool and bring it very close to the edge of the oval shape. Be careful not to select it. When you see a tiny arc symbol form below the pointer, you can click and drag to pull the shape outward or inward (see Figure 19-9). Practice pushing and pulling the edge of the shape until it becomes distorted and the shape changes. If the oval object moves its position instead of changing shape, you have accidentally selected it. Just select Undo from the Edit menu to return it to its original position.



When the Selection tool is touching a line, a right-angle symbol (it looks as if it's a corner) or an arc symbol appears below the cursor. The angle means that if you drag the line at that point, it pulls into a straight angle, and the arc signifies that any manipulating gives the line a curve.

www.it-ebooks.info

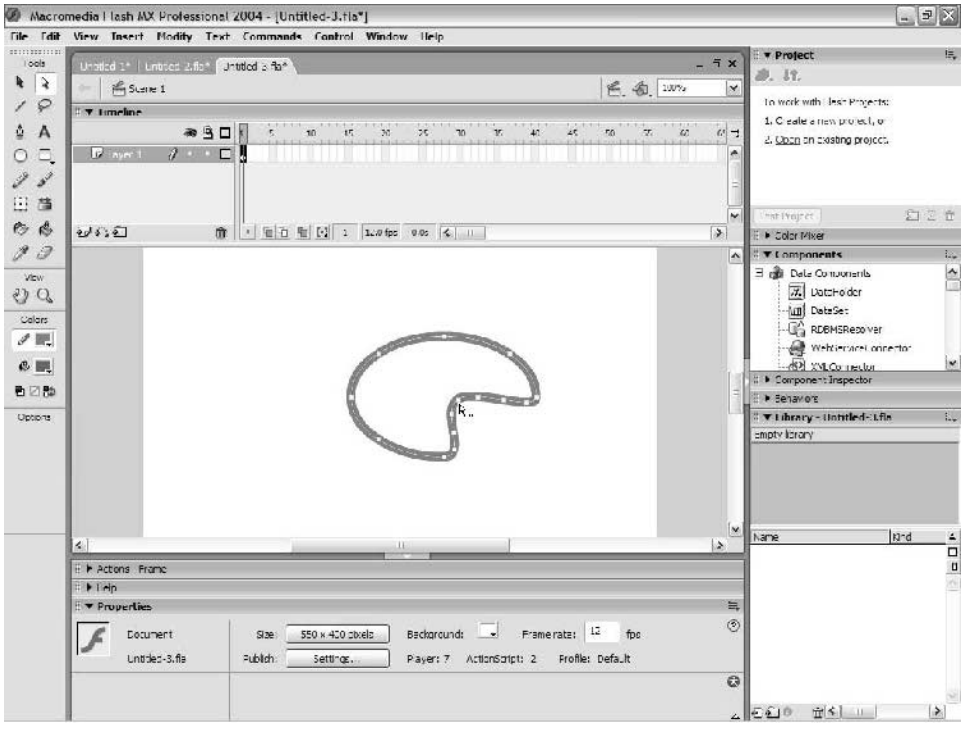


Figure 19-9: Change the oval object's shape.

Flash has a feature that becomes a little tricky when dealing with multiple objects. This is especially true when it catches you by surprise.

With your distorted oval object on the Stage, select the Oval tool again and draw a small oval shape partly on top of the original object; make sure that the new object must overlap the existing one. Now use the Selection tool to select the new oval object. Double-click to verify that the fill and outline are both selected. When selected, click and drag the shape away from the original object. A hole is now revealed in the first oval shape (see Figure 19-10).

When objects come in contact with one another on the same layer, they connect, causing the object on top to displace the covered portion of the bottom object. When you move the top object, the corresponding portion of the bottom object disappears as if it had been displaced by the connection.

Grouping objects

To learn about this feature, clear the Stage. Select the Oval tool and draw a new oval shape. Select Modify ➪ Group to create a group. Now select the Oval tool again and select a new, contrasting fill color from the Colors area of the Tools panel. Draw another oval shape overlapping the first one. Now, select the overlapping oval object and move it away from the original one. This time, the object remains unchanged.

www.it-ebooks.info

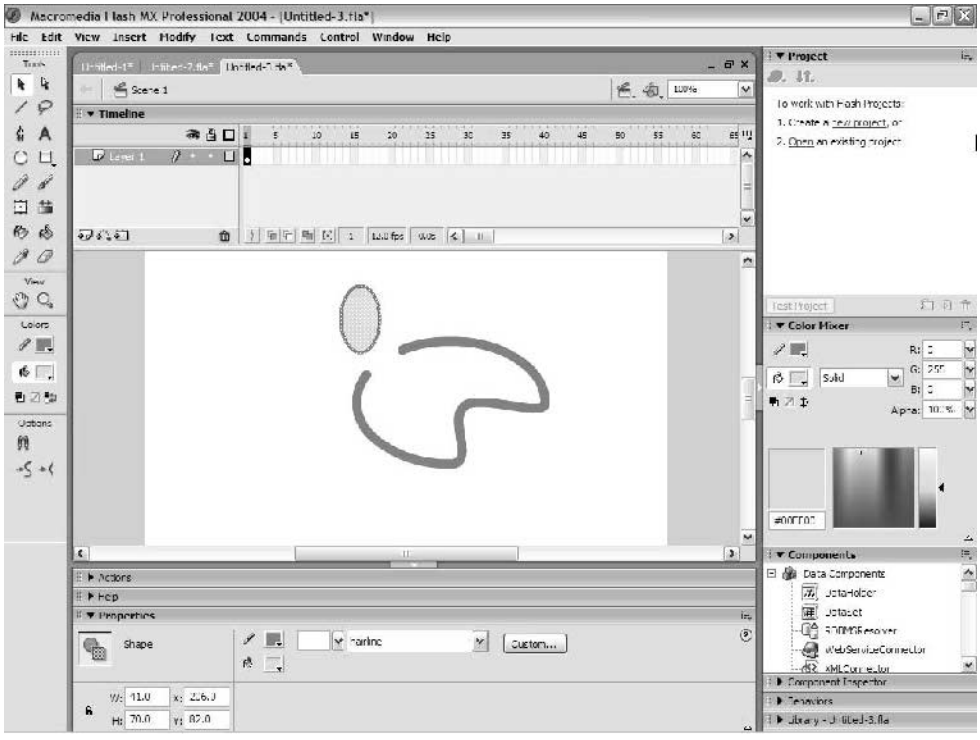


Figure 19-10: Change the old shape by moving the new one.



When grouping objects, those most recently grouped appear to be on top of the other objects.

Scaling

You can scale objects using the Free Transform tool. Select the Oval tool and draw a new oval shape. Select the Paint Bucket tool and select any color you like from the Fill Colors box. Click inside the circle to color it. Choose the Selection tool and double-click the oval object to select both the line and fill.



When both line and fill are selected, the fill appears in a crosshatched pattern and the line becomes darker and thicker.

Select the Scale modifier in the Options area. It displays an icon with two squares, large and small. Small square tabs form around the oval object. Grab one of the corner tabs and drag toward the center of the object to make it smaller (see Figure 19-11).

You also can distort the object into an egg or oval shape by grabbing one of the middle tabs and dragging it toward the center.

www.it-ebooks.info

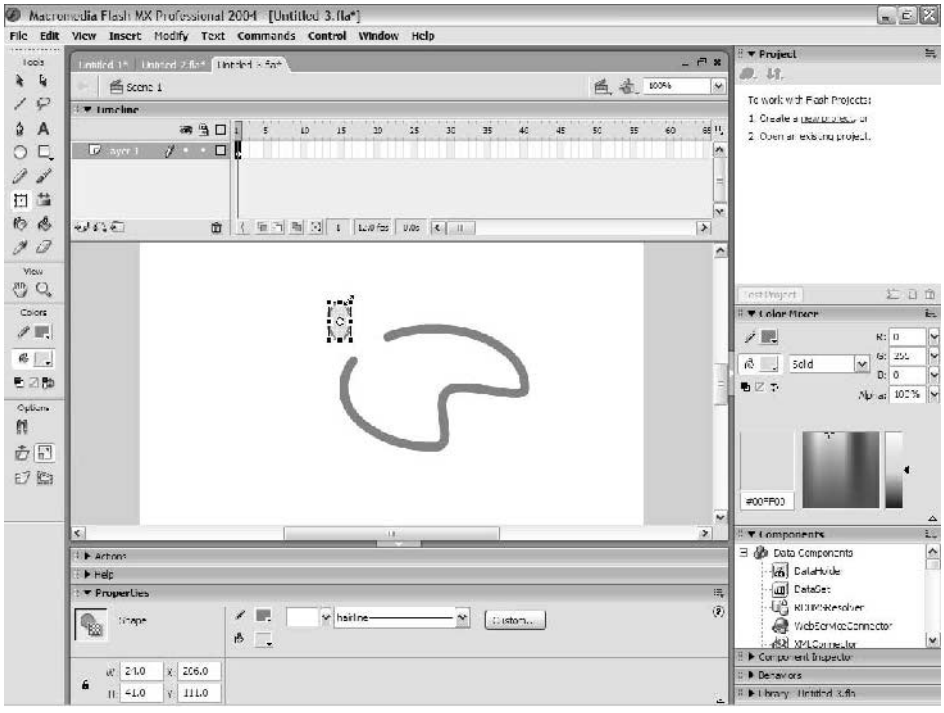


Figure 19-11: Scale the oval.

Rotating and skewing

Rotating an object is similar to scaling it. Select the Free Transform tool and select the Rotate and Skew modifier in the Options area. Grab one of the corner tabs and drag in either clockwise or counterclockwise direction. The oval should now be upended (see Figure 19-12).

Straightening and smoothing

With the Selection tool, you can use the Straighten and Smooth modifiers in the Options area to modify shapes. Clear the Stage and select the Pencil tool. Draw a wavy line across the Stage. Choose the selection tool and select the line you drew. Select the Straighten modifier in the Options area. Select the line and see if it changes. Now, select the Smooth modifier. Select the wavy line. The amount of difference you see in the shape of the line depends on the nature of the line drawn. Depending on how the line was drawn, the Smooth or Straighten modifier changes it more or less. As you can see, learning your way around Flash depends on some experimentation. It is a good idea to experiment a lot before beginning serious projects.

Using gradient fills

So far you have used solid colors for stroke and fill in the objects you created on the Stage, but there is a lot more you can do with color. You can select one of the

www.it-ebooks.info

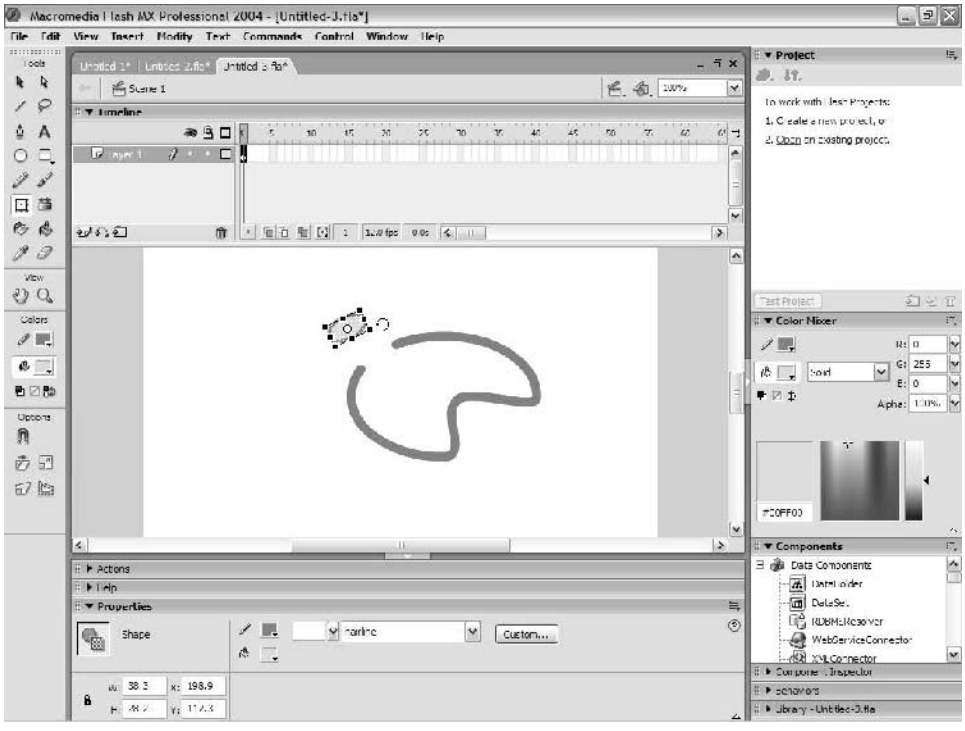


Figure 19-12: Rotate the oval.

premixed gradients by clicking the Fill color button (shown with the Paint Bucket icon) in the Colors area. You also can select Window ➪ Design Panels ➪ Color Mixer to choose a fill type (solid, linear, or radial) and a fill color. At the bottom of the Fill color palette contains a selection of linear and radial gradient mixes.



A linear gradient shades in a straight line from the beginning to the end of the object being filled. A radial gradient shades in a circular pattern.

Here's how to add a gradient:

1. Clear the Stage and select the Rectangle tool to draw a square or rectangle.
2. If the Color Mixer panel is not showing, select Window ➪ Design Panels ➪ Color Mixer.
3. Select Radial gradient from the drop-down list box in the Color Mixer.
4. Select the right-most pointer under the mixer bar, representing the outside of the rectangle. Choose a color by clicking the Color area, setting the basis for the gradient. Now click the pointer on the far left. Select a different color from the palette. Move the pointers displayed below the slider bar to change

www.it-ebooks.info

the color mix until you are satisfied with the gradient sample shown in the Gradient Preview box (see Figure 19-13).

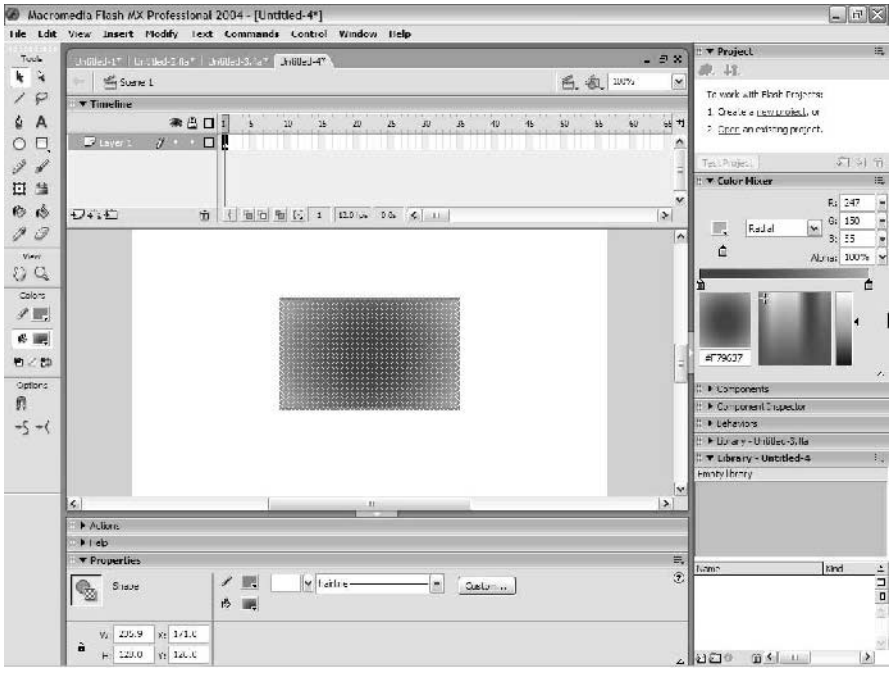


Figure 19-13: Fill the rectangle with a gradient.

Working with bitmap fills

Not only can you fill objects with gradients and solid colors, you also can use bitmaps as fills. You can select a bitmap in the drop-down box on the Fill panel, where you just selected Radial Gradient. First, select the Rectangle tool and draw a rectangle on the Stage.

Here's how to add a bitmap fill:

1. Clear the Stage and select the Rectangle tool to draw a square or rectangle.
2. If the Color Mixer panel is not visible, select Window ➪ Design Panels ➪ Color Mixer.
3. Select Bitmap gradient from the drop-down list box in the Color Mixer.
4. In the Import to Library dialog box, navigate to the bitmap file you want to use, select it, and then click on the Open button to display the imported bitmap.
5. Select the Paint Bucket tool and click inside the rectangle. It will be tiled with the bitmap (see Figure 19-14).

www.it-ebooks.info

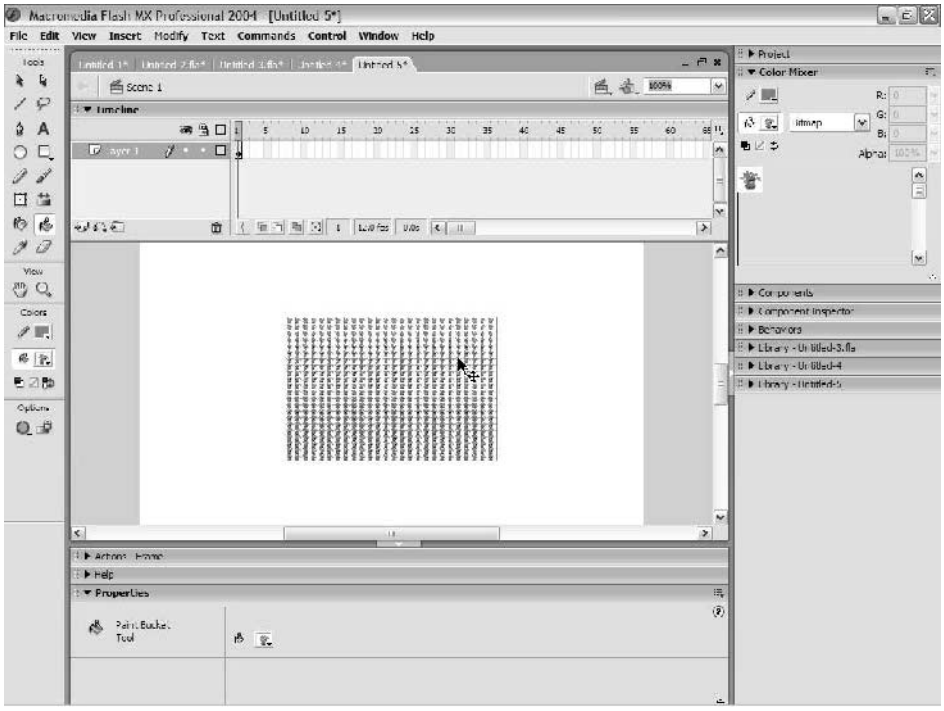


Figure 19-14: Fill an object with a bitmap pattern.

Working with Text

It is easy to add text to your movies. You will find that you can transform text as simply as you can alter lines and shapes. Text is a great way to create stunning effects without creating a movie with a large overhead.

Excellent examples of the uses of text in Flash movies can be found on the Internet. Some examples combine text and sound with no other graphics. Together, these create great effects.

Adding text

You add text using the Text tool on the Tools panel. Buttons and list boxes in the Property inspector enable you to select a font, a font size, tracking, kerning, and normal, superscript, and subscript, as well as the bold and italic styles and font color.

Choosing fonts

When you select a font and publish it with your movie, it will be exported with the Flash Player (`.swf`) file, so that font does not need to be installed on the user's system. Device fonts shown at the top of the fonts list (_sans, _serif, _typewriter_)

www.it-ebooks.info

are not embedded in the exported Flash movie. Using device fonts causes the movie to use the font on the user's system that most closely resembles the selected device font. The purpose of using device fonts is to create a movie with a smaller file size because the font is not embedded in the `.swf` file.

You select a font from the list of fonts you have installed on your system. Sometimes the fonts are installed automatically when you install a new application, so you may not have used many of the fonts available to you. Windows and Mac users usually have many True Type or PostScript fonts from which to choose.

Fonts are available from the slider in sizes from 8 points to 96 points. But size is relative to the size of the Stage in your movie. If you really need bigger fonts, you can type numbers ranging from 0 to 2500 into the Font Size list box instead of using the slider.

Setting font characteristics

Select a font. Enter the point size as 48 in the Property inspector, or click the down arrow to the right of the Font Size list box and use the slider to select a 48-point size. Select a bright color and begin typing on the Stage. You can see that 48 points doesn't look very large when displayed on the default movie Stage (see Figure 19-15).

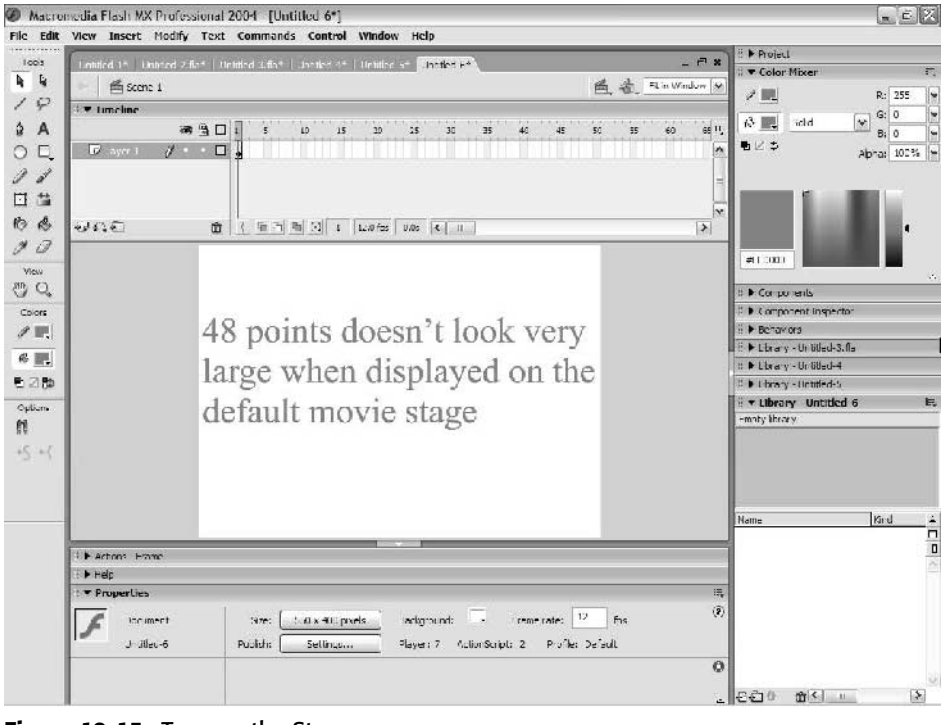


Figure 19-15: Type on the Stage.

In addition to font size and color, you can change the character spacing. An increase in the character spacing increases the space between letters. Use the Selection tool

www.it-ebooks.info

to select the text block just created. Select the down arrow on the Character Spacing list box and use the slider to increase and decrease the value. Figure 19-16 shows the space between the letters when the character spacing value is increased to 18.

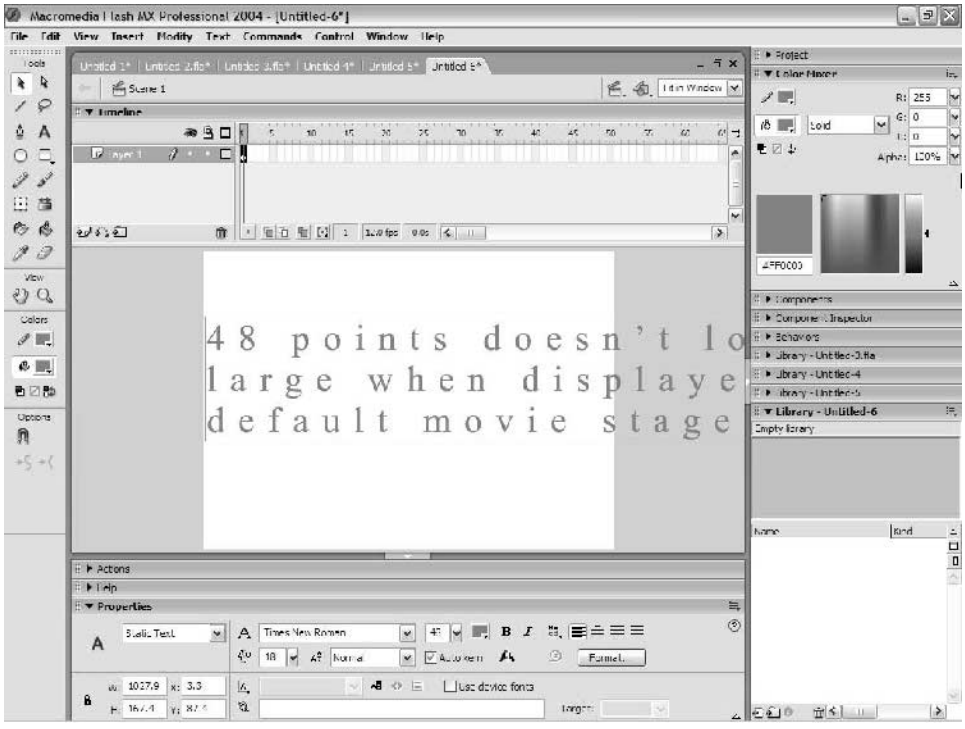


Figure 19-16: Adjust tracking to increase the spacing between characters.

Kerning controls the space between certain letters. In kerned text, differing amounts of space exist between letters, depending on the space the letter takes up. For example, in the preceding sentence, the space used by the letters *t* and *a* in the word *takes* is less than that used by the letters *k* and *e*. For the most part, kerning is not a major concern in Flash movies.

You also can use the selection tool and the Scale modifier to stretch the text block. With the text block selected, use the Selection tool and the Scale modifier and drag one of the tabs formed on corners of the text block to enlarge or decrease the size. Drag a middle tab to scrunch or elongate the text. Dragging the middle tabs distorts the text, whereas dragging a corner tab maintains the aspect ratio while adjusting the size.

Creating text effects

You can create some great effects using the Flash tools, including making pictures out of text by following these instructions:

1. Delete your text to clear the Stage.
2. Select the Text tool, click on the Stage, and type **I LOVE** on one line.

www.it-ebooks.info

3. Click the mouse button on the Stage immediately below that text block to begin a new text block and then type **BOOKS**.
4. Select the I LOVE text block and select Modify ➪ Break Apart to break apart the text block (see Figure 19-17). This is necessary so that you can manipulate individual letters. Now you can change the shapes of each letter using the various transformation tools.

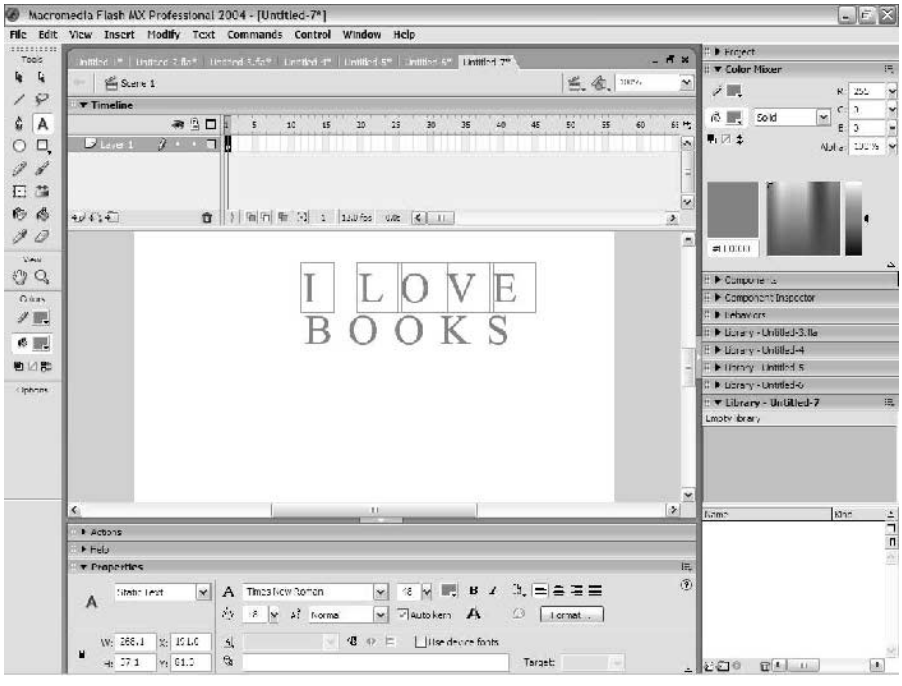


Figure 19-17: Breaking apart the text block yields five text blocks instead of one.

Working with Animation Timelines

The Timeline is the basis for your Flash animation. The area above the Stage and immediately below the Scene title bar shows a ruler containing numbers incremented at regular intervals. The Timeline represents the time span of the movie. By using the Timeline, you can determine the sequence of your objects' movements. You can animate the objects on the Stage by selecting different frames on the Timeline and changing the placement of objects.

Using frames

Each box directly below the numbers on the Timeline is a frame. A frame is a unit of time on the Timeline. For animation to appear, each successive frame must differ from the preceding one. If you ever created your own animated GIF movie frame by frame, you understand the amount of work involved. This is termed frame-by-frame animation, and it requires that you create a series of still images, each slightly

www.it-ebooks.info

different from the others, so that when these are viewed in succession, they appear to move. In fact, a motion picture is really a lot of frames (still photos) displayed in rapid succession. Using the Timeline you can automate this process so that you do not need to create a specific action for each individual frame.

Assigning keyframes

Keyframes are those frames you use to identify events. Each keyframe marks a change in objects on the Stage. You can simplify animation by placing keyframes at strategic points along the Timeline. To create a keyframe, right-click a frame somewhere on the Timeline, and select Insert ➪ Keyframe from the shortcut menu. A black dot appears in that frame to signify it is a keyframe.

Adding layers

Each layer has its own Timeline. Layers are important because they enable you to separate objects in the same scene. When the objects are on different layers, you do not have to worry about object interaction as you did in the shape exercise earlier in this chapter. That is, when objects are on different layers, they cannot affect one another. You add new layers by clicking the plus symbol at the bottom of the Layers area, below the Scene title bar.

Clear the Stage to begin a new movie. Double-click the layer name (Layer 1) and rename it to **Background**. Naming the layers according to their functions is helpful. Click the plus sign in the lower-left corner of the layer area to add a new layer. Double-click the new layer and rename it **Action**. You now have two layers and two Timelines (see Figure 19-18).

Adding objects to the layers

To add content to your document, you need to add objects to the Stage. It is important to identify the layer on which you are adding the objects. The active layer is the one you currently have selected. Select the Action layer and draw an oval shape near the top of the Stage. Use the Paint Bucket tool to select a light green fill. Click inside the oval to color it green (see Figure 19-19).

This is a good time to save your movie. Select File ➪ Save As and name the movie **Test2**. Type the name in the File name box and check the Save box to ensure that you are saving it in the correct folder. If you want to change the folder, click the down arrow next to the Save in box to show all the folders on your system. You can move up one level by clicking the folder icon with the up arrow symbol.

Converting objects to symbols

Symbols can be reused in different scenes and also in different movies. When your project requires a number of similar objects, one symbol can be slightly changed several times to create different instances of the symbol. Each copy of a symbol is referred to as an *instance*.

www.it-ebooks.info

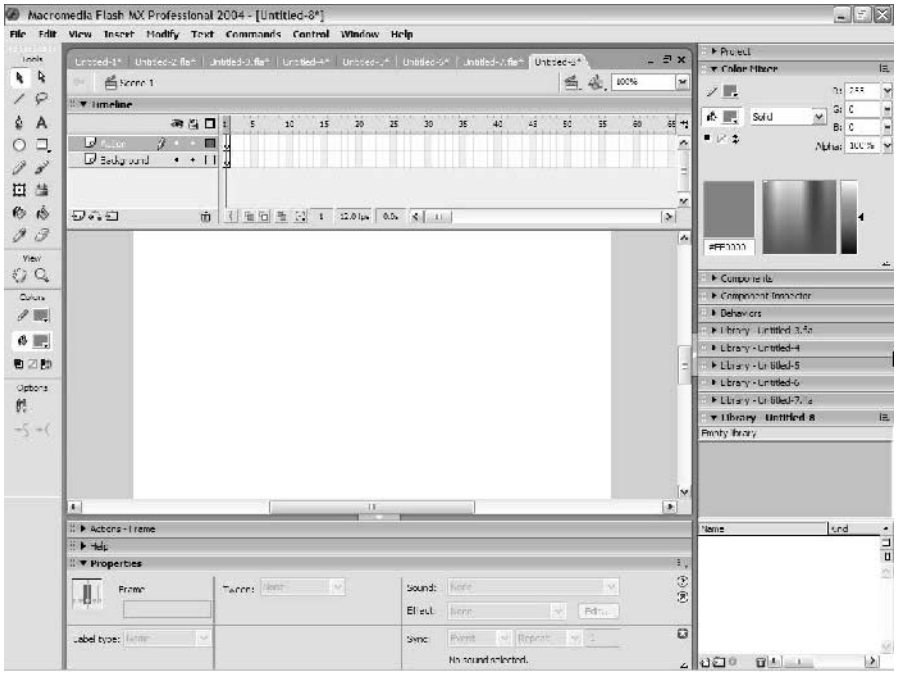


Figure 19-18: Add and name a new layer.

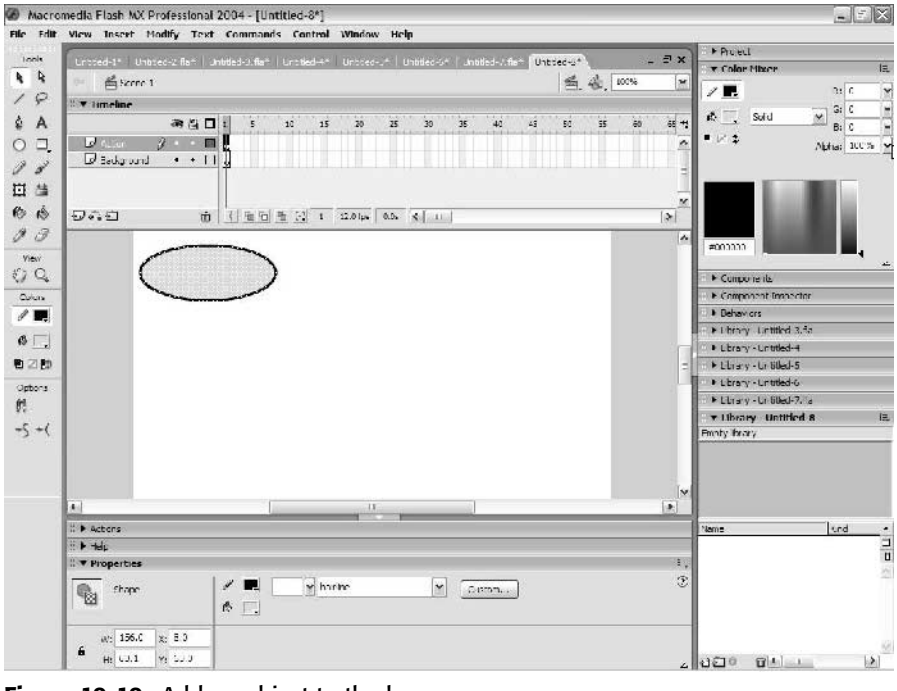


Figure 19-19: Add an object to the layer.

www.it-ebooks.info

In addition to saving time, converting an object to a symbol prevents accidental deletion or distortion. Once converted, symbols also appear in the Library panel and in the Library list. You can click and drag symbols from the Library panel onto the Stage as needed. Actions carried out on a symbol also affect all instances of a symbol.

Select the oval shape you just drew and select Insert ➪ New Symbol. When the Create New Symbol window pops up, rename the symbol to **GreenOval** and click OK. Select Window ➪ Library to verify that the symbol appears on the Library list.

Creating animation

Although you can change objects that have not been converted to symbols, you will find symbols in frames on a Timeline much easier to manage than simple objects. The first keyframe on a Timeline is always created for you when the layer is added. To make the green oval object appear to move and at the same time change color, follow these steps:

1. Right-click the 15th frame on the Action layer Timeline and select Insert ➪ Keyframe from the shortcut menu.



Be sure that you have the correct layer selected before changing the action or attributes of objects or symbols. It is very easy to select the wrong layer, so check to see that the desired layer is highlighted.

2. Drag the oval downward to the center of the Stage.
3. Select a bright pink color in the Property inspector.
4. Play the movie by selecting Control ➪ Play. Make sure the first frame is selected, or you will need to select Control ➪ Rewind then Control ➪ Play. When the movie plays, the oval stays put until the 15th frame, in which it jumps to the new position and changes color (see Figure 19-20).

Now, this may not seem astonishing, but it is animation. If you want to change the action frame by frame, you could just move the oval in short increments by adding keyframes to every frame between 1 and 15. However, there is a much better and easier solution.

Tweening action

The solution to automating animation is called *tweening*. You can have your oval float gracefully down to center stage and change color gradually by tweening the action:

1. Select the first frame on the Action layer Timeline.
2. Choose Insert ➪ Timeline ➪ Create Motion Tween.

   An arrow forms on the Timeline between the first and the 15th keyframe. This signifies motion. In this case, the symbol is moving through space and changing color simultaneously. Both events involve motion tweening.

www.it-ebooks.info

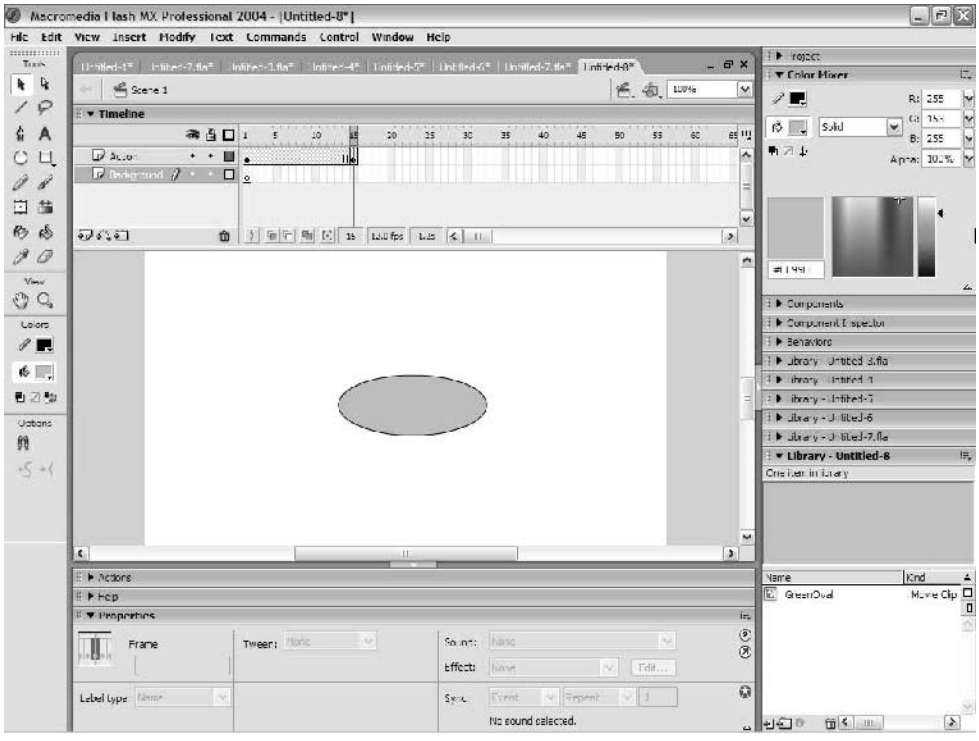


Figure 19-20: Add a new keyframe to the Action layer Timeline.

3. Play the movie by selecting Control ➪ Play. The action should be significantly better than the abrupt change experienced before the action was tweened.
4. You needn't stop with one tweened action. You can extend the action by adding a keyframe to the 40th frame. Right-click the 40th frame of the Action layer and select Insert ➪ Keyframe from the shortcut menu.
5. With the 40th frame selected, drag the oval to the far right of the Stage.
6. Choose a bright blue from the Property inspector.
7. With the 15th frame selected, choose Insert ➪ Timeline ➪ Create Motion Tween.
8. With the first frame selected, use Control ➪ Play to view the movie. The movie should be decidedly better than the last change you made. Now the movie changes colors and moves in different directions and for a longer time span (see Figure 19-21).

Creating more layer effects

Now that you have something going on for the Action layer, you can turn your attention to the Background layer. The movement and color shift of the oval can be more interesting if you create an interesting background.

www.it-ebooks.info

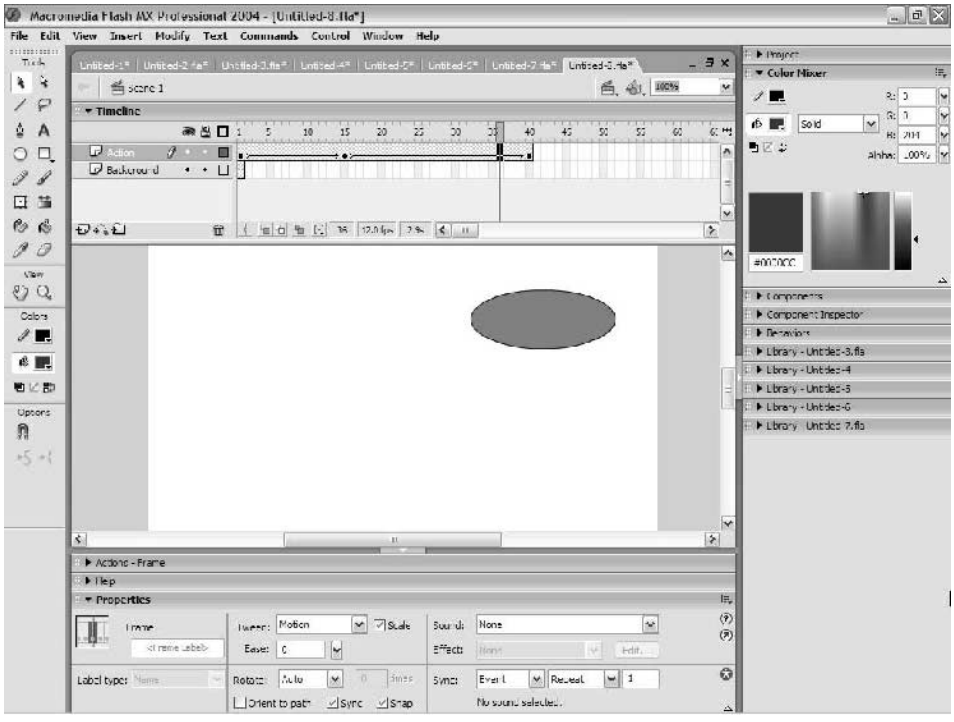


Figure 19-21: Play the tweened movie.

To make a simple background, select View ➪ Zoom Out to view the entire Stage. Select the first frame on the Background layer. Then select the Rectangle tool to draw a rectangle around the area of the Stage where the action takes place. You can then select a fill color and use the Paint Bucket to fill the rectangle with a color or gradient. Add a keyframe to the 40th frame directly below the keyframe for the Action layer. Play the movie, using Control ➪ Play. The oval show now floats over a backdrop (see Figure 19-22).



To prevent making unwanted changes to layers, you can lock layers you are not working on. Click the dot in the column headed by a padlock symbol on the layer you want to lock.

You can add more features to the background if you want. Click the dot in the column headed by the eye symbol on the Action layer. This removes the oval object from view, which is handy when you want to remove the distraction of objects that you are not currently working on.

1. Select View ➪ Zoom In to zoom back into the Stage.
2. With the first frame in the Background layer selected, draw several shapes around the Stage.
3. Play the movie. Now, the oval action plays on a more interesting backdrop (see Figure 19-23).

www.it-ebooks.info

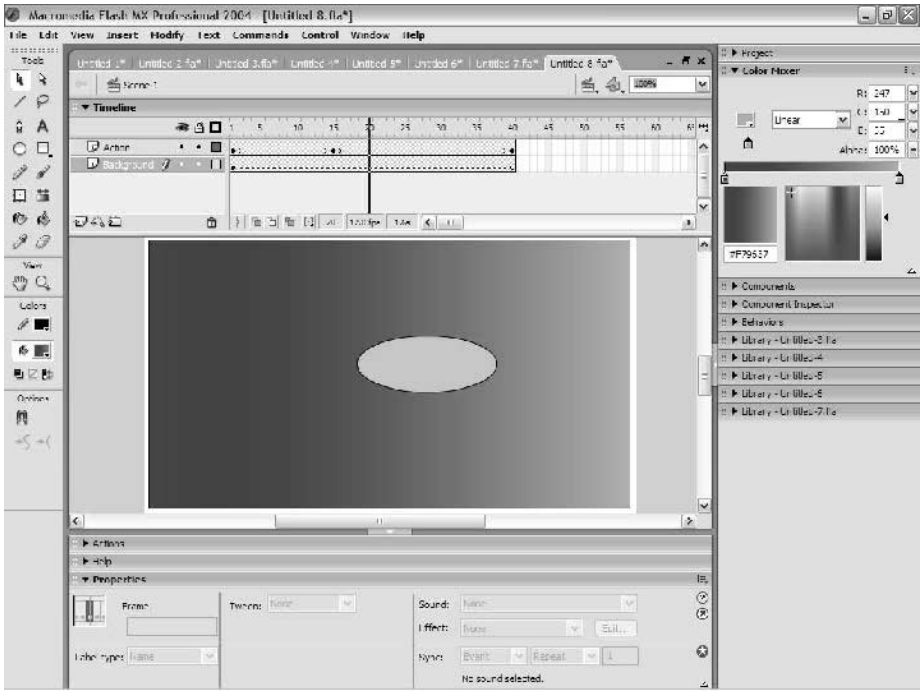


Figure 19-22: View the added background.

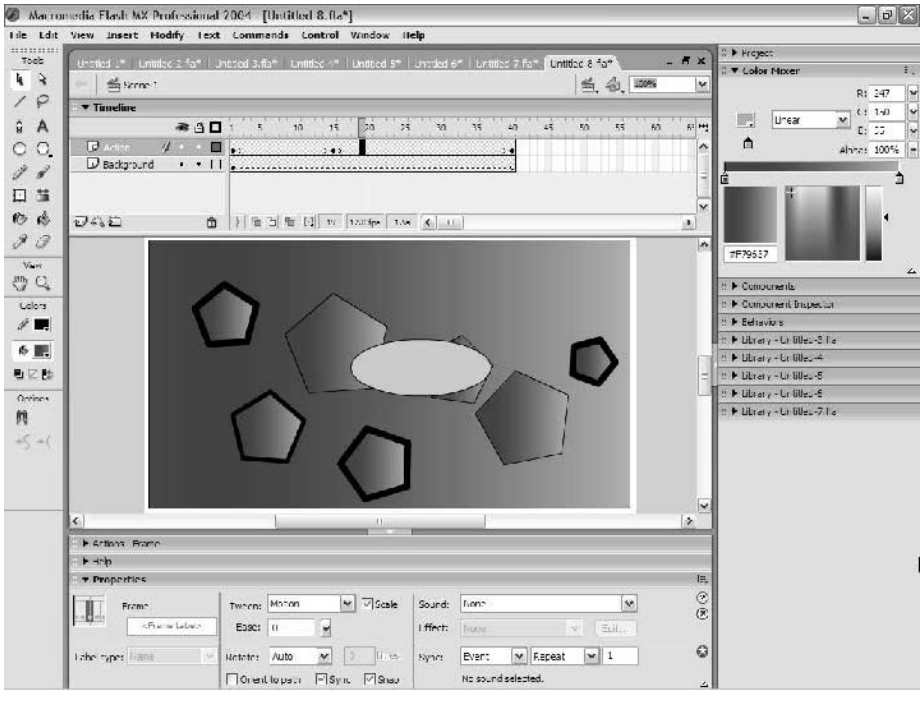


Figure 19-23: Play the completed movie.

www.it-ebooks.info

Using sound

You can import several types of sound files into Flash movies, including the Windows Wave (`.wav`) format, the Macintosh AIFF format, and QuickTime 4 formats. The MP3 format is supported by both platforms. The MP3 format is also compressed to create the smallest import file type, so it is the preferred file type to use in Flash movies.

Sound adds a great deal to Flash movies, as you experience from Web site introductions on the Web. Many Web sites use text and sound without graphics. Presumably, the combination of sound and text has a greater presence than text and graphics. The tradeoff in the form of a smaller file size due to the absence of graphics probably influences the decision not to include them.

Importing sound files is as easy as selecting File ➪ Import ➪ Import to Library. Once imported, the sound files appear in the Library list. When selected, the wave forms of the sound files display in the Library window (see Figure 19-24).

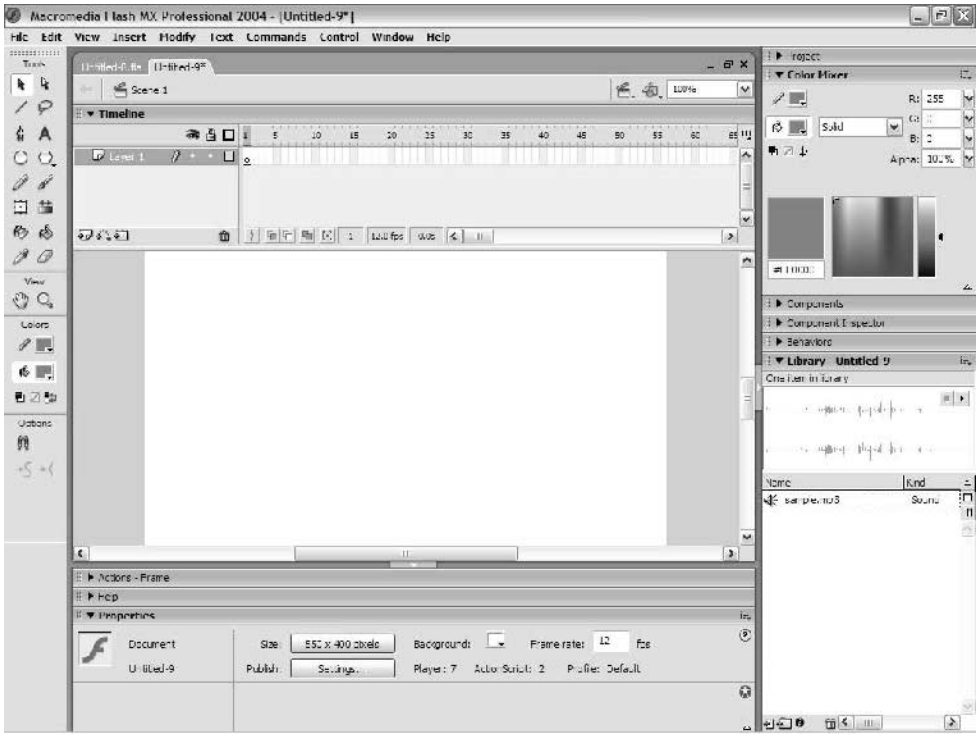


Figure 19-24: Wave forms display in the Library window.

To incorporate the sound in your movie, select the layer and keyframe with which you want the sound associated. Drag the sound file from the Library onto the Stage. The sound begins playing when the associated keyframe is displayed and continues playing until the sound is complete.

www.it-ebooks.info

Adding MP3 sound

Because MP3 sound is a highly desirable format to use in Flash movies, you need to have a way of converting other file formats to MP3. You will also be able to download sound files from the Web that are already in MP3 format. To find file downloads, you can simply perform a search for "sound files" to get a list of potential sites. When you are learning, the actual sound effect is not important because you are just practicing using it in Flash.

If you have a number of Windows Wave files, you can use a sound conversion program to convert them to MP3. These programs are available in trial versions on the Web They can also be purchased in any software store. There are many different sound editing and conversion programs. Most of them will do well, but make sure the product's documentation indicates that it converts your existing sound format into MP3.

Synchronizing sound and action

To synchronize sound with events, select Event from the Sync list box in the Property inspector. The list box also includes a Stream option that can save loading time on the Web. Instead of waiting for the entire sound file to download before playing the movie, you can stream the sound, meaning the sound begins playing almost immediately while continuing to download the rest of the file. Streaming the sound causes Flash to synchronize the sound and animation. If you stream sound, you should not enter a value in the Sound Panel Loops list box, because this lengthens the movie beyond the frames you select.

However, you have to compress (or recompress) MP3 files before publishing your movie. One advantage of using a Wave file instead of an MP3 file is that the Wave file exports without recompression when the movie is published. To recompress the MP3 file, right-click the sound file in the Library panel and select Properties from the shortcut menu. In the Sound Properties Dialog box, select MP3 compression from the Compression list box (see Figure 19-25), then click OK.

Using Advanced Techniques

Once you have tried and mastered the basic skills outlined in this chapter, you are ready to learn some more advanced techniques. It is not essential that you program in ActionScript (the built-in object-oriented programming language) in order to create great Flash movies. However, there are some projects that require these more advanced skills.

Programming with ActionScript

ActionScript is an object-oriented, scripting language similar to other object languages such as JavaScript and Visual Basic. You can build applications in Flash

www.it-ebooks.info

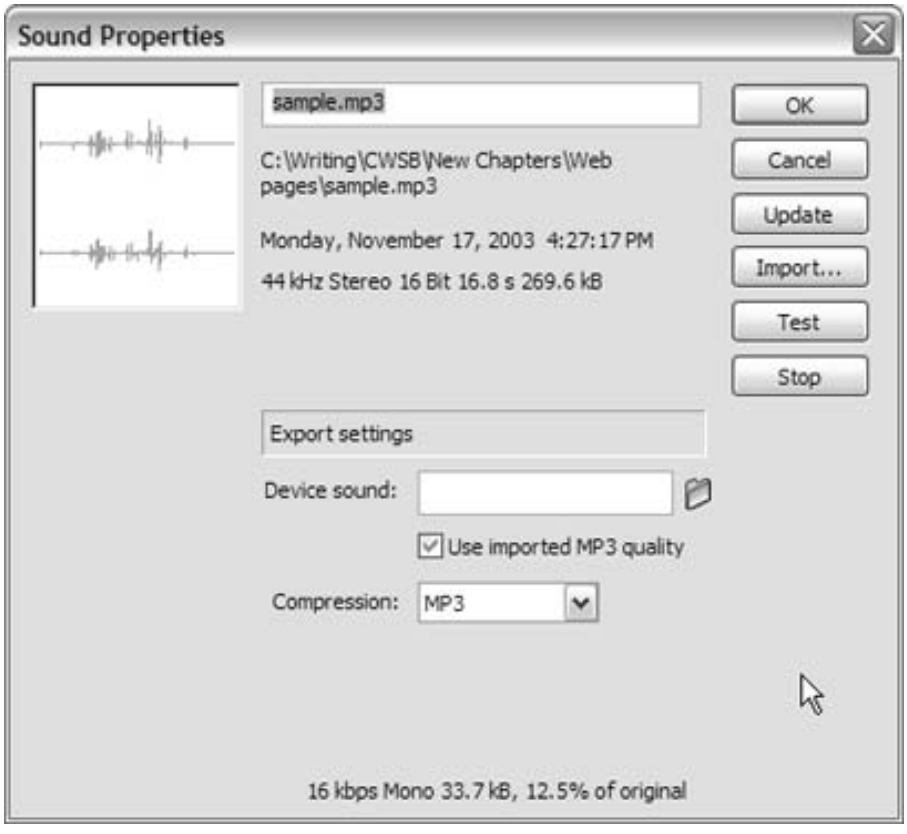


Figure 19-25: Recompress an MP3 file.

that interact with the user. Using ActionScript, you can create scripts that respond to a mouse click by displaying a Web site or playing a Flash movie. You can use this powerful tool in the Normal or Expert mode.

Adding actions

Using the Normal mode (Window ➪ Development Panels ➪ Actions) provides you with a selection of preset scripts. Clicking the right arrow in the upper right-hand corner of the Actions Panel displays a list of options, including the Normal or Expert setting for your ActionScript task. You can choose from options on the Toolbox list, and Flash creates the ActionScript code for you. However, you can edit the code if you want. Using Expert mode, you are required to type code in the text box to the right side of the Action Panel.

Basic Actions provide a list of built-in actions, and when selected, generate the boilerplate code displayed in the text box. The basic actions for which Flash generates the code are shown in Table 19-5.

www.it-ebooks.info

Table 19-5
Basic Actions in ActionScript Action Panel

Action	*Description*
Go To	Go to a frame or scene.
Play	Plays a movie.
Stop	Stops a movie playing.
Toggle High Quality	Adjusts the quality of the movie.
Stop All Sounds	Stops all sounds playing.
Get URL	Displays a URL in a browser.
FSCommand	Controls Flash Player.
Load Movie	Loads and plays a movie from local drive or URL.
Unload Movie	Unloads a loaded movie.
Tell Target	Specify path and control movies.
If Frame is Loaded	Specifies actions taken if a particular frame is loaded.
On Mouse Event	Identifies actions taken when specified mouse event occurs.

Once you have begun using ActionScript, you may find that you need to troubleshoot some errors in running the scripts. The script debugger can be launched from Window ➪ Debugger. To debug a movie in Test Movie mode, you select Control ➪ Debug Movie.

Exporting and Publishing Movies

Exporting is different from publishing. Exporting means that you convert your movie (or a frame from it) to a file format other than Flash. Exporting into some formats will not include the sound files in the exported movie. Publishing converts your Flash movie to the Flash Player format. In a sense, it is compiled. All files incorporated into your movie (such as bitmaps and sound) are combined into the Flash Player file.

Exporting movies

You can export your movies just as you imported images into them. Why would you want to export a movie? There are at least two reasons. First, you might want to export a still image from one of the movie's frames to use in an advertisement or a promotion. Second, you could export your movie to be played in a different player than Flash Player.

www.it-ebooks.info

Flash exports both still image and animated formats. Images from your movies can be exported as bitmap and vector graphic files. Some of the specific formats Flash exports are:

- ✦ JPEG, GIF, PNG, BMP (bitmap)
- ✦ WMF, EPS, or SWF (vector graphic)

When you export an image in your movie as an SWF file (which is the same format as the Flash movie file), the result is a still image of the first frame of the movie.

Mac users cannot use files exported from Windows in the Windows Metafile format. If you are a Mac user exporting to a Windows user, you can export in nearly any format because most image editing programs (PhotoShop, PhotoPaint) can import the Mac PICT format.

Open the `Test2.fla` file that you saved previously. Select File ➪ Export Image. In the Export Image dialog box, select JPEG Image (`*.jpg`) from the Save as type dialog box. Type **test** in the File name box and click the Save button. The `Test2.jpg` file will contain the frame displayed on the Stage at the time it was exported.

You can export movies in movie (animated) formats just as easily as you can export still images. There are some additional consequences you need to be aware of when you export movies. When you export a movie from Flash, instead of publishing it as a Flash Player file, it is going to be because you expect your recipients to play it using software other than Flash.

Exporting in different formats yields different results. Export the file in two or three formats, and if you have more than one movie player you can open it in each player and compare the results. For example, most Windows users have Windows Media Player because it is distributed with Windows. Mac users are likely to have QuickTime because it is an Apple Computers product.



Many player programs for the Windows or Macintosh platform can be downloaded for free from the Internet. Windows Media Player is available at `www.microsoft.com`, QuickTime can be found at `www.apple.com`, and RealPlayer has a free basic player at `www.real.com`.

Among so many formats, how do you know which is best? The answer is that you select the best export format based on the player that will be used to play the movie. The Windows Media Player plays MPEG, MP3, MPG, MP2, AVI, and WMV files. RealPlayer plays MPEG, MPG, MP2, MP3, and MPV formats. QuickTime plays MPEG, MP3, MOV (QuickTime movies), and AVI formats, among others.

To determine which file format to use in exporting a movie, it is best to experiment by exporting and playing the files to test them in the selected player on the platform you intend to use. The Flash Player format works well across the Windows and Macintosh platforms. If possible, it is best to publish (in the Flash Player format) rather than export the movie.

www.it-ebooks.info

A further consideration in movie file exporting is the file size. Some files may be extremely large. For example, movies exported in the AVI format and also in Encapsulated PostScript (EPS) tend to be very large. A Flash movie that is less than 1MB when saved in the FLA format may be over 20MB when exported as an AVI file. Such a large file size will not work on the Web.

Select File ➪ Export ➪ Export Movie. In the Export Movie dialog box, select the Windows AVI under Save As Type. Name the file **Test2.avi**. Accept the defaults in the Export Windows AVI dialog box and click OK to save.

Start QuickTime, Windows Media Player, or RealPlayer. Any of them should be able to play your exported movie. If you have old versions of these programs, you can download the current version of each player from the Internet at `www.microsoft.com`, `www.apple.com`, or `www.real.com`. Next, launch Windows Explorer or use the Macintosh Finder to locate the Test2.avi and the Test2.fla files to compare. The AVI file is probably over 2MB, while the Flash movie file can be well under 100KB. Clearly, you would want to be selective about the movies you export.

Publishing movies

When you publish your movie, all the images, sounds, and symbols that you added to the movie are compiled into the Flash Player file. A Flash Player movie (`.swf`) is the compiled version of the Flash authoring (`.fla`) file. There is no need to copy all of these files to your Web site. Flash takes care of that when the movie is published. You can also select a number of different settings for your published movie.

Select File ➪ Publish Settings to view the Publish Settings window. Select the Formats tab. The file types Flash (.swf) and HTML (.html), are selected by default (see Figure 19-26). You also have the option to create other files during publishing. If you create a Windows Projector file, it can be executed by double-clicking the file name.

Select the Flash tab (see Figure 19-27). You can determine the load order (the order in which objects load from bottom up to top down). You also can generate a size report to show the resources used by each element in the movie. You can protect your product from import online. You can choose to permit or not permit debugging. You can use the JPEG Quality slider bar to adjust the bitmap quality. If you are streaming audio, you can modify these settings. You also can publish in previous versions of Flash.

The HTML tab (see Figure 19-28) affects the HTML file (`Test2.html`) settings. You can determine the template to use, the movie dimensions, the Playback options, and whether Flash Player will display a menu. If you are playing the movie as part of a Web site, you probably won't want a menu showing at the top of the action. You can modify the settings and play back the movie in Flash Player to experiment with the different effects.

Click the Publish button, and then click OK. Locate your `Test2.swf` file on your system and double-click to play it in Flash Player. Does the movie play as it did when you selected Control ➪ Play or Control ➪ Test Movie?

www.it-ebooks.info

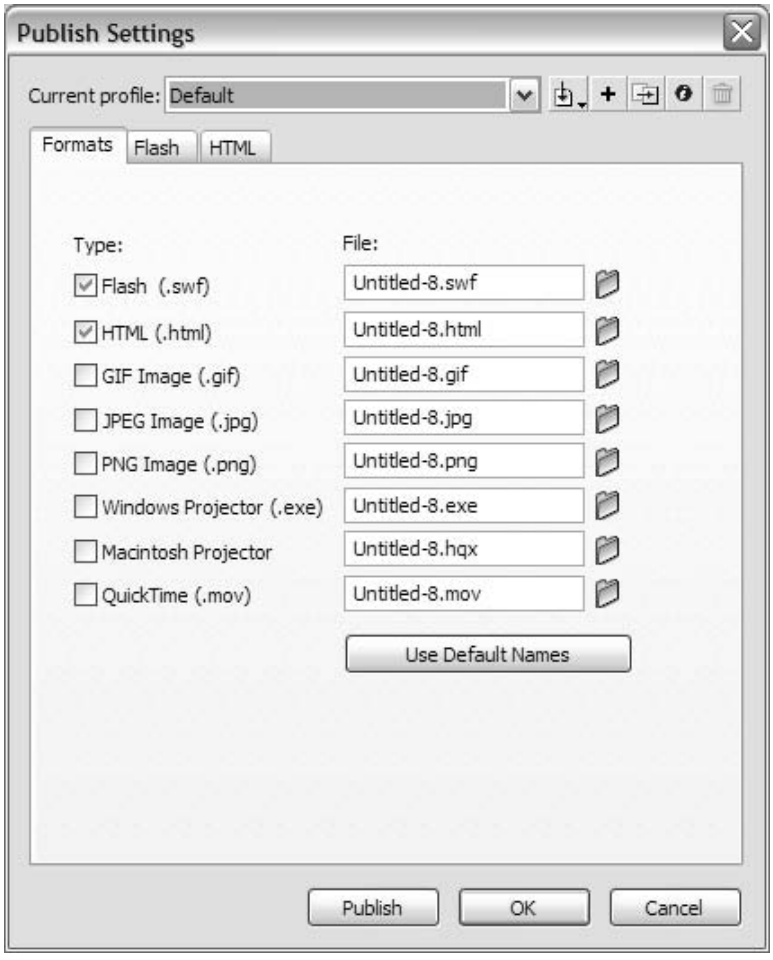


Figure 19-26: Set the file formats in the Publish Settings window.

If you use a Web Publishing application such as Microsoft FrontPage or Macromedia Dreamweaver, you can use it to publish your new Web page containing the Flash movie. Dreamweaver automatically creates the HTML code for your Flash Player movie. You also can use any Word processor, as an HTML editor, that has an option to save the file in HTML format. The `Test2.html` file generated by Flash can be used just as it is by adding the page to an existing Web site.



If you have FTP access, you can upload the SWF file directly to the Web site. Usually you have FTP access if you have your own domain name. FTP (File Transfer Protocol) provides access to file uploading and downloading.

When you publish your movie, the Test2.swf file is created by default as is an HTML file (Test2.html) that handles the HTML code on your Web page. The HTML code is generated for you to copy and paste into the Web page where you want the movie to play (see Figure 19-29). The HTML code can be tailored to specific needs according to the settings you select.

www.it-ebooks.info

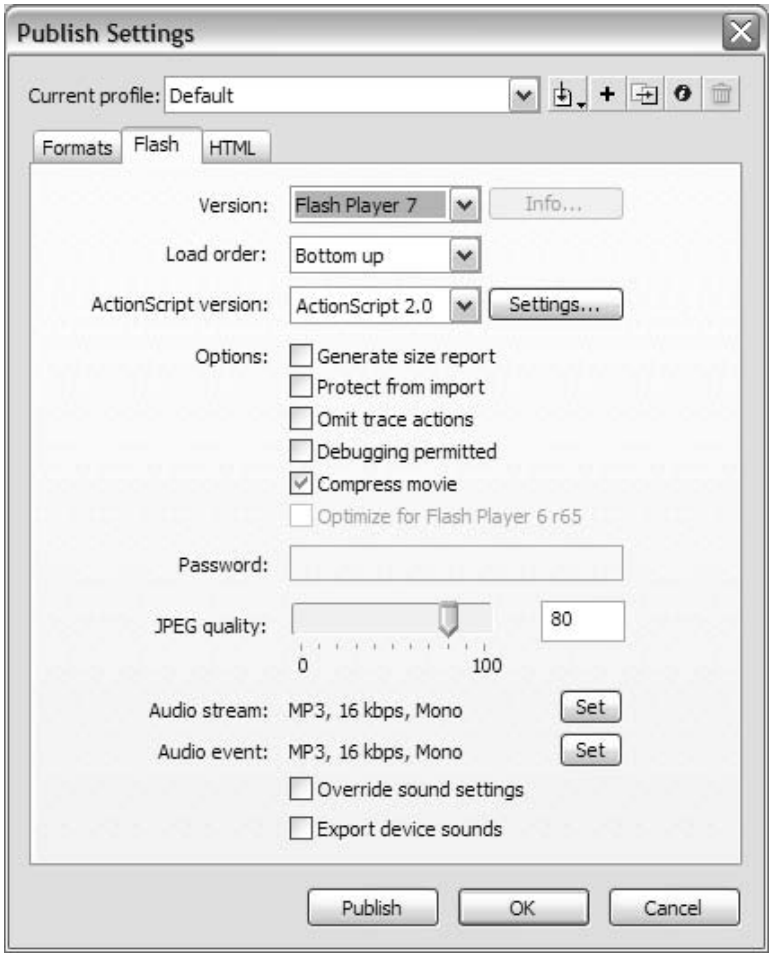


Figure 19-27: View the Flash tab in the Publish Settings window.

When you use Windows Notepad or any text editor to view the `Test2.html` file generated by Flash, you will see that an entire Web page has been generated for you. If you already have a Web page, edit it by selecting the lines that are placed between the HTML Body tags. Select the lines between the HTML tags, `<BODY>` and `</BODY>`. Copy these tags and paste them into the HTML code on your Web page as shown. Be sure to paste the code within the `BODY` element.

Tips for optimizing your movies

There are several things to keep in mind when designing your movies in order to make them run better:

- ✦ Convert all reusable objects to symbols
- ✦ Tween animations

www.it-ebooks.info

- ✦ Don't animate bitmaps
- ✦ Don't embed large font files
- ✦ Use small sound files
- ✦ Use small bitmaps
- ✦ Use vector (WMF) instead of bitmap (GIF or JPG) graphics

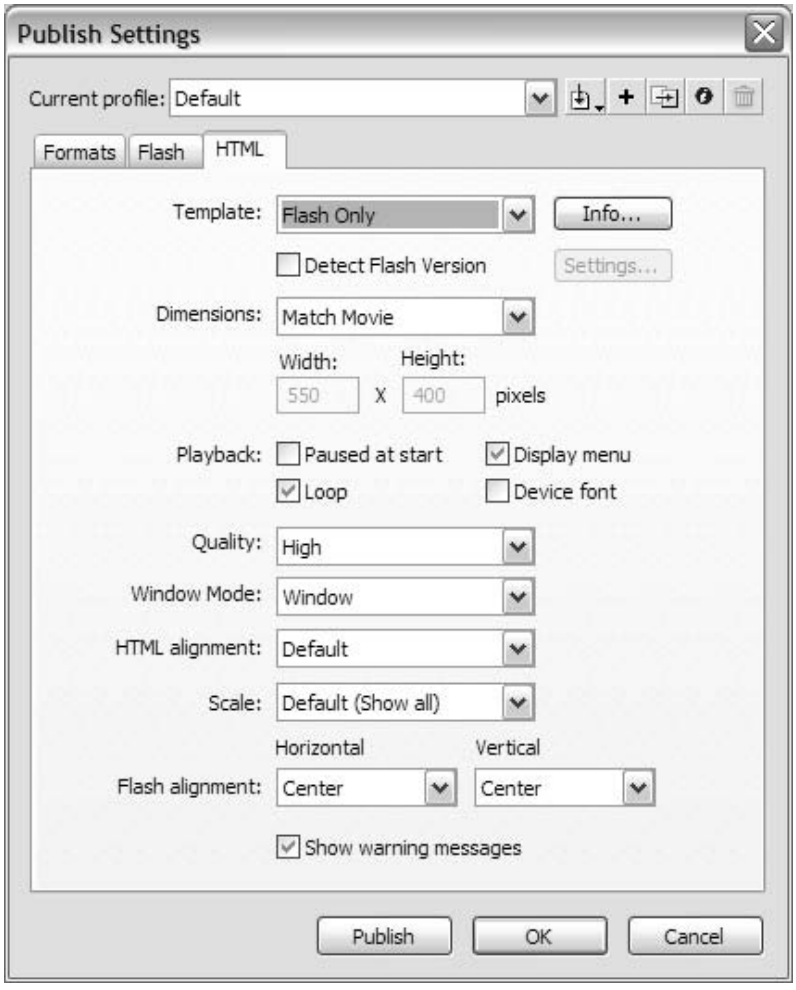


Figure 19-28: Modify the HTML Publish Settings.

Some of these options can be managed by the File ➪ Publish options and some by what you import into the movie. Using the previous suggestions, you can make any necessary modifications to reduce the download time and system requirements. Also, when you design and create new movies, you can keep the memory and connection speed factors in mind.

www.it-ebooks.info

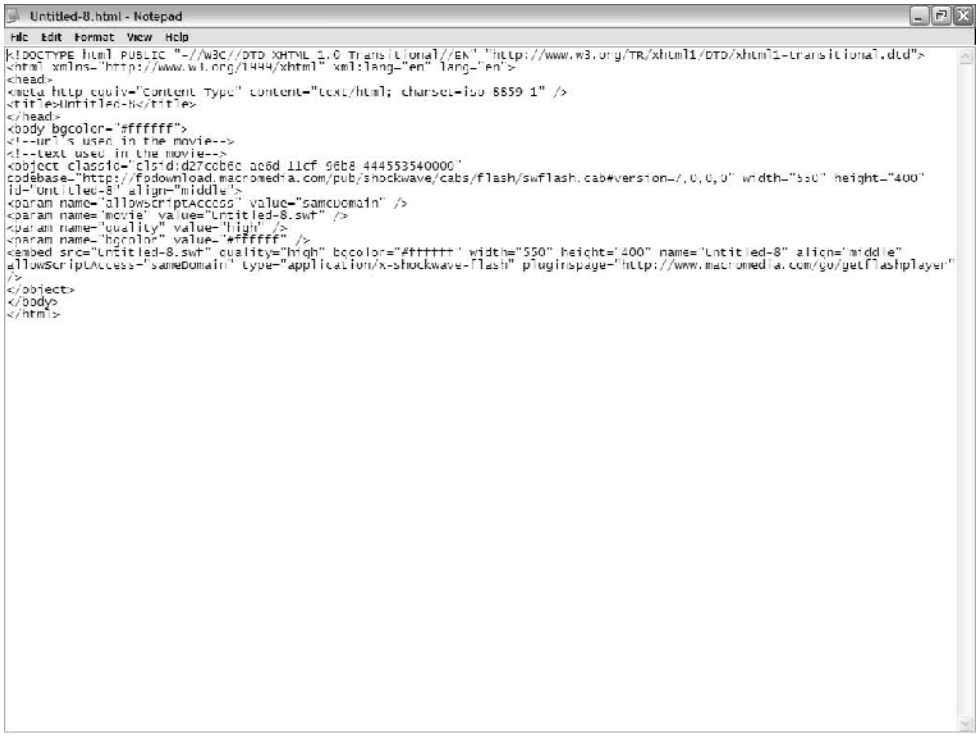


Figure 19-29: View the Test2.html file in Notepad.

Summary

Macromedia Flash is the program of choice for most Web designers when it comes to creating Web animations. This chapter introduced you to the following concepts and topics:

- ✦ Flash movies are made up of scenes.
- ✦ Each scene contains a stage, layers, and timelines.
- ✦ Objects can either be created within Flash or imported into it.
- ✦ Animation is created by assigning events to keyframes.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 20

Adding Multimedia and Other Objects

In This Chapter

Adding audio

Embedding video

Adding Java applets

You can create a really good Web site without any audio, video, or other kind of *plug-in* object. A plug-in is a program that extends the capabilities of a browser. Different plug-ins have varying attributes, so you really need to fully acquaint yourself with any new plug-in, reading all the documentation, in order to use it to the fullest extent possible. However, you can get away with a Web site that doesn't have any Java applets in it. Honest, most of them are done that way. But if you want to explore the outer limits of what you can do with Web pages, you have to dip a toe into these waters.

Adding Audio

Before we get into the details of how to embed an audio file in your Web page—in other words, to make the sound an automatic and integral part of the page—consider that there's a very easy way to add sound. And it's one that avoids a common problem.

One of the most important rules of good Web design is this: if you're going to use background sound on your site, ensure that you give your visitors some way to turn it off. Not everybody appreciates a soundtrack accompanying his or her Web surfing. And even if people do enjoy sound, they grow tired of the same thing repeating endlessly. If you make it impossible for visitors to opt out of the sound, you'll lose some of them. There are ways to do this with embedded sound, as shown later in this chapter, but there's also a simple way to keep everybody happy.

By simply linking to the sound file instead of embedding it, you give your site visitors the option of whether to listen to the sound or not. All you do is create the same kind of link as you

www.it-ebooks.info

would to another Web page or a graphics file. For example, if you wanted to play an MP3 song, you could use this code:

```
<A href="hardrock.mp3">Click here to listen to hard rock!</A>
```

That's all there is to it. Anyone who wants to hear the music can click the link. Anyone who doesn't want to hear it doesn't have to.

Choosing a file type

Plug-ins enable you to play practically any kind of audio file in a Web browser. However, you need to consider just how important the music (or other sound) is compared to how much it can delay the download of your Web page. A sound file in the WAV format, for instance, is hideously large when compared to other options. When it comes to audio file formats on the Web, for the longest time, MIDI files (which end in either `.mid` or `.midi` extensions) were king. MIDI files, rather than being recordings in digital format, are sets of instructions on how to reproduce a sound. This means that they can be very small, making them ideal for use on Web pages.

With the advent of MP3, however, this is changing. MP3 files offer an almost unbelievable capacity for storing CD-quality sound in a relatively tiny file compared to earlier digital recording formats. The key word here is "relative." The longer the music (or other sound), the larger the file. It's not unusual for a two-minute song to take up 3MB, and that adds a stiff penalty to your Web page's download time. Whatever format you use, it's a really good idea to keep the sound duration as short as you can. Unless you're selling digital albums, think in terms of seconds, not minutes.

Embedding audio

We recognize that background sound—sort of a soundtrack to a Web page—does appeal to some Web designers. We'll show you how to make your background sound jump through hoops if you want it to.

Web browsers don't natively play music. Instead, plug-ins do the job. Except for one issue that we cover, you do not need to concern yourself with the particular brand of music plug-in your visitors are using. Many different ones exist, and few Web browsers don't have plug-ins with the capability to play MIDI or MP3 files.

Audio files are embedded in Web pages with the `OBJECT` element or the `EMBED` element. The basics can be handled relatively easily, as the following example demonstrates:

```
<html>
<body>
<object title="A Cool Song" data="Mission_Impossible.mid" >
Sorry I can't play your song!
</object>
</body>
</html>
```

www.it-ebooks.info

```
<EMBED src="url_of_file" width="number" height="number">
</EMBED>
```

This code has the absolute minimum amount of information necessary to add sound. All it does is to specify the location and name of the audio file. If you wanted to play a MIDI file called `softsong.mid`, you could fill in the blanks like this:

```
<EMBED src="softsong.mid" width="200" height="100">
</EMBED><object title="A Cool Song" data="Mission_Impossible.mid" >
```

The `width` and `height` attributes, which are required, seem a bit strange at first—after all, this is a *sound*, not an image or a movie. How can it have dimensions? Well, it's the controls that are used to start, stop, and play the sound, as well as to change its volume, that are at issue here. The problem is that you don't know what plug-in someone's using. That means you don't know what the controls are going to be, let alone how large they'll be. The ones in the example code pretty much cover all the bases, however.

You can actually set things up so that the controls don't show on the Web page. However, you still have to specify the `width` and `height` attributes. Of course, the numbers could be anything you wanted under those circumstances. To make the controls disappear, you use the `hidden` attribute:

```
<EMBED src="softsong.mid" width="200" height="100" hidden="true">
</EMBED>
```

This attribute takes a value of either `true` or `false`, but there's not much point to using `false`—if you don't want the controls hidden, just don't use the `hidden` attribute.



If you hide the controls, you make them inaccessible to your visitors, which means they cannot turn off the sound. This is a fast way to annoy people. It's usually preferable to leave the choice up to them.

A less invasive attribute is `autostart`, which does just what its name implies—it plays the sound file automatically when the page loads into a visitor's Web browser. It also takes values of `true` or `false`, and once again, there's no reason to use `false`, because that just yields the same result as not using `autostart`.

```
<EMBED src="softsong.mid" width="200" height="100" autostart="true">
</EMBED>
```

If you want the sound to play more than once, you can set it to start over again when it finishes. This is done with the `loop` attribute. This attribute takes a value of `true`, `false`, or the number of times to play the sound. As you'd expect, there's no point to specifying `loop="false"` because `false` is the default value. To set a sound to play forever, you use the following code:

```
<EMBED src="softsong.mid" width="200" height="100" loop="true">
</EMBED>
```

www.it-ebooks.info

To set it to play four times, specify that number as the value:

```
<EMBED src="softsong.mid" width="200" height="100" loop="4">
</EMBED>
```



This, as you might imagine, is another touchy area. While some people enjoy listening to a hot new song over and over again, even the best music can get old very quickly if it's repeated endlessly.

Setting volume

The volume of a sound file is only kind of under your control. Although you *can* set how loud you want it to play back, this setting is relative to the volume at which a visitor's speakers are set. In other words, if you set your sound file to play at 100 percent volume, that's just the normal volume at the visitors' end of things. If their speakers are cranked up, your sound plays loudly; if they're turned way down, your sound is gentle. All you can do is to accept this, or lower the volume. Thankfully, Web designers have no way to turn up the volume on a visitor's computer speakers.

The value of the volume attribute sets what percentage of the current full volume your song plays at. A value of `0` means that no sound is heard, whereas a value of `100`, as in the following code, means that the song is played at full volume:

```
<EMBED src="softsong.mid" width="200" height="100" volume="100">
</EMBED>
```



If `volume` is not specified, it defaults to 50.

Using NOEMBED

The `NOEMBED` element enables you to add some message for those people whose Web browsers cannot play the sound, much as with the alt text used with images and other elements. It is placed within an `EMBED` element as follows:

```
<EMBED src="softsong.mid" width="200" height="100" volume="100">
<NOEMBED>
Sorry, your Web browser won't play this song.
</NOEMBED>
</EMBED>
```

Finding digital audio sources

Copyright is always an issue on the World Wide Web, but perhaps nowhere more in recent times than with the music industry. The person who creates a song owns the copyright, which means just what it says—they own the right to make copies. This

www.it-ebooks.info

includes the right to perform the song, either live or on some kind of recording medium. So, if you write the music, you own the rights to it. Otherwise, someone else does. And that someone just might come after you with a battery of lawyers if you don't watch your step, and you infringe on their copyright.

You can, of course, get permission to play popular music by paying money to the big recording studios (some of the money, we understand, even trickles back to the singers, musicians, and songwriters, after the lawyers and executives are finished taking their slices). You can get also permission directly from the artist if you know them. But, even if you happen to be on a first-name basis with a famous recording artist, you're probably still out of luck. Very few of them write their own music and lyrics, which means you'd have to get the composer's and, if applicable, the lyricist's permission, too, along with permission from the singer. It all gets so complicated that not even the lawyers really know what's going on. There are two basic approaches you can take to avoid all these problems, however: You can make your own music or buy music custom made for you by someone else. And the Web puts you on the fast track to getting the music you need, either way.

Making your own music

You can make your own kind of music. If you're an accomplished digital musician, the last thing you need is our advice on what kind of recording studio equipment to use. If you're not, though, and you still want music you can call your own, the answer to your prayers is a Web site in Japan that you should visit: `www.asahi-net.or.jp/~HB9T-KTD/music/English/Soft/`. While you're there, download a copy of Tetsuji Katsuda's Automated Composing System (available in both Windows and Macintosh versions). This precious little lifesaver of a program, shown in Figure 20-1,



Figure 20-1: The Automated Composing System creates a wide variety of tunes.

www.it-ebooks.info

enables you to create your own music by just making a few choices such as whether you want jazz or hard rock, or what kind of instrument(s) you want to simulate—a harmonica or a string quartet.

If you have even a passing knowledge of music (or if you simply enjoy playing around with program settings to see what happens), you can come up with an astonishing variety of digital song files. Because you make them, you own the copyright—and they're unique, which means that nobody else will have the same songs on their Web sites (unless they coincidentally make the same choices for their own settings). Granted, you won't get the same quality as you would get if if you hired a professional digital composer, but the tunes are surprisingly good. And you cannot beat the price with a stick—the Automated Composing System is freeware.

The song files created with the Automated Composing System are in MIDI format, which are fully compatible with use on Web pages. However, you can easily convert them to other formats such as MP3 if you want. Our recommendation for this purpose is a program called Melody Assistant, which can be found at the Guillion Bros. Web site at `http://64.224.241.18/enindex.htm`. This is another great program that, in addition to simply converting digital music formats, is an impressive little music composing system all on its own, including professional features like real-time input from a MIDI keyboard. It requires at least some knowledge of music if you're going to use it for anything but file conversion, but it can be enjoyed by musicians of all levels. And all it costs is $15—about the price of a bucket of fried chicken.

Table 20-1 shows where to find other digital music software.

Table 20-1
Other Sources of Digital Music Software

Site	*Address*
Algorithmic Arts	`http://geneticmusic.com`
AnalogX	`www.analogx.com/contents/download/audio.htm`
Cakewalk	`www.cakewalk.com`
Digidesign	`www.digidesign.com`
Dream Station	`www.dreamstation.de`
Drumtrax	`www.drumtrax.com`
Magix	`www.magix.com/select.html`
Shareware Music Machine	`www.hitsquad.com/smm`
Sonic Foundry	`www.sonicfoundry.com/products`
Squeezer Virtual Band	`www.hmmsoftware.com/squeezer`

www.it-ebooks.info



A kind of in-between approach is also available—using public domain music (songs that nobody owns the copyright to). However, this involves meticulous legal research, where a mistake can land you in court. Plus, you have to enter the sheet music into a composing program, which is nearly as much work as creating your own.

Buying music

If you're looking for custom music, you need to find a composer who is comfortable with creating digital music. So where do you look? The same place that has been in the news so much lately: MP3.com is a marvelous gathering of musicians from every genre (see Figure 20-2). If you're looking for country and western, you'll find it there, along with progressive rock, trance, new age, classical, jazz, and some other kinds you've never heard of before. What does this have to do with Web pages? Simple. MP3.com is your composer supermarket. Go there, spend some time listening, and you may find the sound that you are looking for. Once you do, just click the link to contact the recording artist and tell them what you want. It's a lot faster than advertising for and auditioning composers.



Figure 20-2: MP3.com is a great place to find digital recording artists.

Of course, not everyone who's listed at MP3.com is a musician. You may not, for example, need a songwriter as much as you need a composer; but if you do, all

www.it-ebooks.info

aspects of music creation are represented. And not every composer is interested in whipping up a five-second ditty or a bit of background music for a Web site. But you will find that most starving artists are perfectly willing to jump on the chance to put a little bread on the table by doing what they love most. You make the deal directly with an artist whose work you already respect. No mega-corporations or law firms get in the way, either, so you keep your costs down and the composers maximize their own profits.

You can also check out the Band and Composers Sites listing at `http://midimusic.about.com/musicperform/midimusic/msub15.htm`.



Make sure that you have at least some sort of minimal contract, stipulating exactly what rights you have to use the music. This helps prevent any possible misunderstandings down the road.

If you're willing to settle for music that's not unique, and which your visitors may recognize from other Web sites they visit, you can always buy some prerecorded digital music. Table 20-2 lists the URLs of several sites on the World Wide Web where you can either purchase CDs or download, for a fee, some existing digital music files that can be used on your Web sites. Although you always need to make sure that you double-check the legal details, you can generally use such music without paying any fee beyond the original CD price or download fee.

Table 20-2
Online Sources of Digital Audio

Site	*Address*
Classical Music Archives	`www.prs.net`
EyeWire	`www.eyewire.com/products`
First Wave Media	`http://fwmedia.com/music/cddesc.htm`
Free Up Down Under	`www.royaltyfree.com.au`
musicandsfx.com	`www.musicandsfx.com`
Partners in Rhyme	`www.partnersinrhyme.com`
Web Sounds	`http://pcsounds.com`

Embedding Video

Video is embedded with nearly the identical code you use to add audio files—both use the `EMBED` element, and both are played by the plug-in appropriate to the file type. Video clips, of course, differ from audio in that they are visible, and this means

www.it-ebooks.info

that they have a greater impact on the appearance of your site than just the presence or absence of audio controls.

For video, you'll to know the width and height of the movie. The wrong settings make it look distorted, stretched, or compressed horizontally, vertically, or both. If you recorded the video yourself, there's no problem, because you doubtless know what settings you used. If you're using a commercially available video clip, the documentation should tell you all the parameters you need to set. If you don't have this information, you'll have to rely on either a Web authoring program like GoLive or a graphics program that recognizes that video format to get the information from the file for you.



See Chapter 9 for information on creating your own digital video.

If you do not know the sizes, and your Web authoring or graphics software does not automatically provide it, you can use a standalone media player that recognizes the file type. For example, if you are embedding a video in Apple's popular QuickTime format (.mov), you can find the size in QuickTime Player by following these steps (see Figure 20-3):

1. Select File ➪ Open Movie from the QuickTime Player menu.
2. Navigate to the movie file.
3. Select the filename and click the Open button.
4. Select Movie ➪ Get Info from the QuickTime Player menu.
5. In the dialog box that pops up to the side of the movie, choose Size from the right- hand drop-down list.

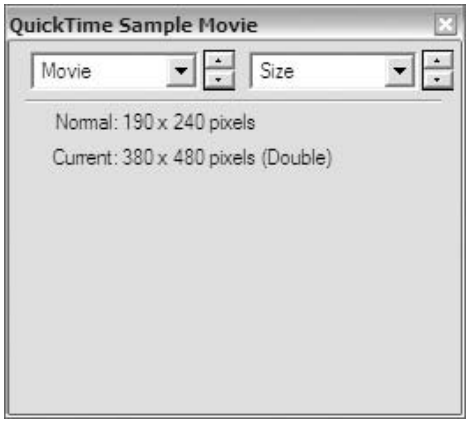


Figure 20-3: You can get the size of a QuickTime movie with QuickTime Player from this dialog box.

www.it-ebooks.info

Otherwise, the only solution is to discover the proper settings for the `width` and `height` attributes by trial and error.

You also should be aware of a couple of slight differences between QuickTime movies and sound files. The `autostart` attribute for sounds is replaced by `autoplay` for movies. And the `controls` attribute becomes the `controller` attribute. Other than the slight difference in name, they work the same way. This is a classic example of the need to check the documentation for any type of plug-in you plan to use.



Do not use the `hidden` attribute with a video clip. If you do, it hides the video.

Running a movie

There are a couple of differences in approach, if not in coding, when dealing with movies. Many video clips *need* to be infinitely looped in order to have the greatest impact. The same code that produces an incredibly annoying perpetual background sound enables a looped video clip to be seamlessly consistent. By the same token, because the video is likely to be an integral part of the Web page, it is fair to hide the controls. Another reason for this is that the video, as with a still image, moves off screen if a visitor scrolls down the page. Background music, on the other hand, is present all the time, regardless of what part of the page you're looking at.

The following simple Web page illustrates how to add a video clip:

```
<HTML>
<HEAD>
<TITLE>Untitled Document</TITLE>
</HEAD>
<BODY bgcolor="#FFFFFF" text="#000000">
<EMBED autoplay="true" controller="false" loop="true" src="RF103T.mov"
width="200" height="200">
<NOEMBED>
<P>This is a video clip of flames.</P>
</NOEMBED>
</EMBED>
</BODY>
</HTML>
```

The results are shown in Figure 20-4.

Finding digital video sources

Table 20-3 gives the URLs of several sites on the World Wide Web where you can find digital video sources.

www.it-ebooks.info

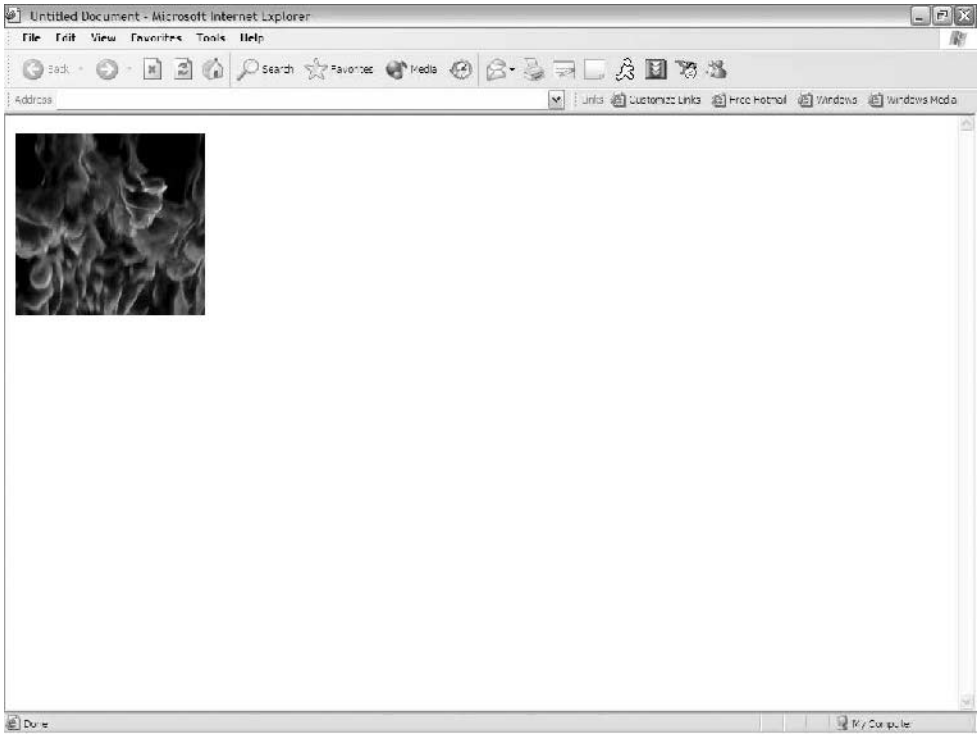


Figure 20-4: Digital video such as this clip from Artbeats.com can add a thrilling bit of dash to your Web pages.

Table 20-3
Online Sources of Digital Video

Site	*Address*
Artbeats	`www.artbeats.com`
Digital Juice	`www.digitaljuice.com`
EyeWire	`www.eyewire.com/products`
Prelude Trilogy	`www.preludetrilogy.com`
Synthetic Aperture	`www.synthetic-ap.com/products/clips`
Wrightwood Laboratories	`www.wrightwood.com`

www.it-ebooks.info

Adding Java Applets

Java applets are small programs that are added to Web pages using, the APPLET element. Java is a full-fledged programming language, and there's not much that a skilled and imaginative Java programmer can't do. The wonderful thing about this is that the capabilities of any Java applet can be easily added to your Web pages. The area occupied by the applet isn't limited to the capabilities of HTML or even JavaScript, but is a separate program running right in your page.



Java has nothing to do with JavaScript, despite the similar names. They are both object-oriented programming languages, but that's as close as they get. See Chapter 16 for information on using JavaScript.

Setting PARAM values

Some applets cannot be customized in any way, but most of them do have some sort of *parameters*—name/value pairs that specify how the applet is supposed to look or function—that you can set. The parameters are set with the PARAM element. PARAM elements are always contained within the APPLET element, as in the following example:

```
<APPLET code="MyApplet.class" width="200" height="100">
<PARAM name="name1" value="value1">
<PARAM name="name2" value="value2">
<PARAM name="name3" value="value3">
<PARAM name="name4" value="value4">
</APPLET>
```

The parameters can be practically anything, depending on what the applet is supposed to accomplish. The following example shows, for instance, four parameters that can be set for an applet displaying some text—the color of the border, the color of the background, the color of the text, and the color of the drop shadow behind the text:

```
<APPLET code="MyApplet.class" width="200" height="100">
<PARAM name="bordercolor" value="0000CD">
<PARAM name="bgcolor" value="FFFFFF">
<PARAM name="textcolor" value="0000FF">
<PARAM name="textshadowcolor" value="D3D3D3">
</APPLET>
```



The eagle-eyed among you may have noticed that the hexadecimal color values in this example lack the usual hash mark. The bordercolor value would most likely be #0000CD, instead of just 0000CD. The manner in which a value needs to be expressed varies from one applet to another. You should carefully check the documentation that comes with any applet to ensure that there are no surprises in store for you.

www.it-ebooks.info

Adding alternative content

A Java-capable Web browser will accept as the content of an APPLET element one or more PARAM elements. If it comes across anything else, it ignores it. Meanwhile, a Web browser that doesn't handle Java (either because it doesn't support Java or because the user has turned off Java support) will fail to recognize the <APPLET> or <PARAM> tags. As a result, if you put anything in an APPLET element that isn't a PARAM element, it will only be displayed by a non-Java browser. This gives you the rough equivalent of a <NOEMBED> tag, or a kind of unlisted alt text that you can use with applets.

If you modified the preceding example to accommodate users of non-Java Web browsers, you can add something such as this:

```
<APPLET>
<PARAM name="bordercolor" value="0000CD">
<PARAM name="bgcolor" value="FFFFFF">
<PARAM name="textcolor" value="0000FF">
<PARAM name="textshadowcolor" value="D3D3D3">
Your computer is not configured to view this Java applet, which displays
colored text against a colored background.
</APPLET>
```

As a result, the visitor using a non-Java browser would see the "Your computer is not . . . " text instead of nothing.

Making it work

So much for theory; time to try all this out on a real Java applet—the PopMenu applet from David Griffith. You can snag a copy of it from www.free-applets.com/PopMenu/PopMenu.html. This applet sets up a navigation bar with the number of buttons that you specify. Each button has an icon and a text label. It uses a rollover so that, when one of your site visitors passes his or her mouse over a button, the icon on it changes color and the button becomes smaller. Clicking one of the buttons sends you to the specified Web page.



See Chapter 17 for more information on site navigation.

Table 20-4 shows the five different parameters the applet takes:



See Chapter 12 for information on using frames on your Web pages.

In the following example, we explore how this Java applet could be used on a personal home page. We also take advantage of a nice little quirk in the APPLET element. Earlier in this chapter, we explained how you can add a bit of explanatory text for your visitors who do not have a Java-capable Web browser. However, you're

www.it-ebooks.info

Table 20-4
Applet Parameters

Parameter	*Function*
`labelpos`	The position of the label text in relation to the icon. Possible values are `right` and `below`.
`frame`	If the applet is used in a framed setting, the value of this parameter is the name of the frame in which the linked page is displayed.
`src`*n*	The source of the icon. The *n* is a placeholder to be replaced by the number of the button.
`text`*n*	The text of the label. The *n* is a placeholder to be replaced by the number of the button.
`href`*n*	The Web page to be linked to. The *n* is a placeholder to be replaced by the number of the button.

not limited to text only—any valid HTML can be used as alternative content (anything that validly includes the `BODY` element, anyway).

In order to make this menu work in both older and newer Web browsers—to "degrade gracefully," as Web designers say poetically—we include within the `APPLET` element a full listing of normal links, each separated by a line break (`BR`) element. As a result, the fancy navigation bar applet shows up in a Java-capable Web browser, whereas a plain set of links that serves the same function (although not as prettily) shows up in a non-Java browser.



See Chapter 4 for more information on setting up links.

The following navigation bar uses five buttons and five corresponding links:

```
<APPLET code="PopMenu" width=400 height=29>

<PARAM name="labelpos" value="right">
<PARAM name="frame" value="mainscreen">

<PARAM name="src0" value="ball1.gif">
<PARAM name="text0" value="Home">
<PARAM name="href0" value="index.html">

<PARAM name="src1" value="ball1.gif">
<PARAM name="text1" value="My Wife">
<PARAM name="href1" value="sandra.html">

<PARAM name="src2" value="ball1.gif">
<PARAM name="text2" value="Our Kids">
<PARAM name="href2" value="children.html">
```

www.it-ebooks.info

```
<PARAM name="src3" value="ball1.gif">
<PARAM name="text3" value="Our Pets">
<PARAM name="href3" value="catsanddogs.html">

<PARAM name="src4" value="ball1.gif">
<PARAM name="text4" value="Vacation Time">
<PARAM name="href4" value="trip.html">

<A href="index.html">Home</A>
<BR>
<A href="sandra.html">My Wife</A>
<BR>
<A href="children.html">Our Kids</A>
<BR>
<A href="catsanddogs.html">Our Pets</A>
<BR>
<A href="trip.html">Vacation Time</A>

</APPLET>
```



The graphics file `ball1.gif` can be found in the same zip file as the applet when you download it from the companion site to this book (`www.wiley.com/compbooks/crowder`). You can, of course, substitute your own graphical icon instead.

Figure 20-5 shows the finished product in a Java-enabled Web browser. Figure 20-6 shows how the alternative, conventional link is displayed in a browser with Java disabled.

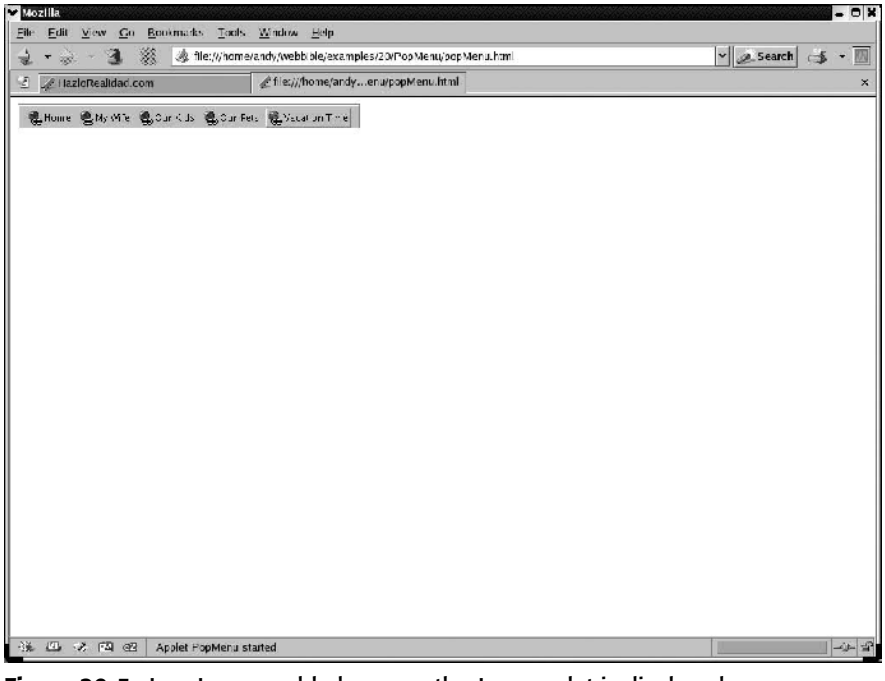


Figure 20-5: In a Java-capable browser, the Java applet is displayed.

www.it-ebooks.info

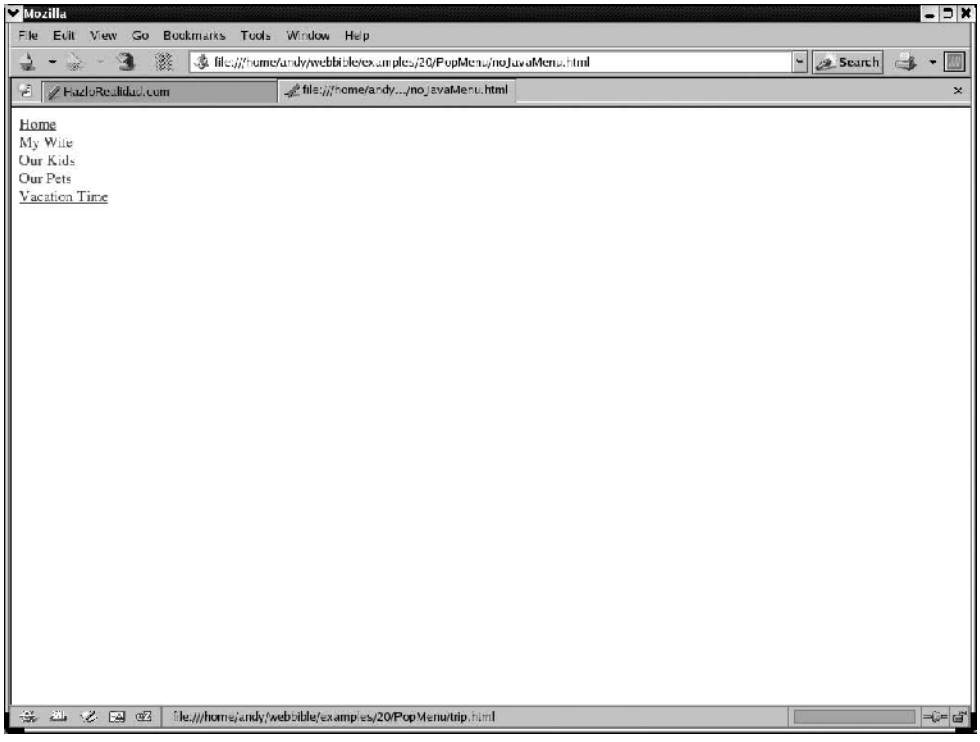


Figure 20-6: In a non-Java browser, the alternative content appears instead.

Applets are objects too

Wouldn't life be simple if any plug-in was represented by the <OBJECT> tag? That's exactly what the members of the W3C think. They have deprecated the <APPLET> tag in favor of the <OBJECT> tag. Let's convert the following file:

```
<APPLET CODE = "MyApplet.class" WIDTH = "200" HEIGHT = "100">
<PARAM NAME = "name1" VALUE="value1">
<PARAM NAME = "name2" VALUE="value2">
<PARAM NAME = "name3" VALUE="value3">
<PARAM NAME = "name4" VALUE="value4">
</APPLET>
```

Using the object tag it would be:

```
<OBJECT classid="clsid:8AD9C840-044E-11D1-B3E9-00805F499D93" WIDTH =
"200" HEIGHT = "100"
codebase="http://java.sun.com/products/plugin/autodl/jinstall-1_4-
windows-i586.cab#Version=1,4,0,0">
    <PARAM NAME = CODE VALUE = "MyApplet.class" >
    <PARAM NAME="type" VALUE="application/x-java-applet;version=1.4">
    <PARAM NAME="scriptable" VALUE="false">
    <PARAM NAME = "name1" VALUE="value1">
```

www.it-ebooks.info

```
    <PARAM NAME = "name2" VALUE="value2">
    <PARAM NAME = "name3" VALUE="value3">
    <PARAM NAME = "name4" VALUE="value4">
</OBJECT>
```

But what is `classid="clsid:8AD9C840-044E-11D1-B3E9-00805F499D93"` for? It starts with 8AD, which stands for *dynamic versioning*. The applet requires at least a certain version to run, in this case Java version 1.4.0.

If you know that your applet only works in one version of Java, then you should use static versioning by specifying a `classid` beginning with `clsid:CAF`. In this case `classid="clsid:CAFEEFAC-0014-0001-0002-ABCDEFFEDCBA"` for Java 1.4.0. If you do this, it will only run in the Java version you specified even if the version you have is higher. You should never need to use this unless there is a problem with a combination of your Java class files and the latest Java virtual machines.

Luckily, Sun Microsystems have created an HTML converter that converts the `<APPLET>` tag to the `<OBJECT>` tag. The converter is located in the directory lib of the Java Development Kit (JDK).

Open a DOS window or Terminal and change directory to lib and run the following command:

```
java -jar htmlconverter.jar -gui
```

The original file is converted to:

```
<!--"CONVERTED_APPLET"-->
<!-- HTML CONVERTER -->
<SCRIPT LANGUAGE="JavaScript"><!--
    var _info = navigator.userAgent;
    var _ns = false;
    var _ns6 = false;
    var _ie = (_info.indexOf("MSIE") > 0 && _info.indexOf("Win") > 0
&& _info.indexOf("Windows 3.1") < 0);
//--></SCRIPT>
    <COMMENT>
        <SCRIPT LANGUAGE="JavaScript1.1"><!--
        var _ns = (navigator.appName.indexOf("Netscape") >= 0 &&
((_info.indexOf("Win") > 0 && _info.indexOf("Win16") < 0 &&
java.lang.System.getProperty("os.version").indexOf("3.5") < 0) ||
(_info.indexOf("Sun") > 0) || (_info.indexOf("Linux") > 0) ||
(_info.indexOf("AIX") > 0) || (_info.indexOf("OS/2") > 0) ||
(_info.indexOf("IRIX") > 0)));
        var _ns6 = ((_ns == true) && (_info.indexOf("Mozilla/5") >=
0));
//--></SCRIPT>
    </COMMENT>
<SCRIPT LANGUAGE="JavaScript"><!--
    if (_ie == true) document.writeln('<OBJECT
classid="clsid:8AD9C840-044E-11D1-B3E9-00805F499D93" WIDTH = "200"
```

www.it-ebooks.info

```
HEIGHT = “100”
codebase=“http://java.sun.com/products/plugin/autodl/jinstall-1_4-
windows-i586.cab#Version=1,4,0,0”><NOEMBED><XMP>’);
    else if (_ns == true && _ns6 == false) document.writeln(‘<EMBED
      type=“application/x-java-applet;version=1.4”
             CODE = “MyApplet.class”
             WIDTH = “200”
             HEIGHT = “100”
             name1 =“value1”
             name2 =“value2”
             name3 =“value3”
             name4 =“value4”
      scriptable=false

pluginspage=“http://java.sun.com/products/plugin/index.html#download”>
<NOEMBED><XMP>’);
//--></SCRIPT>
<APPLET CODE = “MyApplet.class” WIDTH = “200” HEIGHT = “100”></XMP>
    <PARAM NAME = CODE VALUE = “MyApplet.class” >
    <PARAM NAME=“type” VALUE=“application/x-java-applet;version=1.4”>
    <PARAM NAME=“scriptable” VALUE=“false”>
    <PARAM NAME = “name1” VALUE=“value1”>
    <PARAM NAME = “name2” VALUE=“value2”>
    <PARAM NAME = “name3” VALUE=“value3”>
    <PARAM NAME = “name4” VALUE=“value4”>

</APPLET>
</NOEMBED>
</EMBED>
</OBJECT>

<!--
<APPLET CODE = “MyApplet.class” WIDTH = “200” HEIGHT = “100”>
<PARAM NAME = “name1” VALUE=“value1”>
<PARAM NAME = “name2” VALUE=“value2”>
<PARAM NAME = “name3” VALUE=“value3”>
<PARAM NAME = “name4” VALUE=“value4”>

</APPLET>
-->
<!--“END_CONVERTED_APPLET”-->
```

To avoid using JavaScript you can also exploit a browser behavior with regard to the <OBJECT> tag: if recognizes the <OBJECT> tag it ignores the tags inside; if not, then it looks at its content. So you could put the <EMBED> tag inside the <OBJECT> tag. Likewise if the browser doesn’t understand the <EMBED> tag it looks at the <NOEMBED> tag’s contents, so you could put the original <APPLET> tag here. If the browser doesn’t understand the <APPLET> tag, it will display its contents so you can include a text message for the user.

You now have the best of four worlds. Here is the result:

```
<OBJECT classid=“clsid:8AD9C840-044E-11D1-B3E9-00805F499D93” WIDTH =
“200” HEIGHT = “100”
```

www.it-ebooks.info

```
codebase="http://java.sun.com/products/plugin/autodl/jinstall-1_4-
windows-i586.cab#Version=1,4,0,0">
    <PARAM NAME = CODE VALUE = "MyApplet.class" >
    <PARAM NAME="type" VALUE="application/x-java-applet;version=1.4">
    <PARAM NAME="scriptable" VALUE="false">
    <PARAM NAME = "name1" VALUE="value1">
    <PARAM NAME = "name2" VALUE="value2">
    <PARAM NAME = "name3" VALUE="value3">
    <PARAM NAME = "name4" VALUE="value4">
<EMBED
      type"application/x-java-applet;version=1.4"
             CODE = "MyApplet.class"
             WIDTH = "200"
             HEIGHT = "100"
             name1 ="value1"
             name2 ="value2"
             name3 ="value3"
             name4 ="value4"
      scriptable=false

pluginspage="http://java.sun.com/products/plugin/index.html#download">
<NOEMBED>
<APPLET CODE = "MyApplet.class" WIDTH = "200" HEIGHT = "100">
    <PARAM NAME = CODE VALUE = "MyApplet.class" >
    <PARAM NAME="type" VALUE="application/x-java-applet;version=1.4">
    <PARAM NAME="scriptable" VALUE="false">
    <PARAM NAME = "name1" VALUE="value1">
    <PARAM NAME = "name2" VALUE="value2">
    <PARAM NAME = "name3" VALUE="value3">
    <PARAM NAME = "name4" VALUE="value4">

Sorry I can't run your java applet!
</APPLET>
</NOEMBED>
</EMBED>
</OBJECT>
```

You can be the judge of whether you use an <APPLET> tag or adhere to the standards and use an <OBJECT> tag. After all, now when it stops working in the major browsers you know there is a utility at hand to do the conversion.

Finding applet sources

Fortunately for Web designers—especially those who don't want to become master Java programmers—a zillion different Java applets are available for download. Because different people with varying skill levels create these applets, some are marvelous, and others are not so impressive. As with other kinds of software, some Java applets are available for free, whereas others are either shareware or straight commercial ware.

Table 20-5 lists the URLs for several sites on the World Wide Web where you can browse through and download enough Java applets to keep you busy for the rest of the year.

www.it-ebooks.info

Table 20-5
Applet Parameters

Site	*Address*
CodeBrain.com	www.codebrain.com/java
free applets	www.free-applets.com/applets-home.html
FreewareJava.com	http://freewarejava.com/applets/index.shtml
JARS	www.jars.com
Java Applet Store	www.javaappletstore.com/index.html
Java Boutique	http://javaboutique.internet.com
Website Abstraction	http://wsabstract.com/java/index.shtml

Summary

Multimedia—basically, the addition of audio and video to your Web pages—is the secret of lots of the glitz and splash of Web sites. This chapter introduced you to the following concepts and topics:

- ✦ Sound can be added to your Web page as an optional feature—just a link like any other link.
- ✦ The EMBED element can be used to add sound to any Web page regardless of which Web browser is used to access it.
- ✦ The volume you set for a Web page is a percentage of the normal volume your visitors have set for their own speakers.
- ✦ Professional musicians are waiting to provide you with the small amounts of music you need at worthwhile prices. Alternatively, prewritten music may suit your needs.
- ✦ Video can be added in almost the same manner as audio.
- ✦ Prewritten Java applets enable you to add nearly any kind of capability to your Web pages.

✦ ✦ ✦

www.it-ebooks.info

CHAPTER 21

Blogging

In This Chapter

The world of blogging

Using Blogger.com

Alternative blogging tools

Integrating blogging

A *blog*—short for *Web log*—is an online diary. From a technical point of view, a blog is simply a Web page that gets updated every day or so with new thoughts, comments, and links to interesting Web content. But blogs have taken on a life of their own and can be seen as an art form, a creative outlet, and a whole new way of looking at the Web—or even at the world.

Blogging solves the biggest problem people have after they get their initial Web site up: that is, wondering what do I do now? A regular Web site can seem rather complete and finished, and leave you—and the user—wondering where new, fresh content goes. With a blog, new, fresh content is the whole reason for the site—and all you have to do to keep the site alive is update it regularly.

The World of Blogging

Blogs came to many people's attention through politics. The abortive presidential campaign of Howard Dean, Democratic candidate for President of the United States, reached a peak of popularity and hype in late 2003 and early 2004. At the center of the campaign's success in attracting money and volunteers was the Internet—and at the center of the campaign's Internet strategy was the candidate's blog.

The campaign itself became its own Internet "bubble." Many of Howard Dean's advisers went from eyeing office space in the West Wing to looking for another campaign to join in a few short weeks in early 2004. But blogging continued to grow, having shown what a powerful tool it can be for getting people's attention.

Blogs already have their own subculture and vocabulary. *Blogging* is maintaining your blog by adding new content to it; the *blogosphere* is the wonderful world of blogs, which tend to be extensively linked to each other. (Some of the most popular blogs are simply updates about the latest postings on other blogs.) Many people spend several hours a day cruising their favorite blogs and adding new content to their own.

www.it-ebooks.info

But you don't have to devote your life to blogging to get a lot out of it. Even major news outlets post updates, in lively areas such as politics, in blog form, and many Web site creators have added a blog to their existing sites.

One way to use a blog is as a tool for adding news—and new interest—to existing Web or even printed content. Figure 21-1 shows the blog for a textbook, *Fundamentals of Building Construction: Materials and Methods* by Edward Allen and Joseph Iano (John Wiley & Sons).

The blog is hosted on Typepad, a popular tool The blog takes what might seem a rather dry subject and brings it to life with frequent updates.

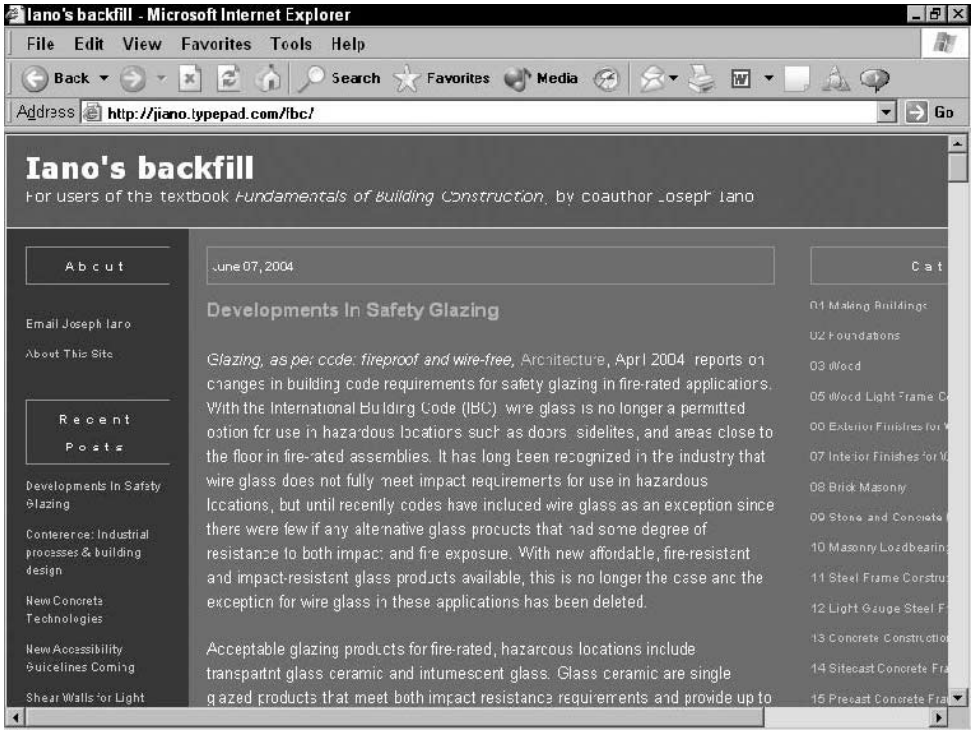


Figure 21-1: A blog can be used to bring a textbook to life.



If you need more information about blogs after reading this chapter, there's a wealth of information online. There are also several good books about blogging. One we can recommend is *We Blog: Publishing Online with Weblogs* by Paul Bausch, Matthew Haughey, and Meg Hourihan (John Wiley & Sons).

This chapter starts by introducing Blogger—the first widely popular tool for blogging. It then introduces other top tools and describes how to add blogging to an existing site, just as the construction example shown in Figure 21-1 does. Use the tools described in this chapter to put a toe in the water of blogging—or to dive all the way in.

www.it-ebooks.info

Using Blogger

For several years now, the number one tool for blogging has been a Web site with an easy name to remember: Blogger (`www.blogger.com`). Blogger makes it very easy to create your own Web site based around a blog of your own.

Blogger is a great way to get started with blogging—or even to make your initial foray into Web publishing. That's because blogging keeps you focused on providing new content—and Blogger makes it very easy to create an initial site and to keep it updated.

A brief history of Blogger

Blogger was started several years ago by Pyra Labs, an energetic small company that recognized the potential of blogging early on. Blogger was begun as an easy-to-use, entry-level tool for creating blogs. It quickly became extremely popular, gaining over a million registered users.

Being this popular, however, created challenges. Being the most popular brand in the blogosphere meant that hundreds of new users a day came to Blogger to get started—but these new users wanted a very easy-to-use tool with a lot of support. The experienced users wanted the tool to have depth so they could take their blogs further. Creating tools that can meet such a wide range of needs is expensive. So Blogger itself needed some support.

Along came Google. Google, the most popular search engine on the Web, was a natural fit for the most popular blogging tool on the Web. Google needed to get better at helping its users search blogs—and Blogger needed Google's resources to keep improving what it offered.

In 2003, Google acquired Pyra Labs, its six employees, and its one million-plus registered users. Shortly afterward, the tool was improved with a spell checker, increased stability and user-friendliness, and other new features—and it continues to be upgraded and improved.

As with many other free Web services, Blogger hosts advertising on your pages—and keeps the revenue for themselves. It's not a bad trade-off for what you get, and you can always move to an alternative service that doesn't include advertising—though you'll probably have to pay for the privilege. Radio Userland (`www.userland.com`) is one alternative paid service to consider if you want an ad-free site.



The setup process described here will put your blog on `blogspot.com`, with a name something like `yourname.blogspot.com`. To have your site hosted elsewhere, use the advanced blog setup at `www.blogger.com/adv-create-blog.g`, described at the end of this chapter. To use the advanced setup, you'll need details such as the FTP server to use; ask for help from your site's host if you need it.

www.it-ebooks.info

Setting up your blog on Blogger

Now that Blogger has been acquired by Google, there are more resources available for making the service easier to use. Blogger has recently updated its interface to make it even easier to create a blog.

However, though the steps are easier, it's still worth taking your time to create your blog carefully. If you like blogging, you may be living with the choices you make here for a long time!

Entering initial information into Blogger

Start by creating your blog by going to `www.blogger.com` (see Figure 21-2).



Figure 21-2: Blogger greets you with a simple starting page.



If you look closely at the Blogger home page, you'll see a link to BlogThis, a quick way to comment on any Web page. Don't worry about this for now—we'll describe it later in this chapter.

From the home page, click the orange arrow with the words, "Create your blog now." You'll then see the Create an account page (see Figure 21-3).

Unfortunately, what you're about to do next looks a little simpler than it is. You have to understand all of these steps to end up with a setup for your blog that you like.

www.it-ebooks.info

Read this section before you actually enter anything, so you can see how each step affects the whole.

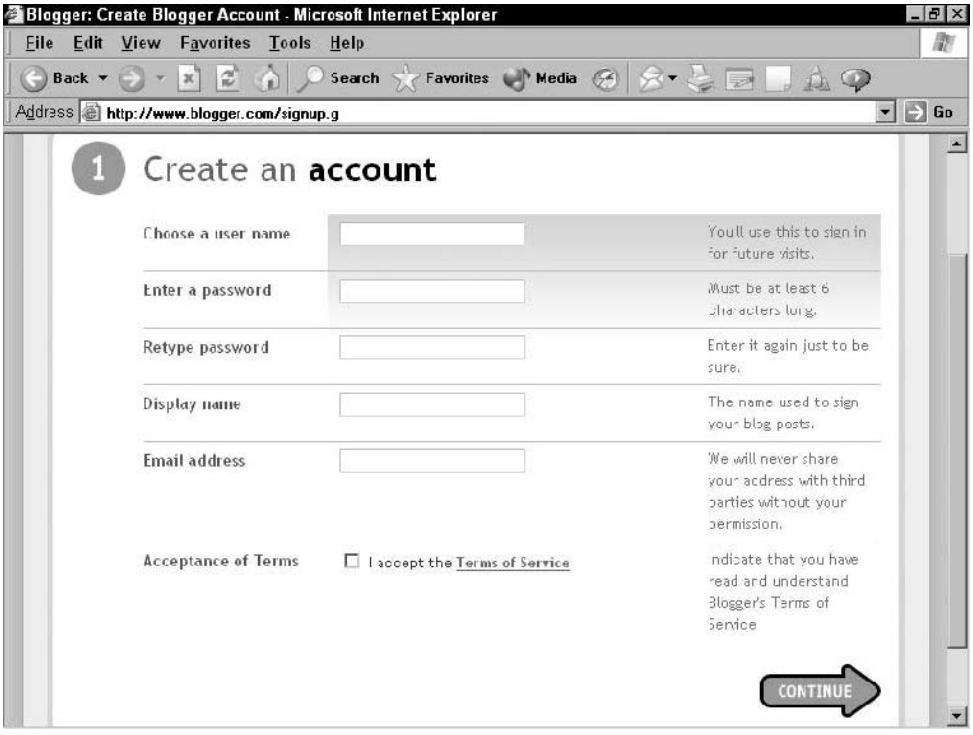


Figure 21-3: Blogger asks for a bit of info to set up your account.

First, enter your proposed user name. This is your sign-on name for Blogger; only you will see it.

Blogger is popular, so many of the choices you might think of, such as your first name followed by your last name, or "blogmaster" or something similar, are likely to be already taken. Choose something that's easy to remember but that's likely to be unique.

Next, enter and retype your password. Be sure to make it hard to guess but easy for you to remember, and keep it secret. Anyone with your user name and password in hand could post content to your blog, which could be quite embarrassing to you.

The next step is where things get interesting: it's time to enter your display name. This name will show up when you post on Blogger, so don't make it too silly. Also, don't choose a display name that gives away your real name. You want to preserve your anonymity, except when you specifically choose to reveal who you are.

You also might want your display name to relate in some way to two things you enter on the next screen: your blog title and blog address.

www.it-ebooks.info

Next, enter your e-mail address. This is only for use by Blogger and Google, who hopefully won't send you too much information unrelated to your use of Blogger.

Next, take a moment to read the Terms of Service agreement. Click the link to read them. They pop up in a new window, so don't worry about losing track of the data you've already entered. After you've read the Terms of Service agreement, click the check box to accept the terms.



It's worth reading the Terms of Service agreement; you'll be entering a lot of personal data into your blog, and it's worthwhile to know what Blogger and Google believe they have the rights to do with it.

Now double-check your choices; you won't get a chance to change them before creating your blog. Then click the Continue arrow.

If you're unlucky, the same screen will re-appear, along with an error message that there's a problem with something you entered. Most likely, your user name is a duplicate of one that's already in use. Try a different one and click Continue again.

When you're fortunate enough to have all the data right, the next screen will appear.

Choosing your title and blog address

After you get all the data right on the first screen and click Continue, you get the Name your blog page (see Figure 21-4).

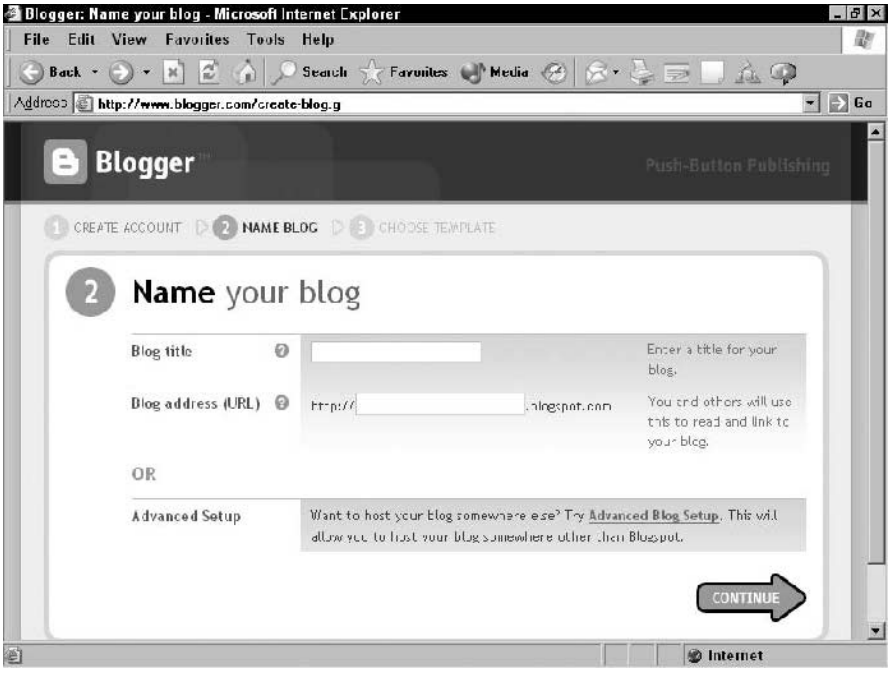


Figure 21-4: Play the name game with your blog.

www.it-ebooks.info

Enter the title for your blog. Give this some thought. Ideally, the title should stand out, should describe something special about your blog, and should relate to your display name (as mentioned previously) and to the blog address you'll enter next, which has to be unique.

Once you have the title right, enter the blog address, the first part of the URL for your blog. The blog address will be added to the domain name `blogspot.com` to form the URL for your blog. (A computer programmer would say the domain name is *prepended by* your blog address.) For example, if you enter *frogwatcher* as your blog address, and it's accepted as unique, your URL will be `frogwatcher.blogspot.com`.

Your blog address, like your user name, must be unique—there's no way two sites can be allowed to have the same URL. A great many desirable names are already taken. So think carefully about a display name, blog title, and blog address that work together and are unique, so as to pass muster.



Google can help you search for interesting blogs on `blogspot.com`, including blogs related to what you want to do. Use the search string "site:blogspot.com *topic*", entering the topic you're interested in. Google will return results with suitable pages. The results won't be perfect—it's hard for Google, or any search engine, to keep up with bloggers! But you're likely to find some interesting reading.

Once you have everything as you want it, check it over. You won't get another chance to change these options. Then, click the Continue arrow. As with the first screen, if you're unlucky, the same screen will re-appear, along with an error message that there's a problem with something you entered—probably a duplicate blog address. Try a different one and click Continue again. Eventually, you'll hit on something unique, and the next screen will appear.

Use a template for your blog's look

Once your second screen of data is accepted, you'll get the Choose a template page (see Figure 21-5). Scroll down to see all the template choices.

Choosing a template is not quite as easy as it first appears. Look at the small template images, called *thumbnails*, to determine which might suit you. To see a preview, mouse over the thumbnail image—your cursor will appear as a pointing finger. Next, click to see a pop-up window with a somewhat larger preview.

To select a template as your choice for the appearance of your blog, click near the option button. But be careful. Once you make a choice, your blog will be created instantly; you won't have a choice to review your choices or change anything first.



You can always change your template later—either in part, by editing the page's HTML, or completely, by choosing a different template. (Blogger will move your data, intact.) But the results of changing your template can be frustrating, so take a bit of extra time in choosing a template initially so as to do all you can to make a good choice.

www.it-ebooks.info



Figure 21-5: Get the look you want.

When you're content with your choice of template, and are ready to create your blog, click Continue. Your blog will be created.

Now's a good time to take one more step. Click the Start Posting button to open your blog so you can start adding content to it. When your blog appears, save the URL in your Web browser's Favorites list. (In Internet Explorer, click Ctrl+D to add the current page to your Favorites.)

Adding to your blog

Although you have to think about the steps, creating a blog is pretty easy. It's keeping it updated regularly with new content that's hard. Yet keeping your blog up-to-date is what enables you to truly claim the title of blogger.

Once you've created your blog in Blogger—try saying "blog in Blogger" three times fast!—you can let it sit, empty, for as long as you'd like. But we recommend that you dive right in and start entering content.

As you do so, you have a lot of options about how to enter text, how to set up options for your blog, and more. In this section we'll describe each of the screens you use to update and manage your blog, so you and the people who visit can get the most out of your blog from the beginning.

www.it-ebooks.info

Posting

You create the posts that appear in your blog using the Posting page (see Figure 21-6). Though you can edit, delete, or rearrange posts later, your blogging life will be easier if you use this page to get your posts right the first time, at least most times.

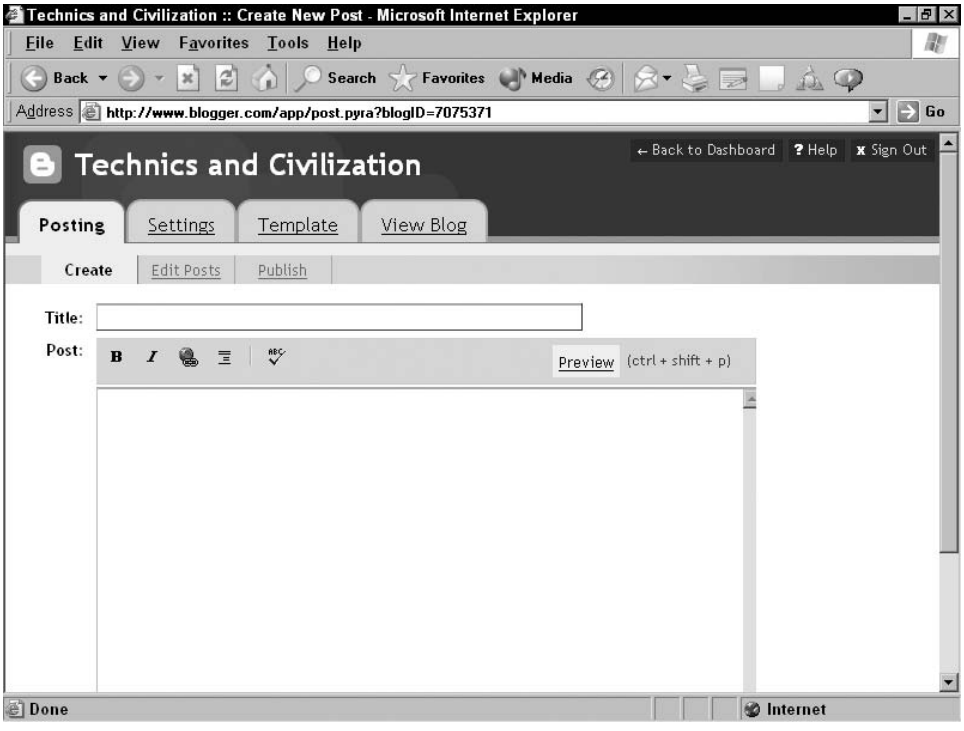


Figure 21-6: Posting is the whole point of blogging.



If you find yourself at your site's Dashboard and want to post, click the link to your page and then click the Create link.

When you enter text on the Posting page, it shows up as plain text, with no formatting. You can cause the text that actually appears as your posting on the Web to be formatted or linked using four options.

The first option is bold. This makes the text you select appear as bold text in the user's browser. First, highlight some text to select it, then choose the bold option. The HTML tags <B> and </B> will surround the text, and the surrounded text will appear bold.

Don't overuse bolding—it makes it appear as if you're yelling at your users. For each phrase that you put in bold, consider whether you can reduce the bolding to a word or two. For each word (or two) that you put in bold, consider whether you can use clearer writing to remove the need for bolding altogether.

www.it-ebooks.info

The next option is italics. This works like bolding, but the text appears italic (of course). Select the text you want, then choose the italics option. The HTML tags <I> and </I> surround the text, showing where italics will begin and end.

Unlike printed italic text, italic text onscreen actually grabs the user's eye less than plain text, as italics make the text harder to read onscreen. So as with bolding, use italic text sparingly, and keep in mind that some of your users may have trouble reading it.

The third option is to link. To link text, highlight the stretch of text that you want to make into a link, then click the globe-and-chain icon. You'll have the opportunity to enter the Web address that you want to link to. To get rid of the link, delete the linked text.

Much of the point of blogging is to provide links to interesting content on the Web—sometimes within a main focus of interest for your blog, sometimes just a personal or a whimsical choice. So link a lot. As with using bold and italic text, it's good to keep the text that you link brief and to the point. (Although it's overused, having the phrase "click here" as a link does tend to get users to click.) But make sure that the linked text, in combination with the text around it, lets the user know what they're likely to see on the other end of the link. Unless part of the point of your blog is to surprise people!

The final option is the blockquote. Text that you make into a blockquote is indented and displayed in a different font. It's usually used to highlight quoted text.

Don't use unending paragraphs of blockquoted text. You may end up with copyright problems if you quote too much, for one thing. Consider using a blockquote in lieu of a link. If there's an interesting few lines of text somewhere, consider blockquoting it and providing a link to the source so that most of your blog visitors can see what you're talking about without necessarily having to follow the link.



Be careful about quoting song lyrics. They're copyrighted, and quoting more than a line or two of them may bring you unwanted attention. Other people post song lyrics on their sites, but there's no reason for you to push—or just plain exceed—the limits of copyright law. (Several sites that contain song lyrics have had to remove them under legal pressure.)

You also have the option of spell-checking your content—a capability added since Google purchased Blogger, and for which those of us with bad spelling can be grateful.

Other Blogger options

Since it's the area where you add new content and keep your blog up-to-date, you're likely to spend most of your time editing your Blogger site in the posting area (hopefully anyway). But Blogger has a lot of other options.

One option you should use a lot is View Blog. All it does is pop up a new window with your blog in it. This is very nice for checking what you currently have.

www.it-ebooks.info

Don't simply use View Blog to view your site after you make changes. You may also want to use this option to see what's currently on your site as you're writing new posts.

The Settings area is nerd heaven. You can change dozens of options on separate pages with Basic, Publishing, Formatting, Comments, Archiving, Site Feed, Email, and Members options.

You should spend time getting to know these options. With them, you can do really cool things like send e-mail that automatically is posted as an addition to your blog. However, don't spend all your time messing with the settings—a blog is there for you to post to!

With the Template area, you can see the HTML code for your template. This may or may not useful to you. You need real HTML and design skill to make useful changes to your template, but if you have that skill, or know someone who does, this may be a good area for you to spend timeon.

You can also choose a different pre-existing template in this area. This doesn't take any special skill—your whole blog just automagically appears the way you want it.

The Dashboard is the screen you'll typically see when you first sign in to Blogger. From the Dashboard, you can manage your entire blog. Figure 21-7 shows how the Dashboard appeared for one of the authors' blogs.



Figure 21-7: The Dashboard is a control panel for your blog.

www.it-ebooks.info

The BlogThis! option is a particularly powerful one. It enables you to put a new icon in the toolbar of your Web browser, which you can use to quickly add a post to your blog that links to the site you're visiting at that moment. This is a great tool for bulking up your blog quickly with interesting content.

There's a lot you can do with your blog. You can allow other people to add comments to it, for instance. Enjoy your blog—and keep adding to it. It's the gradual accumulation of comments and links over time that makes a blog worthwhile.

As soon as you've set up your blog and chosen your basic settings, tell people about it. You can form a whole new online community around your very own blog.

Alternative Blogging Tools

Blogger is not the only tool out there. In fact, as you spend time in the blogosphere and note where various blogs are hosted, you'll run into a variety of different sites that host blogs. You'll also run into many "roll your own" blogs, created with a tool or simply with pure HTML, that live on individual users' domains.

It's worth knowing something about the various blogging tools and add-ons that are out there so you can make an informed choice of the best way to get started with blogging. Perhaps you can even make a different choice for the best tool to use once you've achieved some momentum as a blogger.



If you want your blog to automatically include content from other blogs—or to serve as a source that publishes automatically to others—check out the description of RSS and Atom in the Userland description later in this chapter.

There's a wide variety of blogging tools out there. Each has its own advantages and disadvantages as well as its own thriving community of users.

AOL's Hometown

If you know anything about AOL, which has always focused on user, it may not surprise you to know that AOL includes a strong blogging feature. What might surprise you is to find out that the tool, hosted at AOL's Hometown Web-building site, is free—that is, no charge beyond your AOL membership—to AOL members.

AOL calls a blog a "journal", as shown in Figure 21-8. AOL blogs are very easy to use and fairly capable—and if you're already an AOL user, an AOL blog might be just about an automatic first choice for you.

AOL enables you to add posts directly to your journal or to send them in using instant messenging. AOLbyPhone is another way to add posts—you just call your post in by phone, and it appears as an audio clip.

You can make it easy for people to add comments—even non-AOL members. And you can make your journal public or private. AOL Alerts let you know when someone comments, or let readers of your journal know when you've posted.

www.it-ebooks.info



Figure 21-8: AOL offers a Hometown for blogs—and Web pages.

There are certainly limits on AOL Journals—older posts disappear from the screen fairly quickly, for instance. But you can get around these limits by using other AOL capabilities, or combining your blog with a Hometown Web page or the "You've Got Photos" feature. These capabilities, and the support—from other users, and from AOL itself—that AOL is known for, may keep you an AOL Journals user for a long time.

blogs.com and TypePad

`Blogs.com` is home to the TypePad Weblogging service (see Figure 21-9). TypePad is a hosted service, much like Blogger but without the ads—and with a set of features that may lead you to prefer it to rivals.

Cost is an issue. You may get a certain amount of introductory time free, but in mid-2004 charges for the service ranged from about $5 a month to as high as nearly $15, depending on how many of the service's capabilities you use. The charge isn't bad if you're serious about your blog, but if you only post once in a blue moon, or want to save your money for other purposes, a free service may be a better choice.

For simply adding posts to your blog, TypePad is quite capable. You can allow or disallow comments on each individual post you make, and you can have comments wait in a queue for you to approve or disapprove them. (The first time someone

www.it-ebooks.info

comments on one of your posts with a string of obscenities you may run, not walk, to turn on this feature.)



Figure 21-9: You'll see some great blogs at Typepad.

TypePad is also good for hosting photographs. You can add photo albums to your blog, with photo thumbnails and display built-in pages. Links to Amazon.com for books or music are generated automatically within a list that appears alongside your posts.

Keeping your blog up-to-date is easy too. You can post using a browser, e-mail, or a WAP-enabled phone.

You need to know a little HTML to get the most out of TypePad, but it's easy to pick up the little bit that's needed. And TypePad will produce a nice-looking blog that you'll be proud to have people link to.



If you are technically savvy, or know someone who is, Movable Type may be for you. Movable Type is the engine behind TypePad, and you can install it on your own site. If you know Perl—the scripting language widely used within Web sites—or know someone who does, check out Movable Type.

LiveJournal

Want to join over half a million people in an exciting online community based around blogging? LiveJournal (`www.livejournal.com`) may be the place for you to do it (see Figure 21-10).

www.it-ebooks.info

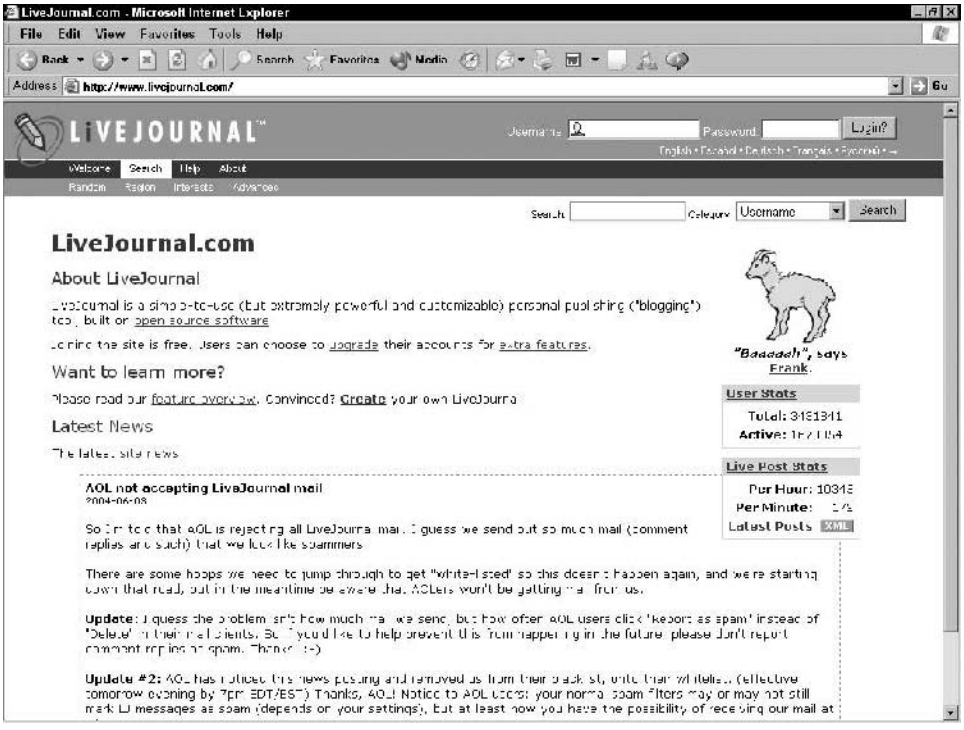


Figure 21-10: LiveJournal is a strong blogging community.

LiveJournal is very community oriented. In fact, when we tried the service, the only way to get a free account was to have a current member invite you. An invitation you may want to take, if you get one. Otherwise, the service recently charged a reasonable $25 a year.

Comments on posts are as much a part of the LiveJournal scene as posts themselves. There's a strong discussion board capability built in, and you can get an e-mail notice sent to you when someone responds to your comment. It's easy for bloggers to link their blogs to one another, and you can add polls to your blog to get quick feedback.

You can post using a browser or a simple tool, though photos weren't supported when we checked the site out recently. (You can always post photos on another service and link to them from your blog.) But LiveJournal continues to grow in activity and popularity.

Radio UserLand

Userland is a powerful tool well-suited to creating a serious blog—one that includes other users' content through an automatic stream, and that can easily be streamed into other sites. Figure 21-11 shows the Radio UserLand site.

You need some technical knowledge to get the most out of UserLand—and a little dough. A UserLand blog costs just under $40 a year. As for technical knowledge, the

www.it-ebooks.info

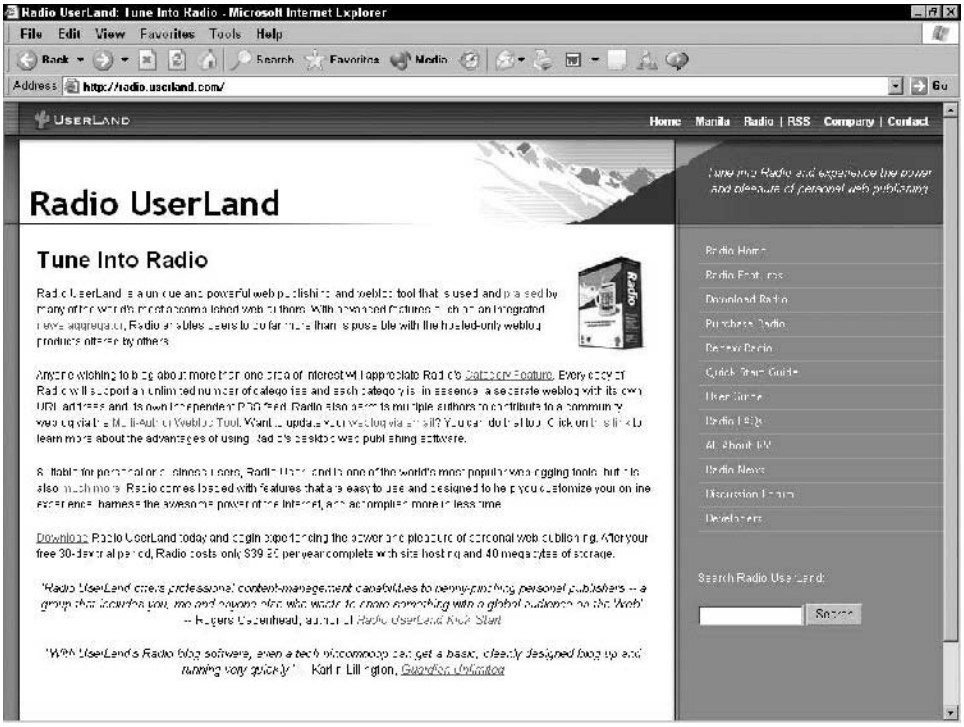


Figure 21-11: Radio UserLand makes you a blogging broadcaster.

higher-end capabilities of UserLand are based on RSS, a specification for using Web site content as a *syndication source* or *feed* into other Web sites. If this sounds like fun to you, then UserLand may be your best choice. If not, you may want to start with a simpler service.

To create your Radio UserLand blog, you use desktop Web server software so you can edit your blog while disconnected from the Internet. Once online, you *upstream* your additions to the Web, leaving you with a local copy of your site's content. You also have the option of posting from a browser or via e-mail, and you can post pictures as well as text. You can allow visitors to your site to add comments on your post.

The real power of UserLand kicks in if you use RSS. UserLand manages the RSS 2.0 specification, and you can automatically take in other RSS content as a feed to your site and enable others to take your own site as a feed to theirs. If you want to have all the latest content on a topic, or if you have a lot of people who want to republish what you say, RSS and UserLand may be the way to go.

However, all of this is not simple. Even the acronym is controversial. Some say it means Really Simple Syndication; others, Rich Site Summary. In any event, RSS may be replaced soon by an updated specification called Atom.

www.it-ebooks.info

But the specifics of RSS may not be a problem for you. Increasingly, this kind of capability is being built into hosted sites or made available through easy-to-use tools. So you may be able to take advantages of the capabilities of RSS without worrying about what the acronym means.

Blogging add-ons

There's a tremendous range of blogging add-ons for everything from adding capabilities to your blog, to keeping up with new posts, to contributing to the latest and greatest technical specifications underlying blogging.

Two of the best resources for blogging add-ons are:

`www.weblogdesign.com/tools.php`

`www.writerswrite.com/blogging/software.htm`

Each of these sources has dozens of resources. Perhaps a brief profile of a few of them will encourage you to go check out the range of what's available yourself:

- **Blogback:** Blogback is a tool that enables visitors to your Weblog to comment on your Weblog posts, if you don't already have that capability yourself. Blogback is free and fun. Check it out at `www.blogback.com`. Figure 21-12 shows the home page.

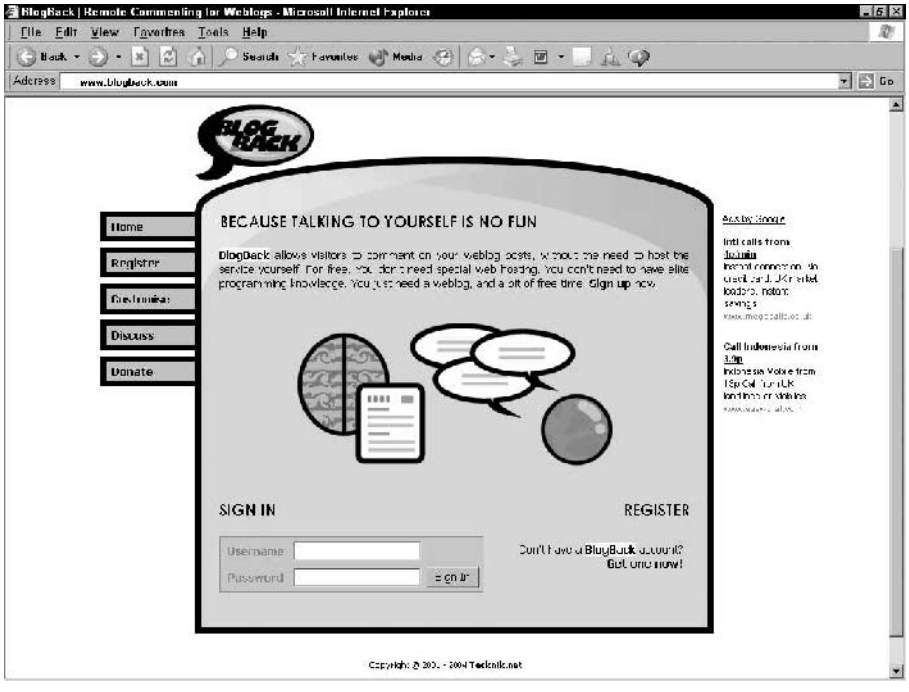


Figure 21-12: Blogback is just one of the scores of tools out there.

www.it-ebooks.info

- **blogBuddy:** blogBuddy is a tool that enables you to access the capabilities found in the Blogger API. The idea is that you can get access to added capabilities, before they're implemented in Blogger or elsewhere. The blogBuddy news page, which you can visit at `http://blogbuddy.sourceforge.net/`, hasn't been updated for a while, but if you're technically oriented, this might be a good place to find out something about what's possible with blogs.
- **blogdex:** Calling itself "the Weblog diffusion index," blogdex tracks the most-quoted stories in the blogosphere. The contents don't always reflect well on the blogging community and sometimes cover controversial topics, but they do tell you what's going on out there. Visit `http://blogdex.net` to see for yourself.

Integrating Blogging

Integrating blog content into a regular, non-blog Web site is not that hard and can add life and interest to a site that might otherwise become stale. There are several ways to do this, from the easy to the complicated, but any of them can bring the power of blogging to a Web site near you.

Blogging as a writing style

Many mainstream news sites, such as `msnbc.com`, use blogging as a style of writing. Just add a new header and a few paragraphs of text to the top of your Web page every so often, commenting on whatever's happened recently, and link to relevant resources. Presto—instant blog!

You can have one or several such blogs on a recent Web site. If you don't mind cranking up your HTML editing program, or using a Web site host's tools for each update, this is a very easy way to make your site more lively and interesting.

Integration by reference

You can create a blog using any of the tools described in this chapter, or others, then link to it from your Web site. For instance, you can create a full blog in Blogger (described previously), posting everything you can think of in it, and inviting comment from all and sundry.

Then cut and paste the "best of Blogger" from your blog into your regular Web site. Link to the whole thing for those who just can't get enough. Using this approach, you might end up with the best of both worlds—a very good blog and a very good Web site.

Full integration

You can fully integrate a Web log into your site through several different means. For instance, Blogger includes an option called Advanced Blog Setup that enables you to put a fully capable Blogger blog right onto a page of your Web site (see Figure 21-13).

www.it-ebooks.info

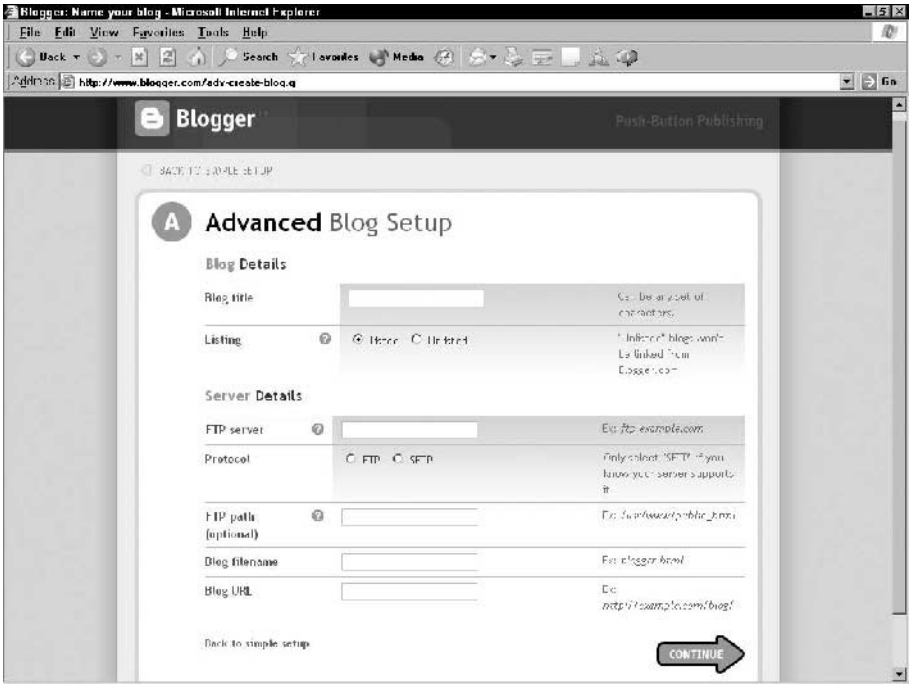


Figure 21-13: Blogger offers one way to integrate a blog.

You'll need FTP access to your Web site and a few details of where Blogger needs to send updates. Ask your Web host for help if you need it. But with this capability, you end up with all the power of Blogger, right in the middle of a Web site over which you have full control.

RSS is another way to integrate your blog into your site into your blog, or just about anything else you want to do. See the earlier section on Radio Userland for details.

Summary

Blogging is a breakthrough in keeping a Web site up-to-date and interesting. There are many ways to create your own blog or to add a blog to an existing site. You can even use a blog as a stream of data into your own site, or others' sites.

This chapter introduced you to several blog-related concepts and topics:

- ✦ Blogging makes it easy for anyone to be heard on the Web.
- ✦ Blogger, now owned by Google, is one popular way to start your own blog.
- ✦ Providers ranging from AOL to Radio UserLand and beyond provide a wide variety of tools for blogging.
- ✦ It's easy to integrate a blog into your existing site.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

P A R T

V

Cashing In on E-Commerce

In This Part

Chapter 22
Setting Up Your Store

Chapter 23
Selling on eBay

Chapter 24
Getting Paid

Chapter 25
Using Advertising

Chapter 26
Covering All the Bases

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 22

Setting Up Your Store

In This Chapter

Choosing your e-commerce setup

Setting up shopping carts

Taking payment

Riding Internet buying patterns

E-commerce is not some big, mysterious new thing. The goals for e-businesses are the same as the goals of traditional retailers—the customer is simply reached through a new and different channel. You're selling either a product or a service to the same people you'd be selling to if there were no such thing as the Internet. Of course, with the worldwide reach of the Internet, you have access to many more potential customers or clients than ever before. And your customers can shop at your e-commerce store 24/7.

Choosing Your E-Commerce Setup

Depending on what you're selling and how you want to market it, your storefront needs will fall into one of five categories:

- Single product sites and specialty stores
- General stores
- Service sites
- Shopping malls
- Affiliate stores

Single product sites and specialty stores

Single product sales tend to be the province of the smallest kind of business—a single individual or small group of people. The product being sold is often a craft item or a book.

A specialty store is, at first glance, not very different from a single product site. Both feature a very limited stock focused

www.it-ebooks.info

on one thing. A specialty store, however, carries more than one item, although those items themselves fall into a certain easy classification:

- ✦ Bookstores
- ✦ Shoe stores
- ✦ Jewelers
- ✦ Automotive supplies

Some stores fall into kind of a gray area between the two classifications. The Web site for Jerry Stocking and Karen Bates (`www.achoiceexperience.org`), for instance, sells both books and tapes, but by a single author (see Figure 22-1). Thus, we classify it as a single product site instead of a specialty store.

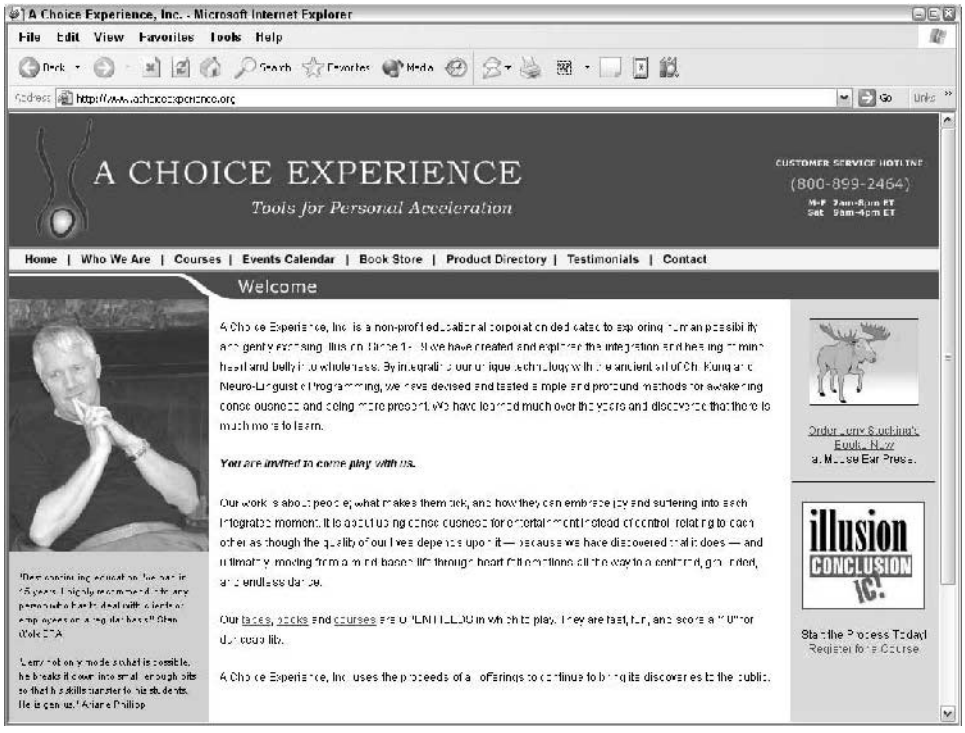


Figure 22-1: Achoiceexperience.org is a good example of a single product Web site.

General stores

The era of the general store seemed to have pretty much passed many years ago, with the exception of a few large department stores. Even those have severely limited their stock, and if you want anything other than a few limited lines like clothing, jewelry, or perfume, you have to turn to their catalogs. Catalog sales,

www.it-ebooks.info

however, are practically tailor-made for the Internet. As a result, the new technology has brought about a return of the general store.

Even the big department stores like JC Penney offer more on their Web sites than they do in their typical stores (see Figure 22-2). Why? Because Web sites don't take up any physical space, which means that a huge site costs very little more than the simplest, one-page site does. When it comes to physical real estate, every square inch must be totally cost-effective. But a Web site can have several pages that don't draw much business without increasing the cost of the site.

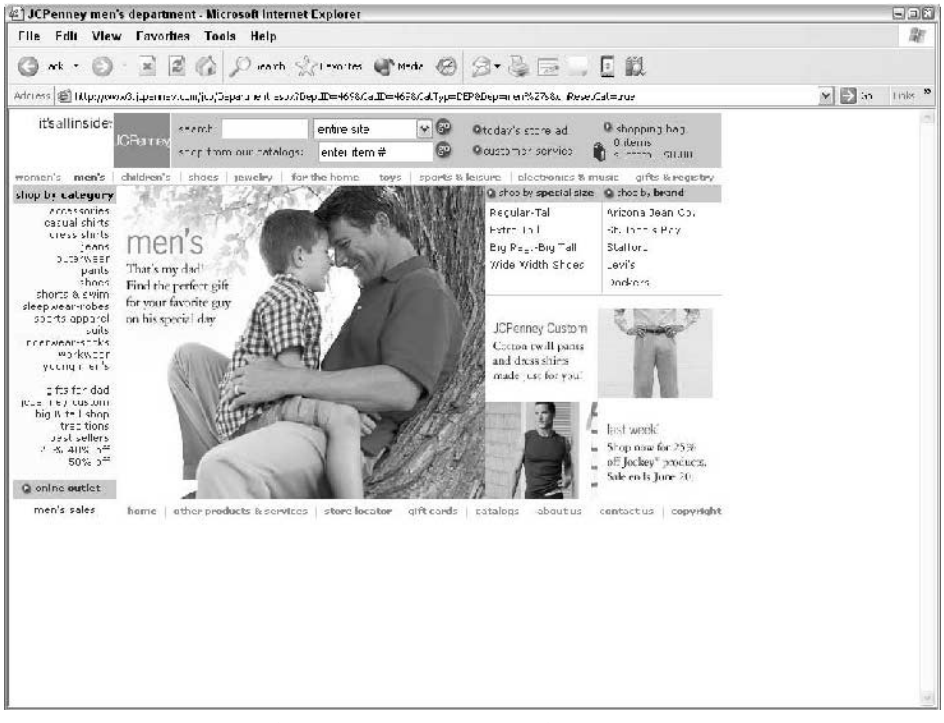


Figure 22-2: You can find more at JCPenney.com than you can in most JC Penney physical stores.

Online superstores

Technology is advancing at an amazing pace. New gadgets, computer-related products, and other technology is finding its way into many mainstream stores. Retailers such as Best Buy have devoted a large portion of their showroom to the latest digital technology: computers, camcorders, digital cameras, PDAs, mp3 players, and much more. In addition to being a thoroughly stocked retailer, Best Buy also maintains a huge online presence at `www.bestbuy.com` (see Figure 22-3). You can browse their entire inventory online, order products, and check local inventory.

www.it-ebooks.info



Figure 22-3: You can find a wide variety of technology at bestbuy.com.

In addition to Best Buy, two other technology-based superstores can be found online: Comp USA (`www.compusa.com`) and Circuit City (`www.circuitcity.com`). Circuit City features detailed product descriptions, as well as product reviews on its Web site. The Comp USA site gives you the option of searching inventory in local stores as well as ordering online for store pickup (see Figure 22-4).

Online computer stores

Most computer manufacturers have their own Web sites. You can browse through their pages, read model specifications, and even order a system online. Dell revolutionized computer buying on the Internet. In fact, Dell computers can only be purchased through the Internet. When you visit the Dell Web site (see Figure 22-5), you build your system online and then place the order.

Service sites

The vast majority of people consider e-commerce to be the sale of some sort of product. However, the trend of the new economy—even before the advent of the Internet—has been largely toward services and information. In this regard you'll find Web sites where you can purchase insurance, refinance your home, plan

www.it-ebooks.info



Figure 22-4: You can reserve items for in-store pickup from the Comp USA Web site.

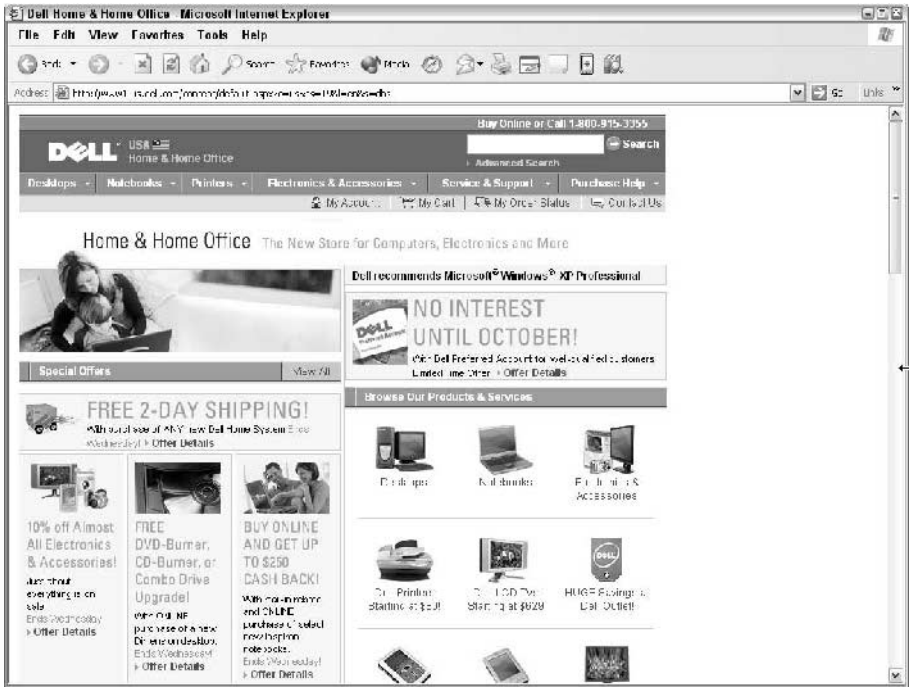


Figure 22-5: You can custom build your computer at the Dell Web site.

www.it-ebooks.info

vacations, and so on. Expedia.com is an excellent example of this type of service (see Figure 22-6). From the company's Web site, you can check airfares and hotel accommodations as well as rent cars.

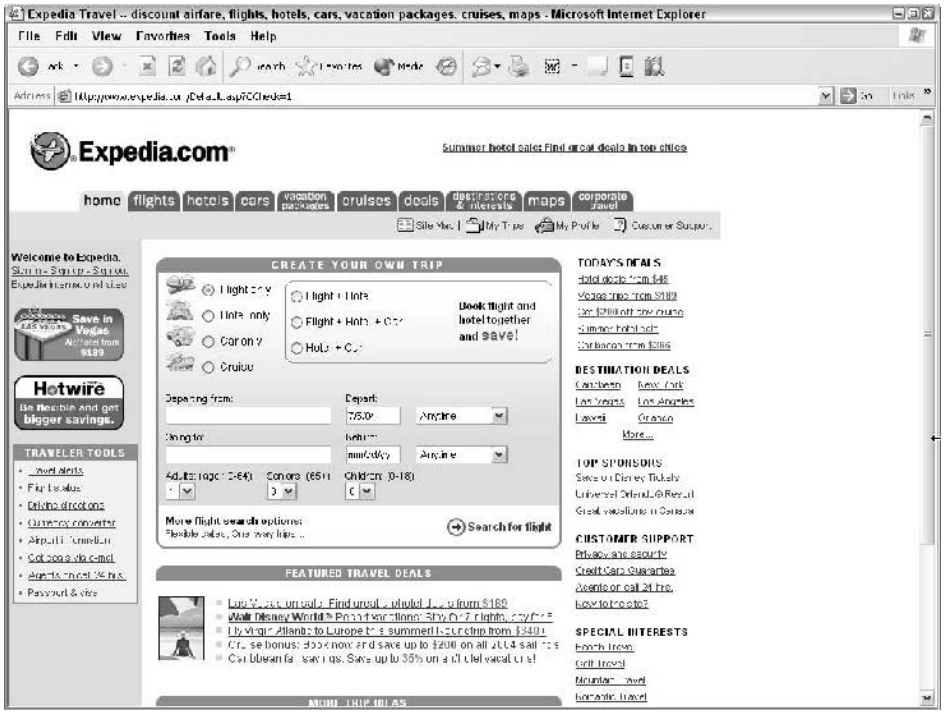


Figure 22-6: Expedia.com is a good example of a service-based Web site.

Some common examples of service-based sites are:

- ✦ Travel agencies
- ✦ Stock trading firms
- ✦ Insurance agencies
- ✦ Banks
- ✦ Auction houses
- ✦ Mortgage companies

Service sites can offer visitors the opportunity to use—and purchase—the services on the spot, or they can be nothing but glorified advertisements for the brick-and-mortar version of the business.

Shopping malls

Running an online shopping mall requires an entirely different approach than opening any kind of store. If you're going to open a mall, you're not in the retail

www.it-ebooks.info

business, you're in the province of property management instead—virtual property, that is. In that case, you're in the B2B (business to business) market instead of the B2C (business to consumer) line. Shopping malls such as ExciteStores (`shopping.excite.com`) provide a variety of services to their merchants, ranging from Web design assistance to payment processing (see Figure 22-7).

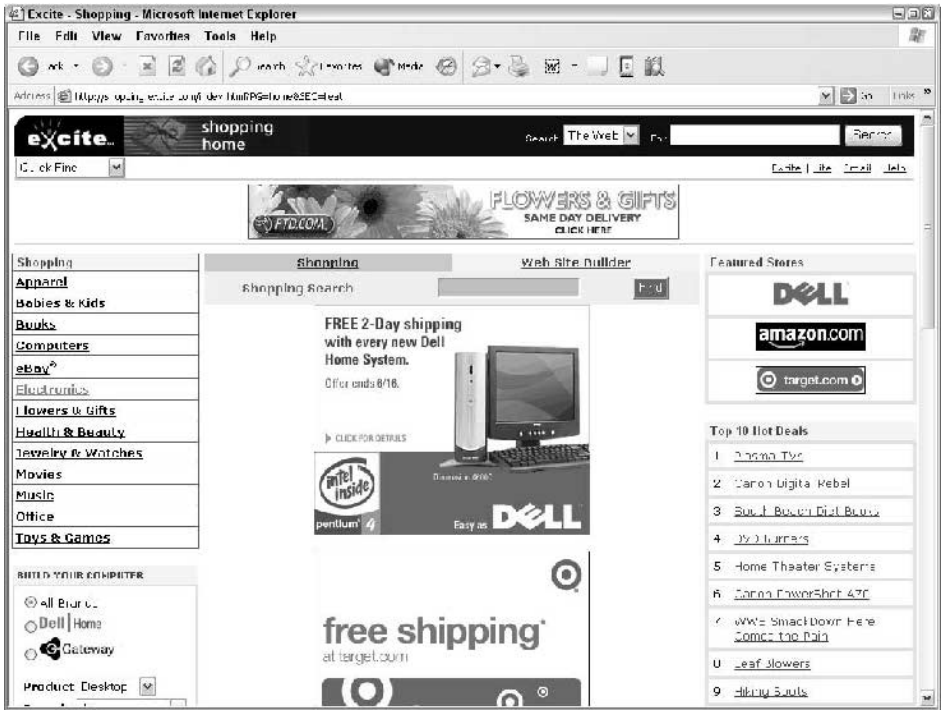


Figure 22-7: ExciteStores is a major online shopping mall.

The majority of online shopping malls, however, are really not what you think they are. They are simply sites whose Webmasters have signed up for a vast number of affiliate programs. The "stores" in them are nothing but links to the companies they're affiliated with.

Setting Up Shopping Carts

A *shopping cart* is an interface that enables your customers to choose which products they want to buy. The term is a good one because the analogy to an actual physical shopping cart is quite accurate. You wander up and down the aisles of a physical store, pick up items from the shelves, and drop them into a real shopping cart; then you take the cart to a checkout line and buy the items. Along the way, you may change your mind and remove some items from your cart.

Shopping cart programs work just like that, except that the purchaser is wandering around a Web site instead of a brick-and-mortar store. The shopping cart software

www.it-ebooks.info

uses a combination of Web page forms and a database to keep track of the items a site visitor wants.



See Chapter 15 for information on how forms work.

After you choose the items you want to purchase, you proceed to the checkout. Most online shopping carts use some kind of secure server. This assures the purchaser that the credit card information they're entering online cannot be intercepted by a third party. Other online retailers use services such as PayPal (`www.paypal.com`). This cuts down on the retailer's investment, as they don't need to be hosted by a secure server, nor do they need to deal with a bank. PayPal is the secure server, and they take a cut of the transaction for their services.

Addressing concerns about security

The heart of any transaction on the World Wide Web is security. Without that, your visitors won't want to purchase your product or service. The security is provided, appropriately enough, by a *secure server*, which is simply a Web server that is capable of sending and receiving encrypted information back and forth between itself and a Web browser.

If the idea of *digital certificates*—a complex creation of coded information that identifies one entity as opposed to another—is a little intimidating to you, you'll be happy to know that you don't have to set it up yourself. VeriSign is the best-known issuers of digital certificates (see Figure 22-8). DigiSign is another issuer of digital certificates. However, you don't have to arrange to set up your own digital certificate. That's one of those techie details that you can leave up to your Web hosting service. If your provider can't set it up for you, get a new one.



Just because you have a secure server doesn't mean a particular Web page is secure. The page must be saved with an `.shtml` extension to be secure.

Joining mall-provided programs

The simplest way to get involved in e-commerce is to join an existing online shopping mall such as Yahoo! Store (see Figure 22-9). Other online shopping malls are Jos. A. Bank (men's clothes), Zappos (shoes), and Lands' End (men's and women's clothing). Each one will have its own software already in place, and all you have to do is upload your own Web pages to be in business. Of course, the mall will want something for its troubles, which means that it will cost you.

However, an online mall receives a lot of traffic, traffic you would have to generate if you set up your online store through your own Web site. Generating Web traffic is an art that online shopping malls have down to a science. If you decide to set up shop in an online shopping mall, you can be assured of getting some traffic.

www.it-ebooks.info

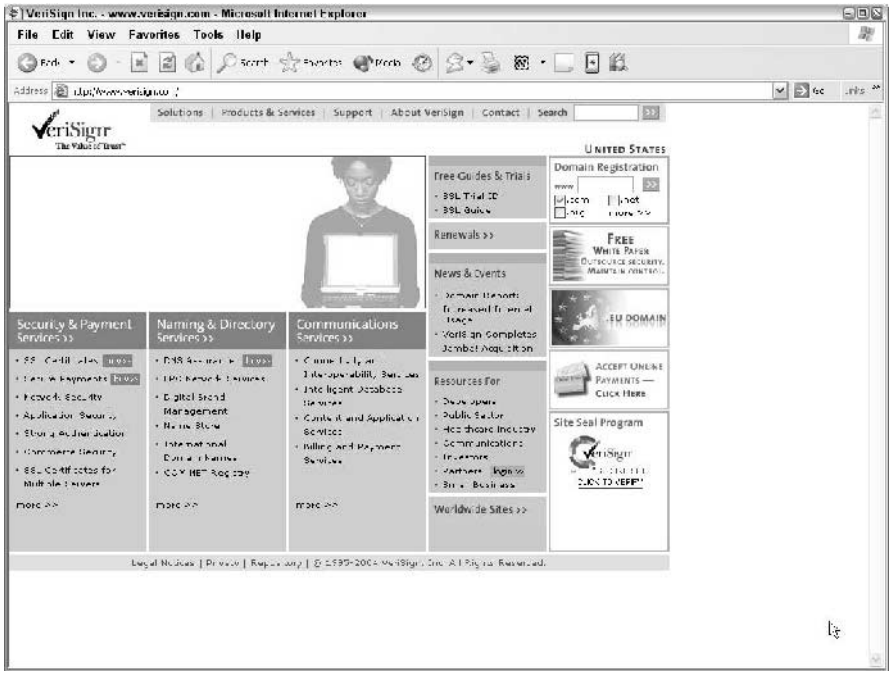


Figure 22-8: Verisign takes care of most of the Web's digital certificates.



Figure 22-9: Yahoo! Store is one of the fastest ways to get an online store up and running.

www.it-ebooks.info

Check out the details of the agreement carefully. Some malls charge a percentage of sales, others a regular monthly rate. Which one you go for is largely a matter of personal preference, but make sure that you understand exactly what your own obligations are. If you pay a percentage, for example, is there a minimum amount per month? If so, you can find yourself with no sales one month, yet you still owe money for your online presence.

Table 22-1 provides the Internet addresses of several shopping malls.

Table 22-1
Online Shopping Malls

Mall	*URL*
Award Winning Mall of Treasures	malloftreasures.com
ExciteStores	shopping.excite.com
ishop	ishop.co.uk/index.shtml
MSN	shopping.msn.com/
Yahoo! Store	store.yahoo.com

Getting commercial programs

If you do want to go it alone and set up your own free-standing Web site, you still need a shopping cart. As you might expect, this is a growing market, and the more people become used to dealing with e-commerce, the more choices you're going to have.

Many Web hosting companies can set you up with a shopping cart program. The Web hosting business has become so competitive that companies need to offer additional services to stay in the marketplace as shown in Figure 22-10.

Or if you prefer, you can get software to set up an online shopping cart at your existing Web site. Table 22-2 provides the URLs of companies that provide shopping cart software.



Some of the companies also provide e-commerce Web hosting.

Using open source programs

Open source software is freely available and often comparable to—or even superior to—similar commercially available programs. Most often, open source programs run on Unix systems and are particularly common for Linux, the Unix-like operating system that is, itself, an open source project (see Figure 22-11).

www.it-ebooks.info

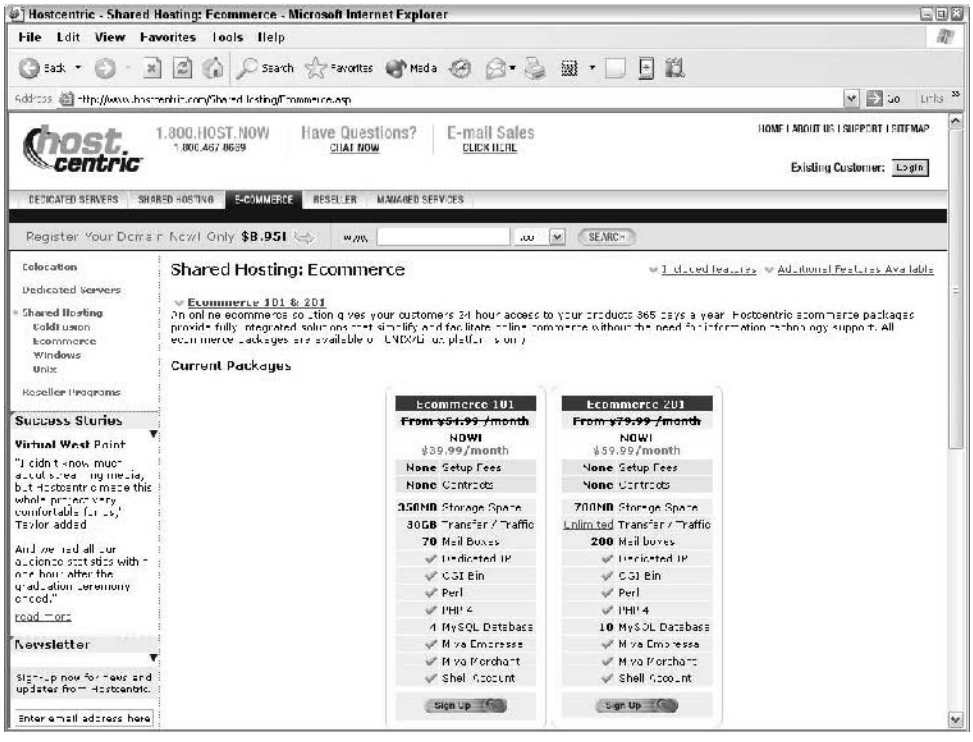


Figure 22-10: Web hosting companies can provide you with an online shopping cart system.

Table 22-2
Commercial Shopping Carts

Software	*URL*
Americart	www.cartserver.com/americart
Electrinic Merchant Systems	www.emscorporate.com
Hypermart	hypermart.net
In The Q	intheq.com
MerchandiZer eCommerce	merchandizer.com
Miva	miva.com/products
ShopCart	shopcart.com
ShopFactory	shopfactory.com

www.it-ebooks.info

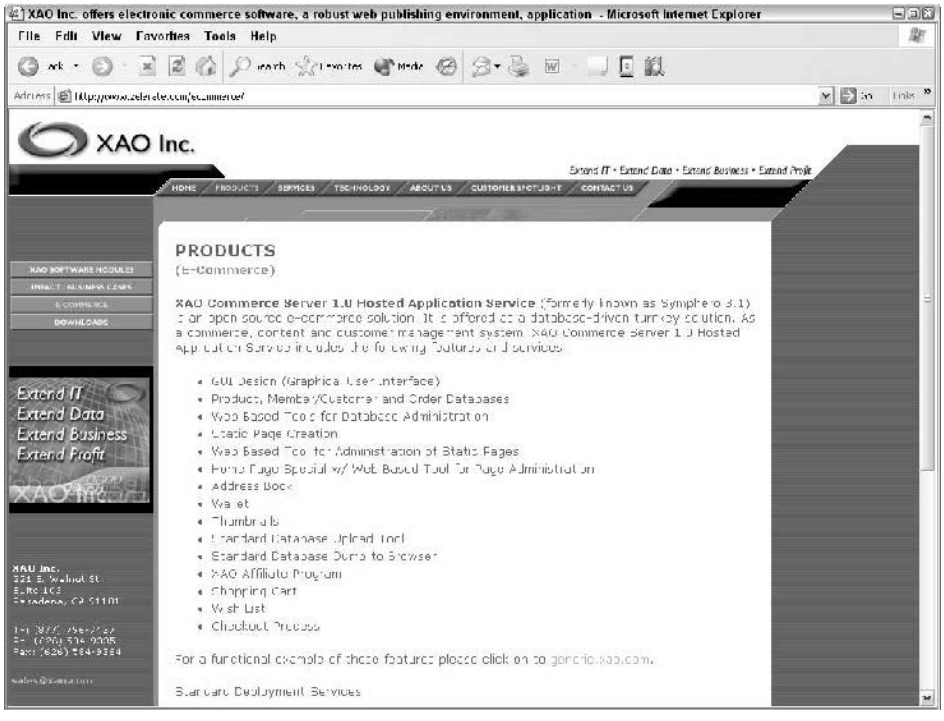


Figure 22-11: XAO Commerce Server from Zelerate is a fine open source e-commerce program.

Table 22-3 gives the addresses of some Web sites where you can find open source shopping carts.

Table 22-3
Open Source Shopping Carts

Software	*URL*
FishCart SQL	fishcart.org
IMS First Shopping Cart	1stcart.net
Open Source Shopping Cart	javaboutique.internet.com/articles/shoppingcart
Zelerate	www.zelerate.org/html/eng/home.shtml

Taking Payment

The obvious difference between e-commerce and a brick-and-mortar store is that you cannot come face to face with your customers. There's just no way in the world that they can hand you cash for what they're buying so you have to have some way

www.it-ebooks.info

of accepting non-cash payments. There are three ways you can get paid: credit and debit cards, checks, and e-cash.

Credit and debit cards

The single most common way of purchasing on the World Wide Web is with credit cards (see Figure 22-12). Although debit cards—the kind used to access ATMs—are gaining popularity as well, they will probably never surpass the tried-and-true standard. No matter what kind of Web space provision you decide on, the odds are pretty good that the Web hosting service provides credit card acceptance. It's only the purely tech providers who are missing out on the opportunity to grab a few extra bucks by adding basic e-commerce services to their roster.

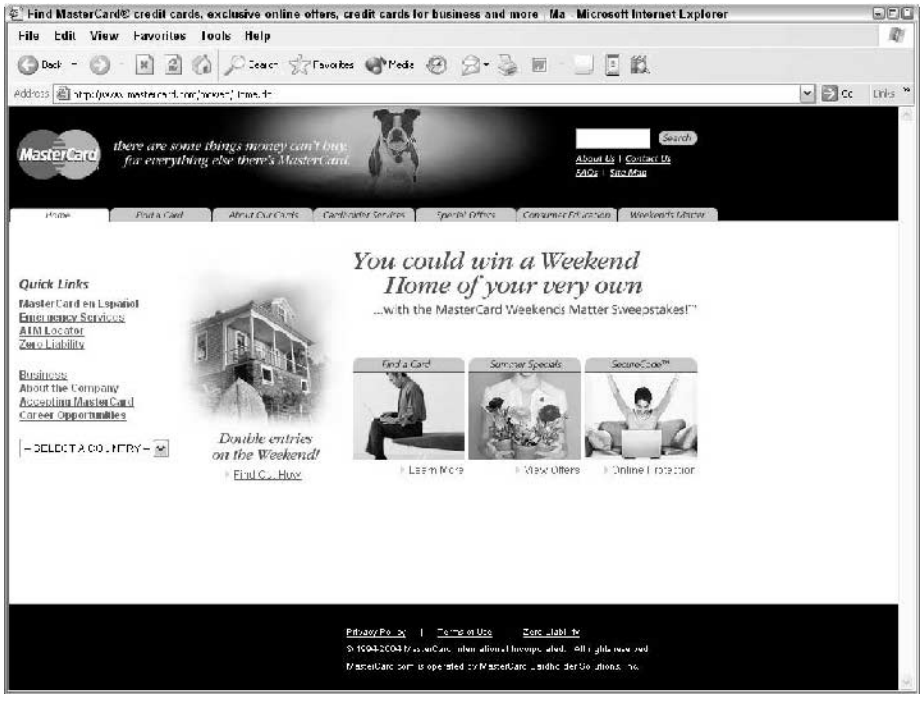


Figure 22-12: Credit cards are a necessity for any online business.

If, however, your Web hosting service doesn't provide credit card acceptance, all you have to do is go to the source. Table 22-4 lists the URLs for the major credit card companies.

Checks

You can accept checks for your online business in two different ways. The first, of course, is the old-fashioned way: someone simply prints out an order form, fills it in, and mails it along with a paper check. The second possibility is for you to print the

www.it-ebooks.info

check and deposit it just as if it had been sent in the mail. Yes, it's legal—as long as you have the buyer's permission, that is.

Table 22-4
Credit Card Companies

Company	*URL*
American Express	americanexpress.com
Discover Card	discovercard.com
MasterCard International	mastercard.com
VISA International	visa.com

If you're not familiar with this approach, take a look at the bottom of the next TV ad that's trying to sell you some fabulous gadget you just can't live without. Somewhere in the payment details, it'll doubtless say that you can pay with a check over the telephone. All you have to do is call them up and give them the information on the bottom of your check. This identifies two important things—your bank and the number of your checking account. Armed with that information, a check called a *paper draft* can be created. Checker is one company that offers software you can use to create paper drafts (see Figure 22-13). Instant Check by EasyDesk Software and Smart Checker are other sources for paper draft software.

This may sound scary to you—what's to keep them from whipping up as many checks as they want to and emptying your bank account? That's a good question, and you should carefully consider to whom you give that information. But everyone who ever wrote a check for anything already gave that information to everyone they ever gave or sent a check to. Remember, it's right there for anyone to read. You may want to point this out to your customers if they seem antsy about using the technique.

What's the real value of this capability? Simple—people who don't have credit cards can place an online order just as easily as if they had a wallet full of them.

Table 22-5 lists sources for paper draft software.

E-cash

E-cash is one of those ideas that sounds so good, you wonder why it hasn't taken off. In theory, it will one day be fully integrated with the online world. Some day, the visionaries say, your Web browser will interface with your bank account, and everything all along the line will be protected by the same kind of digital certificates that make secure Web servers work. When that day comes, all you will have to do to

www.it-ebooks.info

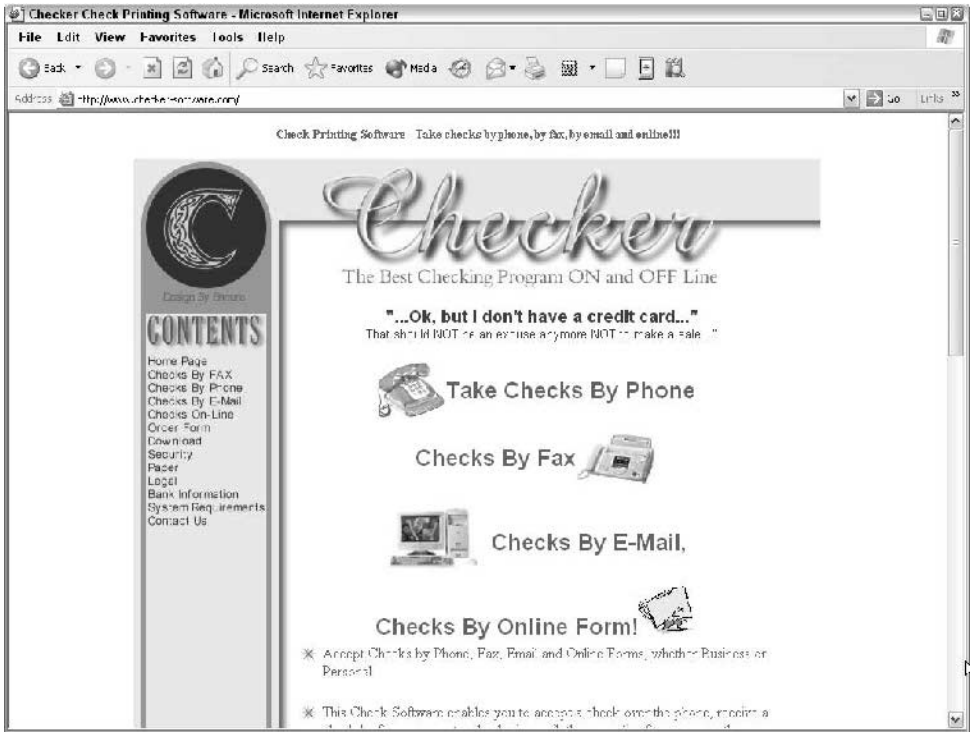


Figure 22-13: Checker enables you to create paper drafts that work just like regular checks.

Table 22-5
Paper Draft Programs

Program	*URL*
Checks By Fax	www.checksbyfax.net
Checker	www.checker-software.com
Smart-Checker	www.web-store.net/checker

buy anything on the Web will be to say you want it. The money will be instantly debited, and the merchant will be instantly paid.

The concept, however, is far from a reality for a variety of reasons. First of all, nobody can agree on an e-cash standard. In addition, the concept of digital certificates is difficult to get across to the public. Remember, many of the folks who will presumably one day use e-cash are still having trouble navigating a Web browser to the site they want.

www.it-ebooks.info

And the same fundamental human problem exists with e-cash as with debit cards. When people use credit cards, they can wait to pay them off. With e-cash, they lose their money on the spot. We're not holding our breath on this one.

Setting up an affiliate store

If you own a Web site that provides information about a specific topic, you're drawing a steady stream of traffic. An excellent way to augment your income is setting up an affiliate store that sells products related to your Web site. When you set up an affiliate store, you create links to products at other Web sites. When a visitor from your site purchases a product, you get a percentage of the sale.

The amount of money you can earn depends on the exact arrangement. Some affiliate programs track how many people visit them from your site and pay per click, just as with banner advertising. Others pay only if your referred visitor actually makes a purchase. Generally, a pay-per-click arrangement doesn't pay very much—pennies at the most. Commissions on purchases, however, can add up to a tidy sum.

How often you get paid also varies widely. Some companies pay monthly, but the vast majority pays quarterly, so you can only expect payment once every three months. And you may not receive a payment then either unless you pass the minimum threshold. Every such program has some minimum amount you must generate in order for them to bother with cutting and mailing a check. Usually, it's a pretty small amount such as $10 or $20. And, if you're not making that much, you need to take a good, hard look at either the program or your own site because something's wrong somewhere along the line.

Amazon (`www.amazon.com`) is an excellent example of online merchandise you can use for your affiliate store. To become an Amazon affiliate (see Figure 22-14), you fill out an online form. After your application is accepted, you have access to a section of Amazon's Web site where you set up your links, view traffic from your online store, and view your earning reports.

The best way to find out which affiliate program or programs are available is to check out some of the online guides that keep track of the current state of affairs in the affiliate world. Many of them have top ten lists that tell you which programs are paying the best. Some also have reviews and commentary by people who have used the programs. Most of them also make money by referring you, so be sure to take a good look at several of them and don't just sign up right away with whichever site says is best. Table 22-6 lists the URLs of several different affiliate program directory sites.

Riding Internet Buying Patterns

E-commerce began to be taken seriously in 1999, and all signs point to it being a major force in the economy of the twenty-first century. As surely as the sun rises and sets, as surely as spring follows winter, the calendar sets the peaks and valleys of Internet commerce.

www.it-ebooks.info

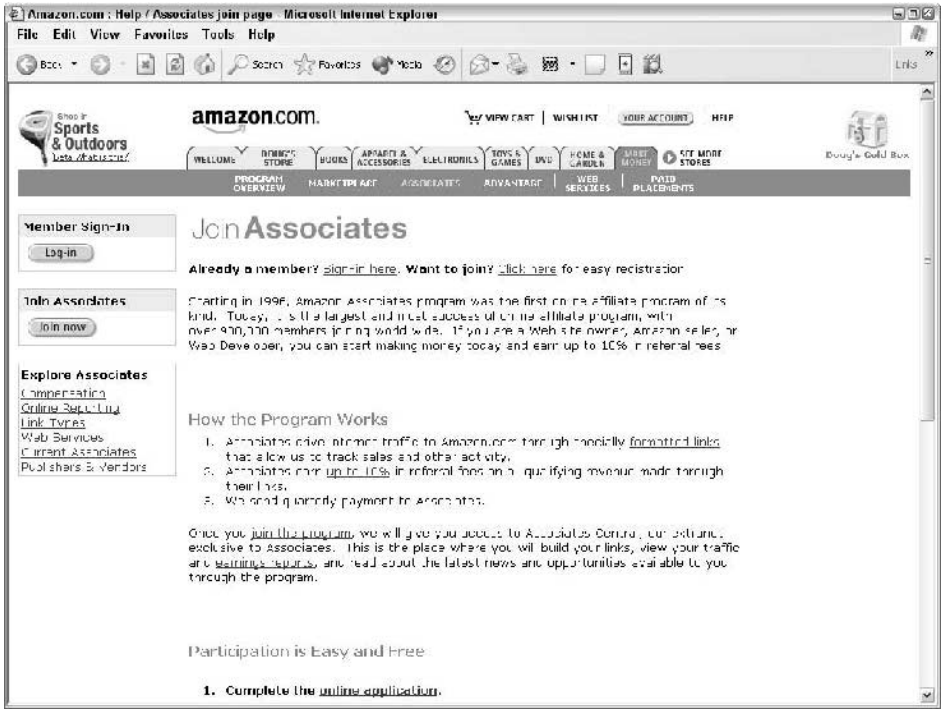


Figure 22-14: You can become an Amazon Associate by filling out an online form.

Table 22-6
Affiliate Guides

Company	*URL*
AffiliateMatch.com	affiliatematch.com
AffiliateWorld.com	affiliateworld.com
ClickQuick.com	clickquick.com
ClicksLink.com	clickslink.com
Commission Junction	cj.com
LinkShare	linkshare.com
SimplytheBest	simplythebest.net/affiliates
WebWorkers	referralincome.com

www.it-ebooks.info

Without question, the biggest buying spree on the Internet is during the December holiday season. A great many people leave their holiday shopping to the last minute, and the World Wide Web is the best place for them to get their shopping down. The increase in e-commerce makes it easy to find almost any product online. Buyers also find many unique gifts they could not purchase locally.

ClickZ Network (`www.clickz.com`) is one of the many online marketing firms dealing with e-commerce and has observed several interesting patterns developing over the past couple of years (see Figure 22-15).

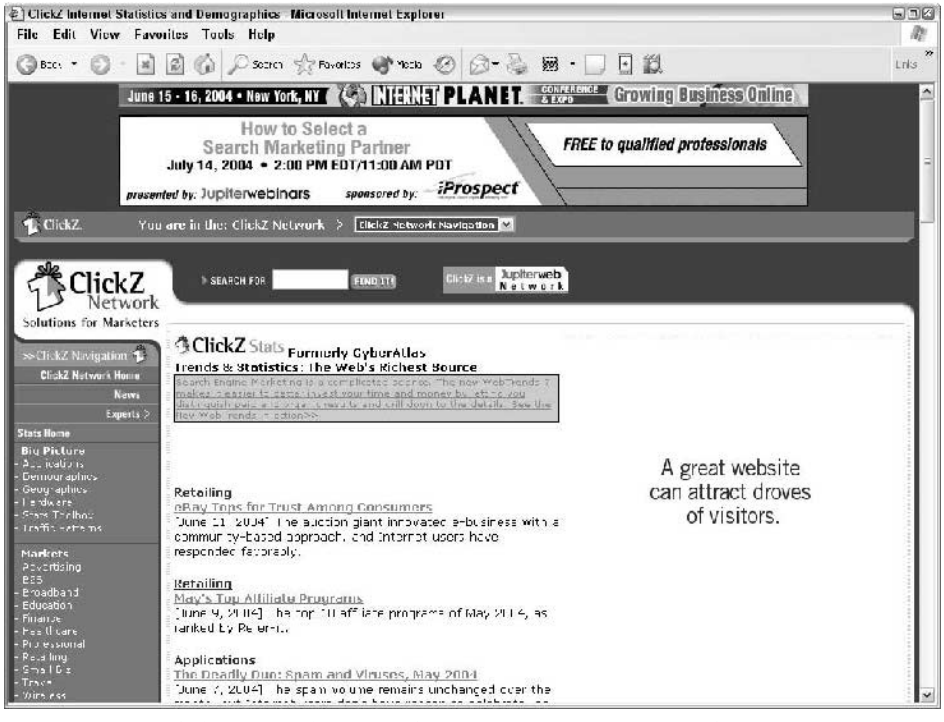


Figure 22-15: ClickZ Network is an online source of e-commerce data.

Table 22-7 shows ClickZ Networks analysis of e-commerce purchasing by quarter since 1999. Notice that online sales have steadily increased and the busiest buying period is the 4th quarter when people do their holiday shopping.

It's important, however, to remember that nothing changes faster than the Internet. If you're going to be in business on it, you have to make it your business to keep up to date on the latest facts and figures.

www.it-ebooks.info

Table 22-7
Online Purchasing Trends

Sales Period	*Total Retail Sales (in Billions)*	*e-Commerce Sales (in Billions)*	*% of Retail Sales*
Q4 1999	$787,212	$5,335	0.7%
Q1 2000	$714,561	$5,663	0.8%
Q2 2000	$774,677	$6,185	0.8%
Q3 2000	$768,139	$7,009	0.9%
Q4 2000	$812,809	$9,143	1.1%
Q1 2001	$724,731	$7,893	1.1%
Q2 2001	$802,662	$7,794	1.0%
Q3 2001	$779,096	$7,821	1.0%
Q4 2001	$850,265	$10,755	1.3%
Q1 2002	$738,185	$9,549	1.3%
Q2 2002	$814,626	$10,005	1.2%
Q3 2002	$818,061	$10,734	1.3%
Q4 2002	$859,250	$13,999	1.6%
Q1 2003	$767,433	$12,115	1.6%
Q2 2003	$852,760	$12,718	1.5%
Q3 2003	$867,242	$13,651	1.6%
Q4 2003	$912,109	$17,512	1.9%

Summary

Many people see e-commerce as the most important aspect of the World Wide Web. It promises to revolutionize marketing. This chapter introduced you to the following concepts and topics:

- ✦ The type of product or service you offer will determine the kind of Web site you need to present it.
- ✦ Shopping carts are software designed to enable your site's visitors to browse and purchase whatever you're selling.
- ✦ Web hosting services can provide e-commerce solutions.

www.it-ebooks.info

- ✦ Affiliate programs make it possible to sell products online without maintaining an inventory.
- ✦ Credit cards are the most common form of payment on the World Wide Web.
- ✦ Paper drafts enable you to take checks instead of credit cards via your Web site.
- ✦ The winter holiday season is the biggest spending season on the Web.

✦ ✦ ✦

www.it-ebooks.info

CHAPTER 23

Selling on eBay

In This Chapter

- Setting up an eBay account
- Inserting an item into eBay
- Using the Buy Now option
- Setting up a Dutch auction
- Creating an effective eBay auction
- Setting up an eBay store
- Accepting payment

Whether you need to sell a few tech items that have been gathering dust in your closet, or you need to augment your online store, eBay is the place to go. Many people consider eBay the world's largest online flea market. At eBay you'll find everything from the most obscure gadget, to commonly used items such as computers, camera lenses and so on. And eBay receives thousands of visitors per hour, from the four corners of the globe. What is construed as trash by some, is treasure to others. Therefore, you can sell just about anything on eBay. Figure 23-1 shows the eBay home page, and will give you an idea of the wide variety of merchandise you can expect to find, and sell on eBay. In this chapter, you'll learn the ins and outs of selling on eBay.



All prices quoted in this chapter are current as of the date this chapter was written. For the most up-to-date prices, refer to `www.eBay.com`.

Working with eBay

When you decide to sell something on eBay, you must set up an account. Setting up the account is free. You can set up an eBay account online in only a few minutes. After you set up an eBay account, you can put up objects for auction. eBay charges an insertion fee for each item you put up for auction. You'll also be required to pay a fee when your auction ends. The upcoming sections will give you all the information you need to know about setting up an eBay account, as well as the fees you'll be charged for selling on eBay. However, the Internet is in a constant state of flux. If you have any questions or concerns that are not covered in this chapter, you can refer to eBay's extensive library of online help by clicking the Help link you'll find at the top of many eBay pages (see Figure 23-2).



You can contact eBay directly through their Contact Us page. However, eBay does not respond promptly to general questions.

www.it-ebooks.info

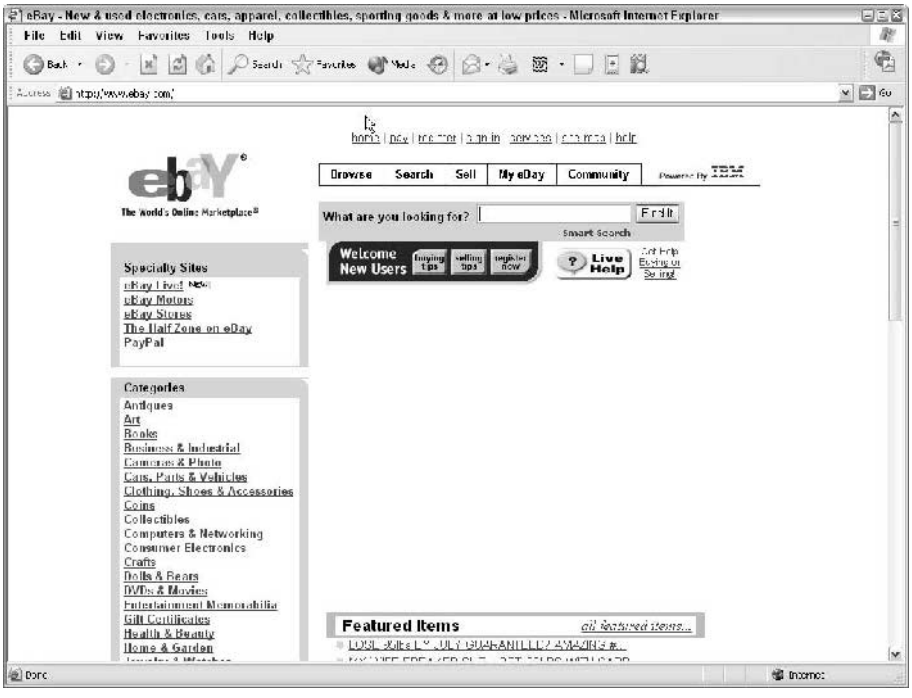


Figure 23-1: Many people consider eBay the world's largest online flea market.

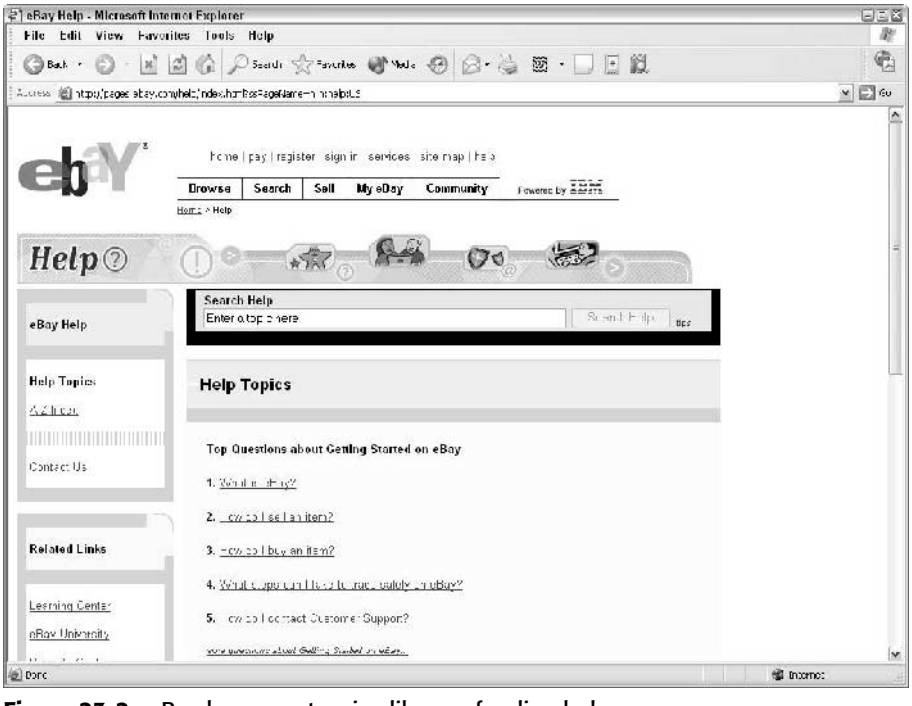


Figure 23-2: eBay has an extensive library of online help.

www.it-ebooks.info

Setting up an eBay account

To begin selling on eBay, you'll have to register as an eBay user. After you register, you can set up a seller's account by filling out the online questionnaire. To set up an eBay account:

1. Navigate to the eBay home page at `www.eBay.com`
2. Click the Sell link at the top of the home page.
3. Sign in using your eBay User ID and Password. If you don't have an eBay User ID and Password, click the Register button to set one up. After signing in, the first page of the eBay Seller's Account questionnaire appears.
4. Verify your information. The Verify page lists the information you supplied when you registered with eBay.
5. After verifying the information, click the Continue button to display the Provide Credit Card Information page (see Figure 23-3).

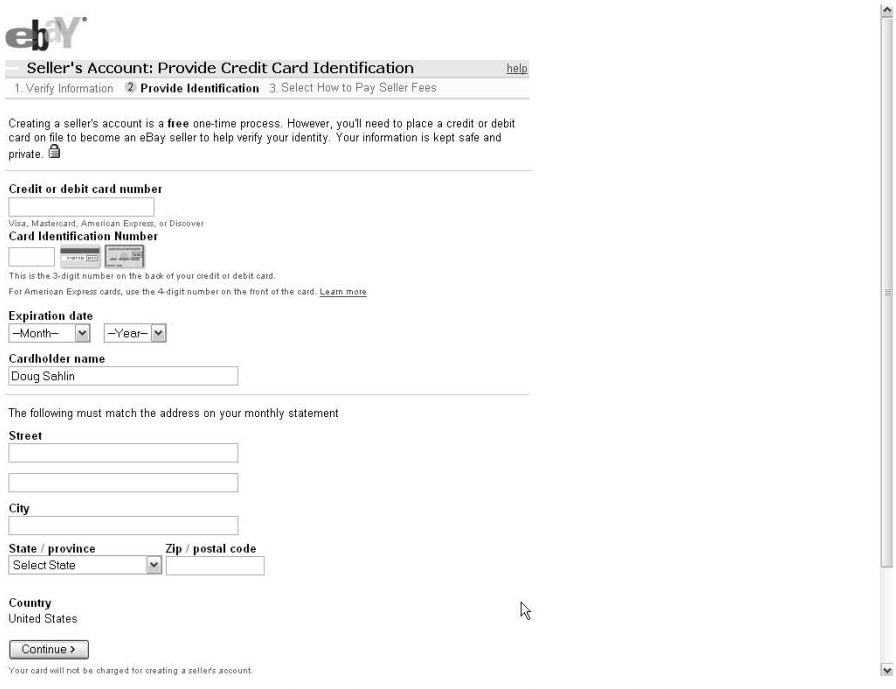


Figure 23-3: Providing credit card information is the second phase of setting up a seller account.



You will not be charged for setting up an eBay seller account. The credit card information is used only when you incur selling fees. All information you provide is secure.

6. Click Continue to display the Provide Check Identification page. On this page, you're required to enter your checking account information, including routing

www.it-ebooks.info

number and checking account number. This information is required by eBay as an additional method to verify your identification and insure overall site security. eBay will not debit your account for fees unless you instruct them to do so.

7. After providing your banking information, click Continue to display the Select How to Pay Seller Fees page.
8. Click the radio button to specify how you will pay eBay selling fees. Note that your banking account is selected by default.
9. Click the Continue button to complete the process. eBay displays the Sell Your Item page (see Figure 23-4) which is the first step to selling an item online.



Figure 23-4: After setting up an account, you can begin selling online at eBay.

Fees for selling on eBay

After you set up an account on eBay, you can sell items on eBay. Detailed information on how to set up an eBay auction will be covered later in this chapter. As a seller, you are liable for certain fees for eBay services. Table 23-1 lists the fees for selling items on eBay as of the time of writing.

As you can see, the fee for putting an item up for auction on eBay is rather minimal. In fact it's a downright bargain when you consider the amount of traffic the eBay site receives.

www.it-ebooks.info

Table 23-1
Insertion Fees for Selling on eBay

Starting Price, Opening Value, or Reserve Price	*Selling Fee*
$0.01–$0.99	$0.30
$1.00–$9.99	$0.35
$10.00–$24.99	$0.60
$25.00–$49.99	$1.20
$50.00–$199.99	$2.40
$200.00–$499.99	$3.60
$500.00 and up	$4.80



Don't sell any item with an opening price of less than $10.00. Remember, you'll have shipping fees, plus e-mail correspondence with potential buyers as well as the winner. Your time is worth money. In this regard, you should set your opening bid for a minimum of $10.00. Don't sell yourself, or your item short. If the item is worth less than $10.00, save it for a spring or fall garage sale with any other merchandise you can't list on eBay for $10.00 or more.

There are additional fees for selling on eBay. For example, if you auction real estate on eBay, there are additional fees for the listing type and duration. In addition, there are additional fees for selling vehicles on eBay. You'll also incur additional fees when you sell an item with a Reserve (the minimum price you're willing to accept for the item). These fees are $1.00 for an item with a Reserve price of less than $49.99, $2.00 for an item with a Reserve price of less than $199.99, and 1 percent of any Reserve price higher than $200.00 with a ceiling of $100.00.

When your auction ends, you'll also incur a Final Value fee based on the final selling price of your item. Table 23-2 lists the Final Value fees you'll incur when an item is awarded to a bidder.

Table 23-2
Final Value Fees

Final Selling Price	*Final Value Fee*
$0.00 to $25.00	5.25% of the closing value
$25.00 to $1000.00	5.25 % of the first $25.00 ($1.31), plus 2.75% of the remaining final closing value ($25.01 to $1000.00)
Over $1000.00	5.25 % of the first $25.00 ($1.31), plus 2.75% of the remaining final closing value ($25.01 to $1000.00), plus 1.5% of the remaining final closing value ($1000.01 to final closing value)

www.it-ebooks.info



The Final Value fee is due whether the seller pays you or not. It is your responsibility to collect the money from the seller.

Setting Up an eBay Auction

After setting up an eBay account, you can begin selling items online. When you sell items online, you must choose whether to sell the item as an online auction, sell the item at a fixed price, or sell the item through your eBay store. But before you actually sell the item online, you should consider the following:

- ✦ In which category will you list the item?
- ✦ By what title will you list the item?
- ✦ What description will you use to list the item?
- ✦ What photos will you use to help sell the item?
- ✦ How will you price your item?



The item you are selling must be legal. Prohibited, questionable, or infringing items may not be sold on eBay. For a complete list of prohibited sales items for eBay, refer to `http://pages.ebay.com/help/sell/item_allowed.html`.

To sell an item on eBay, follow these steps:

1. Navigate to `eBay.com`.
2. Click the Sell link to display the Sell Your Item: Sign In page.
3. Enter your eBay ID and Password, then click the Sign In button to display the Sell Your Item: Choose a Selling Format (see Figure 23-5).
4. Choose the desired selling format by clicking the applicable option button and then click the Continue button to display the Sell Your Item: Select Category page (see Figure 23-6).
5. Choose the desired selling category by clicking the applicable option button. Alternatively, you can enter a keyword in the Search field that applies to your item, and then choose the most likely category from the returned search items.



You can only select one selling category per auction. If the item falls into multiple categories, choose the most likely category. If in doubt, use the Search feature mentioned in step 5 and list the item in the category that has the top ranking for the keyword you think is most representative of the object you are selling.

6. Click the Continue button to reveal the Sell Your Item: Select Subcategory page. Depending on the subcategory you choose, you may be prompted to

www.it-ebooks.info

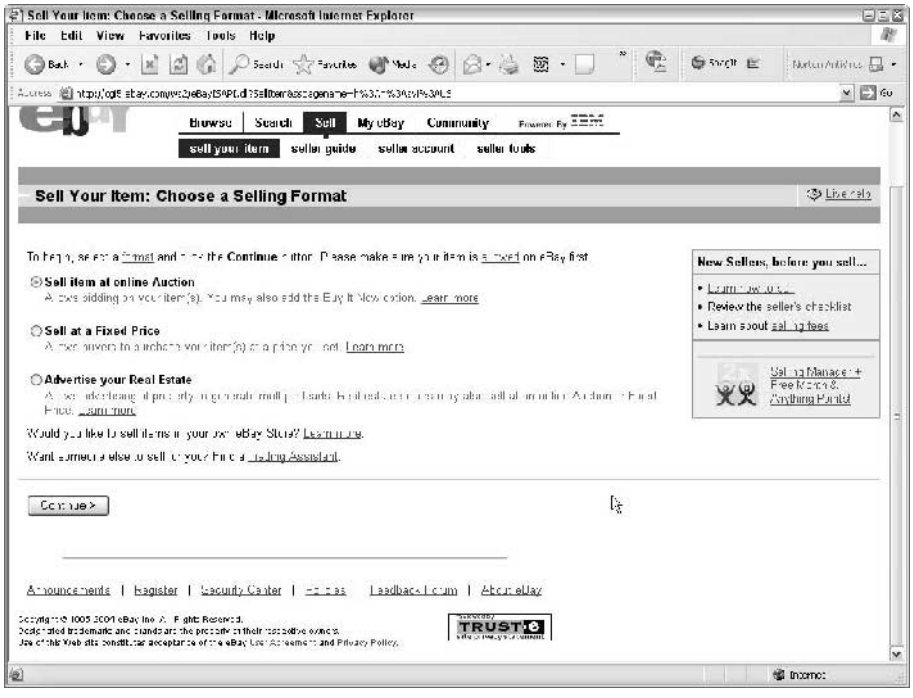


Figure 23-5: Choose a selling format.

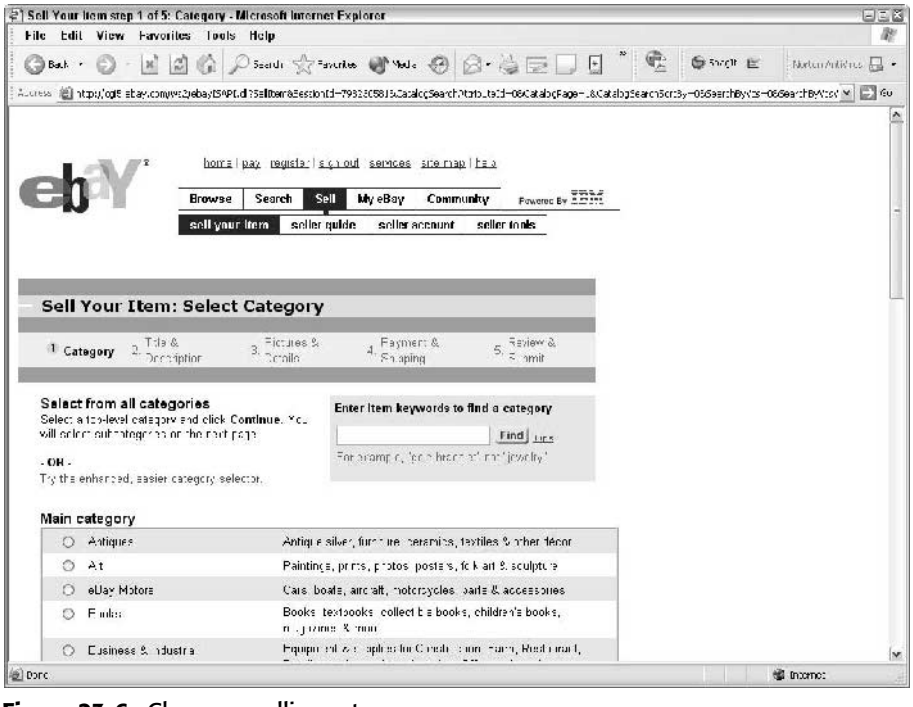


Figure 23-6: Choose a selling category.

www.it-ebooks.info

choose additional subcategories. If a subcategory has a line with a right-pointing arrow there are additional subcategories from which to choose (see Figure 23-7).

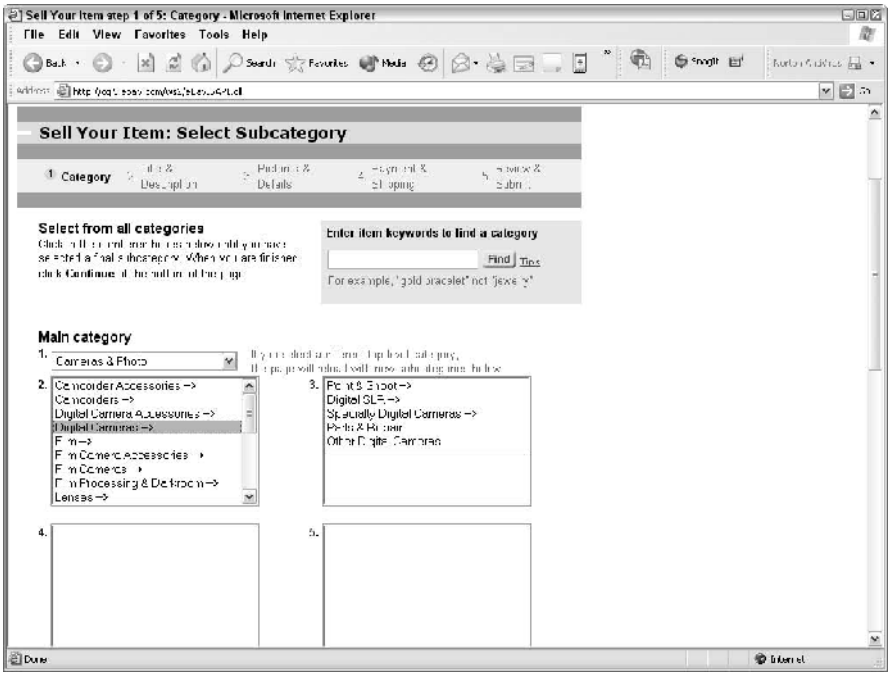


Figure 23-7: Choose a subcategory.

After selecting the main category and subcategories, the remaining subcategory boxes are filled with the message: Main Category Selected. Continue below.

7. Click the Continue button to display the Sell Your Item: Choose a Listing Option. You can choose to list your item with pre-filled item information, or list it the standard way (see Figure 23-8). If you choose the first option, you can choose listing options from the drop-down menus. The available options will pertain to the item you are selling.

8. Click Continue to display the Sell Your Item page where you have the option to select a model based on existing auctions. Alternatively, click the Continue button in the List the Standard Way section of the listing option page to display the Sell Your Item: Describe Your Item page.

9. Enter the information you think best describes the item you are putting up for sale in the Item Title field. Alternatively, you can enter additional information in the Item Specifics section of the page. If you want, for an additional $0.50 you can add a subtitle that gives potential buyers additional information about the object you are selling. This information is searchable using the advanced search options where potential buyers enter additional search information in the Item Description field.

www.it-ebooks.info

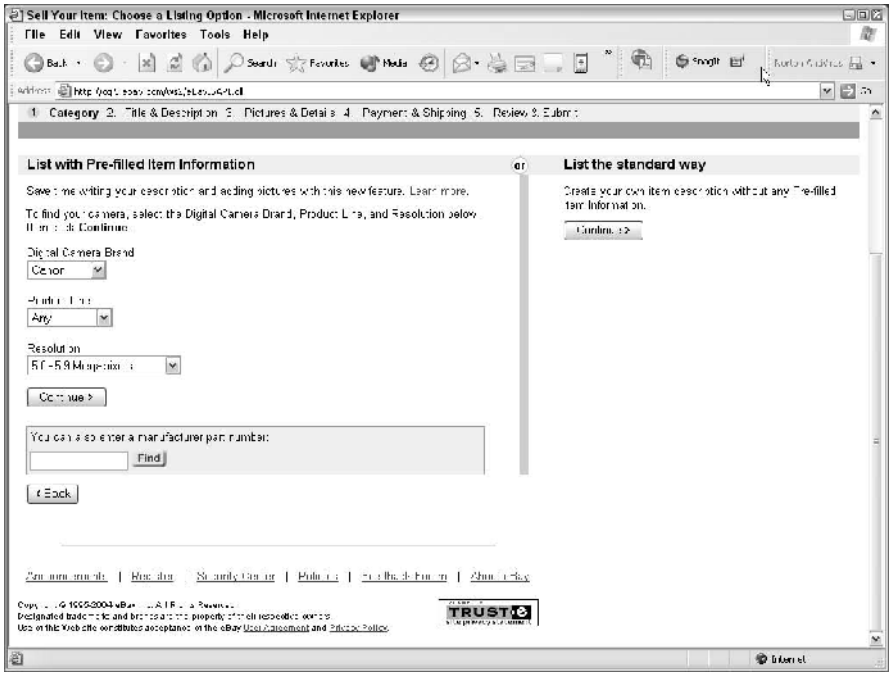


Figure 23-8: Choose a Listing option.



Scroll to the bottom of the page and you can choose the font name, size, and color for the item description from drop-down lists. You can also click the Inserts button to add additional inserts in your auction, such as Add To Favorites List, View Sellers Other Items and so on.

10. Click the Preview Description button to preview your description in another window. After previewing your description, close the window and then click the Continue button to reveal the Sell Your Item: Enter Picture and Item Details page (see Figure 23-9).

11. Enter the pertinent information about the item you are selling, such as the item pricing information, the auction duration, start time, the location from where you are selling the item. In this section you'll also choose options for the pictures you are including. You can use eBay as a host for your pictures and upload them with the auction, or click the Your Own Web Hosting tab, which enables you to enter the URL for each image you are uploading with the auction.



eBay Picture Services is the quick and easy way to go. Enter the path to each image you want to upload with your auction. The first picture is free and each additional picture is $.15. After submitting your item for sale, the pictures are automatically uploaded to eBay, and are linked to your auction.

12. (Optional) Choose additional options in the Picture option section of the Sell Your Item: Enter Picture and Item Details page. You can choose to supersize

www.it-ebooks.info

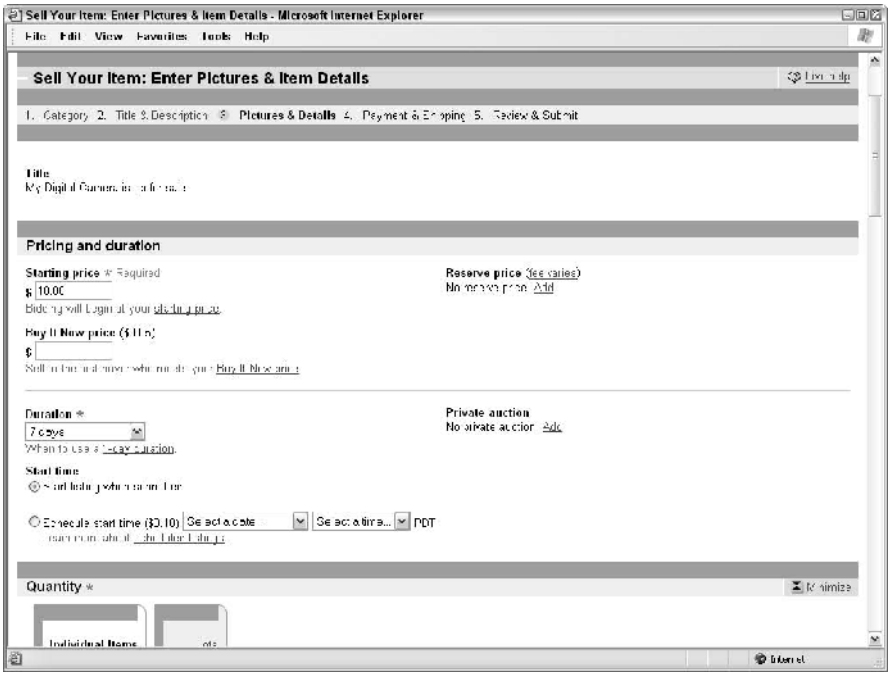


Figure 23-9: Enter picture and item details about the object you are selling.

your pictures to appear within a 500 × 375 pixel space on the auction page for an additional $0.75, choose to display your pictures as a slide show in a 400 × 300 pixel area for an additional $0.75, or choose to create a Picture Pack. A Picture Pack displays up to 6 pictures for $1.00, or 7 to 12 pictures for $1.50.

13. In the Listing Designer section of the Sell Your Item: Enter Picture and Item Details page, you can choose a theme and layout for your pictures from the drop-down lists. This service costs an additional $0.10.



After choosing a theme, you can preview your listing by clicking the Preview Listing link at the bottom of the Listing Designer window. If you don't like the layout of your pictures, you can remove them or choose a different design theme.

14. (Optional) Choose options for the Increase Your Visibility section of the Sell Your Item: Enter Picture and Item Details page (see Figure 23-10).
15. Click Continue to display the Sell Your Item: Shipping and Payment Details page (see Figure 23-11). In this section you can choose payment and shipping information, which we will discuss in detail Later in this chapter.
16. Click the Continue button to display the Sell Your Item: Review and Submit Listing page (see Figure 23-12). At this stage, you can review the listing, as it will appear on the eBay Web site. If something isn't right, you edit the information by choosing the edit button in the applicable section. The insertion fee for your auction is displayed near the bottom of the page.

www.it-ebooks.info

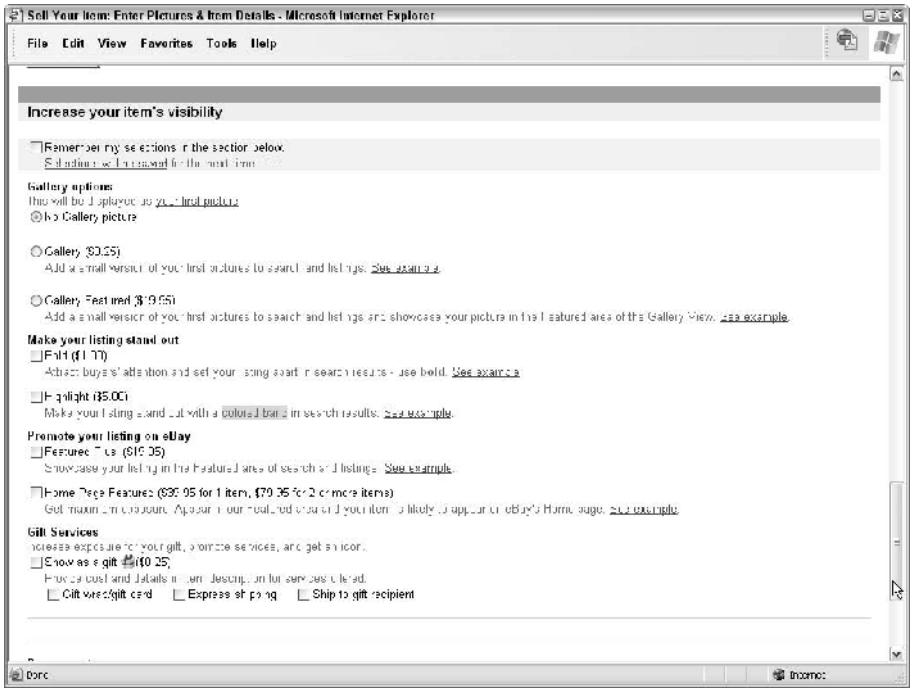


Figure 23-10: Choose options to increase your visibility.

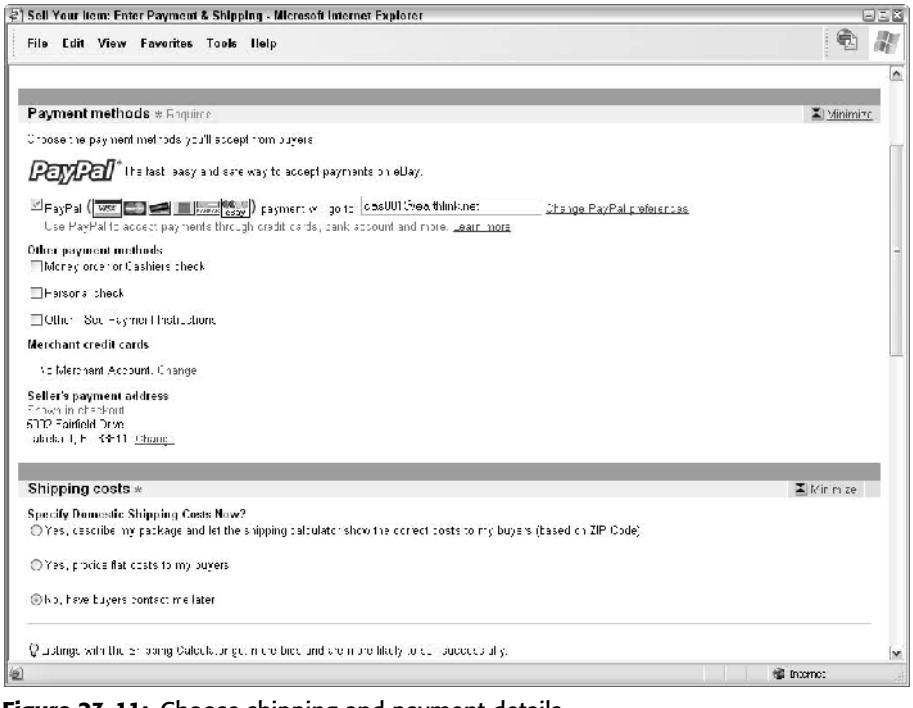


Figure 23-11: Choose shipping and payment details.

www.it-ebooks.info

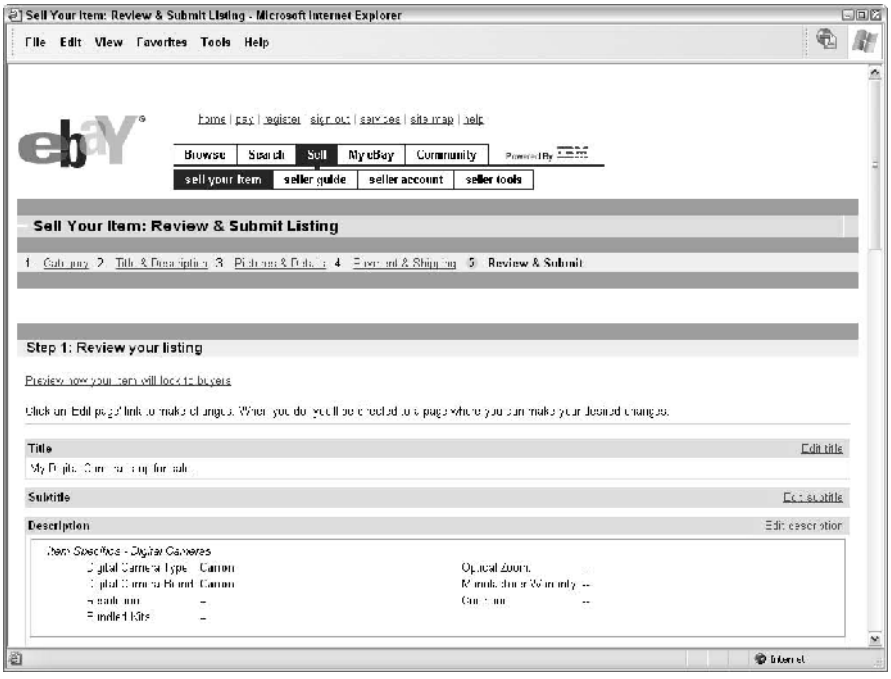


Figure 23-12: Review your listing.

17. If all is in order, click the Submit Listing button at the bottom of the Sell Your Item: Review and Submit Listing page. Your item is now listed for auction on eBay.

Using the Buy Now option

In the Pricing and Duration section of the Sell Your Item: Pictures and Item Details page, there is an option for entering a Buy Now price. This option costs you a mere $0.05 and could be the best nickel you've ever spent. When you enable the Buy Now option, you can set a price that you're willing to accept for your item. The Buy Now option gives the buyer the option of purchasing your item for the Buy Now price, before the end of the auction. This gives the buyer instant gratification. If your item is something a buyer cannot possibly live without, he or she can buy it immediately. If you've got a lot of bidders for your item, an anxious bidder is more likely to pay the Buy Now price than lose out on the bid, which ensures you of a sale at a higher price than the auction may have yielded. Of course, if the bidding is fast and furious, the Buy Now price may be too low. To be on the safe side, research similar items that have sold on eBay and use the ending auction price as the basis for your Buy Now price.

Setting up a Dutch (multiple item) auction

If you have a lot of items, say 10 widgets, you can combine all items in what is known as a Dutch auction. When you set up a Dutch auction, you set the beginning price for

www.it-ebooks.info

the items. A bidder enters a bid for the number of items he or she wants to buy. In a sample scenario with 10 widgets that you list for an opening bid of $9.99 you have the following bids at auction close:

- Bidder A enters a bid for $9.99 for six widgets.
- Bidder B enters a bid for $11.99 for six widgets.
- Bidder C enters a bid for $12.99 for two widgets.

The widgets are awarded as follows:

- Bidder A is awarded two widgets and pays a price of $9.99 per widget.
- Bidder B is awarded six widgets and pays a price of $11.99 per widget.
- Bidder C is awarded two widgets and pays a price of $12.99 per widget.

A Dutch auction is an efficient way of selling several identical items. For example, if you purchased 10 identical Barbie dolls at a flea market, you can list all of them at eBay using the Dutch auction feature.

Payment and Shipping Options

If your auction is successful, you've sold the item. But for all intents and purposes, a sale is never made until the item is paid for. You are responsible to eBay for the insertion and ending value fees, regardless of whether your buyer pays you or not. There are several ways you can take payment from a successful bidder that will be covered in upcoming sections.

Shipping is another concern. In order to establish yourself as a reputable eBay auctioneer, you should ship items promptly after payment has been received. There is also the matter of what to charge for shipping. You'll also have to concern yourself with properly packing the merchandise to thwart the many fingers and vehicles that will be handling the merchandise you ship.

Setting up a PayPal account

eBay recommends PayPal as a preferred method of payment. PayPal requires a considerable amount of information to set up an account, and a portion of your proceeds goes to PayPal for handling the payment. However, when a bidder pays using PayPal, you're assured that you'll get the payment and can promptly ship the item.

One of the options on the Sell Your Item: Enter Payment and Shipping page is to accept payment through PayPal. You can set up an account with PayPal (an eBay company), by following the prompts if you accept the default PayPal option. Alternatively, you can go to the PayPal Web site at `www.paypal.com` to set up your account (see Figure 23-13), or manage an existing account.

www.it-ebooks.info

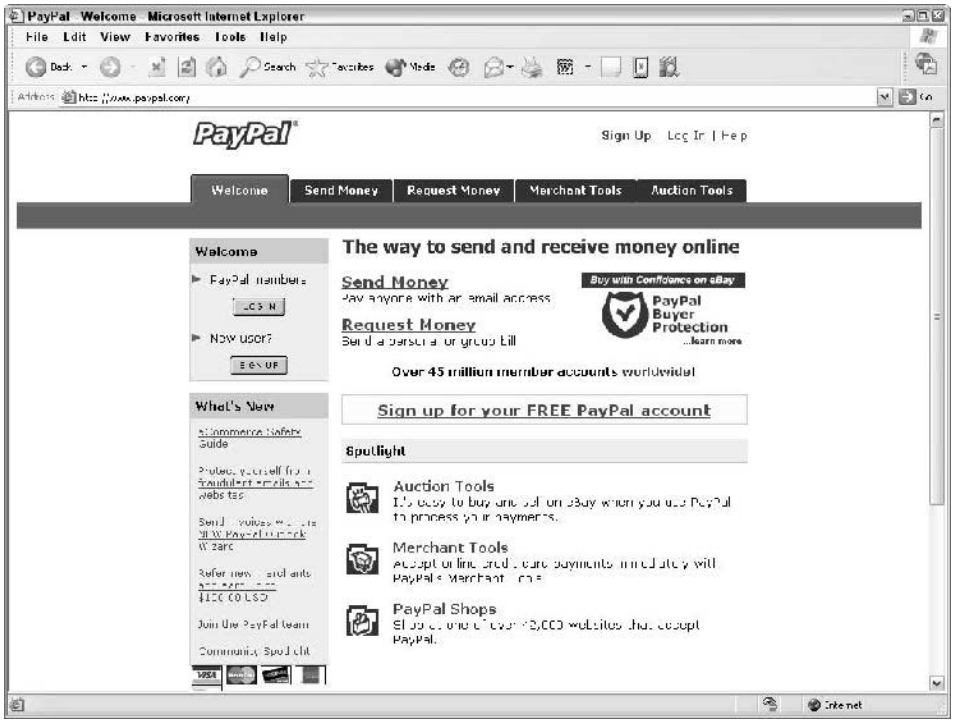


Figure 23-13: You can set up a PayPal account at `www.paypal.com`.

You pay no charge to open a Personal or Premier/Business PayPal account. If you open a personal account, you may not accept a debit or credit card payment. If you set up a Premier/Business PayPal account, when a buyer pays via PayPal you incur a fee anywhere from 2.2 percent of the selling price plus $0.30 USD and 2.9 percent of the selling price plus $0.30 USD. Additional fees are incurred if you accept payment from a buyer not in the United States, or if a currency transaction is involved.



To avoid any unforeseen loss of profit when accepting payment via PayPal, clearly point out in your auction whether or not you will accept payment from foreign countries. If you do decide to accept PayPal payment from foreign buyers, be sure to mention any additional charges you will tack onto the winning bid.

Getting paid with checks

If you decide to accept personal checks from successful bidders, it is advisable to hold the merchandise until the check clears your bank. If you do decide to accept personal checks and heed this advise, be sure to notify your buyer that you will hold the merchandise until his or her check clears.

Certified checks are another method of accepting payment. However, certain cashier's checks can only be cashed by the company that issued the check. If the

www.it-ebooks.info

check vendor does not have an office in your area, you may have to travel a considerable distance in order to receive your funds. If that is an issue be sure to point out which forms of cashier's checks you will accept, and which you won't.



Make sure to point out clearly which forms of payment you will and will not accept. If you prefer not to accept payments through PayPal, be sure to point this out in your auction information.



If you have a repeat customer whose checks have always cleared, you may want to consider shipping merchandise immediately upon receiving the "preferred customer's" check. This may help establish an ongoing business relationship.

Shipping Merchandise

Shipping is another important consideration when putting an item up for auction on eBay. You should clearly point out the manner in which you'll ship the item. You also need to point out what you'll charge the buyer to ship the item. If the item you're shipping is small, fairly compact, and weighs only a few pounds, you may be able to ship it through U.S. Postal Service Priority Mail. You can find out exactly how much it will cost to ship the item by packing it and weighing it on an accurate scale. You can get the latest postal rates by visiting `www.usps.com` and then clicking the Calculate Postage link.

Another option for shipping the item is UPS or FedEx. Each carrier offers many options for shipping. You can calculate UPS shipping rates by visiting `wwwapps.ups.com/calTimeCost?loc=en_US` or FedEx shipping rates by visiting `www.fedex.com/ratefinder/home?cc=US&language=en`.



UPS and FedEx rate estimates are just that. If you use different sized packaging, the rate may fluctuate. Pad your shipping charges accordingly if you get a rate estimate at either shipper's Web site.

If the item you're shipping is expensive, insurance is another consideration. Insurance should be added to buyer's costs, as they will be making any potential claim from the carrier. Check with your desired shipper for current insurance costs.

Make sure you pack the merchandise securely. If you plop fragile items into a cavernous box and stuff some loosely wadded newspaper around it, your buyer is going to receive damaged and perhaps unusable merchandise. Not only will this damage your reputation as an eBay auctioneer, you'll end up having to take the merchandise back and refund your buyer. After the package is returned, what was once profit will now be relegated to the trash bin.



If you frequently sell items on eBay, create a boilerplate document in your favorite word processing application that clearly points out your method of shipping, forms of payment you'll accept, and other text you frequently include in your auctions. Cut and paste the text from the document into the applicable sections of the eBay insertion form.

www.it-ebooks.info

Getting the Most from eBay

eBay can be a wonderful way to enhance your online store, or to simply get rid of unwanted, but still desirable merchandise, trinkets, antiques, or other baubles that may be gathering dust on your bookshelves. If you plan to repeatedly sell merchandise on eBay, you can greatly enhance your chances by minding your business—in this case, your eBay auctions.

Establishing yourself as a reputable dealer

People buy from dealers who sell quality merchandise and deliver first-class service. Many people who buy on eBay expect to get a bargain, but they still expect the bargain to be quality merchandise and the transaction to be handled in an efficient businesslike manner.

Establishing a good reputation is easy at eBay. When you list an item on eBay, note that you will give positive feedback for prompt payment. After an auction closes, and you ship the merchandise, reiterate through e-mail that you will give the buyer positive feedback, and request that the buyer do the same for you. A buyer or seller can leave feedback at the eBay Feedback forum (see Figure 23-14).

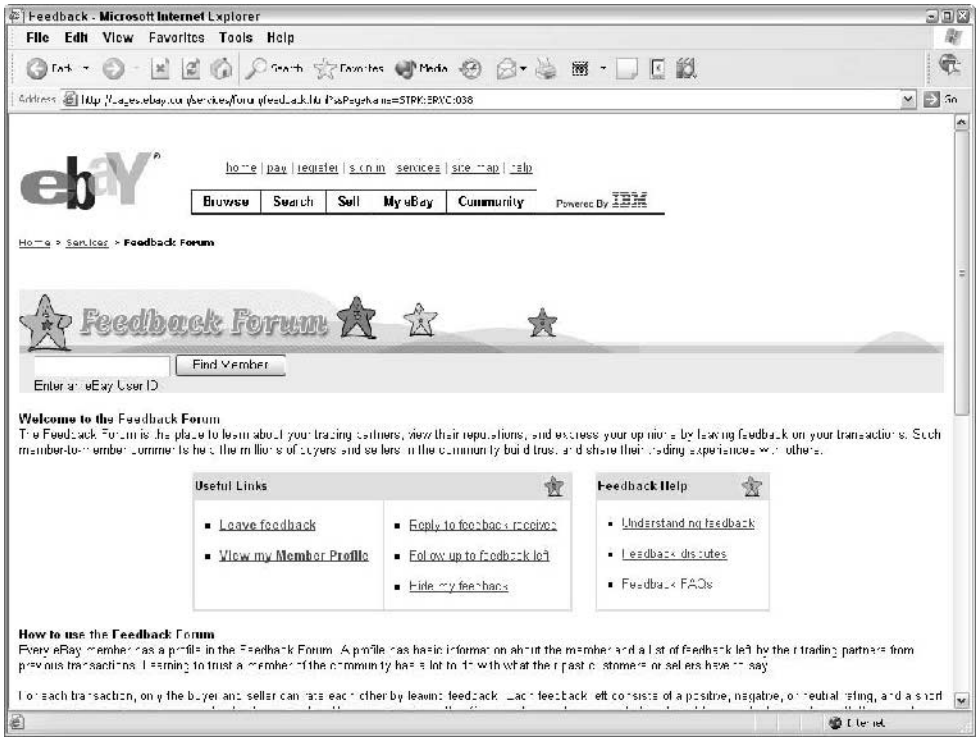


Figure 23-14: Buyers and sellers can leave feedback at the eBay Feedback Forum.

www.it-ebooks.info

Another thing you can do to establish yourself as a reputable dealer is to create a reasonable exchange or return policy. This may not be applicable if you're selling an antique or gadget. However, if you regularly sell on eBay, or sell from an eBay store, creating an exchange or return policy will ensure you don't get negative feedback. Be sure to post your return or exchange policy prominently on every auction you insert.



When you're just starting out on eBay, you have no reputation as a dealer. You can, however, establish yourself as a reputable person, by purchasing items from eBay dealers and then requesting feedback when you pay promptly. After you've gained a dozen or so positive feedbacks, you're ready to set up shop as an auctioneer. The positive feedbacks from your previous transactions are posted with your auctions and help soothe potential buyer fear.

Setting a start and finish time

When you decide to insert an item for auction on eBay, you can accept the default start time, which is the time of insertion, and the default end time, which is seven days later. However, for an additional $0.50, you can specify the start time. When you specify a start time, by default the end time is seven days later.

You can enhance your chances of selling your item by specifying a start time. For example, imagine the item you are selling is a surfboard. Consider California, the part of the country where a surfboard is a hot commodity. Set the start time for Sunday between 7 and 10 PM when people are unwinding from a busy weekend and checking for eBay treasures before retiring for the evening. The reason for setting 10 PM as the latest hour for auction end is that experienced bidders will often work an item right down to the wire. If you set the end auction time at a reasonable hour for the time zone where your item is most likely to sell, you'll entice more buyers to hang in until the bitter end in order to win your auction.

Monitoring your auction

When you insert an item on eBay for auction, you receive a hit counter by default. You can visit your own auction and view the hit counter. In fact, visiting your own auction will increase the number of hits and make it look like your auction is receiving more action then it's actually getting.

If you have a desirable item up for auction, and a hit counter with high numbers, it might encourage impulse bidding. Or if you've added the Buy Now option, an impulsive buyer may purchase the item immediately instead of waiting for the auction to end.

After the auction

When an auction is successful, the first thing you should do is to communicate with the buyer. Thank him or her for the bid and provide information. Tell the buyer that you will confirm through e-mail when payment is received and you will send another e-mail when the item is shipped. If you can provide tracking information with the

www.it-ebooks.info

shipping confirmation, you enable the buyer to track the shipment, which is a sign of professionalism that will enhance your reputation. Communicating with the buyer establishes you as a reputable dealer and increases your chances of repeat business with the buyer.

Enhancing your chances for success

Selling merchandise on eBay can be exciting, and quite profitable. You can enhance your chances for success if you use a few tricks of the trade. The following list provides a few pointers for a successful auction on eBay:

- Create an effective title. eBay gives you 45 characters including spaces to create a title. The eBay search engine uses the information in each auction title to determine whether or not the item matches the buyer's query. Use product names and any other buzzwords associated with the item you are selling.
- Do not go overboard and create a title that uses superlatives like "Best", "Greatest" or "Must See" in an attempt to catch a buyer's eye when your item is returned in a search. Experienced bidders are hip to this ploy and often pass over items with suspect titles.
- Make sure your title and description are grammatically correct. A spelling or grammatical error marks you as an amateur who may be selling merchandise of inferior quality.
- Make sure the image or images with your auction are sharp and clearly in focus. If you're selling a shiny item, be careful as to what shows up in the reflection. If your item reflects a sloppily dressed cameraman and a cluttered background, your merchandise will be regarded in the same manner, and many bidders will pass your auction by.
- Optimize each image you use in an auction. If you host your image with eBay, the image will be sized to fit a 400 × 300 pixel space (standard size), or a 500 × 375 pixel space (supersize). However the image file size is still the same as you upload. Resample your image to the applicable size with a resolution of 72 ppi. This ensures you a small file size, which means your auction images will download quicker. You can use applications like Macromedia Fireworks MX 2004 and Adobe Photoshop Image Ready CS to optimize your images for an eBay auction.
- Visit eBay services (see Figure 23-15) to get the latest information on eBay services. You'll also find tutorials and helpful hints in this section of the eBay site. Mine this treasure trove for all its worth, and you'll find some competitive edges to put you on the fast lane to success.

Using the eBay template

When you insert an item into eBay for auction, you follow the prompts on each insertion page. This determines the title, description, and images used to create your auction page. It's pretty much an enter-data-click-and-press-Continue process to create an auction page. You do, however, have some choices in the matter. You can

www.it-ebooks.info

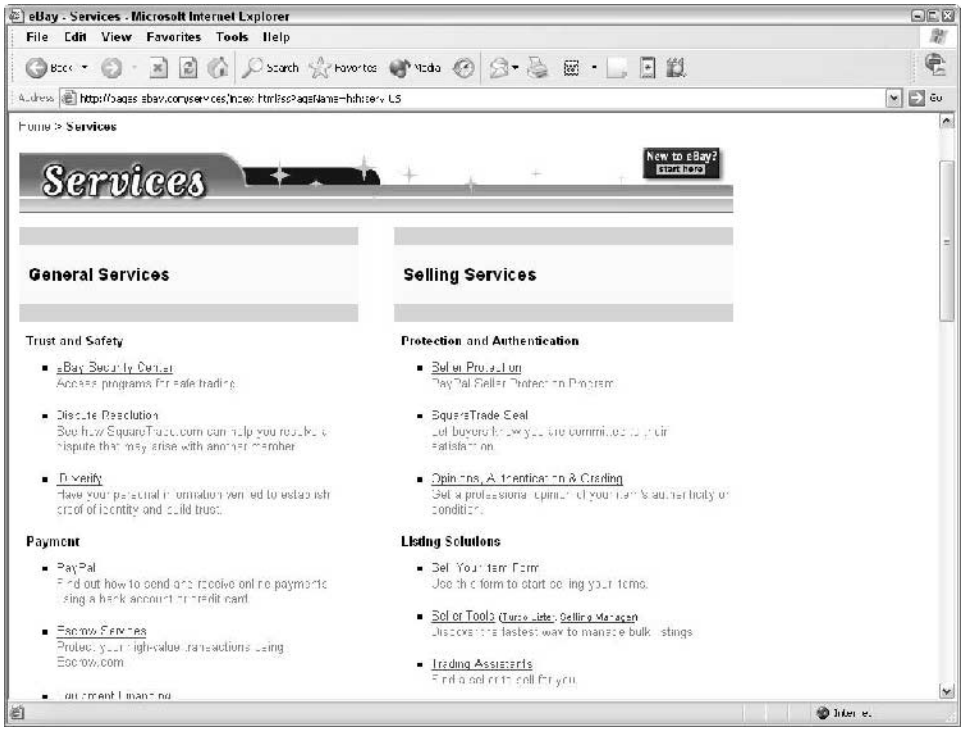


Figure 23-15: eBay Services has the most up-to-date information on all aspects of eBay.

modify the title text to make your item stand out when displayed in a search return. You can use multiple images to display the item you have for sale, as well as supersize the images so they stand out. You can also create a slide show. Use the elements eBay provides to create a catchy auction page.

Modifying the eBay template

You can modify the basic eBay template by choosing a theme from the Listing Designer section of the Sell Your Item: Enter Pictures and Item Details page. Choose a theme that's a good match for the item you're selling. For example, if you're selling a surfboard, the Summer-Surf theme may be just the ticket. In addition to choosing a theme, for $0.25 you can add the Gallery Option, which displays a small version of your first image at the top of the auction page, as well as in the search results. Remember, you can always preview your auction before leaving the Sell Your Item: Enter Pictures and Item Details page. If you don't like what you see, choose a different theme.



If you want your listing to stand out on a search results page, add the Bold option for $1.00. You can make your item stand out even more in search results, by adding the Band highlight for $5.00, which displays a colored band around the listing.

www.it-ebooks.info

Setting Up an eBay Store

If you sell merchandise from your Web site, setting up an eBay store may drive additional traffic to your online store. You can create a unique eBay store with a look and feel that mirrors your current online store. With your eBay store, you can create multiple listings that last longer and are less expensive than regular eBay listings. You can take advantage of eBay tools to cross-market your eBay store and your Web site store. As of this writing, you can open an eBay store on a free one-month trial basis. To set up an eBay store:

1. Navigate to `eBay.com`.
2. Click the eBay Stores link to display the eBay stores page.
3. Click the Open Your Store Now link. eBay prompts you to log on.
4. Log in using your eBay user name and password. eBay displays the user agreement.
5. After reading the user agreement, click the I Accept the eBay User Agreement button to display the Basic Store Information page.



If you don't have a user feedback of at least 20, you will not be able to open an eBay store unless you click the Become ID Verified link and provide the required information.

6. Enter the store name, type of products sold, and other pertinent information. Your next stop is the Store Colors and Graphics page.
7. Personalize your eBay store by choosing your favorite colors and preset eBay graphics. You can customize your eBay store by uploading a custom logo. If you want, you can create your own stamp of approval on your eBay store by designing your storefront using HTML.
8. Review your store and make any necessary changes.
9. Confirm your store.
10. After your eBay store is up and running, you can use the eBay store options to begin cross-marketing with your other online ventures.

Summary

eBay is the world's largest online auction. You can buy and sell a wide variety of items on eBay. This chapter introduced you to the following concepts and topics:

- ✦ You can set up an eBay account for free.
- ✦ You can insert an item for auction by following the prompts.

www.it-ebooks.info

- You can use eBay to host any images associated with your auctions.
- You can use of themes to customize the eBay template.
- A Dutch auction makes it possible for you to sell a lot of items.
- Accepting payment through PayPal is quick and easy.
- You can cross-market your online store with an eBay store.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 24

Getting Paid



In This Chapter

Becoming a credit card merchant

Working with paper drafts

E-cash and your store



The obvious difference between e-commerce and a brick-and-mortar store is that you cannot come face to face with your customers. There's just no way in the world that they can hand you cash for what they're buying, so you have to have some way of accepting non-cash payments. There are three ways you can get paid: credit and debit cards, checks, and e-cash.

Credit and debit cards

The single most common way of purchasing on the World Wide Web is with credit cards (see Figure 24-1). Although debit cards—the kind used to access ATMs—are gaining popularity as well, they will probably never surpass the tried and true standard. No matter what kind of Web space provision you decide upon, the odds are pretty good that the Web space provider itself is an easy channel to credit card acceptance. It's only the purely tech types who are missing out on the opportunity to grab a few extra bucks by adding basic e-commerce services to their roster.

If, however, your Web space provider is a little on the slack side, all you have to do is go to the source. Table 24-1 lists the URLs for the major credit card companies.

The sites in Table 24-1 will give you a lot of good information. You can't, however, get the credit card merchant account from the credit card companies themselves (except for American Express). Instead, they'll refer you to an outside company called a *merchant acquirer*, with whom you will actually sign your merchant agreement. Table 24-2 lists several such companies.

When you talk to these merchant acquirers, there are a few details you need to nail down. In a nutshell, they all come under the heading of "how much is this going to cost me?" The answer to that question, however, isn't as simple as you might

www.it-ebooks.info



Figure 24-1: Credit cards are a necessity for any online business.

Table 24-1
Credit Card Companies

Company	*URL*
American Express	americanexpress.com
Discover Card	discovercard.com
MasterCard International	mastercard.com
VISA International	visa.com

think. Here are the points you need to cover:

- ✦ **Application fee:** Is there a charge for submitting the application form? If so, do you get your money back if the application is denied?
- ✦ **Setup fee:** Does it cost you anything for the account to be initiated once you are accepted? What does that fee include?
- ✦ **Statement fee:** What is the monthly charge for sending you a statement of your earnings and fees?

www.it-ebooks.info

Table 24-2
Merchant Account Acquirers

Company	*URL*
AMS Merchant Account Services	www.merchant-accounts.com
Best Payment Solutions	www.best-payment.com
Discover Business Services	www.discoverbiz.com
ePayment Solutions	www.epaymentsolutions.com
EZ Merchant Accounts	www.ezmerchantaccounts.com
MerchantAccount.com	www.merchantaccount.com
Merchant Systems	www.merchant-systems.com
Monster Merchant Account	www.monstermerchantaccount.com
Paymentech	www.paymentech.com
Total Merchant Services	www.totalmerchantservices.com

- **Discount rate:** This is one of the most important points. The discount is the percentage of your sales that the credit card company will take for its cut. The lower the discount rate, the better. However, most Internet companies cannot get the kind of low discount rate that conventional companies can.
- **Per-transaction fee:** Is there an additional flat fee that is charged for each transaction? If you sell high-ticket items like customized Ferraris, then the additional amount (maybe 30 to 50 cents) will mean nothing. But you understand why most stores don't like taking credit cards for low-priced items.
- **Monthly minimum fee:** If your sales don't take off as anticipated, you may have to pay the credit card company anyway. It's their way of making sure that, no matter what happens to your business, they always have a little something coming in.

Establishing Return and Refund Policies

Before you ever ship your first order, you need to establish (and clearly post on your Web site) your policy regarding returns and refunds. Without it, you're asking for trouble from day one. As a credit card merchant, you have a few things you can do to protect yourself from losses, and this policy is one of the most important, yet often overlooked, ones.

Why? Because if you have a clearly stated and obviously posted return and refund and return policy, then the credit card company will abide by those posted rules in handling any complaints that your customers file with them. You can find a good example of a refund and return policy at www.microcenter.com/how_to_shop/return_policy.html (see Figure 24-2).

www.it-ebooks.info

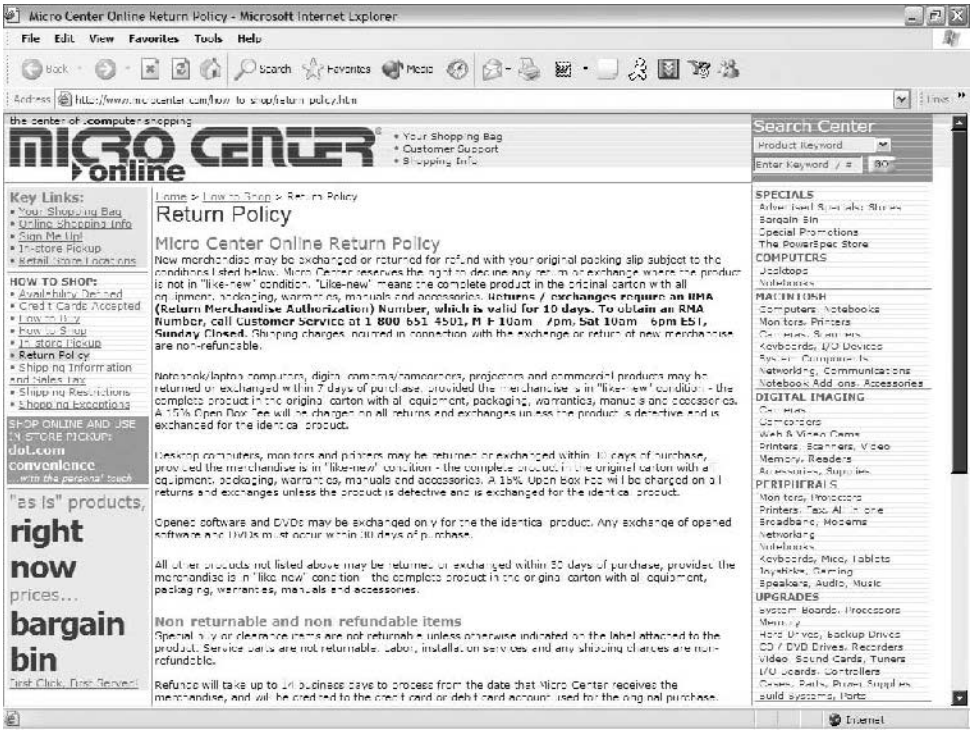


Figure 24-2: A clear return and refund policy is essential for a credit card merchant.

Here are some tips for what to keep in mind while creating your site's refund and return policy:

- **Time limits:** You should specify the length of time during which a product may be returned. Is it 30 days, 90 days, a year, a lifetime? Does the time limit vary for particular products or classes of products?
- **Valid reasons:** For what reasons will you accept a return or issue a refund? Are there any special conditions for different products? For example, do you need one policy for electronics and another for housewares?
- **Return authorization:** Can your customers just send it back, or do they need to get an authorization form from you for the return? If the latter, then you need to assign responsibility for handling the return approval process. Is it the job of customer service, technical support, accounting, or some other department?
- **Packaging:** If the item has been opened (and that is usually the case), does it have to be returned in its original packaging? What if the UPC code is missing from the box?
- **Documentation:** What proofs of purchase do you accept? If you're dealing solely online, you don't have a signed receipt. Any form of receipt you sent via e-mail, or that the buyer printed out from your Web site, can be easily forged. Therefore, you have to rely on your own sales records for your own protection.

www.it-ebooks.info

- **Bricks and mortar:** If you operate a conventional, physical store in conjunction with your Web site, does the same policy apply to purchases from both? What about if someone tries to return a product at your store that was purchased from your Web site?
- **Shipping charges:** You need to factor shipping costs into your return calculations. Depending on your product, they may be a serious impediment to the return process. Make sure you state who pays for what.



There may be specific limitations that apply to you, depending on where you live and what industry you are in. For example, you may find that you have to lengthen your time limit or make exceptions for residents of other states. Check with your lawyer to ensure that the policy you adopt does not violate applicable laws.

Watching out for chargebacks

Sooner or later, you're going to run into some situation in which your happiness over your credit card sale turns into anything from surprise to dismay as the sale goes sour. Your profit drops as you face a *chargeback*—the credit card company taking the money back from you.

There are many possible reasons for a credit card sale to fail; it can even happen if the whole approval process runs smoothly. A cranky customer who didn't read your return policy (or who did, but doesn't want to admit it) writes to MasterCard that you defrauded them and suddenly you have to defend your right to get paid. Another typed in the wrong credit card number and it was charged anyway. Stay in business for a while, and you can write your own list of problems easily enough.

Credit card fraud, however, is the one bugaboo that hits most companies the hardest. It's a multizillion dollar industry, and it's not going to go away. Your merchant account provider will doubtless be fully on your side in helping you to prevent it, but make sure to ask for specifics before you sign anything.

The most important item to have on your side is an *address verification system* (AVS). This is a service that compares the address your buyer gave on the order form with the address that the credit card company sends the bills to. If they're not the same, then the chances that the credit card is being used by someone other than its owner go way up.

Of course, there are lots of cases where somebody legitimately wants to order something to be sent to a different address. The florist business, for example, would wither overnight if it insisted on delivering roses to your house instead of to your valentine's house. Nonetheless, if you are in a more common line of business, you should consider conflicting addresses to be a red flag. If you are willing to accept shipments to an address other than the billing address, then at least have separate address sections on your order form—one for the shipping address and one for the credit card billing address. That way, you can still run the AVS on the billing address. If it matches, then you may feel a bit easier about shipping to the other address.

www.it-ebooks.info

Checks and Paper Drafts

You can accept checks for your online business in two different ways. The first, of course, is the old-fashioned way: someone simply prints out an order form, fills it in, and mails it along with a paper check. The second possibility is for you to print the check and deposit it just as if it had been sent in the mail. Yes, it's legal—as long as you have the buyer's permission, that is.

If you're not familiar with this approach, take a look at the bottom of the next TV ad that's trying to sell you some fabulous gadget you just can't live without. Somewhere in the payment details, it'll doubtless say that you can pay with a check over the telephone. All you have to do is call them up and give them the information on the bottom of your check. This identifies two important things—your bank and the number of your checking account. Armed with that information, a check called a *paper draft* can be created (see Figure 24-3).

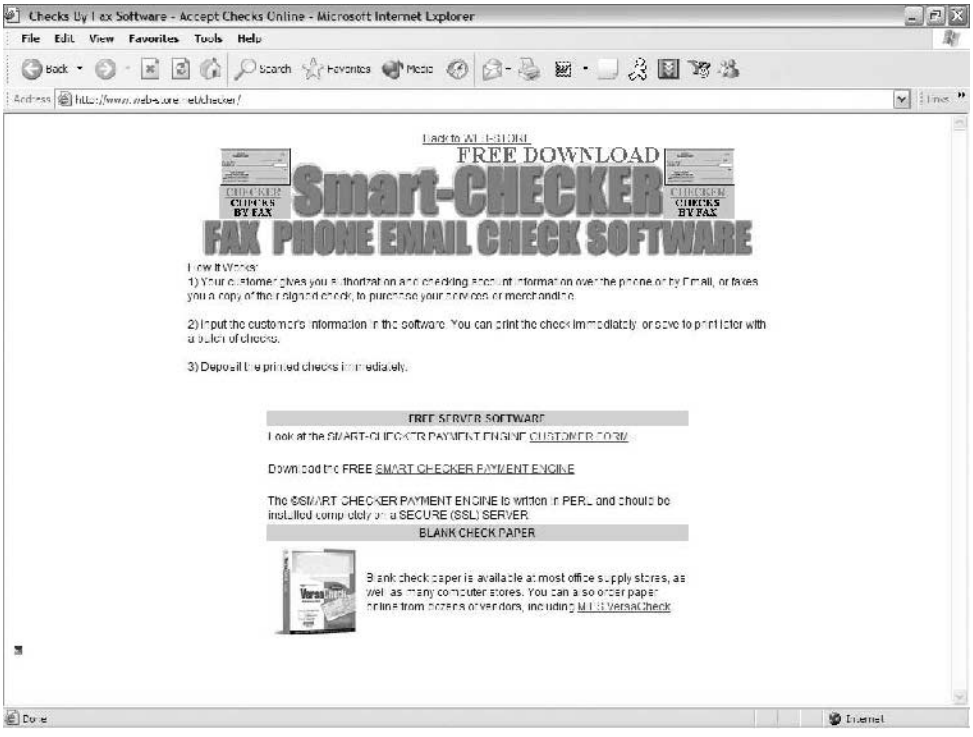


Figure 24-3: Smart-Checker lets you create paper drafts that work just like regular checks.

This may sound scary to you—what's to keep them from whipping up as many checks as they want to and emptying your bank account? That's a good question, and you should carefully consider to whom you give that information. But everyone who ever wrote a check for anything already gave that information to everyone

www.it-ebooks.info

they ever gave or sent a check to. Remember, it's right there for anyone to read. You may want to point this out to your customers if they seem antsy about using the technique.

What's the real value of this capability? Simple—people who don't have credit cards can place an online order just as easily as if they had a wallet full of them.

Table 24-3 lists sources for paper draft software.

Table 24-3
Paper Draft Programs

Program	*URL*
Checks By Fax	checksbyfax.net
CHEXpress	berryhillfinancial.com/chexped.htm
Draft Creator	www.advancemeants.com/draft/
Smart-Checker	www.web-store.net/checker

E-cash

E-cash is one of those ideas that sounds so good, you wonder why it hasn't taken off. In theory, it will one day be fully integrated with the online world. Some day, the pundits say, your Web browser will interface with your bank account, and everything all along the line will be protected by the same kind of digital certificates that make secure Web servers work. When that day comes, all you will have to do to buy anything on the Web will be to say you want it. The money will be instantly debited, and the merchant will be instantly paid.

The concept, however, is far from a reality for a variety of reasons. First of all, nobody can agree on an e-cash standard. In addition, the concept of digital certificates is difficult to get across to the public. Remember, many of the folks who presumably will use e-cash one day are still having trouble navigating a Web browser to the site they want.

And the same fundamental human problem exists with e-cash as with debit cards. When people use credit cards, they can wait to pay them off. With e-cash, they lose their money on the spot. I'm not holding my breath on this one. In fact, if it weren't for one exception, I'd be willing to relegate the whole thing to wherever grand dreams that never happened go. That exception is PayPal, a company that managed to connect bank accounts, credit cards, debit cards, and most of the world's major currencies—and made the whole shebang easy to use for both merchants and customers. That's quite an accomplishment, and you can see the details of how to integrate PayPal into your own Web site in Chapter 22.

www.it-ebooks.info

Summary

Many people see e-commerce as the most important aspect of the World Wide Web. It promises to revolutionize marketing. This chapter introduced you to the following concepts and topics:

- ✦ Credit cards are the most common form of payment on the World Wide Web.
- ✦ Paper drafts allow you to take checks instead of credit cards via your Web site.
- ✦ E-cash, except for one company (PayPal), has not taken off.

✦ ✦ ✦

www.it-ebooks.info

CHAPTER 25

Using Advertising



In This Chapter

Advertising with search engines

Banner ads

Working with advertisement agencies

Using affiliates

Investigating marketing alternatives



If you're serious about marketing online, you may want to consider advertising. When you advertise online, you reach out and contact more potential buyers or clients. You have various options for advertising online. You can pay a fee to search engines to boost your visibility when visitors search using keywords related to your product or service. In addition to advertising online, you can work with online marketing firms to enhance your visibility on the net.

Advertising with Google and Other Search Engines

As of this writing, Google is undoubtedly one of the most widely used search engines. It's used by a wide variety of people from all over the world. The search engine is very user-friendly, and delivers accurate results. As an Internet entrepreneur, you can significantly increase your chances of success by advertising with a search engine. In the following sections, you'll learn how to advertise your business using Google and other search engines.

Advertising with Google

When a user types a keyword or key phrase into the Google search engine, relevant results appear quickly. The top results (descriptions and URLs for Web pages) for the keyword or key phrase appear on the first search results page and subsequent search results pages contain less relevant hits. The right border of Google's search engine results contains sponsored links from paid advertisers (see Figure 25-1).

You can add your Web site to Google's sponsored links by using a service known as Google AdWords. When you use this service, you create your own ads and choose keywords to match your advertisement with potential buyers of your services or products.

www.it-ebooks.info

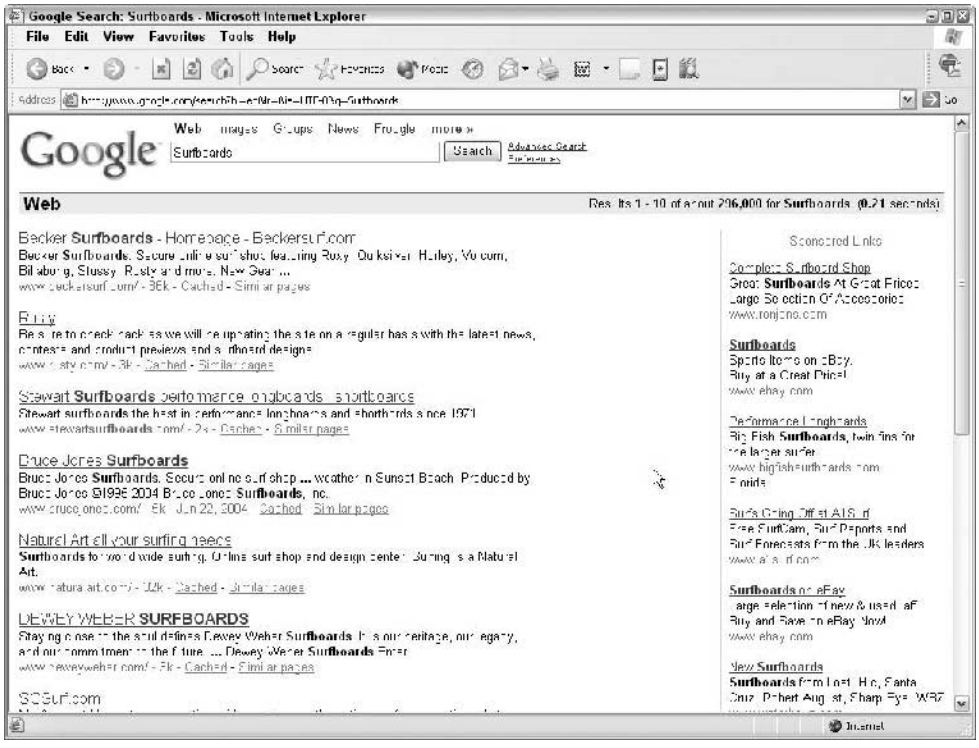


Figure 25-1: Paid advertisers appear on the right side of a Google search results page.

The fee for setting up a Google AdWords account is $5.00 USD. It takes approximately 15 minutes to set up a Google AdWords account. To set up a Google account with help from a guided tutorial, navigate to `https://adwords.google.com/select/` and click the sign-up tutorial link under the Sign Up Now button (see Figure 25-2). This launches the Google AdWords Guided Tour page (see Figure 25-3).

The signup process consists of four steps:

1. Choosing the target language and region.
2. Creating advertisement groups.
3. Setting your daily budget.
4. Creating your account.

The following explains the process in detail:

1. To begin the signup process, click the Learn How button for Step 1 to launch the Step 1 animated tutorial (see Figure 25-4). The tutorial explains all the intricacies of choosing the target language and country for your Google ad. During this step, you'll choose your target language, and the region in which your Google advertisement will be displayed.

www.it-ebooks.info

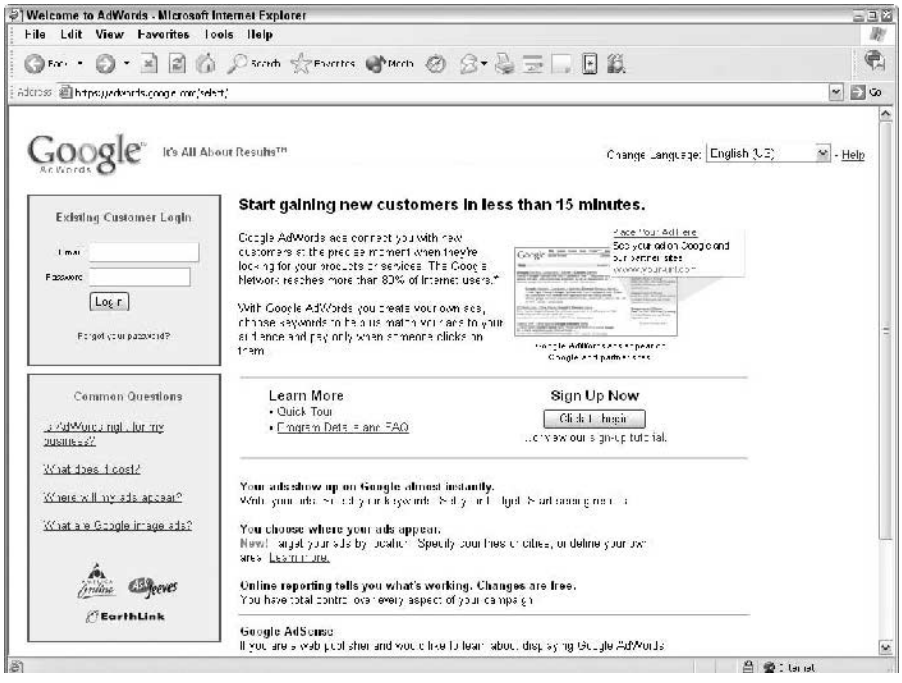


Figure 25-2: The AdWords home page provides a link to a step-by-step signup tutorial.

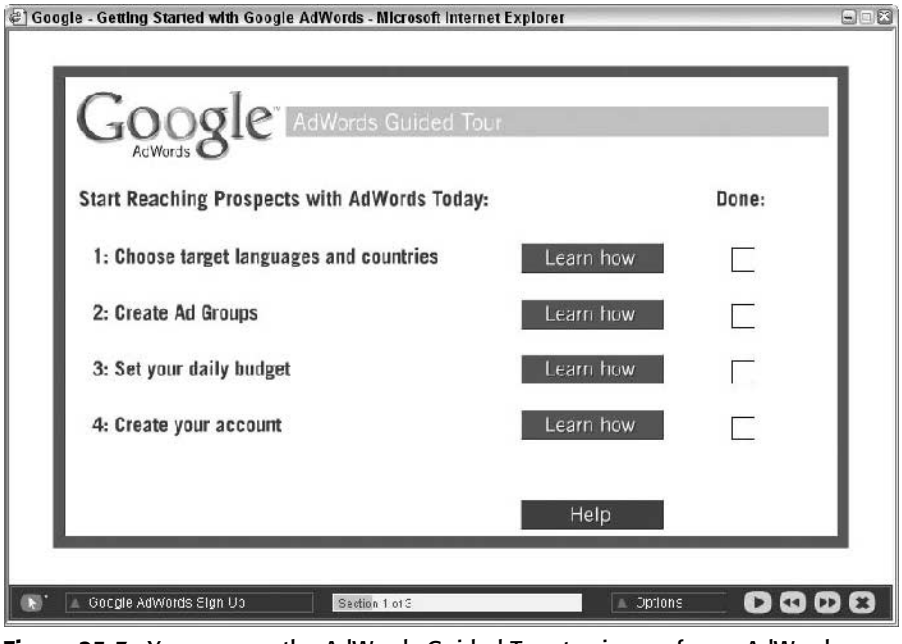


Figure 25-3: You can use the AdWords Guided Tour to sign up for an AdWords account.

www.it-ebooks.info

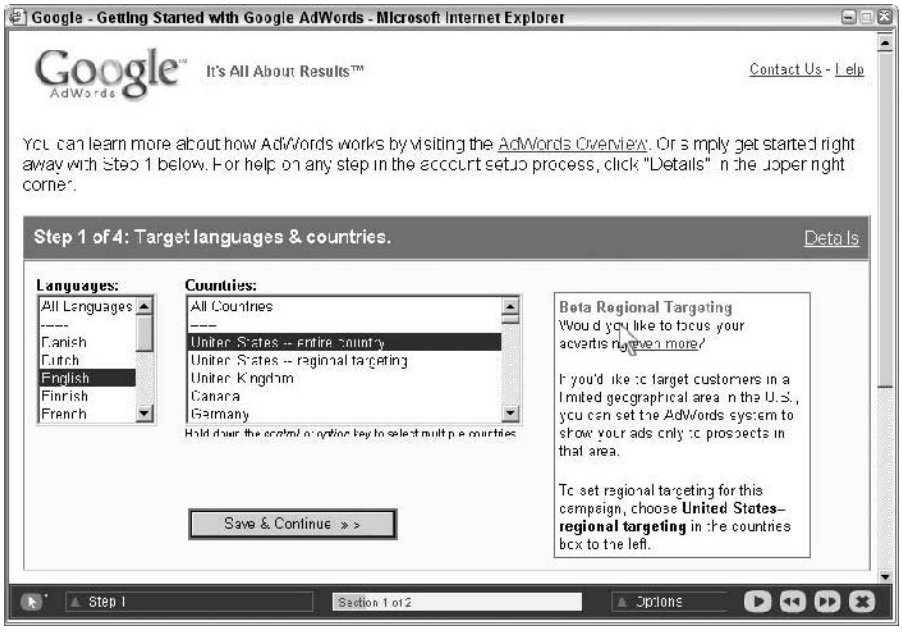


Figure 25-4: Your first step is to choose your target language and country.

If you're targeting multiple languages, you must create a different advertisement for each language, and the advertisement must be written in the language you are targeting. After the tutorial for Step 1 finishes, you'll prompted to go to a URL, where you'll complete step 1.



You can change your target language and region at any time after setting up your Google Ad.

2. After selecting your target language and region, return to the tutorial and click the Learn How button for Step 2 to launch the Step 2 tutorial (see Figure 25-5). In Step 2, you create your advertisement group and choose your keywords and maximum cost per click. When you name the advertisement group and create the ad, think outside the envelope and consider what would make a potential customer click the link to view your Web site and potentially purchase your products.



When you put your advertisement together, avoid superlatives like "best", "greatest", and so on. Ads with too many superlatives are red flags to experienced buyers.

After viewing the tutorial, you're prompted to choose your keywords and maximum cost per click.

3. After choosing your keywords and maximum cost per click, return to the tutorial and click the Learn How button for Step 3 to launch the Step 3 tutorial (see Figure 25-6). In this step, you'll also set the maximum amount you want to spend, which helps you control your advertising budget.

www.it-ebooks.info

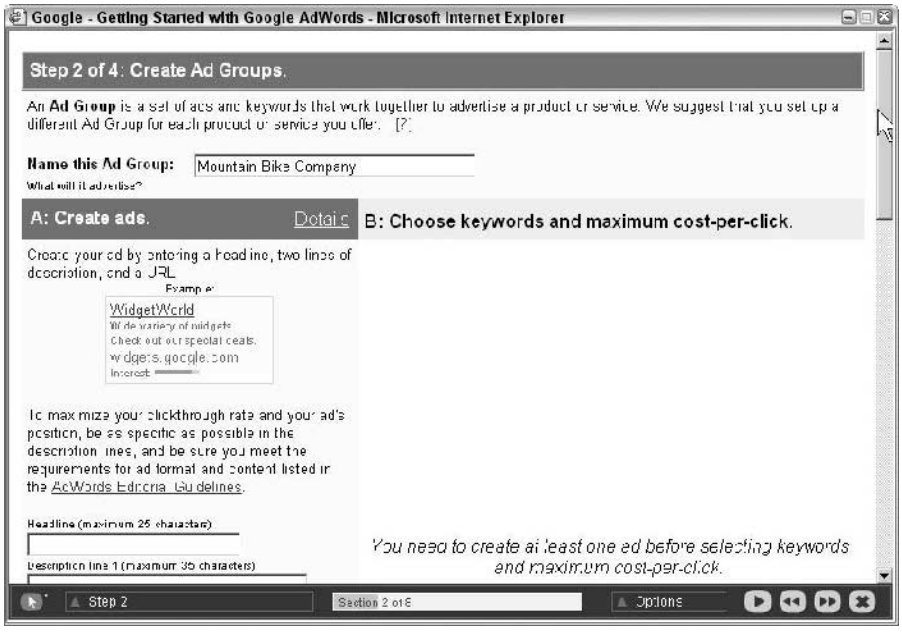


Figure 25-5: Your second step is to choose keywords and set your Google AdWords budget.

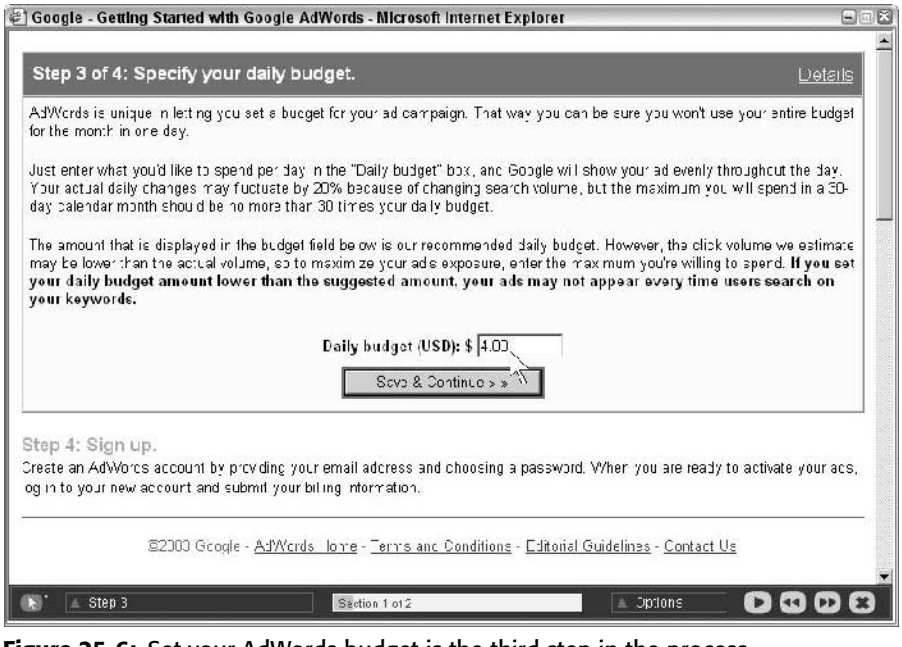


Figure 25-6: Set your AdWords budget is the third step in the process.

www.it-ebooks.info

Google doesn't require a maximum or minimum advertising budget. You determine the amount you want to spend per month, divide it by 30, and enter that figure for your maximum daily budget. During this step Google recommends a daily budget. If you decide to accept Google's recommended budget and find it isn't right for your business, you can change it (or a budget you specify) at any time.



If you specify a budget lower than the Google's recommendation, your advertisement might not be displayed each time your keyword is entered into Google's search engine.

After setting your maximum daily AdWords budget, you're returned to the tutorial.

4. Click the Learn How button for Step 4 to launch the Step 4 tutorial (see Figure 25-7). In this step, you'll enter the e-mail account you use to conduct business, as well as set a password. You'll also be prompted to accept the AdWords Terms and Conditions of Service.

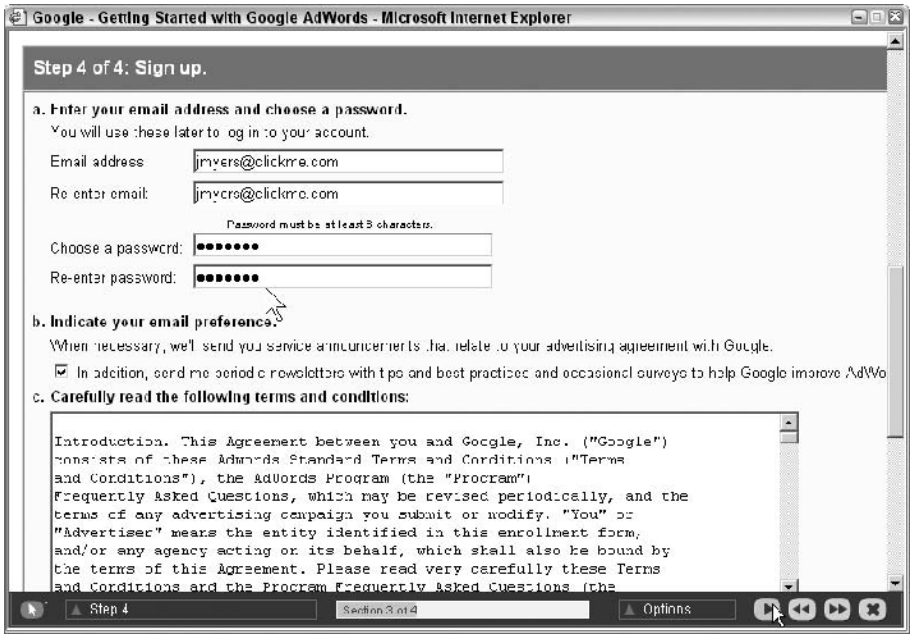


Figure 25-7: Create your AdWords account.

After setting up the AdWords account, Google will send you an e-mail asking you to confirm the account by clicking a link within the e-mail. After that step, your account is active and ready to reach thousands of potential buyers. At the end of the tutorial, you'll see a page that lists links to which you can refer for additional information and help on your AdWords account.

www.it-ebooks.info

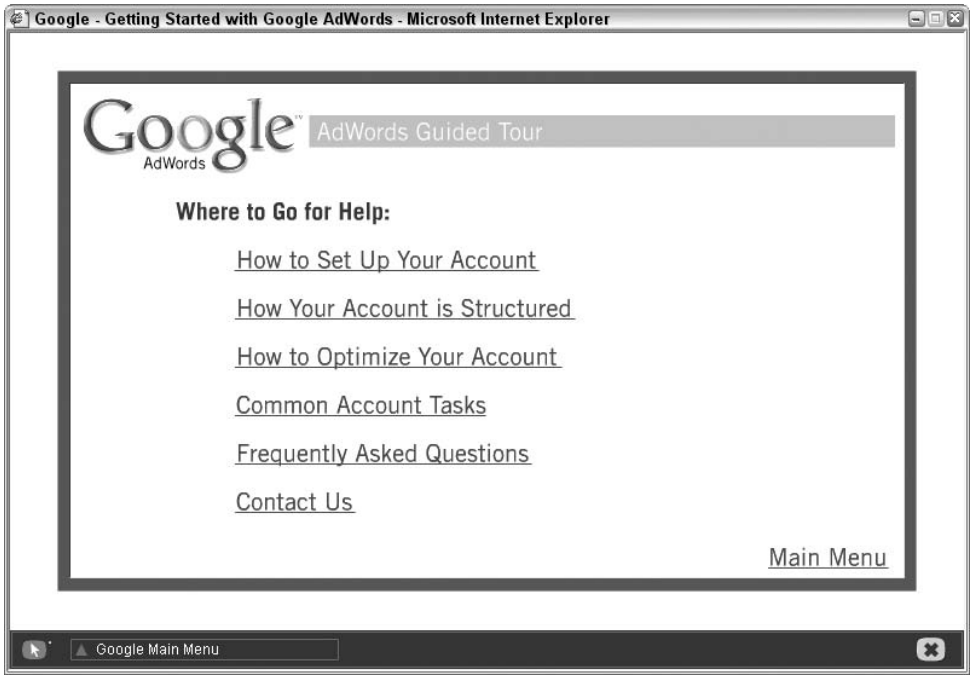


Figure 25-8: You can browse additional links for help on setting up and managing your AdWords account.

After you set up a Google AdWords account, you can monitor the account to see how much the advertisement is actually costing you, how many click-throughs your advertisement is generating, cost per click and so on. Figure 25-9 shows sample statistics for an account with three campaigns.

Advertising with other search engines

Google is not the only search engine where you can pay to have your site turn up in a search results page. Many major search engines offer this service. The majority of these are pay-per-click ads and the sign up process is similar to Google. Different search engines have different criteria for the manner in which they return results. If a search engine continually returns Web sites that feature services or products similar to those that you're marketing, it may be advisable for you to advertise with that search engine.

Lycos (`www.lycos.com`) is an example of a major search engine where you can advertise your site on a pay-per click basis. To advertise with Lycos, click the Advertise link on the companies home page and follow the prompts to set up an account (see Figure 25-10).

Table 25-1 provides an overview of major search engines that accept advertising.

www.it-ebooks.info

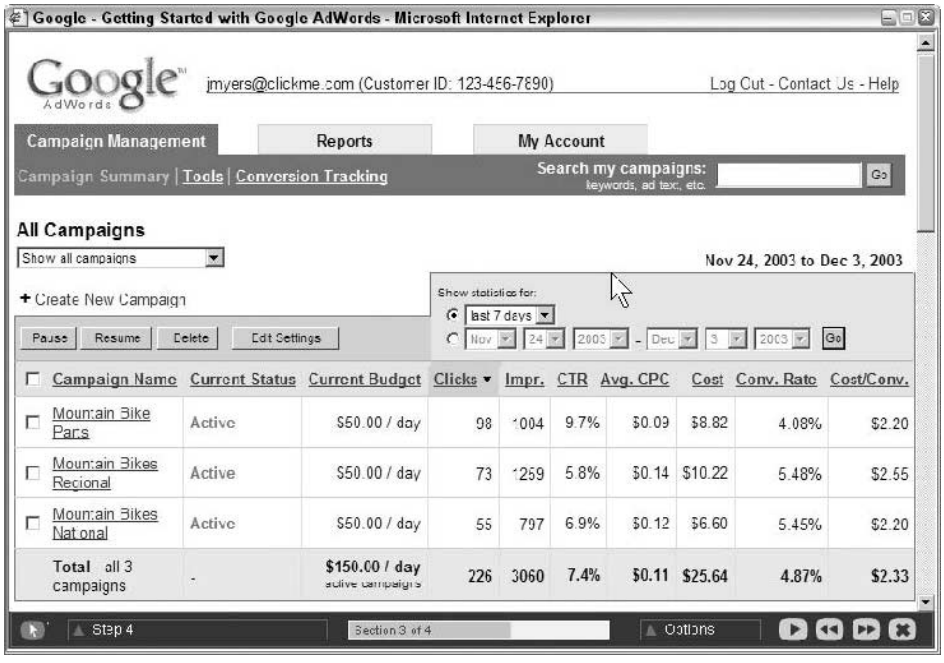


Figure 25-9: This figure shows a sample AdWords account with three campaigns.

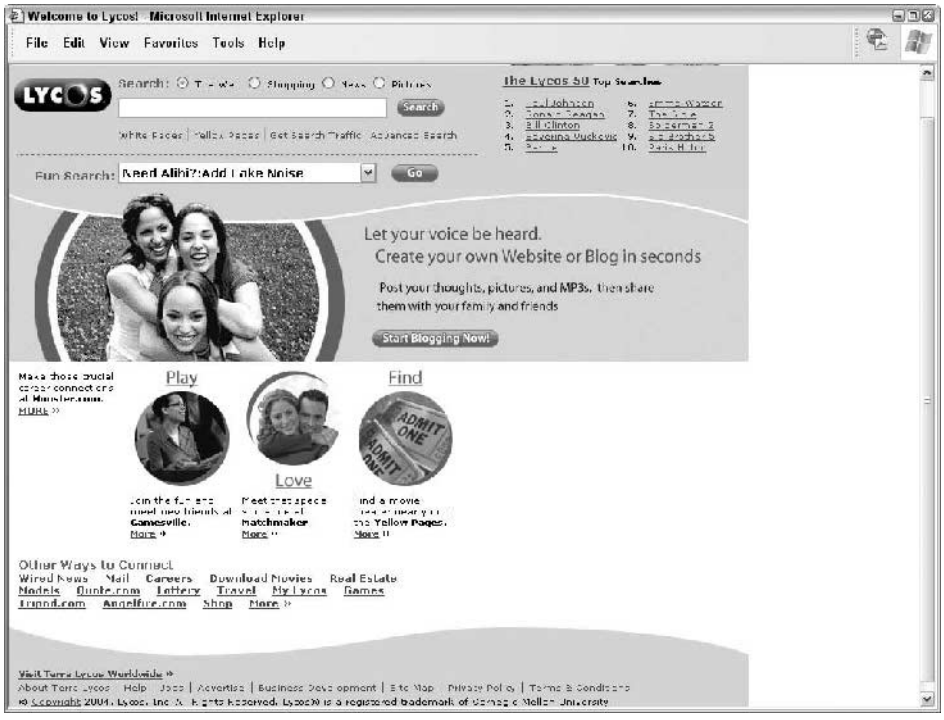


Figure 25-10: You can set up pay-per-click advertising at Lycos.com.

www.it-ebooks.info

Table 25-1
Search Engines That Accept Advertising

Search Engine	*URL*
About	www.about.com
Ask Jeeves	www.askjeeves.com
Google	www.google.com
Lycos	www.lycos.com
Yahoo!	www.yahoo.com

Each search engine company has its own policy and requirements. You can find additional information by navigating to the home page of the search engine you're interested in advertising with, and clicking the site's Advertise link on the home page.

Advertising with banners

In addition to vaulting your Web site to a position of prominence by advertising with a search engine on a pay-per-click basis, you can also advertise with banners. You can advertise with a banner exchange (see Figure 25-11), a service where you agree to post banners from other Web sites on your Web site, and your banner is displayed

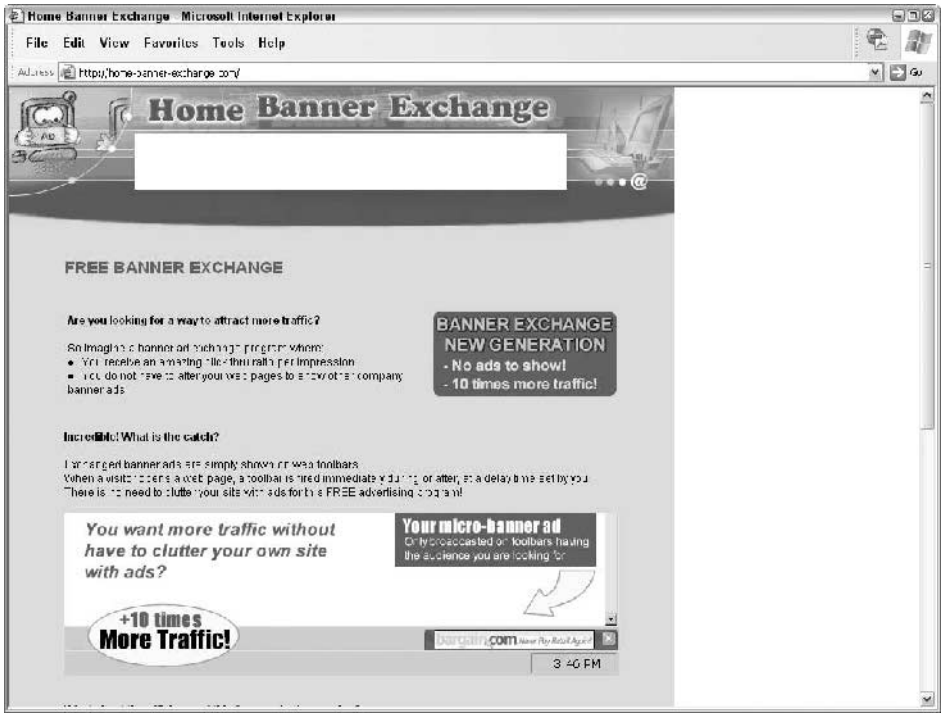


Figure 25-11: You can promote your Web site using a banner exchange.

www.it-ebooks.info

on other Web sites within the exchange. This form of banner advertising can be effective. However, you are advised to check into any free banner exchange. Remember: "There are no free lunches."

If you opt to use a free banner exchange, you may find yourself promoting other peoples services without getting much in return because the Web sites in the banner exchange don't get much traffic. When you research a banner exchange, consider the quality of the Web sites participating in the banner exchange and the products they offer. Also consider whether or not you really want to advertise the products offered by the other banner exchange participants. If possible, contact other banner exchange participants to see if they are satisfied with the service.

Other Web sites will require you to pay to display a banner on their site. The cost of the banner depends on the prominence of the site, and the page in the site on which you choose to display your banner. Frequency will also determine the cost of your banner ad. Many popular Web sites, such as MSN.com, are divided into sections by interest group. You can target a specific section of sites like this to maximize the impact of your banner ad.

Working with Advertisement Agencies

An Internet business is no different than its brick-and-mortar equivalent. As such you can grow your business by being a savvy marketer. However, when you're just beginning, or your business has grown to the point where you don't have time to concern yourself with marketing, it may be in your best interest to employ a professional—an advertisement agency that specializes in promoting Internet businesses.

Finding an Internet advertisement agency

The easiest way to find an Internet advertisement agency is to do a search in your favorite search engine. You'll find plenty of candidates from which to choose. In fact the sheer quantity of returns you get from a search engine query may be quite intimidating. So many agencies, so little time.

The type of services offered by Internet advertisement agencies vary widely. You'll find agencies that do little more than optimize your site for a search engine and then submit it as well as full-fledged agencies that will do the aforementioned and provide other services such as banner creation, banner advertisement serving, e-mail marketing and more. Figure 25-12 shows a listing of Internet advertisement agencies at one Web site.

When you peruse the Web sites of online advertisement agencies, here are a few things to consider:

- ✦ Does the Web site look professional?
- ✦ What services does the agency offer?
- ✦ Does the agency list any case studies?

www.it-ebooks.info

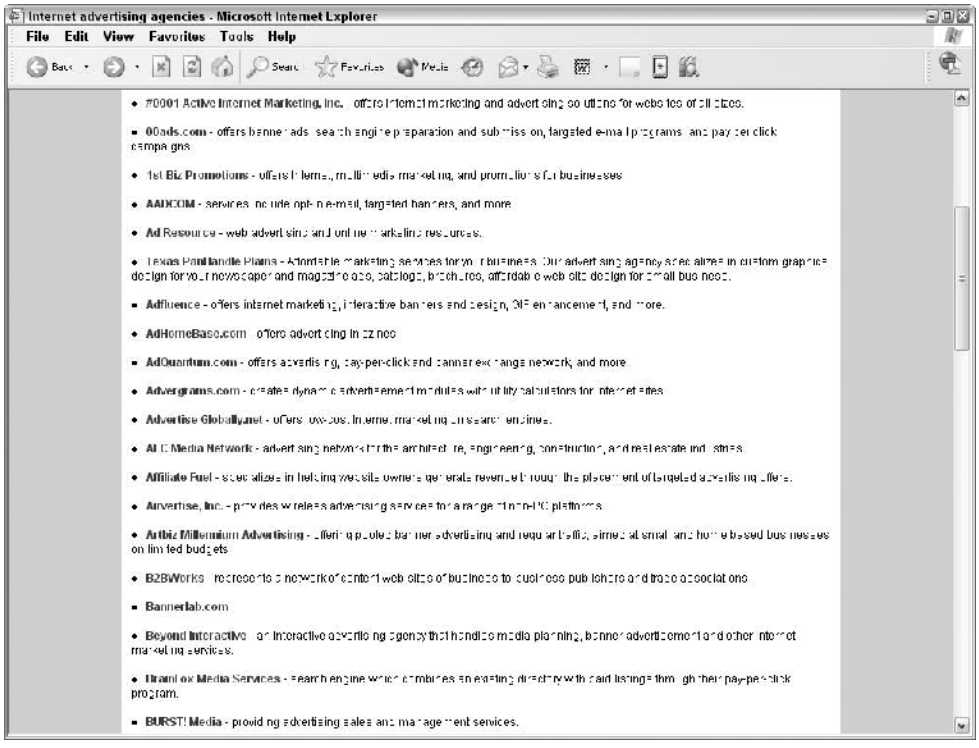


Figure 25-12: You'll find a wide variety of Internet advertisement agencies from which to choose.

- ✦ Are there testimonials from satisfied clients? If so, does the agency provide links or contact information of the satisfied client? Can you contact the satisfied client?
- ✦ Does the site offer an FAQ page?
- ✦ Does the site list information about the representatives and key personnel? This attention to detail is the sign of a professional agency.
- ✦ Does the site have a contact page? If so, is the contact through e-mail only, or can you talk to a representative?



If an advertisement agency lists only e-mail contact information, proceed with caution. If you do contact an advertisement agency with e-mail–only contact, and you don't get a response within an hour or two, move on to another agency. We prefer phone contact because you can get immediate answers, and size up the representative by his or her knowledge and tone of voice.

Working with affiliates

Another alternative you may consider is affiliate marketing. Affiliate marketing is the practice of sharing revenue between multiple Web sites. When you join a group of affiliates, you create links to the affiliate sites. The link coding tells the affiliate from which site the click-through came. Affiliates are paid on a per-click basis.

www.it-ebooks.info

Another form of affiliate marketing is an online catalog where you feature merchandise that you do not inventory. When a visitor to your Web site orders a product, the product is shipped from the affiliate marketer and you get a percentage of the sale. Amazon.com is an example of an affiliate marketer. The coding in the link informs Amazon.com which affiliate merchant generated the sale. Payment for an affiliate marketer varies depending on the company with whom you are affiliating. As a rule, payment is not issued until the affiliate has accrued a certain dollar amount. Payments are generally issued per quarter.

Table 25-2 shows the name and URLs of some Internet affiliate marketing companies.

Table 25-2
Internet Affiliate Marketing Companies

Marketing Company	*Web site*
BBL Internet Media	www.bblmedia.com
Internet Affiliate Marketing Program	www.iafma.org/foyer/foyer.html
Site Build It	www.affiliatemarketing.sitesell.com/
Leadhound Network	www.leadhound.com/
Clix Galore	www.clixgalore.com/
Share A Sale	www.zyra.net/sharsal.htm
Marketing Paradise	www.marketingparadise.net

Before investing time and energy in any affiliate program, research the program thoroughly. Use the same criteria listed in the "Finding an Internet advertisement agency" section earlier in this chapter to determine if the affiliate program suits your needs.

Investigating Alternative Approaches

If you have an entrepreneurial spirit, you probably like to do for yourself, rather then to have others do for you. In this regard, there are several things you can do to boost your Internet business and increase the visibility of yourself, your Web site, and your products. The upcoming sections show some proven methods for promoting yourself, and your offerings.

E-mail advertising

E-mail advertising is a touchy business. These days Internet users are barraged with huge quantities of junk e-mail, (also known as *spam*) for every type of product or service imaginable. We're not suggesting you resort to spam. If you send unwanted e-mail, it will actually harm your chances for success. However, a well-designed e-mail advertisement sent to buyers who have elected to receive information from

www.it-ebooks.info

your company can go a long way towards increasing your business. You can send a plain-Jane text e-mail advertisement or a full-fledged HTML advertisement complete with images and hyperlinks to your Web site. Figure 25-13 shows an example of an HTML e-mail advertisement with images.



Figure 25-13: This figure shows a well-designed e-mail advertisement with images and hyperlinks.

Amassing a mailing list

As mentioned in the previous section, we're firm advocates against delivering any form of unwanted e-mail. So how do you get the e-mail addresses of qualified prospects that are interested in receiving information about your products or services? One method you can use is to buy a mailing list. If you enter *e-mail mailing list* in your favorite search engine, you'll have the URLs to many companies that specialize in selling mailing lists. However, this still involves the problem of unwanted e-mail. After you buy a mailing list and begin distributing your e-mail advertisements to names on the mailing list, you're going to get some nasty return e-mail messages telling you to cease and desist, while your other efforts will end up in your intended recipient's spam folder, never to be read. All in all, not a good return on your time or investment. So if you do decide to purchase an e-mail mailing list, make sure the people on the list are at least qualified prospects.



Many states have an anti-spam law. Before sending out mass e-mails, check your local laws to make sure you're not violating them. Many anti-spam laws require qualified advertisers to put the abbreviation ADV in the subject line of a mass e-mail advertisement.

www.it-ebooks.info

A better method for building a list of qualified e-mail addresses is to collect them from your Web site. Many Web sites use a hover advertisement that drops in from the top of the home page when loaded and asks the visitor to sign up for notification when the Web site is updated. The site Webmaster, or sole proprietor, then adds the e-mail address to a mail group in the address book of an e-mail application or into a database.



Hover advertisements are created using software. You can find companies that sell hover advertisement software by doing an Internet search. You may be able to download a trial version of the software from some vendors.

Another alternative for adding e-mail addresses to your address book or database is to give something away. For example, many Web sites that specialize in services such as promotion or marketing will give away an eBook. In order to download the free eBook, the visitor must provide his or her e-mail address. Clearly point out that the visitor will be receiving communication from you in the future, when new products or services are available.



You'll alleviate the fear of leaving an e-mail address at your Web site if you post a link to your privacy policy on the home page of your site and on any page that request information from your Web site visitor. For more information on privacy policies, refer to Chapter 26.

Sending newsletters

Even though you do an exemplary job of selling quality merchandise or services, and you follow up to make sure your customers are satisfied after the sale, your efforts are quickly forgotten in the humdrum buzz of everyday life. A newsletter is one method many marketers use to keep their name and products in front of their customers. You can ask Web site visitors to subscribe to your newsletter using one of the methods outlined in the previous section. If you do ask visitors to sign up for a newsletter, tell them how often you'll deliver the newsletter, and make sure you adhere to the schedule. Sending a newsletter once a month is generally all it takes to keep your customers informed and interested in your products or services. Alternatively, you can assume customers on your mailing list will welcome your newsletter and just send it to all recipients on your mailing list. Figure 23-14 shows an example of an HTML newsletter sent through e-mail.

Summary

You can promote your Internet business using various methods. This chapter introduced you to the following concepts and topics:

- ✦ You can draw buyers to your Web site using Google AdWords.
- ✦ You can advertise with major search engines on a pay-per-click basis.
- ✦ You can advertise your product or services using banners.

www.it-ebooks.info


Videoguys' Insider - New training DVDs, Adobe Upgrades, Avid Xpress Studio, SmartSound Palettes, Vegas 5, Boris Red and more!

Vol4 Is 7 June 24th, 2004

Dear DOJG SAHLIN,

Summer s here and things are definitely heating up.
- Last month, the new Adobe Video titles started shipping. We have Collections, upgrades and individual titles in stock. We are your source for all Adobe upgrades!
- We have received word from Avid that Xpress Studio is on schedule to begin shipping next week. Get your upgrade orders in now and save big bucks!
- Vegas 5 is in stock and it's getting rave reviews from video editors, magazines and websites. We now have the Vasst Vegas Fasst training DVDs in stock and we're running a really cool training promotion with Vegas 5 +DVD.
- Our web team worked with the gang at SmartSound to create a new page that will allow you to browse, listen to samples and order SmartSound Palettes.
- We have also put together a fantastic promotion with Class On Demand training and Boris Red. For just $1 more you can add Class On Demand "Boris Basics, Tips & Tricks" to any Boris RED or Boris Motion Graphics Pack purchase

New training DVDs from Vasst, Class on Demand and Total Training.

This year one of our goals was to become the industry's leading source for training materials. We have joined forces with all the leading training producers. Whenever possible we're working with

Figure 25-14: An e-mail newsletter keeps your customers informed.

- ✦ You can promote your business using Internet advertisement agencies.
- ✦ Internet Affiliate marketing companies are available to aid in your marketing efforts.
- ✦ E-mail advertising is effective when directed to recipients who have requested information from you or your company.
- ✦ You can create an e-mail list of qualified prospects by requesting an e-mail address in exchange for a free eBook.
- ✦ E-mail newsletters are an effective way to keep your customers and prospects informed.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 26

Covering All the Bases



In This Chapter

Using domain names and trademarks

Soothing common fears

Establishing return and refund policies

Investigating alternative approaches

Optimizing your site for search engines



Well, there just had to be a chapter where all the little, miscellaneous bits were covered, and this one is it. No matter what kind of Web site you're running or contemplating running, there's something in this chapter that you'll need to take a look at.

From choosing a domain name that doesn't conflict with old-style business models and trademarks to figuring out how to fit the latest techniques of online advertising into your business model, we've got you covered.

Using Domain Names and Trademarks

A domain name is often thought to be the same thing as a company name. In most people's minds, a dot-com name is as much a corporate name as Nabisco, Ford, or Sears. But there's a tremendous difference in the legal sense.



We do not pretend to be lawyers, and this book does not offer legal advice but merely presents a summary of situations that are of concern to Web developers. Always seek the advice of a qualified attorney with experience in matters pertaining to the Internet before making any decision or committing to any course of action relating to trademarks.

A trademark is a symbol that indicates the source of a product or service. It has a tangible value because it establishes one brand as different from all others. A domain name is, legally speaking, an address on the Internet and nothing more. This may change over time as more judges become familiar with technical matters that are, so far, very new territory in law. But, as things stand right now, whether or not a domain name can be considered property is even disputed.

www.it-ebooks.info

Why does such an esoteric question matter? According to the law, people and companies have property rights. What you own can't be appropriated by anyone else unless they break the law. Nobody can take your car or use your trademark, both of which are your property, without your permission. But, if you don't own your domain name, how can you have any rights to it?

Domain name disputes

Even if you find that a domain name isn't taken yet, that doesn't necessarily mean you're home free. You can certainly register it in your name at one of the registrars such as Network Solutions at `www.networksolutions.com` (see Figure 26-1) or the registrars listed at `icann.org/registrars/accreditation-qualified-list.html`, but there are some situations in which you may not be able to actually use it.

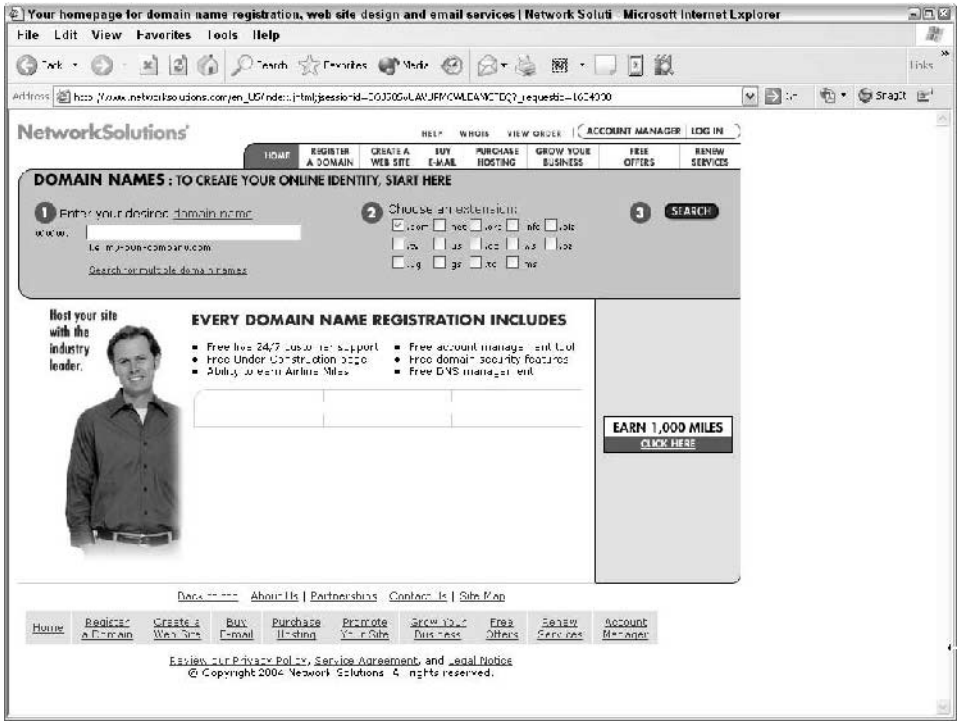


Figure 26-1: Network Solutions is one the best-known registrars.



See Chapter 4 for information on getting a domain name.

The most common dispute situation occurs when someone registers a domain name that is the same or very similar to an existing trademarked name by another company. As soon as the company notices that your Web site is using their name,

www.it-ebooks.info

the trouble starts. Under current U.S. trademark laws, they *have* to go after anyone using their trademarked names or likenesses without their permission. If they don't defend their right to the trademark, the law assumes they're surrendering it.

All it takes to lose a trademark is for the company to not object loudly enough to its use as anything but a name for their product or service. Some examples of trademarks that died this way include aspirin, ping pong, tarmac, gunk, escalator, kerosene, and yo-yo. Since very few companies are willing to lose their trademarks, you can expect to receive a "cease and desist" order, commanding you to stop using the trademarked name online.

Under trademark law, they can't take the actual dot-com domain name away from you just because they own the trademark. But they can tell you to stop using it. You can, of course, fight the trademark owner. With the awesome power of the Internet, numerous campaigns have been waged against such situations. Many individuals have won the public opinion judgment, which is far more important to corporate PR departments. The other recourse is to go to court and prove that you have a legitimate claim to your domain name, for instance, if it's a variation on your own name, if it's a separate trademark for a company in a different industry, if it was registered before the trademark was registered, or if it's an.org registration. People owning dot-org domains have started to lose them to companies holding trademarks, but there are grounds for a fight.

Most importantly, however, if you get a *cease and desist* order that you plan to combat, contact your domain name registrar and ask them not to shut off service. Many domain name registrars shut off your domain name registry entries if they receive a copy of the cease and desist order, even though such an order is in no way a court order to the registrar. Read your terms of service agreement carefully to see if the registrar retains the right to halt or interrupt service in a trademark or domain name dispute.

Another sticky issue is using a celebrity's name as the basis for a domain name. While this will undoubtedly draw traffic to your site, as soon as the celebrity finds out, you're going to receive some attention from the celebrity's manager and attorney.

There are a couple of well-publicized cases where domain names using celebrity names have been purchased. A person purchased `juliaroberts.com` and put the domain name up for auction. Although the famous actress did not have her name registered as a trademark, the ruling cited that her name had secondary meaning and therefore the actress possessed "trademark rights" to her name. Administrative panels from the Uniform Domain Name Resolution Policy (UDRP) and the Anti-Cybersquatting Consumer Protection Act (ACPA) transferred ownership of the domain name to the actress.

Another case involving a celebrity name is `jimihendrix.com`. The estate of Jimi Hendrix complained under the UDRP that a Web site created by a supposed fan was actually being used to sell other celebrity domain names. The panel decided to transfer the domain name to the estate of the deceased rock star, because the "fan club" was not registered in the state in which the domain owner lived and that the domain name was being used in bad faith.

www.it-ebooks.info

Under the UDRP when a domain name complaint is filed, the domain name owner is obligated to attend a mandatory hearing conducted by an administrative dispute resolution service approved by ICANN (Internet Corporation for Assigned Names and Numbers).

A complaint may result in action against the domain name holder if the resolution service finds that the domain name is identical or similar to a trademark name to which the plaintiff has rights, if the domain name owner has no rights or legitimate interest in the domain name, or if the domain name has been registered and is being used in bad faith. During the administrative hearing the plaintiff must prove all three elements.

Anti-Cybersquatting Consumer Protection Act

In November 1999, President Bill Clinton signed the Anti-Cybersquatting Consumer Protection Act into law. This law gave citizens the right to seek legal action if a trademark name or famous name is being used in bad faith. This act amends section 43 of the Trademark law that prohibits bad faith registration or the use of a domain name that is identical to a registered trademark, identical or disconcertingly similar to a distinctive mark (registered or not), or is misleadingly similar or damaging to a famous mark or name.

When a case is ruled in a plaintiff's favor, rewards issued under the Trademark act are available. Alternatively, the plaintiff may elect an award of statutory damage between $1,000 and $100,000 per domain name (the amount is awarded at the discretion of the presiding judge). The plaintiff may also be awarded the transfer of the domain name. Under the Anti-Cybersquatting Consumer Protection Act, a trademark holder may file suit in federal court under an explicit provision of the act without having to resort to a traditional trademark infringement or dilution action.

Researching domain names online

When you decide to set up an online presence, the domain name you choose can bolster your Internet presence. If you choose a short domain name that fits the service or product you're intending to promote on the Internet, it's the equivalent of an online trademark. Using a trendy domain name is fine as long as you don't try to intentionally create a reasonable facsimile of an existing trademark, or use a celebrity's name. When choosing a domain name, it's a good idea to come up with a couple of derivatives of the domain name and then research whether or not the name is available. You can research a domain name at Better-Whois.com (`www.betterwhois.com`) (see Figure 26-2).

Maintaining your domain name

You register a domain name for a given period of time, the shortest period being one year. After your registration expires, your domain name becomes available. Even if you don't set up a Web site with the domain name prior to the name expiring, it's a

www.it-ebooks.info

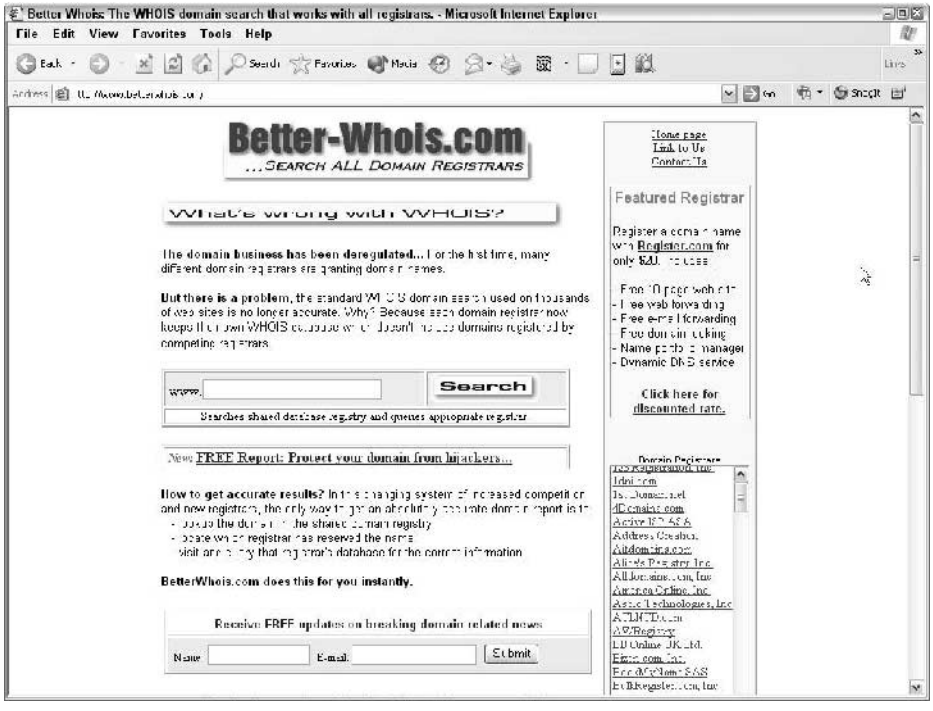


Figure 26-2: You can find out if a domain name is available at Better-Whois.com.

good idea to renew the registration, especially if it's a good domain name. It's especially important to renew the domain name if you've been running a Web business under the domain and have temporarily taken the Web site down. It has been a common practice of Internet porn sites to scoop up domain names when they become available and point the domain name toward the server of their site. If your domain name lapses and so-called porn pirates purchase your domain, your previous customers will be in for a shock.

When you register a domain name, you may also want to consider registering derivatives of the name. If your site becomes popular, porn pirates can register derivatives or your domain. When people mistype your domain name, the porn site appears in their browser.

Another common practice is to register the domain name with the other domain extensions. In addition to registering your site with the `.com` extension, you should consider registering it with a `.net` and `.biz` extension. You may also want to consider registering your domain name with the `.org` or `.info` extensions.

Researching trademarks online

To avoid getting into a trademark/domain name dispute in the first place, you could hire a law firm specializing in trademark law to investigate all the possibilities for you. But that's an expensive proposition. It's far better to do the basic research

www.it-ebooks.info

yourself. Then, if you think you have a trademark-free domain name, have the lawyer confirm it—and trademark it for you, just to cover all the bases.

The first step in doing the trademark research is the most simple and obvious—go to any major search engine and punch in the name you want to use. If the search results show that someone is already using that name for their product or service, then it's back to the drawing board, unless your product or service is radically different, and that's a matter for a qualified attorney to decide.

If you can't find anyone already using the name, you're not out of the woods yet. You still need to check out the listings of current trademarks, such as the one at the U.S. Patent and Trademark Office (`www.uspto.gov`) (see Figure 26-3). Table 26-1 lists the addresses of several trademark databases, including official and international ones. While some are free, others are fee-based. Be sure to read all the fine print before committing to using any of them.

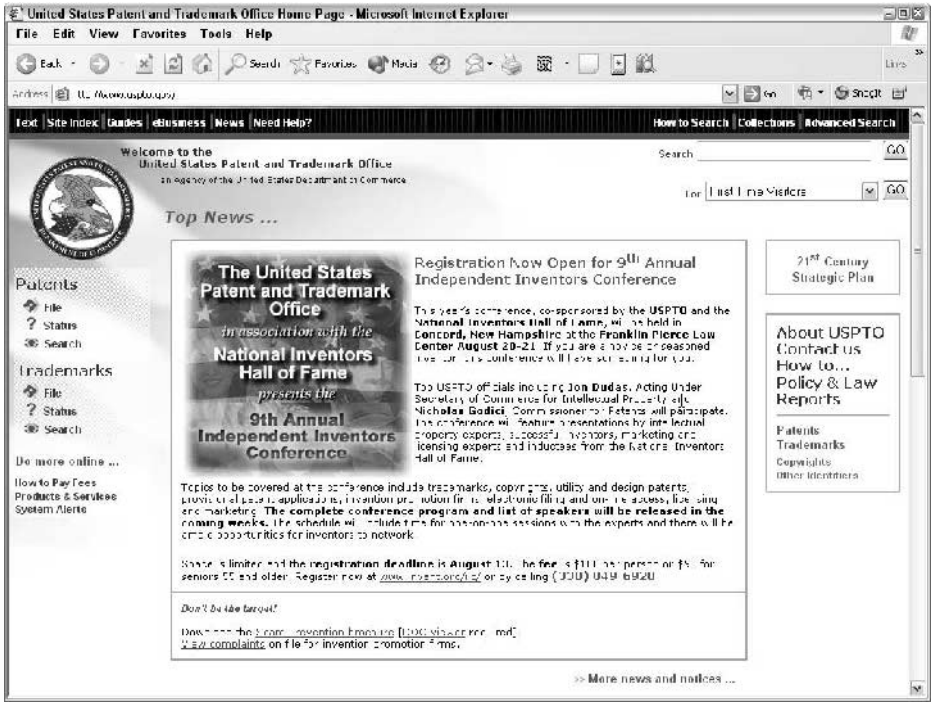


Figure 26-3: The Trademark Office provides a variety of search options.

Soothing Common Fears

While you may be perfectly comfortable with computers, networks, and all the latest technological innovations, the Internet is a strange and scary place for many people. As more and more nontechnical people become Internet users, addressing this common fear is bound to become more of a factor than it is even today.

www.it-ebooks.info

Table 26-1
Trademark Research Sites

Company	*URL*
Australian Trademark On-line Search System	pericles.ipaustralia.gov.au/atmoss/falcon.application_start
Canadian Trademark Database (English)	strategis.ic.gc.ca/cgi-bin/sc_consu/trade-marks/search_e.pl
Canadian Trademark Database (French)	strategis.ic.gc.ca/cgi-bin/sc_consu/trade-marks/search_f.pl
DontNeedALawyer.com Free Trademark Search	dontneedalawyer.com/freetmsearch.html
European Community Trademark Consultation Service	oami.eu.int/search/trademark/la/en_tm_search.cfm
Intellectual Property Office of New Zealand	www.iponz.govt.nz/search/cad/dbssiten.main
Japan Patent Office	www.jpo.go.jp/
The American Trademark Company	trademrk.com
U.K. Patent Office	webdb4.patent.gov.uk/tm/number
U.S. Patent and Trademark Office	www.uspto.gov/

Publicizing your security

Except for the rare cases where a Webmaster is just plain crazy, all Internet sites have a fair degree of security. If you run your own Web server, you've doubtless seen to all the details yourself. If you're leasing space on a virtual server or other Web hosting service, presumably standard security measures are a part of the package. If not, move on to a better provider—you're not getting what you're paying for.

The general public, fed by Hollywood's "vision" of the Internet as a place where anyone who feels like it can break into any system, often doesn't understand that something as simple as encrypting an order in transit is all they need to be protected. Yes, there are plenty of talented hackers out there who can slip past just about any security setup. But most of them aren't malicious and few of them are criminally inclined. They're far more likely to try and break into the United States government's computers just to prove they can do it than trying to find out who's ordering kippered herrings from an online grocery store.

Of course, the real fear most people have is that someone will get their credit card information. Why this fear has become associated with the Internet is anyone's guess because you give that information to strangers every time you use your credit card anywhere. Thousands of "someones" have access to your credit card information already, even without shopping online.

www.it-ebooks.info

And you probably literally throw it away every time you take out the trash. Seriously. Take a good look at your receipts. Some of them are done up right, listing only partial information, just enough to identify the credit card—such as the last four digits—in case you need to make a return. Many others, however, give your full name, credit card number, and expiration date. When you wad up that shopping bag and toss it into the same trashcan you throw your junk mail into, you've given any trash raider all they need to buy anything they want—on your credit card. It's much more likely that criminals will go through your trash than intercept an online transaction or raid a database (this is not a joke—ask any detective who handles financial crimes).



Although this book's about the Internet, identity theft is a very real issue. To thwart the trash raiders, buy a paper shredder and shred anything that can be used to usurp your identity, including the endless barrage of credit card offers that appear in your mailbox.

Nonetheless, as a Webmaster, you have to deal with the public perception, not the reality. And that means making a big deal about how secure your site is. Lay it on as thick as you honestly can. Display the logo of the company, such as VeriSign (`www.verisign.com`), that issues your digital certificate (see Figure 26-4). If your site uses 128-bit encryption, tell everyone that it's so good that the U.S. government won't allow it out of the country (it gets out anyway, of course, but that's another

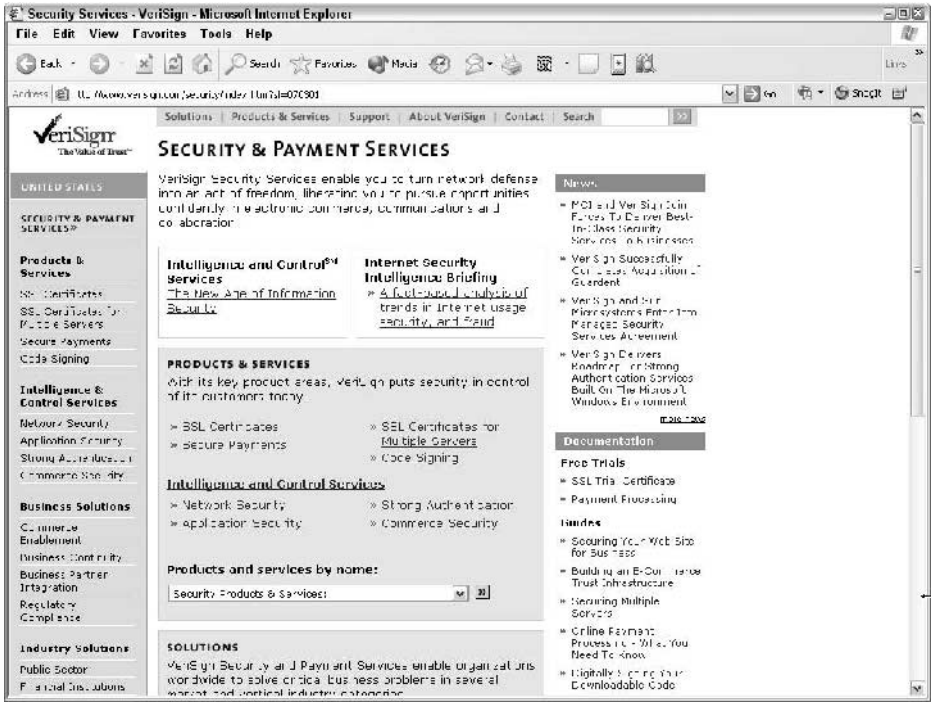


Figure 26-4: VeriSign provides digital site identification.

www.it-ebooks.info

story). Make a special page with links to every technical report on encryption you can find—the more obtuse, the better. Your aim here is not technical education, but consolation. Just let visitors know that they're as protected as it is possible to be.

Creating a privacy policy

The first thing we look for before we give any personal information to any Web site is the site's privacy policy. Why? If the site doesn't specifically state that they keep your information confidential, you don't know who will end up with it.

We're hardly the only people with this attitude—Internet privacy is one of the hottest issues going. You're well advised to give serious consideration to your own site's privacy policy and to ensure that your visitors are well aware of it. Even if you don't run a commercial site, even if you don't sell anything to anyone, if you ask your visitors for any information at all, you need a publicly posted privacy policy.



Put two links to your privacy policy on every page where you ask for personal information—maybe even on every page in your entire site. Put one at the top of the page and the other at the bottom. That makes it easy for anyone to find one of them quickly.

Why ask?

An astonishing number of Web sites ask their visitors to provide personal information when they have no legitimate use for it. Or they require their visitors to register, selecting a user name and password in order to access the site, even though the site is free.

Think long and hard before you do either of these things. Ask yourself, "What am I gaining by doing this?" Then ask yourself, "Is that worth putting my visitors through an extra hassle?" Because everything that they have to do that puts an unnecessary step between them and why they came to your site is a hassle.



Instead of creating your own privacy statement, you can save a lot of work by using the Direct Marketing Association's Privacy Policy Generator, which can be found at `the-dma.org/privacy/creating.shtml` (see Figure 26-5).

Table 26-2 lists the Internet addresses for several privacy policies representing a variety of different companies and organizations.

Establishing Return and Refund Policies

If you run any kind of commercial site, you will, sooner or later, have to deal with some dissatisfied consumers. That ultimately means, whether you offer products or services, that you have to handle returns, exchanges, and/or refunds. If you have no

www.it-ebooks.info

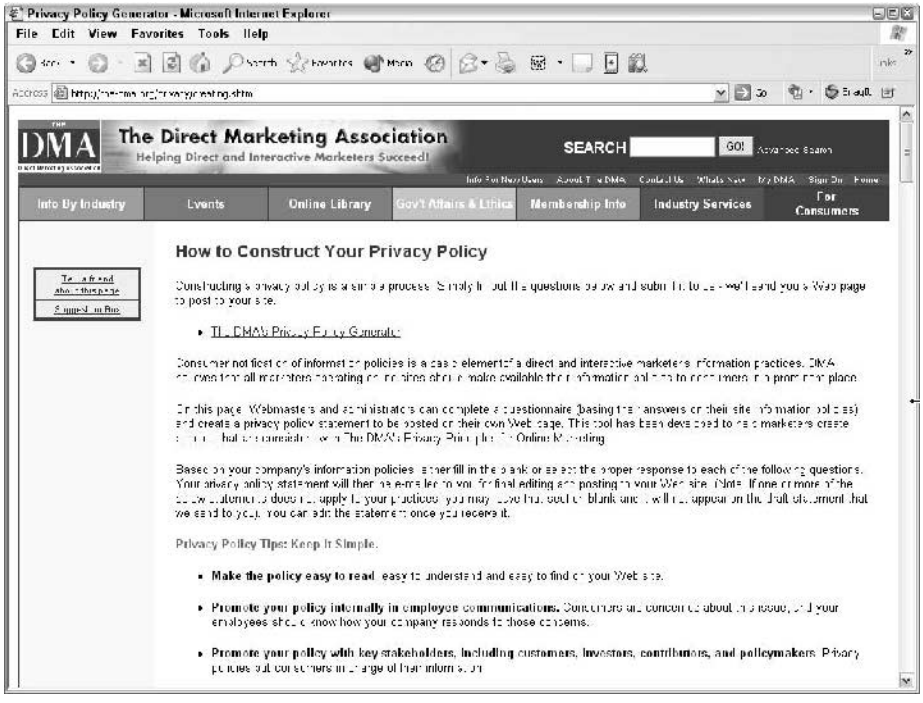


Figure 26-5: The Direct Marketing Association helps you create a privacy policy.

Table 26-2
Example Privacy Policies

Company	*Policy URL*
Bloomberg.com	www.bloomberg.com/notices/privacy.html
Gibson Research Corporation	grc.com/privacy.htm
JobBankUSA	jobbankusa.com/privacy.html
Lucent Technologies	lucent.com/privacy.html
Marriott Rewards	http://marriott.com/privacy.mi
NASA	http://image.gsfc.nasa.gov/image_policy.html
Red Hat Linux	redhat.com/legal/privacy_statement.html
SportsLine.com	www.pgatour.com/info/privacy
The Free Garage Sale	freegaragesale.net/privacy.htm
The Internet Society	isoc.org/help/privacy
Tucows	tucows.com/privacy.html
United Parcel Service	www.ups.com/about/privacy.html

www.it-ebooks.info

policy in place for handling these situations, your bottom line could suffer, both because you may have to issue refunds you hadn't planned on and because many people will be reluctant to buy from you at all. You need to decide not only if you will accept returns, but also details such as the following:

- **Type of settlement:** On returns, will you offer a full refund or store credit only? Do you want to charge a restocking fee? Will you only allow exchanges for identical products in the event of a defect? Will you charge a repackaging fee for opened items?
- **Time limits:** You probably don't want every sale to perpetually be a potential return. Set some kind of time limit on it, so that you don't have to maintain a large cash set-aside to cover returns long after the sale is made. Many merchants traditionally give 30 days for returns. Some go as high as 90 days, and only the most confident (and those with the deepest pockets) set no limits. Certain types of items that are particularly susceptible to return fraud are often given very short time limits. The number of camcorders people attempt to return right after graduation day, for instance, is incredible. Unless you actually plan to run an equipment rental facility, set tight limits on anything that is vulnerable to use on a short-term basis.
- **Shipping charges:** Are you going to "eat" the shipping charges for a returned item? Some companies, in the interest of maintaining good customer relations, make this a practice. Others, in the interest of keeping their bank accounts full, avoid it like the plague.
- **Authorization:** You don't want to be caught off guard by returns. Most online companies only allow returns if the customer first contacts the company and explains the situation. Many times, any problem the customer has can be handled at this point, and the return avoided (minor details such as "Have you pressed the On button, sir?"). If the return is actually necessary, the company issues a Return Material Authorization (RMA) number. This number must be clearly written on the outside of the box when the customer returns it. Return departments automatically refuse to accept delivery of any item without an RMA number on it. And of course you'll want the customer to include a copy of the invoice with his or her return.

Whatever you decide to do in regard to your return policy, it should be prominently posted on your Web site. Presumably, most of your sales will be with credit card, and credit card companies automatically back up your return policy in the event of a complaint from the cardholder—but only if it is publicly available.



Make sure that you check out applicable state laws regarding returns. You may have to modify your return policy because of them.

If you have decided to set up shop in an established mall such as Yahoo!, you'll have to accept the Yahoo! terms of service agreement (see Figure 26-6). Violation of a mall's requirements generally results in losing your membership, which effectively means shutting down your business. Some malls, however, allow each merchant to

www.it-ebooks.info

set their own individual policies. Check out all the details before you decide to open a store in a mall.

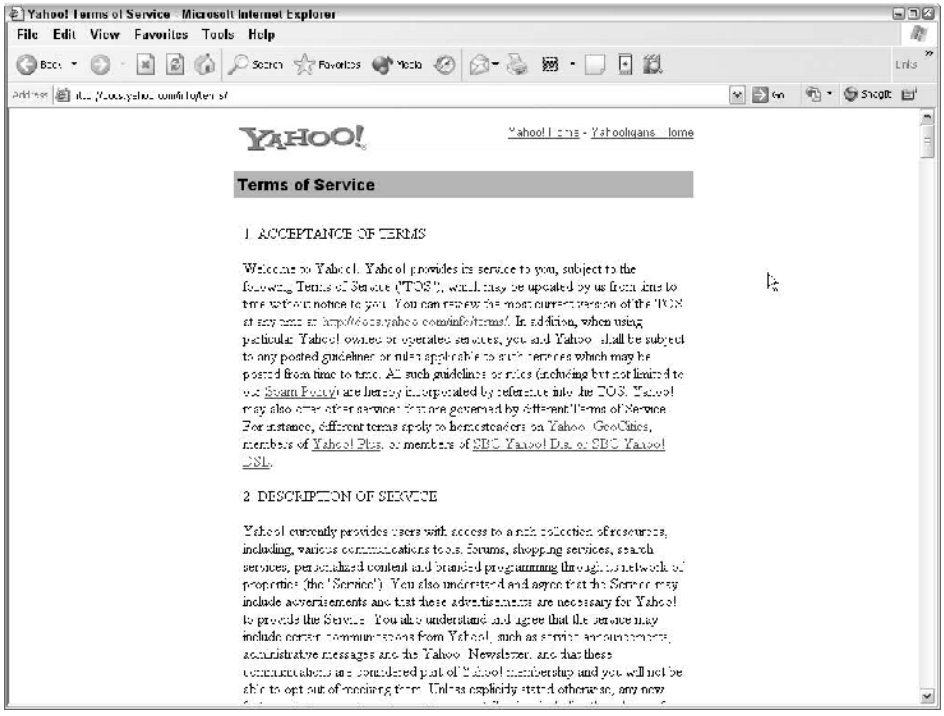


Figure 26-6: Yahoo! Shops includes a terms of service policy.

Table 26-3 lists the locations of a number of existing return policies that you can browse through for inspiration.

Table 26-3
Example Return Policies

Company	*Policy URL*
AvShop	avshop.com/avshop/returnpolicy.html
Bakery Equipment.com	bakeryequipment.com/Customer-Service/Sales-Agreement.htm
DreamRetail.com	dreamretail.com/returns.html
Eddie Bauer Outlet	https://www.eddiebaueroutlet.com/custserv/eb_returnpolicies.asp
Hotel Supplies-Online	hotelsupplies-online.com/returnpolicies.htm
Radar City	radarcity.com/static?file=refund.html

www.it-ebooks.info

Investigating Alternative Approaches

If you don't intend to run a commercial site, but still enjoy the idea of getting at least some money out of your Web site, you can take a couple of simple approaches. The first is to display advertising, making your site into a sort of online bus stop or billboard. The other is to refer your site's visitors to another company's products or services. Both are a type of advertising, of course, but they work slightly differently.

Accepting advertising

Banner advertising has long been an accepted part of the World Wide Web. Some banner advertising arrangements, called *exchanges*, are formed by groups of companies or even individual Webmasters, who agree to display small advertisements for each other's sites. The banners are rotated automatically, and the number of sites your banner is displayed on depends on the number of other members' banners you display on your own site.

Banner exchanges tend to have, by and large, pretty short lives and are of questionable usefulness. Too often, about all you end up doing is showing ads for the exchange itself.

You can, however, pay for banner ad arrangements, such as those provided by Advertising.com, one of the most professionally run outfits in Internet advertising (see Figure 26-7). They work both ends of the street, bringing together serious advertisers with both significant Web sites and opt-in e-newsletters.

Companies such as this save you the trouble of finding advertisers who are interested in buying your space. These companies assist you in finding the right advertisers for your site's target audience. Usually, you have to demonstrate that you have a certain amount of steady traffic on your site in order to attract any kind of serious advertising. Just as a magazine must prove circulation figures, a Web site must be able to provide server logs upon request that show how many visitors come to the site each month. The number varies depending on the advertiser, but if you're not generating over 50,000 visits a month, you're not even going to be close to making any money.

Table 26-4 lists several paying banner advertising resources for Webmasters.

Optimize Your Site for Search Engines

If you build it, they will come. Well not quite. In order to assure a steady stream of traffic to your Web site, you'll have to take certain steps. The obvious steps are to include the URL for your Web site in every piece of literature you mail, on your business cards, and in every e-mail you send. The other step you should consider taking is optimizing your Web site for search engines.

In the old days, Web developers resorted to all manner of chicanery to boost their Web sites as close to the top of the first results page when a given keyword is

www.it-ebooks.info

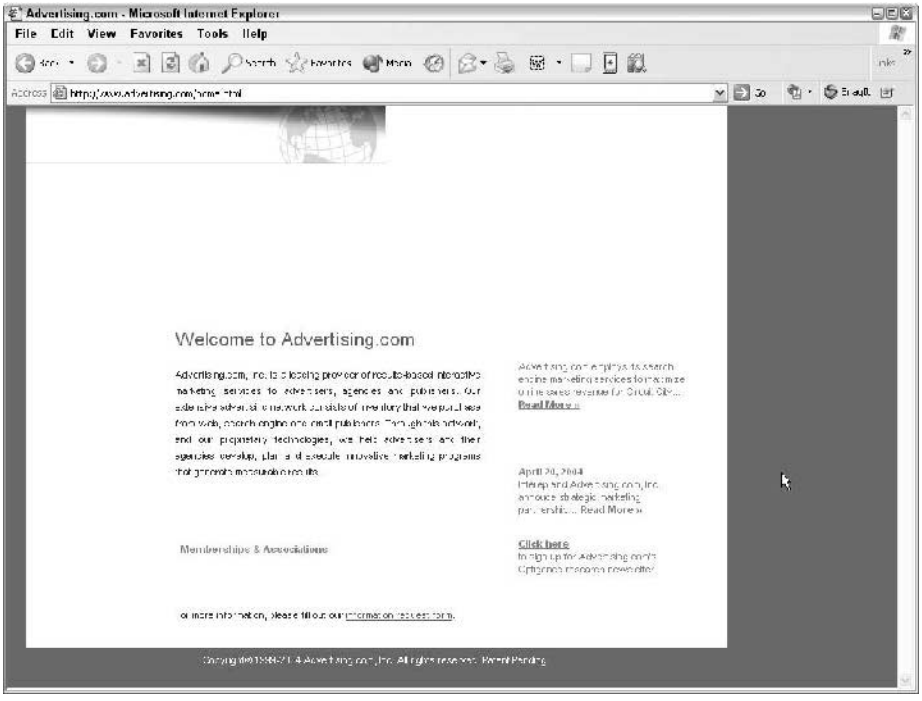


Figure 26-7: Advertising.com provides a variety of services to Internet publishers and advertisers alike.

Table 26-4
Banner Advertising Resources

Company	*URL*
Advertising.com	`Advertising.com`
PennyWeb	`home.pennyweb.com`
The Financial Ad Trader	`adtrader.com`
ValueClick	`valueclick.com`

entered into a search engine search field. The methods were rather blatant, repeating the same keyword over and over on the home page and using a color that was so close to the background color, visitors couldn't see the keyword. But the search engine robots could, and therefore the Web page jumped to the top of the hit parade every time that keyword was entered in a search engine.

Times have changed and search engine companies are wise to the tricks of the trade. Sites that resort to this type of chicanery are now booted from a search engine database. So how do you optimize your site for popular search engines?

www.it-ebooks.info

You can purchase software to optimize your Web site. One popular search engine optimization tool is called Web Position Gold. You can download a trial version of the software at `www.webposition.com` (see Figure 26-8). The standard edition of Web Position Gold supports up to 5 different Web sites and retails for $149.00. The professional edition supports unlimited domains and retails for $349.00.

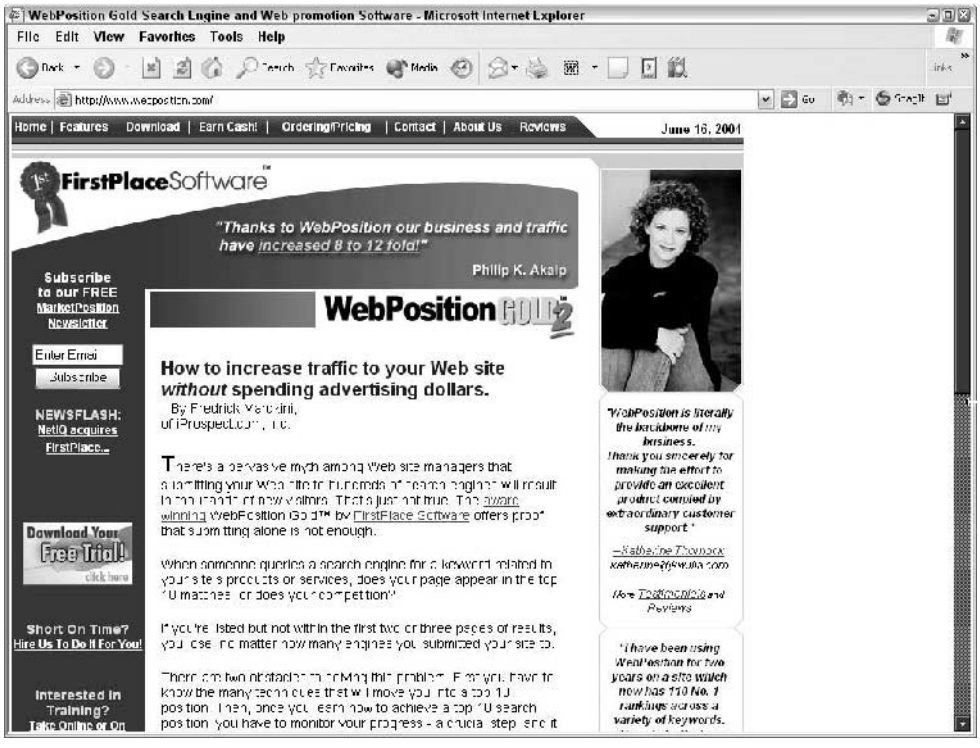


Figure 26-8: You can download a trial version of Web Position Gold.

If you don't like the idea of working with a piece of software, you can hire a company to optimize your Web site, and then submit it to the most popular search engines. The options vary depending on the company. Some will repeatedly submit your Web site to popular search engines, while others will offer tips for optimizing the keywords and home page text to boost your search engine rating. Table 26-5 lists some companies that provide search engine optimization services. The cost of submission and optimization services vary depending on the number of search engines the site is submitted to. You can find submission services for as low as $39.00.



Before hiring any search optimization company, it's a good idea to ask the company for referrals before plunking down your hard-earned cash.

www.it-ebooks.info

Table 26-5
Search Engine Optimization Companies

Company	*URL*
360i.com	www.360i.com
dMedia	www.dmediallc.com
HighRankings.com	www.highrankings.com
Reinventbusiness.com	reinventbusiness.com
SEO	www.seoinc.com
Submit Express	www.submitexpress.com
SubmitToday.com	www.submittoday.com
WebPublicitee.com	www.webpublicitee.com
WPromote.com	www.wpromote.com

Summary

The new Web business model isn't something that's suddenly going to overwhelm the old one. It needs to fit into the existing one.

This chapter introduced you to the following concepts and topics:

- ✦ Trademarks and domain names aren't the same thing. If you want to trademark your domain name, you'd better talk to an attorney who specializes in that sort of thing.
- ✦ Online security is a major concern with the majority of shoppers. If you offer online security, but don't actively publicize it, you're making a major mistake.
- ✦ Make sure that you keep your visitors' private information private—and make sure that you tell them up front that you're going to do so.
- ✦ Your return policies must be stated clearly and plainly, and they must be in accord with all the state laws of the states you're doing business with.
- ✦ You can add to your site's revenues by accepting advertising—either direct or via affiliate programs.
- ✦ You can boost your site's ranking in search engines by using optimization software or by hiring online optimization companies.

✦ ✦ ✦

www.it-ebooks.info

PART

VI

Maintaining and Updating Your Site

In This Part

Chapter 27
Maintaining Your Site

Chapter 28
Updating Your Site

Chapter 29
Designing with XML

Chapter 30
Making Your Site Mobile with WAP/WML

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 27

Maintaining Your Site

In This Chapter

Testing your site

Coping with visitors' changing your site

Maintaining consistency on your site

Even though the bulk of the work you perform on a typical Web development project is contained within the actual creation of the site, a fair amount of time still needs to be devoted to site maintenance. In fact, it is the maintenance of a site that can often make or break it. This chapter and the following one focus on ways to lengthen your site's functional life span through effective site maintenance and updating.

Testing Your Site

Before uploading your site to a server and taking it *live* for all to see on the Internet, it is important to do some thorough testing. At this point, you want to determine that everything is working as you anticipated by ensuring the following:

- There are no broken links.
- The download times are acceptable for the targeted audience.
- All content is legible and functional in targeted browsers.

When you begin testing your site, it is helpful to have a table or spreadsheet containing information about your testing. For instance, you can create a table that lists your targeted browsers across the top and targeted platforms down the left side. After you test each one, record your findings in the appropriate spaces. Table 27-1 gives you an idea of how this displys.

Not all of the listed browsers and platforms will be appropriate for your testing situation. Ultimately, the browsers and platforms you test should be representative of your target audience. To help determine the most important browsers and platforms for your testing scheme, prioritize them according to which constitutes the widest portion of your audience.

www.it-ebooks.info

Table 27-1
Testing Spreadsheet

	Netscape 4.x	*Netscape 6.x*	*Netscape 7.x*	*IE 4.x*	*IE 5.x*	*IE 6.x*	*AOL*
Windows 95							
Windows 98							
Windows 2000							
Windows XP							
Mac OS 8.*x*							
Mac OS 9.*x*							
Mac OS 10.*x*							



For a review of Internet demographics, refer to Chapter 1.

You may find that certain newer technologies fail to perform in older browsers. For example, while the use of cascading styles sheets (CSS) provides more attractive pages in the newest browsers, you must also ensure that your pages "fail gracefully" in browsers that do not recognize or support CSS. If users of older browsers cannot see the newer technologies you're using, they should still be able to read and view the content in some form.



Not sure how to test multiple platforms when you have only one or two computers? In the section "Finding Outside Testers," you'll learn ways around this.

In addition to browsers and platforms, you must also test a variety of screen resolutions, as dictated by your target audience. At the very least, you should test your site under the following resolutions, using both Internet Explorer and Netscape.

- 640 × 480
- 800 × 600
- 1024 × 768

Visit your site as a stranger

In Chapter 1, we discussed Internet demographics, specifically as they relate to knowing your target audience. When you test your site, you must do so with that target audience in mind.

www.it-ebooks.info

Take a break from your site for a few days and go back to visit it as if you were seeing it for the first time. One way to make this task easier is to assign a role for yourself within the target audience, often referred to as a *user scenario* (see the sidebar for a typical example). For instance, if you are creating a site for a toy company whose target audience includes children ages 8 to 12, you might assign yourself the role of a nine-year-old child who is visiting the site in search of ideas for birthday presents.

User scenario

Web Site: Toy Store

Role Name: Tommy Jones

Role Description: Tommy is a nine-year-old boy who is visiting the site for the first time. Tommy has used many similar Web sites, mostly with the help of his parents. For this particular after-school visit, Tommy is looking for things to put on his birthday wish list. He plans to share the wish list with his parents at dinner tonight.

It can be quite useful (and even fun) to visit your Web site with the fresh perspective of a user scenario. By putting yourself in the shoes of a member of your target audience, you can identify elements that don't work as you intended, or sections that need to be better clarified. In our own work, we prefer to use several user scenarios, representing each major group in the target audience.

Finding outside testers

Once you've performed your own testing, be sure to enlist the help of others. This can be accomplished through formal or informal user testing, depending on the size and complexity of your company and your Web site.

In formal user testing, you hire users from the target audience to perform certain tasks on the new Web site. So in the toy store scenario, you may decide to hire several parent/child teams to test your site from both users' perspectives. A large number of testers are not required. A group of 5 to 10 people can provide realistic results or recommendations to greatly improve your site's usability.

Testers are typically paid somewhere between $25 to $50 for an hour or two of testing. You can usually find the best testers by advertising to your target audience. If your site is for a toy store, consider advertising at a local toy store or in the local newspaper, or simply ask friends and co-workers.

Prior to hiring the testers, create a list of key tasks, in order of importance, that you expect typical users to accomplish on your site. As your testers follow the key test list, observe them as they attempt to complete each task and record any relevant information. The sidebar "Record from user testing" gives a sample record from a user test performed on the fictional toy store Web site we've been using.

www.it-ebooks.info

Record from user testing

Web Site: Toy Store

Tester: Tyler Hall, age 10

Task #1: Locate a pinball game for Nintendo 64 and add it to your shopping basket.

Observations: *From the home page (which was already loaded on the screen), Tyler quickly clicked the link for Video Games. After the Video Games page was loaded (it took 6 seconds), he read the page for 10 seconds before clicking a link for Nintendo 64. The Nintendo 64 page took 5 seconds to load. From here, Tyler couldn't decide where to go. He read the category titles, but none matched "pinball." He also saw an alphabetical listing and finally decided to click "p" for pinball. When the "p" page was loaded, he read through all the titles listed (approximately two minutes) before giving up. Observer note: None of the pinball titles for Nintendo 64* ***begin*** *with the word "pinball," which is why they were not in the "p" section. Tyler could have found it by performing a search but when asked why he hadn't done so, he replied, "I didn't see the search box."*

To create the best situation, you should have the testers in the same room as your observers so you can control the computer setups, interruptions, and environment. Even better is to have a testing room with a mirrored glass wall behind which observers can sit, so they don't distract the testers. You may be able to rent such a room in your area.

In a more informal approach to user testing, you may simply ask a number of your friends or co-workers to test the site for you. This type of testing can be extremely helpful, particularly if you need to test your site under the AOL browser on a Mac but you have only a PC, for instance. You can also walk through the site with the person over the phone and record his or her comments in a similar fashion as for the formal testing.

Yet another way to use outside testers is through surveys or polls on your Web site. You may have seen something similar on other Web sites, particularly if you've done any shopping on the Web. Many online stores ask you to fill out an evaluation form, which is similar to comment cards you obtain at traditional stores.

One of the most popular evaluation systems for online stores is BizRate. Many stores use this company's services to gather feedback from shoppers. Then, potential shoppers can view the company's *rate card* at BizRate to determine whether or not to do business with it. A typical rate card at BizRate includes the following information from previous shoppers:

- ✦ Overall rating
- ✦ Ease of ordering
- ✦ Product selection
- ✦ Product information

www.it-ebooks.info

- ✦ Price
- ✦ Web site performance
- ✦ On-time delivery
- ✦ Product representation
- ✦ Customer support
- ✦ Order tracking
- ✦ Shipping and handling



For more information about BizRate, visit `www.bizrate.com`.

If you decide to create your own online form for a similar purpose, consider asking the same type of questions of shoppers. If you ask specific questions, you're more likely to receive specific answers. In other words, it is more effective to post a form with questions such as "How would you rate the download speed of our new site?" instead of "What do you think about our new site?"

If you're interested in asking a single question of your visitors, consider using a poll. If you don't have the experience to write a CGI script to process your poll or survey, plenty of free or inexpensive options are available online. For example, companies such as Sparklit or Alxnet offer free polls and surveys (among other types of scripts) that are easy to use and customize.



For more information on Sparklit or Alxnet, visit www.pollit.com or www.alxnet.com. For additional links to similar companies, visit http://dir.yahoo.com/Business_and_Economy/Business_to_Business/Computers/Software/Surveys_and_Polling/.

Evaluating input from testers and visitors

Once you've gathered information from testers and visitors, it's important to evaluate the findings and follow up with changes to your site as needed. The best way to prepare for an evaluation is to set up a series of success metrics prior to launching your test, poll, survey, or other feedback form.

Success metrics identify how you plan to gauge whether or not your site has achieved its goals. To determine the success metrics for your site, first review the goals for your site to identify the intended purpose. The following list shows several examples of success metrics for Web sites:

- ✦ Improve consumer awareness of the company among 18- to 24-year-old female Internet users to at least 50 percent, as measured by independent surveys of customers.
- ✦ Register 250 paying members within the first three months of the launch.

www.it-ebooks.info

- ✦ Generate online sales of merchandise totaling $500,000 after six months of operation.
- ✦ Enhance satisfactory customer support for shoppers, as measured through independent surveys of customers.

Once you've identified what you consider to be successful for your site, you can evaluate any feedback you receive. For example, if you have not met your targeted goal of $500,000 in sales after 6 months, you can ask your users why they don't make purchases. In many similar situations, customer satisfaction surveys show that 68 percent of customers feel the ordering process is too difficult. After companies listen to users and alter the ordering process, sales are more likely to reach their targeted levels.

Another key point in evaluating feedback is to set a specific time frame in which you will gather the feedback, and another time frame for evaluating it. If you evaluate responses one by one, it is more difficult to get a clear picture of the results than if you evaluate all of the feedback together. In other words, consider evaluating your site feedback on a weekly or monthly basis, instead of daily.

Finally, don't consider having an e-mail address on your Web site as your only form of gathering visitor feedback. While you may indeed get some feedback through a simple mailto link on your site, you're likely to receive more complaints than constructive feedback. The best way to effectively gather and evaluate feedback is by asking direct questions of targeted users.



Don't forget about your site's server logs! They may also be a source of feedback because they identify concerns such as how many people received errors when visiting your site, how long typical visitors spent on your site, and where they came from. For more information, refer to Chapter 5.

Coping with Visitors' Changing Your Site

One common frustration among Web designers is that visitors to a site have the ultimate authority on how it is displayed. In other designed media such as print advertising, every viewer sees the same advertisement, at the same size, with the same colors, and with the same layout.

The opposite is true for Web design because a single Web page can be interpreted in many different ways, according to the viewer's settings and preferences.

Browser preference settings

Most browsers give viewers the option to adjust preferences related to the display of Web pages. For example, Netscape enables users to specify in which font faces, sizes, and colors they want to view Web sites. Figures 27-1 and 27-2 show how Netscape 7.1 on the PC handles these preferences.

www.it-ebooks.info

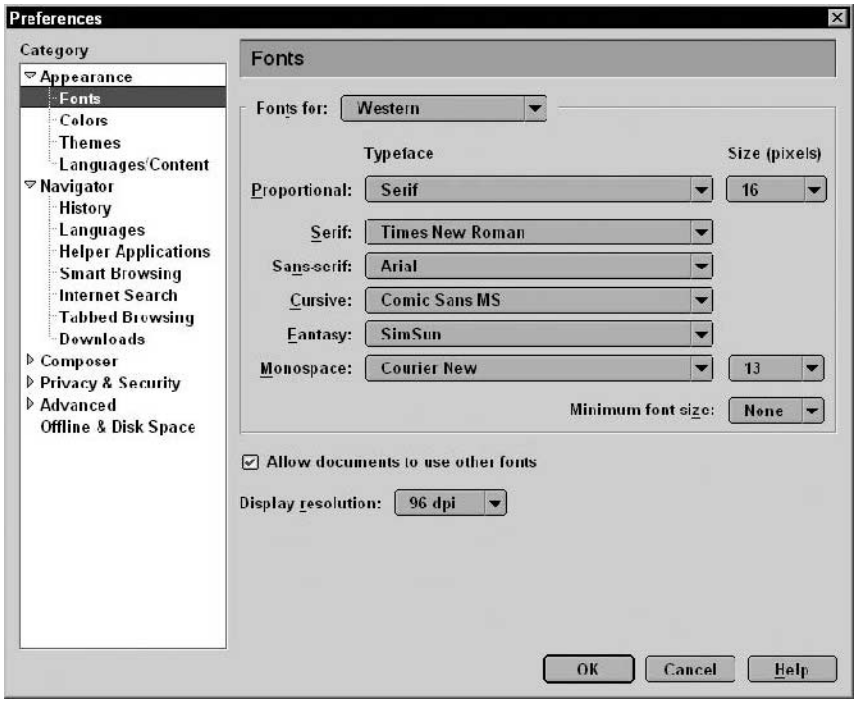


Figure 27-1: Users of Netscape 7.1 can customize font settings.

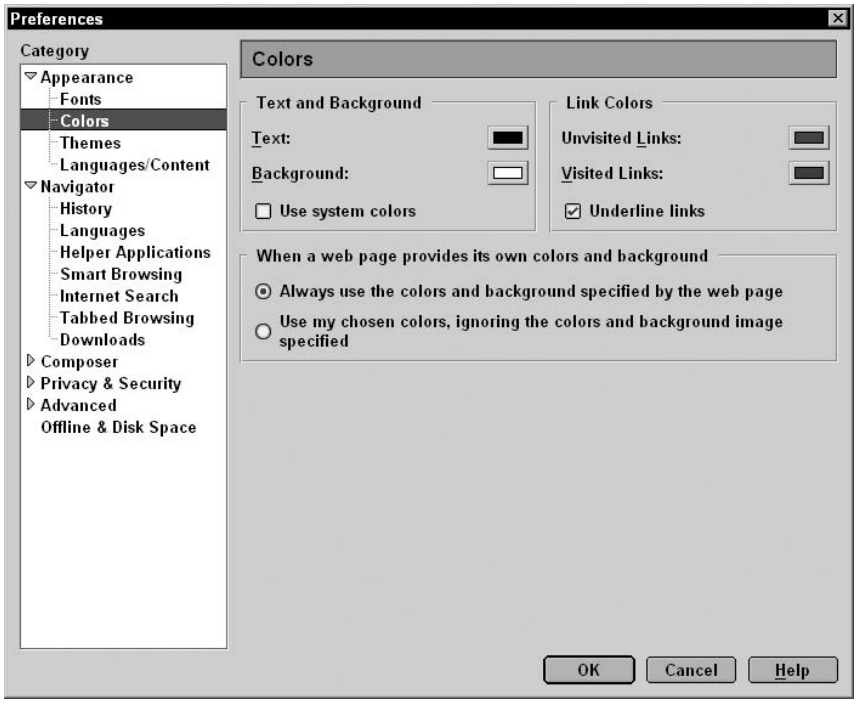


Figure 27-2: Users of Netscape 7.1 can customize color settings.

www.it-ebooks.info

Internet Explorer users can also customize their browsers, although in a slightly different way than Netscape users. Figures 27-3 and 27-4 show some examples of the Internet Explorer preferences in Internet Explorer 6 on the PC.

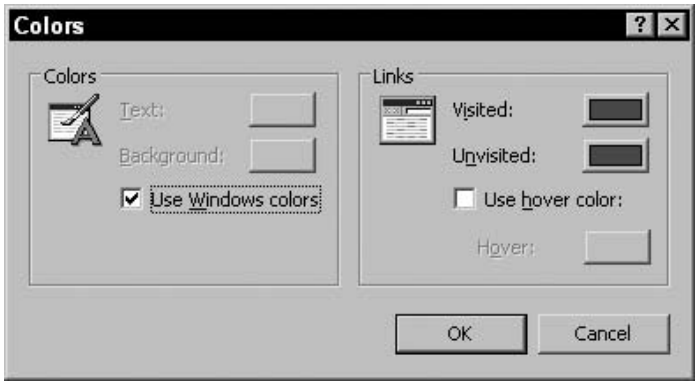


Figure 27-3: Users of Internet Explorer 6 can customize color settings.

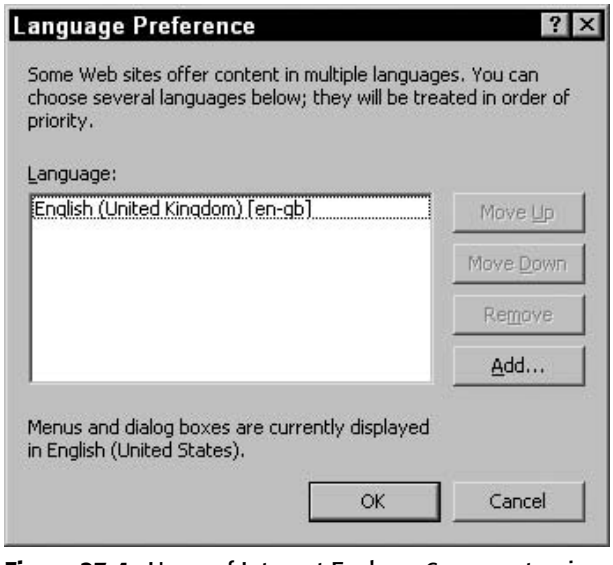


Figure 27-4: Users of Internet Explorer 6 can customize these language settings.

Internet Explorer users can also set up their own personal style sheets to manage options such as the font size, style, and color of the displayed text content.

Our impression is that the majority of Internet users do not customize their browsers in a way that greatly affects the display of your pages. Perhaps the most commonly adjusted setting in the browser is the font size, making it easier for visitors to see

www.it-ebooks.info

pages according to their own needs. As a result, it is probably worthwhile to test your site under several different font settings to ensure that all text is still legible.

Maintaining Consistency on Your Site

One of the beauties of the Web is the ease with which things can be changed. Altough it may be tempting to change your site each time you learn of a new tool, technology, or feature this approach may not necessarily benefit your visitors.

Visitor comfort

A constantly changing Web site cuts into your visitors' comfort level with the site. Maintaining the comfort level on your site is important for two reasons. First, you don't want to adopt any new Web technologies so early that no one knows how to use or comprehend them. And second, you don't want to change the elements of your site so quickly that your visitors never really learn how to use them. For example, if your local grocery store switched the locations of your favorite foods weekly or monthly, you'd probably find shopping there quite frustrating. The same is true of Web sites that change structure and layout too often.

For these reasons, it is worthwhile to limit major upgrades to the structure and layout of your site to once or twice a year. In the interim, focus on the actual content of your site, ensuring that it is user-friendly and timely. Gradually updating the content of your site over time and fixing minor problems as you encounter them is called *site maintenance*.

Common maintenance tasks

The following list provides an overview of some common maintenance tasks:

- ✦ Archive or update news/press releases
- ✦ Update job listings
- ✦ Update contact information (such as e-mail addresses, mailing addresses, and phone and fax numbers)
- ✦ Revise home page copy (such as a daily/weekly/monthly feature or article)
- ✦ Publicize events (perhaps on a calendar)
- ✦ Add new links
- ✦ Verify existing links
- ✦ Remove outdated links
- ✦ Promote new services/features (within the Web site)
- ✦ Update photos
- ✦ Promote the Web site (on the Internet)

www.it-ebooks.info

The question of whose responsibility it is to perform maintenance on a Web site is not easily answered. Large companies often subcontract the work to consultants or other firms. Smaller companies sometimes add this task to an existing employee's responsibilities. However, you can hire a company that focuses specifically on Web site maintenance tasks.

Site improvement tools

If you decide to perform your own site maintenance, many tools are available to assist you. To make changes to your HTML, you'll need some type of HTML editor. Many HTML editors offer specific site maintenance tools such as link and HTML validators, so make sure you check for those before looking for (and possibly paying for) other tools. In addition, don't forget to check with your host company—many of these companies offer maintenance tools with hosting packages.



Services in the improvement category of maintenance tools include a full range of tools from link validators to traffic analyzers.

Several companies offer such services. Here's a list of some of the more popular ones:

- Doctor HTML (`www2.imagiware.com/RxHTML/`)
- LinkAlarm (`www.linkalarm.com`)
- Net Mechanic (`www.netmechanic.com`)

So far, the tools we've listed are run online. However, you can purchase several quality products to install on your own system. Typically, products such as the following are used by Webmasters, who want to manage their own site maintenance:

- WebTrends by NetIQ (`www.netiq.com/webtrends`)
- WebQA by Watchfire (`www.watchfire.com/products/desktop/webqa`)
- WebMaster by Coast Software Inc. (`www.coast.com`)
- CyberSpyder by Aman Software (`www.cyberspyder.com`)

Server monitors

A relatively new breed of online maintenance tools involves third-party server monitoring. Companies such as @Watch (`www.atwatch.com`) monitor your Web server, checking at regular intervals to ensure that everything works correctly. If they detect an outage, you are notified within five minutes.

One of the biggest advantages of these services is the independent nature of the monitors. Because the monitoring company is not the hosting company, you may be more likely to be notified in a timely manner about lapses in service.

www.it-ebooks.info

Summary

It's not enough to just whip up a good-looking site and walk away from it. You need to thoroughly test it, too. This chapter introduced you to the following concepts and topics:

- ✦ Broken links are one of the most common and irritating problems with Web sites.
- ✦ Test your site with every available computer platform and browser.
- ✦ Ask outsiders to test your site and provide you with feedback.
- ✦ Keep your site consistent and fresh.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 28

Updating Your Site

In This Chapter

Keeping your site fresh

Adding extra value

Facing the XML future

Transitioning to XHTML

The immediacy of the Internet demands that a site must be updated regularly. Creating, publishing, and publicizing a Web site are only the initial steps in the evolution of your site. When customers arrive at your site, they believe that the content is *live*, meaning that at any moment the site can change to reveal new and engaging content. This belief is impressed upon them by sites such as CNN.com, CNET.com, eBay and the thousands of other sites that are driven by rapidly changing content.

Not only must Web sites be prepared for rapidly changing content, they must also be prepared for rapidly changing technology. Three years ago a site would need to leverage HTML and JavaScript in its construction, whereas one year ago HTML 4.0 (HTML, cascading style sheets, and Document Object Model Script) would have been the flavor of code required for the construction of a site. Today, the technologies required for building a site range from HTML 4.01 to WML (the Wireless Markup Language) to XML (the eXtensible Markup Language) to XHTML (eXtensible HTML) to the dozen of XML-based languages. The new languages are needed because of the massive deluge of Internet appliances.

Keeping Your Site Fresh

There are a number of ways to provide new and fresh content on a site. One simple approach is to set up a schedule for news releases. Updated material ignites communication between a company or another kind of organization and the customer. As an alternative, a Web site can freshen its content by developing encouraging ways for customers to communicate with each other from the site.

www.it-ebooks.info

Using incremental uploads

It is better to have no Web site than to have a site that has not been updated. The Internet is littered with tens of thousands of sites that have not been updated for a year or more. Not updating a site for more than a week should be considered a cyber crime. For a high-traffic site, with perhaps a million page views a week, several changes to the front page of the site should be done each and every week.

Your customers' demands create the need for so many changes to a site. The single reason for the remarkable success of the Internet is that it offers immediate access to knowledge (hence the *information superhighway*). A site that does not provide a customer with the knowledge they are seeking—accurate and updated nearly up to the minute—will not keep that customer.

Information-rich sites, such as Amazon.com (`www.amazon.com`), Barnes & Noble (`www.bn.com`), and CNET (`www.cnet.com`), keep and retain loyal customers not because they necessarily offer the lowest prices but because they are able to provide the most information about a product. At Amazon.com, a user can read reviews of a book. Barnes & Noble provides free education on literature. CNET offers the most up-to-date information and resources for computer users and people in the computer industry.

The online magazine, Slate (`slate.msn.com`), discovered that the traffic to its site increased when the articles for a month were spread out over the course of a month. Customers could wait a day between fresh articles. But a whole month? It just doesn't work on the Internet. A site must be designed with the capability for easily and quickly inserting fresh content.

Establishing a schedule

An enormous amount of work goes into the creation of a Web site. That's why it's a crime when the owners of a site sit back on their laurels and expect the site to run forever without any changes.

Change is part of the Internet. Thus, change must be part of your site. But unlike change on the Internet as a whole, change on a Web site can be managed. The subject matter and content of a site determines the amount of change.

Often, the key to managing change is to produce a schedule for change. Humans are creatures of habit. If a user knows that a favorite Web site changes content on Monday, the user may make it part of their routine to check the site every Monday. Choosing a schedule is dependent on the type of content on a site; for example, a football site will want to synchronize its schedule with key football games.

In creating a schedule, consider the following questions:

- ✦ Do you publish a weekly magazine or newsletter? If you do, break the articles up and publish them to the site during the week.

www.it-ebooks.info

- ✦ Do your customers expect information? Not every site requires daily updates. A company that manufactures zippers may only need to update monthly.
- ✦ Is your site linked to time-sensitive material? A site dedicated to a political candidate will host a considerable number of news releases prior to any voting.

Finding new material

With a schedule defined for delivering material, the next challenge is to find the content. Again, the material is going to be strongly defined by the type of site you have developed.

In creating new material, consider what you sell or otherwise provide on your site. Is your site for a manufacturing company, a medical business, or a financial concern? Each type of site has tools, products, and services unique to that business. Examine each product, tool, and service by defining the what, why, where, and how of each. If your company sells financial services ask these questions for each service. When a customer's question is answered before it can even be asked, the customer's confidence with your product is increased.

A great feature for any site is a "Frequently Asked Questions" (FAQ) section. Ask any help desk or support center for the most popular questions asked by customers. Then list the most popular questions and place them on the Web site. As the site grows, break down the FAQs for each service or product to increase the knowledge base of each of them.

Having your visitors provide material

Creating and generating new material is time consuming and expensive. While new material is necessary, a site does not need to rely on it alone for generating content. Unlike magazines and books, the Internet is computer driven. With this in mind, savvy sites leverage this power to encourage e-customers to add their two cents to a site.

Companies, such as Amazon.com and eBay, learned early that enabling people to comment on products offered on their site empowers the customer. At Amazon.com, a customer doesn't only have to judge a book by a professional book critic's review. Amazon enables users to add their own comments to a book review. The customer is empowered by telling the world her likes and dislikes about a book. And this encourages other people to join the conversation. Before long, the product on sale is a talking point. Amazon capitalized on the Internet's ability to build a community.

Developing online communities that talk and interact with each is a vital part of the Internet. The single largest Internet activity is e-mail. Why? People want to communicate with each other. Community building can be done on any Web site.

Certainly, an excellent technique is to provide an opportunity for any product sold on the site to be not only a selling point but also a talking point. In many ways, acting

www.it-ebooks.info

as a talking point is a litmus test for any product. If the product really stands up to the sales and marketing, the comments posted will be positive and encourage sales. If the comments are negative, the feedback can be used to enhance the product.

Another way to encourage users to comment on the site is through online chat rooms and bulletin boards. This technology is nothing new and proven. Sites such as iVillage.com, the Women's Network, have built their entire business around enabling people to meet, talk, and find information on numerous subjects. The success of iVillage.com is largely due to the categorization of content on its site (see in Figure 28-1). Content is not haphazard. Communities are grouped together. In turn, members are encouraged to stay focused on the subject of the specific community's content. For each community, a specific person is allotted as the site's gatekeeper. This person's role is to help keep the conversation moving if it has stalled, to encourage users to stick to the group's topic, and to police unwanted criticism (such as a user verbally attacking other users).

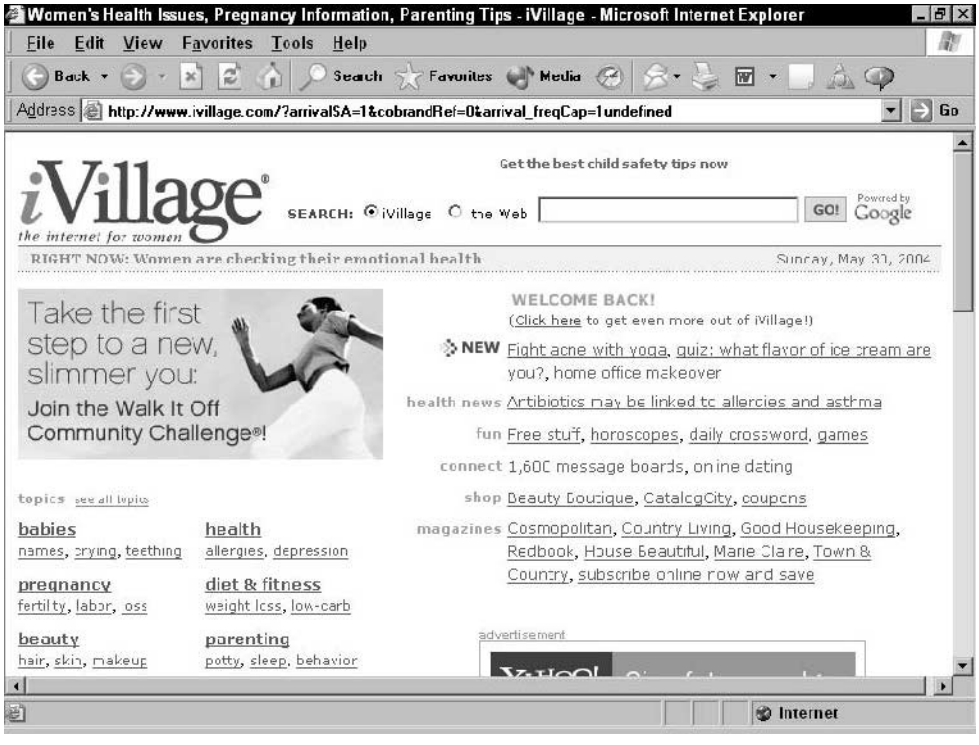


Figure 28-1: The power of giving customers a "voice" has driven the success of iVillage.com.

The iVillage technique is very successful and should be employed by other sites. Creating online chats where vice presidents and technologists talk to customers helps reinforce the company image. A customer who knows they can talk directly to the company's president is more likely to be a patron.

www.it-ebooks.info

Many companies wrestle with the idea of allowing anyone to add comments about its products on its Web sites. Let's face it; no one enjoys getting bad press. A site that manufactures water pumps would not like to see a user complaining that the pumps leak. By allowing customers to voice their feelings, the end result is generally positive. Building a community is a valuable asset that is only just beginning to be fully realized.

Adding Extra Value

These steps—adding articles to a site and enabling customers to add their own comments—are only the first toward increasing the amount of information a site has available to share with its customers. Successful sites are composed of a slew of tools that are employed to bring customers back to the site over and over again.

Producing newsletters and e-zines

A relatively inexpensive tool used by many companies because it is relatively inexpensive is producing newsletters and e-zines. Both of these products are essentially the same: information on products and services is sent to customers in e-mail. While this may sound as if it's *spam*, the Internet equivalent of junk mail, it is not. The difference is that the customer requests the newsletter by filling out a check box on a form (the already notorious "check here to receive more information on our products" check box) or by registering with a site. A user is far more willing to receive e-mail from a company if they have requested it.

Newsletter and e-zines can take many forms. Rich HTML can be used to format the newsletter. But this should be done cautiously. Not all e-mail programs can read HTML (such as AOL 5 and earlier). The alternative is to use plain text, and although it's not as pretty, it does guarantee that everyone receiving the e-mail can read it.

The only rule to bear in mind, when creating a newsletter to be sent through e-mail, is to keep it brief. The average Internet customer receives 25 to 100 e-mails a day. No one has time to read through lengthy newsletters. Indeed, many readers may give up after the first few paragraphs, the first page as viewed on a normal computer screen. If you have important information, place it at the top of the letter.

Successful newsletters from companies such as DevX, RealNetworks, and Netscape Netcenter are used as tools to drive users back to the sites. The newsletter may contain news information, stock quotes, or press releases, but the intent is to get users to the sites to begin viewing advertisements or buying.

All articles in a newsletter should be very short, one to three paragraphs. If an article is longer than that, provide a link back to the main site for the full article. Important product news, such as a recall or news release, should be placed as the first article. The next article in the newsletter should be related to the main article, but with a twist. For instance, the main article may be for a new car, whereas the second article can discuss the fuel savings of the new car. The third article should be part of some

www.it-ebooks.info

sort of series. For instance, the third article for our car newsletter could be a series focused on maintenance.

The idea is to create a consistent format that customers come to expect. Creating newsletters that change format with each issue only frustrates the reader and drives them away from future newsletters.

Newsletters are a great way to get product information placed in your customer's inbox. It also allows for the timely dissemination of information to customers, instead of relying on them to come to your site.

Writing product reviews

Whether it is a product or a service, the single driving need for almost every Web site is to sell, sell, sell.

The Web is unparalleled in its ability to reach and deliver to an enormously diverse range of customers. With this in mind, customers coming to your site expect to find product and service information that is fresh and up-to-date. Generally, customers search a company's site to obtain information on a product before making a purchase. This is particularly true of big-ticket items such as TVs, game systems, motorbikes, cars, and trucks. By providing timely, accurate, and concise information, the customers can make an informed decision about a product before they enter the show room. Indeed, when an informed customer arrives at a show room to make a purchase, the sale is made much faster.

Perhaps nothing is more infuriating to a consumer than arriving at a site to find old, outdated information. This drives customers away. To ensure that a product is given proper visibility, work with a launch schedule, such as the following:

- ✦ Understand the product or service being launched.
- ✦ Create press releases for the new product.
- ✦ Leverage the content of the press release on the company Web site.
- ✦ Create a section on the site for the new product (the larger the product, the larger the section).
- ✦ Create a newsletter to be delivered through e-mail informing customers of the new product and providing a link directly to the site, where they can read additional information about the product.

By defining a new product launch schedule, a process is created whereby the Internet site is part and parcel of the whole advertising process.

Creating tutorials

Offering online education is an emerging technique being employed to keep customers coming back to a site. Today, many products, especially software, are complicated. Meanwhile, other products such as books are not necessarily complicated, but

www.it-ebooks.info

they still require interpretation. Online tutorials and classes are a great vehicle to further extend the customer relationship. In providing training, your company is extending a hand to your customers to enable them to become product specialists.

One company, Barnes & Noble, is pushing the use of online training. At Barnes & Noble.com (`www.bn.com`), a customer can browse and purchase a book or software. With a purchase in hand, so to speak, this same user can stay at the Barnes & Noble.com site and sign up for free training. The training covers a wide range of topics from computers to classic books. A specialist not only develops each class but also is available to answer a student's questions.

This form of training or education is now being referred to as *edutainment*. The education is an extension of the site's traditional services. One goal of edutainment is to keep customers at the site longer so they have the opportunity to make additional purchases.

Facing the XML Future

A new era is dawning. The humble Web browser is undergoing a major change. A new, lean cousin is replacing HTML; the lingua franca of the Web is now becoming XML.

XML, the eXtensible Markup Language, is a lean scripting environment in which the reliance on a browser's support for specific codes is eradicated. A page no longer needs to be developed to work in Netscape Navigator 3, 4, and 6, and to work with the many flavors of Microsoft Internet Explorer. XML takes the support of standards away from the browser and places it into the hands of the developer. Okay, so how does it do that?

The structure of XML is very similar to HTML. Pages are formed of tagged content. The tags are used to categorize the content. The tags do not format the content. Cascading style sheets (CSS) and style sheets built with eXtensible Style Language (XSL) format tags on the page. The browser, independent of default standards, is forced to present the XML document as defined by the developer.

The following example shows a section of an XML document:

```
<?xml version="1.0"?>
<newsfeed>
<channel cid="about.entertainment.hl"
title="About.com: Entertainment"
date="05/03/2000"
time="11:17 PST">
<headline
href="http://log.isyndicate.com:8880/pscripts/hit/kixbiqvkczawz%2526i
jwcb.mvbmzbiqvumvb.pt%2526pbbx%253a%252f%252fzilqw.ijwcb.kwu%252fmvbm
zbiqvumvb%252fzilqw%252ftqjzizg%252femmstg%252fi597355i.pbu%25258nXU
%253d10_656_B"
date="Thu, May 04, 2000 11:05 AM"
time="11:05 PST">Webcasters of the World Unite</headline>
<headline
```

www.it-ebooks.info

```
href="http://log.isyndicate.com:8880/pscripts/hit/kixbiqvkczawz%2526i
jwcb.mvbmzbiqvumvb.pt%2526pbbx%253a%252f%252femjewzab.ijwcb.kwu%252fm
vbmzbiqvumvb%252femjewzab%252ftqjzizg%252femmstg%252fii505755i.pbu%25
258nXU%253d10_657_B"
date="Thu, May 04, 2000 11:05 AM"
time="11:05 PST">Elements of Bad Design</headline>
<headline
href="http://log.isyndicate.com:8880/pscripts/hit/kixbiqvkczawz%2526i
jwcb.mvbmzbiqvumvb.pt%2526pbbx%253a%252f%252fktiaaqkbd.ijwcb.kwu%252f
mvbmzbiqvumvb%252fktiaaqkbd%252ftqjzizg%252femmstg%252fii505655i.pbu%
25258nXU%253d10_656_B"
date="Wed, May 03, 2000 11:17 AM"
time="11:17 PST">Senator Coleman?</headline>
</channel>
<channel cid="hollywood.news"
title="Hollywood Online: News"
date="05/04/2000"
time="04:03 PST">
<headline
href="http://log.isyndicate.com:8880/pscripts/hit/kixbiqvkczawz%2526p
wttgewwl.vmea%2526pbbx%253a%252f%252feee.pwttgewwl.kwu%252fvmea%252fb
wxabwzqma%252f50-59-7555%252fpbut%252f6-7.pbut"
date="Thu, May 04, 2000 04:03 PM"
time="04:03 PST">Whatever Happened to Shania Twain?</headline>
<headline
href="http://log.isyndicate.com:8880/pscripts/hit/kixbiqvkczawz%2526p
wttgewwl.vmea%2526pbbx%253a%252f%252feee.pwttgewwl.kwu%252fvmea%252fb
wxabwzqma%252f50-59-7555%252fpbut%252f6-8.pbut"
date="Thu, May 04, 2000 04:03 PM"
time="04:03 PST">EXTRA: All About the &apos;Disgusting&apos;
Rap</headline>
<headline
href="http://log.isyndicate.com:8880/pscripts/hit/kixbiqvkczawz%2526p
wttgewwl.vmea%2526pbbx%253a%252f%252feee.pwttgewwl.kwu%252fvmea%252fb
wxabwzqma%252f50-59-7555%252fpbut%252f6-9.pbut"
date="Thu, May 04, 2000 04:03 PM"
time="04:03 PST">B.O. FORECAST: &apos;Gladiator&apos; Steps Into
Ring</headline>
</channel>
</newsfeed>
```

This is a basic XML newsfeed. This document has not been formatted, which will be done by the cascading style sheet in the browser.

XML is made up of the following features:

- ✦ Classified content
- ✦ CSS and XSL
- ✦ Document type definition

XML is the first language that can truly work on different operating systems, Internet devices, and products. A properly formatted XML document can display in a Web browser such as Netscape Navigator 6, or on a Nokia cell phone (through WML, the Wireless Markup Language), or on a Motorola pager, or a Palm PDA.

www.it-ebooks.info

Classifying content versus defining presentation

The content on an XML page is classified through the use of tags. The structure of a simple XML document is as follows:

```
<?xml version="1.0"?>
<library>
<Book>
<Title>Creating Web Pages Bible</Title>
<Author>David and Rhonda Crowder</Author>
<Category>Web Publishing </Category>
</Book>
<Book>
<Title>Mastering Dreamweaver</Title>
<Author>David and Rhonda Crowder</Author>
<Category>Web Publishing </Category>
</Book>
<Book>
<Title>Harry Potter and the Goblet of Fire</Title>
<Author>J.K. Rowling</Author>
<Category>Childrens/Adventure</Category>
</Book>
</library>
```

The document declares itself as an XML document with the opening line `<?xml version="1.0"?>`. For any XML documents to be properly executed, this opening statement must be added.

The structure of XML appears to be very similar to HTML. The content is structured in a readable format, not machine code. Each section is separated by what appears to be ordinary HTML tags. The reality, however, is that none of the tags are HTML tags. If this document was run through Netscape Navigator 4, the browser would not recognize the document and could not format it.

The power of the document is that it does not format the data. The tags can be managed and modified with scripting by either the server delivering the content or through JavaScript and CSS in a Web browser.

Keeping the content separate enables the document to be delivered to a multitude of programs. The same XML file can be delivered to Internet Explorer and Macromedia Flash Player.



Recent versions of Internet Explorer have the best support for XML, with limited support in Internet Explorer 4.01+. XML is also supported in Netscape Navigator 6.0 and 7.0.

Customizing tags

The "extensible" in XML is derived from the flexibility of the language. The previous example is XML displayed at its most simple: a declaration and some simple tags. Richer information can be added by extending the document. In many ways, it is like categorizing content into different fields in a database table. The following

www.it-ebooks.info

example shows how easily a new tag, called <Description>, can be inserted into the file:

```
<?xml version="1.0"?>
<library>
<Book>
<Title>Web Publishing Bible </Title>
<Author>David and Rhonda Crowder</Author>
<Description>The Web Publishing Bible provides the designer with the
complete set of tools required to build, promote and manage a Web site.
</Description>
<Category>Web Publishing </Category>
</Book>
<Book>
<Title>Mastering Dreamweaver</Title>
<Author>David and Rhonda Crowder</Author>
<Description>The Dreamweaver takes the reader on a tour de force
examination of every feature in Macromedia's Dreamweaver.</Description>
<Category>Web Publishing </Category>
</Book>
<Book>
<Title>Harry Potter and the Goblet of Fire</Title>
<Author>J.K. Rowling</Author>
<Description>Harry Potter is now in his fourth year at Hogwarts.
Prepare yourself for his most exciting adventures yet!
</Description>
<Category>Childrens/Adventure</Category>
</Book>
</library>
```

Specifying changes with a document type definition

Providing a free-form language, in the form of XML, is powerful. However, standard scripting formats such as HTML have great strengths as well. With HTML, large groups of users can build Web sites and communicate, which is key.

XML supports a standard called Document Type Definition (DTD), sometimes called Name Spaces. The DTD is a document that details how specific tags should be displayed, the type of content that can be entered into a given tag, and the method used to lay out tags. In many ways, it is the same as creating a custom HTML language that users can agree to. To date, there are over 40 different DTDs.

The following is an example of a DTD-declaring CIM Element Declarations:

```
<!-- This is a WMI extension used for context objects -->
<!ENTITY % VTType "VTTYPE (VT_I4|VT_R8|VT_BOOL|VT_BSTR|VT_UNKNOWN)">

<!ENTITY % QualifierFlavor "OVERRIDABLE   (true|false) 'true'
                            TOSUBCLASS    (true|false) 'true'
                            TOINSTANCE    (true|false) 'false'
```

www.it-ebooks.info

```
                              AMENDED       (true|false) 'false'
                              TRANSLATABLE  (true|false) 'false'">

<!ENTITY % ClassOrigin      "CLASSORIGIN        CDATA        #IMPLIED">

<!ENTITY % Propagated       "PROPAGATED         (true|false) 'false'">

<!ENTITY % ArraySize        "ARRAYSIZE          CDATA        #IMPLIED">

<!ENTITY % SuperClass       "SUPERCLASS         CDATA        #IMPLIED">

<!ENTITY % ClassName        "CLASSNAME          CDATA        #REQUIRED">

<!ENTITY % ReferenceClass   "REFERENCECLASS     CDATA        #IMPLIED">
```

An XML document that calls any of the following XML tags can be correctly formatted and presented by the WMI DTD:

```
<VTType><QualifierFlavor><ClassOrigin><Propagated><ArraySize><SuperClass>
<ClassName><ReferenceClass>.
```

The DTD is inserted into the top of an XML document. The content of the XML document is then formatted in accordance with the rules of the DTD. The declaration that enables the Web Browser to interpret HTML 4.0 from a DTD is placed in the `<HTML>` tag at the top of the Web page:

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

The `!DOCTYPE` provides the path the browser must take to find the DTD file that is used to interpret the XML tags on the page. An alternative method is to force the browser to locate a stored DTD within the browser program and execute that against the XML file. The following example is of Vector Markup Language (VML), a language developed by Microsoft, embedded within an HTML document. VML is an XML language that, when translated with the correct DTD, takes the scripted VML tags and creates a vector image to display in the Web browser. The DTD is stored within Internet Explorer and must be called from the browser:

```
<html xmlns:v="urn:schemas-microsoft-com:vml"
xmlns:o="urn:schemas-microsoft-com:office:office"
xmlns:w="urn:schemas-microsoft-com:office:word"
xmlns="http://www.w3.org/TR/REC-html40">
<head>
<link rel=File-List href="./Document2_files/filelist.xml">
<style>
v\:* {behavior:url(#default#VML);}
o\:* {behavior:url(#default#VML);}
w\:* {behavior:url(#default#VML);}
.shape {behavior:url(#default#VML);}
</style>
</head>
<body lang=EN-US style='tab-interval:.5in'>
<div class=Section1>
```

www.it-ebooks.info

```
<v:shapetype id="_x0000_t96" coordsize="21600,21600"
 o:spt="96" adj="17520"
path="m10800,0qx0,10800,10800,21600,21600,10800,10800,0xem7340,6445qx
6215,7570,7340,8695,8465,7570,7340,6445xnfem14260,6445qx13135,7570,14
260,8695,15385,7570,14260,6445xnfem4960@0c8853@3,12747@3,16640@0nfe">
 <v:formulas>
  <v:f eqn="sum 33030 0 #0"/>
  <v:f eqn="prod #0 4 3"/>
  <v:f eqn="prod @0 1 3"/>
  <v:f eqn="sum @1 0 @2"/>
 </v:formulas>
 <v:path o:extrusionok="f" gradientshapeok="t" o:connecttype="custom"

o:connectlocs="10800,0;3163,3163;0,10800;3163,18437;10800,21600;18437
,18437;21600,10800;18437,3163"
  textboxrect="3163,3163,18437,18437"/>
 <v:handles>
  <v:h position="center,#0" yrange="15510,17520"/>
 </v:handles>
 <o:complex v:ext="view"/>
</v:shapetype><v:shape id="_x0000_s1026" type="#_x0000_t96"
style='position:absolute;
 margin-left:112.05pt;margin-top:36.2pt;width:1in;height:1in;z-index:1'/>
<table cellpadding=0 cellspacing=0 align=left>
 <tr>
  <td width=149 height=48></td>
 </tr>
 <tr>
  <td></td>
  <td><img width=97 height=97 src="./Document2_files/image001.gif"
v:shapes="_x0000_s1026"></td>
 </tr>
</table>
</span>
</body>
</html>
```

Web browsers do not recognize the VML tags unless they have the DTD for VML installed. The tags can only be displayed correctly with the inclusion of `xmlns:v="urn:schemas-microsoft-com:vml"` in the HTML tag.

DTDs provide programmers with the flexibility to create their own tags and, through linking to the DTD, to share this with other users. This enables e-commerce between companies to be greatly extended. No longer are expensive EDI solutions required to exchange information. The content in an XML document can be easily and efficiently shared between disparate systems.

Getting different XML vocabularies to work together

Many emerging XML vocabularies are already in widespread use. These vocabularies can be managed and displayed in a single Web page. All that is needed is the correct

www.it-ebooks.info

declaration for the DTD. The following example demonstrates four different XML vocabularies being declared in a single HTML page:

```
<html xmlns:v="urn:schemas-microsoft-com:vml"
xmlns:o="urn:schemas-microsoft-com:office:office"
xmlns:w="urn:schemas-microsoft-com:office:word"
xmlns="http://www.w3.org/TR/REC-html40">
```

Here, the programmer is declaring the XML schemas for HTML 4.0, Word 2000, Office 2000, and VML. Such leverage enables richer and more informative content within any single Web page.

Further information on XML can be found at `www.xml.org`, `www.biztalk.org`, and `www.w3.org`. Each site provides a wealth of information on the latest DTDs, XML vocabularies, and XML support in third-party products. The XML revolution is coming, and it will be here sooner rather than later.

Transitioning to XHTML 1.0

December 19, 2000 marked the day when HTML made a bold move. Tim Berners-Lee, author of the original World Wide Web standards and Director of the World Wide Web Consortium (`www.w3.org`) announced his support for XHTML 1.0, the eXtensible HyperText Markup Language. XHMTL addresses the essential need of many developers: the creation of an HTML standard that is XML based. XHMTL is the first major revision of HTML since 1997.

Making HTML XML-compliant

To make HTML XML-compliant and modular, a number of syntactical rules must be adhered to. The most important rule to follow is to properly manage tags. Browsers have only loosely supported strict HTML. The proper closure of tags such as the `<H1>` or the `<P>` tag is not always needed. The following script shows an HTML example that will be presented accurately in any current Web browser:

```
<html>
<body bgcolor="#FFFFFF" text="#000000">
<p><font face="Arial" size="4">The following text will be displayed
correctly in an HTML Web.
<p>Even though some of the tags are not correctly closed.
</body>
</html>
```

Strict XHTML would not display this script. To begin with, the `<p>` and `<font>` tags are not correctly closed with a `</p>` and `</font>`. The strict nature of tag management within XML requires that all tags be correctly closed. In addition, the `<font>` is not supported in XHTML. Only cascading style sheets can be used to format the display of XHTML. The final statement that is missing from this document is the XML declaration at the top of the document identifying and defining the type of XML document and the location of the DTD file to translate the XML tags.

www.it-ebooks.info

To have the previous code work correctly as XHMTL, it has to resemble the following:

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>XHTML Page</title>
<style type="text/css" xml:space="preserve">
 body {
  background-color: #FFFFFF;
  color: #000000;
 }
 P {
  Font-Family: Arial;
  Font-Size: 14pt;
 }
</style>
</head>
</body>
<p>The following text will be displayed correctly in an XHTML Web.</p>
</html>
```

Several new features are added to the document. The first immediate change is the addition of the XML declaration at the top of the document. The insertion `<?xml version="1.0"?>` declares that the following document is an XML document and should be treated with the rules used with any XML document.

Following the XML declaration is a `DOCTYPE` that declares what DTD is being used. Following the strict translation of XHTML, the `DOCTYPE` is "`html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN`".

There are three types of XHTML: Strict, Transitional, and Frames. XHTML-Strict is the implementation of XHTML as defined in the complete XHTML 1.0 standard. XHTML-Transitional is a definition of XHTML that enables HTML 4.0 tags to be used within the syntax, enabling a transition from one language to the next. XHTML-Frames is a version of the language that enables Netscape Frames to be used within XHTML-formatted Web pages.

The line `http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd` directs the browser to download the appropriate DTD file to translate the document.

The XHMTL name space is defined within the `<HTML>` tag, identifying where the name space is located on the Internet.

Formatting can't be done with the `<font>` and `<body>` tags in XHTML. A cascading style sheet replaces the tags in the first version of the document. The `<body>` background color and default text color are identified as white (`#FFFFFF`) and black (`#000000`), respectively. The presentation style for the `<P>` tag is defined as font Arial and size 14 point.

www.it-ebooks.info

The rest of the document is presented as it would be in HTML, but without the formatting tags.

These are but a few of the number of changes that must be made to a document for it to be XHTML compliant. Many tags that have been used by HTML designers have been removed. As has already been noted, the <font> tag must be replaced with CSS. The same is true for the tag. This tag must now be a CSS style. The italics tag, <I>, must be replaced with the equivalent <EM> tag, and the bold tag <B> must be replaced with <STRONG>.

A complete description of how to make a document XHTML compliant is detailed at the World Wide Web Consortium Group's Web site (www.w3.org/MarkUp). The site contains comprehensive documentation on the new standard, including links to tools that convert HTML 4.0 pages to XHTML.

The W3C provides a set of tools to prepare Web pages for XHTML. The tools can be located at the W3C HTML Validation Service site (http://validator.w3.org/). One particular tool, HTML Tidy, is very useful in converting poorly written HTML documents, such as those generated by Microsoft Word, and making them compliant with XHTML standards.

Contrasting XHTML with HTML 4.0

HTML 4.0 bears many similarities to XHTML. Both languages require the use of CSS for formatting, and both require the closure of tags for paragraphs. In many ways, XHMTL is a reformulation of HTML 4.0 in order to follow the rules of XML. By reproducing HTML 4.0 as an XML language, many of the extensions that are added to XHTML can be done successfully in a modular format.

XHTML 1.0 enables the massive release of new Internet access programs, such as PDAs, Game Consoles, and Interactive TV boxes, which can be kept current with the latest Web page standards. All that is needed is a browser that understands XML and DTDs. A designer can still use text-editing tools to develop Web pages.

With the rapidly changing environment of the Internet, the inclusion of a modular and extensible language, such as XHTML, is a natural fit.

Facing browser compatibility problems

When working with new and emerging technologies, any Web designer or programmer is always faced with this challenging question: Should I use a new feature in the latest browser even though many customers may not have that browser, or should I design for the lowest-common browser?

A good way to answer this question is to examine the Web server log files for your site. Each Web server generates text files, called logs, that record information about each and every visitor to your site, such as which Web browser customers are using.

www.it-ebooks.info

Programs such as WebTrends interpret the data in log files and provide visual usage graphs.

Currently, companies such as Nielsen//Netratings (`www.nielsennetraings.com`) are declaring that Internet Explorer is dominating the browser landscape with over 90 percent penetration, of which 90 percent of that number represents users using Internet Explorer 5.0 and later. As for the remaining 10 percent penetration rate, most users are browsing with Netscape Navigator 4.0 and later, and the final few percent are using Netscape Navigator 1–3, AOL, Mozilla, Netscape 6, Opera, and other Web browsers. With this information in hand, it can be clearly surmised that a Web site built for Netscape Navigator 4 and Internet Explorer 5 reaches the largest potential audience. Only a few percent will not be able to view the pages.

Perhaps a more precise phrasing of the earlier question would be to ask: Is it worth developing a site for all browsers? Do the final 5 percent of users carry enough value to make it worth designing the site for use by all browsers? Ask yourself this when deciding on a site design strategy.

An easier task is designing Web sites for intranets. Many companies have standardized, using a single Web browser for the entire company. Whether the browser is Netscape Navigator 2 or Internet Explorer 6 does not matter. Knowing the program that is used to present the material on the Web page increases the number of tools a developer can leverage. For instance, if Internet Explorer 6 is being used on an entire Intranet, a designer can be certain that fonts embedded for page formatting into a cascading style sheet will display correctly.

When developing for the Internet, a developer must always be aware which browsers are most commonly used. In a single recent year Netscape Navigator 6, Internet Explorer 6, and MSDN Explorer were released. Each browser had a massive impact on the design and construction of Web pages but was then replaced by updated versions. Knowing which browser your audience is using enables you to make an educated guess at how to build the pages on your site.

Summary

Your Web site needs updating in two ways. First, the content must be kept fresh. Second, the technology must keep pace. This chapter introduced you to the following concepts and topics:

- ✦ Establish a routine schedule for updating material such as articles.
- ✦ Invite your visitors to participate in providing material.
- ✦ Opt-in e-zines are a great way to keep in touch with your site visitors and build community.
- ✦ XML, the eXtensible Markup Language, is the latest evolution in Web page creation. It allows for the creation of extended element definitions.

✦ ✦ ✦

www.it-ebooks.info

CHAPTER 29

Designing with XML

In This Chapter

Designing with XML

Transitioning to XHTML

Web sites today must not only be prepared for rapidly changing content; they also must be prepared for rapidly changing technology. Three years ago a site would need to leverage HTML and JavaScript in its construction, whereas one year ago HTML 4.0 (HTML, cascading style sheets, and Document Object Model Script) would have been the flavor of code required for the construction of a site. Today, the technologies required for building a site range from HTML 4.01 to WML (Wireless Markup Language) to XML (eXtensible Markup Language) to XHTML to the dozen of XML-based languages. The new languages are necessitated because of the massive deluge of Internet appliances.

Designing with XML

A new era is dawning. The humble Web browser is undergoing a major change. A new, lean cousin is replacing HTML; the lingua franca of the Web is now XML.

XML, the eXtensible Markup Language, is a lean scripting environment in which the reliance on a browser's support for specific codes is eradicated. A page no longer needs to be developed to work in Netscape Navigator , Mozilla, Opera, and the many flavors of Microsoft Internet Explorer. XML takes the support of standards away from the browser and places it into the hands of the developer. How does it do that?

The structure of XML is very similar to HTML. Pages are formed of tagged content. The tags are used to categorize the content. The tags do not format the content. Cascading style sheets (CSS) and eXtensible stylesheet language (XSL) format tags on the page. The browser, independent of default standards, is forced to present the XML document as defined by the developer.

www.it-ebooks.info

XML is made up of the following features:

- ✦ Classified content
- ✦ CSS and XSL
- ✦ Document type definition

XML is the first language that can truly work on different operating systems, Internet devices, and products. A properly formatted XML document can display in a Web browser such as Netscape Navigator 6, or on a Nokia cell phone (through WML), a Motorola pager, or a Palm PDA.

XML Rules

Any XML document must adhere to a set of simple rules:

- ✦ The tags and attribute values are case sensitive.
- ✦ There can only be one root element, all the rest of the tags must be inside this main element, (`<html></html>`).
- ✦ The document must be well formed, which means:
  - Non empty elements must have closing tags (`<p></p>`).
  - Empty elements must use a special syntax (`<br />`).
  - Tags must be properly nested (`<p><a> </a></p>`).
- ✦ Attribute values must always be quoted with matched double or single quotes (`<p class="test">< /p>`).
- ✦ Attribute values can not be omitted (`<dl compact="compact"> </dl>`).

If you don't follow these rules, the browser will report an error and will not display the page. However, working with XML is not that difficult. If you understand the way HTML tags work, then you are more than halfway to understanding XML. In fact XHTML is HTML written according to the restrictions of XML.

Getting different XML vocabularies to work together—Namespaces

Many emerging XML vocabularies are already in mainstream use. These vocabularies can be used in the same XML file or managed and displayed in a single Web page. The browser requires the correct DTD declaration in order to support the XML vocabulary used. Technically speaking the vocabulary is called a namespace. The following example demonstrates four different XML namespaces declared in a single HTML page:

```
<html xmlns:v="urn:schemas-microsoft-com:vml"
xmlns:o="urn:schemas-microsoft-com:office:office"
```

www.it-ebooks.info

```
xmlns:w="urn:schemas-microsoft-com:office:word"
xmlns="http://www.w3.org/TR/REC-html40">
```

In these examples, the programmer is declaring the XML schemas for HTML 4.0, Word 2000, Office 2000, and Vector Markup Language. Such leverage enables richer and more informative content within any single Web page.

The tags within the document can specify which namespace they are in by prefixing the tag with the abbreviation for the namespace, for example `<v:path ...>`. The default namespace is the one that starts `xmlns=""`; any tag that has no namespace defined is assumed to be from this vocabulary. We will provide more information about namespaces later in this chapter.

Classifying content versus defining presentation

The content on an XML page is classified through the use of tags. The structure of a simple XML document is as follows:

```
<?xml version="1.0"?>
<library>
<Book>
<Title>Creating Web Sites Bible</Title>
<Author>David Crowder and Andrew Bailey</Author>
<Category>Web Publishing </Category>
</Book>
<Book>
<Title>Mastering Dreamweaver</Title>
<Author>David Crowder</Author>
<Category>Web Publishing </Category>
</Book>
<Book>
<Title>Harry Potter and the Goblet of Fire</Title>
<Author>J.K. Rowling</Author>
<Category>Children/Adventure</Category>
</Book>
</library>
```

The document declares itself as an XML document with the opening line `<?xml version="1.0"?>`. For any XML documents to be properly executed, this opening statement must be added.

The structure of XML appears to be very similar to HTML. The content is structured in a readable format, not machine code. Each section is separated by what appears to be ordinary HTML tags. The reality, however, is that none of the tags are HTML tags. If this document was run through a browser, it would not know how to format the document. What most browsers do is to show the document as is, adding check boxes for each element to show or hide its children (see Figure 29-1).

The power of the document is that it does not format the data. The tags can be managed and modified with scripting by either the server delivering the content or through JavaScript and CSS on the Web browser.

www.it-ebooks.info

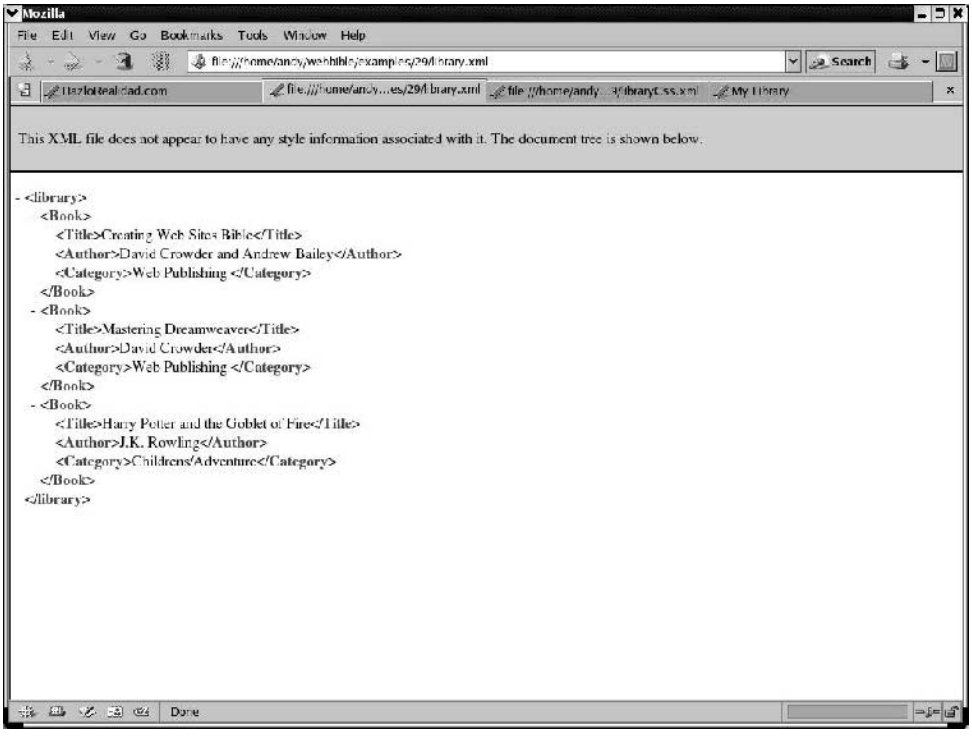


Figure 29-1: An XML document viewed in a browser with no style sheet attached.

Specifying a CSS

Now let's format the document in the previous example using CSS technology. To do this we insert the following line after the first line:

```
<?xml-stylesheet type="text/css" href="library.css"?>
```

The first three lines are now:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css" href="library.css"?>
<library>
....
```

Next, we need to create the library.css reference file, which contains the style definitions that control the appearance of the individual content elements:

```
library { font-size: 14 pt; color: red; display: block; }
Book { font-size: 30pt; color: red; display: block;
  margin: 1em; background-color: yellow; }
Title { font-size: 30pt; color: blue; display: inline; }
Author { font-size: 20pt; font-weight:bold; color: red;
   display: inline; }
Category {font-size: 15pt; color: green; display: block;
  margin: 0 0 0 1em }
```

www.it-ebooks.info



See Chapter 13 for information on CSS.

Figure 29-2 shows the result the library.css file has on the XML document. It now looks like a normal HTML document.



Figure 29-2: An XML document formatted with CSS.

We are restricted by CSS in that it only allows us to specify the format of text already present. We cannot add text and we also cannot change the order of the content in the original document. To overcome this problem, the W3C invented XSL. The good news is that this also works like a CSS. The bad news is that as it is more powerful it is also more complicated.

Specifying an XSL style sheet

For this example, we change the second line of the previous XML file to:

```
<?xml-stylesheet type="text/xsl" href="library.xsl"?>
```

Now all that remains to do is create the style sheet library.xsl:

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html"/>
```

www.it-ebooks.info

```
<xsl:template match="library">
<html>
<title>My Library</title>
<link rel="stylesheet" type="text/css" href="style.css" />
<body>
<h1>My Library</h1>
<table border="1">
<tr>
<th>Title</th>
<th>Author</th>
<th>Category</th>
</tr>
<xsl:for-each select="Book">
<tr>
<td class="Title"><xsl:value-of select="Title"/></td>
<td class="Author"><xsl:value-of select="Author"/></td>
<td class="Category"><xsl:value-of select="Category"/></td>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>

</xsl:stylesheet>
```

Figure 29-3 shows the result of this XSL transformations.

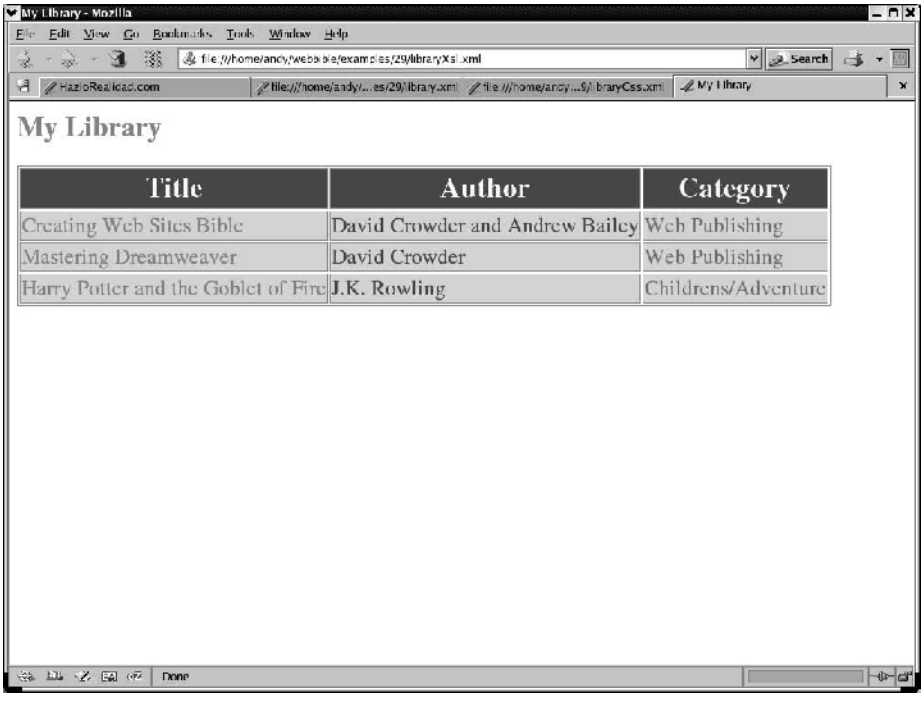


Figure 29-3: An XML document formatted with an XSL style sheet.

www.it-ebooks.info

Note that now we have added three extra text elements: a title, amain heading, and column headings.

XSL is very close to a programming language. It has `for` loops that repeat an action or number of actions for each element in the list:

```
<xsl:for-each select="list of elements">
<!-- Actions -->
</xsl:for-each>
```

It has `if` statements that execute one or more actions if the condition is true. The syntax is as follows:

```
<xsl:if test="condition">
<!-- Actions -->
</xsl:if>
```

It also has a statement to choose one action from a set of actions depending on the conditions specified for each action:

```
<xsl:choose>
  <xsl:when test="condition1">
  <!-- Action 1 -->
  </xsl:when>
  <xsl:when test="condition2">
  <!-- Action 2 -->
  </xsl:when>
  <xsl:otherwise>
  <!-- Default Action -->
  </xsl:otherwise>
</xsl:choose>
```

The conditions are evaluated in turn. If a condition is true then the associated action is performed and then the next instruction after the `choose` statement is executed. If none of the conditions are true then the default action inside the `otherwise` tag is carried out.

If you know a programming language this is equivalent to a multiple if statement, an example in C or Java would be:

```
if (condition1) {
  //action 1
} else if (condition2) {
  //action 2
} else {
  //default actions
}
```

However, the syntax is more similar to a switch, case, default statement in C or Java.

XSL is very complicated at first sight, because it uses a different concept of programming. Taking a page from object-oriented programming, one might call this type of programming template-oriented programming.

www.it-ebooks.info

To understand XSL you need to think of an XML document as a tree, but not a tree as you know it. It's an upside down tree with the root at the top. The root element in our case is `library`. This element has three branches (or elements), which are the three books. Each book, in turn, has three branches: `Title`, `Author`, and `Category`. And each of the branches has a leaf—the actual text content.

Let's go step by step through the style sheet. First comes the instruction that denotes that this is an XML document:

```
<?xml version="1.0"?>
```

The following is the root element of the document; the element's name is `stylesheet`. The `xsl:` prefix denotes that it is in the namespace `xsl`.

```
<xsl:stylesheet version="1.0"
        xmlns="http://www.w3.org/TR/REC-html40"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

Because anyone can define XML tag names, namespaces were invented in order to avoid collisions. In this document we are mixing HTML with XSL. Collisions occur when two elements have the same tag name but belong to a different context. To distinguish between two tags that have the same name, say HTML `title` and XSL `title`, all you have to do is specify the namespace for one of them: for example, `<xsl:title>`.

In this case XSL doesn't have an element called title, it's just there to illustrate the point. The default namespace is declared by `xmlns="(where to find the definition on Internet)"`. Other namespaces used are declared by `xmlns:prefixUsed="(definition)"`. Note that any prefix could be used, but it's best to use the standard one. Next, we tell the style sheet that it should output HTML:

```
<xsl:output method="html"/>
```

Other valid output options are XML or text.

```
<xsl:template match="library">
```

Every time you come across an element called `library`, apply the following template. Start with pure HTML to create the Web page:

```
<html>
<title>My Library</title>
<link rel="stylesheet" type="text/css" href="style.css" />
```

Note that we also use CSS formatting:

```
<body>
<h1>My Library</h1>
<table border="1">
<tr>
<th>Title</th>
<th>Author</th>
<th>Category</th>
</tr>
<xsl:for-each select="Book">
```

www.it-ebooks.info

Remember that the library has three branches—the three books. For each child element that is a book, we do the following:

```
<tr>
<td class="Title"><xsl:value-of select="Title"/></td>
```

The XSL element `value-of` inserts the value of the expression defined in the select attribute. In this case the text content of the `Title` element of the book. Translated into English this means that it inserts the title. The next two lines are similar:

```
<td class="Author"><xsl:value-of select="Author"/></td>
<td class="Category"><xsl:value-of select="Category"/></td>
</tr>
</xsl:for-each>
```

The end of the loop:

```
</table>
</body>
</html>
</xsl:template>
```

The following is the end of the first and in this case only template. However, there can be any number of templates in an XSL document.

```
</xsl:stylesheet>
```



Some browsers do not support XSL style sheets (for example, Opera 7.50).

Even if your browser doesn't support XSL transformations directly, you can use a tool to transform the XML content to HTML and then publish the result on the Web or do it in real time with server-side processing. Many XSL transformation programs exist. Free ones include xsltproc (included in most Linux distributions), Saxon (available at `http://saxon.sourceforge.net`), or Xalan (`http://xml.apache.org/xalan-j`). Xalan is also included as part of the Java Development Kit version 1.4 and later, freely available at `http://java.sun.com`. Both Xalan and Saxon are written in Java and work on Windows, Mac OS, or Linux machines.

Keeping the content separate enables the previous document to be delivered to a multitude of programs. The same XML file can be delivered to any browser.



Microsoft Internet Explorer 5+ and Mozilla have good support for XML, with limited support in Internet Explorer 4.01+. XML is also supported in Netscape Navigator 6.0.

Defining the structure of an XML document

For the previous style sheet, we assumed that the library had books as child elements and that each book had `Title`, `Author`, and `Category` child elements. If this isn't the case, then the style sheet won,t produce an error—it will just do what

www.it-ebooks.info

it's instructed to do. For example, if the library has children that are shelves, which, in turn, contain the book elements, the previous style sheet would not show any books, because we instructed it to match only books that are children of the library element.

How can we define the structure of the document to make sure it is valid? There are two solutions, a Document Type Definition (DTD) or XML Schema. A DTD is much simpler but not as powerful as the XML Schema.

The DTD is a document that details how specific tags should be displayed, the type of content that can be entered in a given tag, and the method used to lay out tags. In many ways, it is the same as creating a custom HTML language that users can agree to.

Here is the file again (this time with a line specifying the structure):

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE library SYSTEM "library.dtd">
<library>
<Book>
<Title>Creating Web Pages Bible</Title>
<Author>David Crowder</Author>
<Category>Web Publishing </Category>
</Book>
<Book>
<Title>Mastering Dreamweaver</Title>
<Author>David Crowder</Author>
<Category>Web Publishing </Category>
</Book>
<Book>
<Title>Harry Potter and the Goblet of Fire</Title>
<Author>J.K. Rowling</Author>
<Category>Childrens/Adventure</Category>
</Book>
</library>
```

The second line of the file specifies that the root element of the XML document is `library` and that the structure is specified in the file on the local file system `library.dtd`.

The DTD called `library.dtd` referenced in the previous XML file is as follows:

```
<!ELEMENT library (Book)+>
<!ELEMENT Book (Title,Author,Category)>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Author (#PCDATA)>
<!ELEMENT Category (#PCDATA)>
```

Once again let's go though this step by step:

```
<!ELEMENT library (Book)+>
```

This line specifies that a library has one or more books as children. If we had wanted to allow a library with zero or more books, we could have said `(Book)*`.

www.it-ebooks.info

The brackets group an element or any number of elements. The + represents one or more of the previous, the * represents zero or more.

```
<!ELEMENT Book (Title,Author,Category)>
```

The book has a group of elements: `Title`, `Author`, and `Category` in that order.

```
<!ELEMENT Title (#PCDATA)>
```

`PCDATA` stands for parsed character data, which means text. In English, a `Title` element has text content, and the same applies for the `Author` and `Category` elements:

```
<!ELEMENT Author (#PCDATA)>
<!ELEMENT Category (#PCDATA)>
```

Customizing tags

The "extensible" in XML is derived from the flexibility of the language. The previous example is XML displayed at its most simple: a declaration and some simple tags. The document can be easily extended. Richer information can be added by extending the document. In many ways, it is like categorizing content into different fields in a database table. The following example shows how easily a brand-new tag, called `<Description>`, can be inserted into the file:

```
<?xml version="1.0"?>
<!DOCTYPE library SYSTEM "library.dtd">
<library>
<Book>
<Title>Web Sites Bible </Title>
<Author>David Crowder and Andrew Bailey</Author>
<Description>The Web Publishing Bible provides the designer with the
complete set of tools required to build, promote and manage a Web site.
</Description>
<Category>Web Publishing </Category>
</Book>
<Book>
<Title>Mastering Dreamweaver</Title>
<Author>David Crowder</Author>
<Description>The Dreamweaver takes the reader on a tour de force
examination of every feature in Macromedia's
Dreamweaver.</Description>
<Category>Web Publishing </Category>
</Book>
<Book>
<Title>Harry Potter and the Goblet of Fire</Title>
<Author>J.K. Rowling</Author>
<Description>Harry Potter is now in his fourth year at Hogwarts.
Prepare yourself for his most exciting adventures yet!
</Description>
<Category>Children/Adventure</Category>
</Book>
</library>
```

www.it-ebooks.info

More on DTD

Providing a free-form language, in XML, is powerful. However, standard scripting formats such as HTML have great strengths as well. With HTML, large groups of users can build Web sites and communicate, which is key.

As mentioned previously, the DTD is inserted into the top of an XML document. The content of the XML document is then validated in accordance with the rules of the DTD. The declaration that enables Web browser to interpret HTML 4.0 from a DTD is placed in the <HTML> tag at the top of the Web page:

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

The !DOCTYPE provides the path the browser must take to find the DTD file that is used to interpret the XML tags on the page. An alternative method is to force the browser to locate a stored DTD within the browser program and execute that against the XML file. The following example is of Microsoft Vector Markup Language (VML) embedded within an HTML document. VML is a version of XML that, when translated with the correct DTD, takes the scripted VML tags and creates a vector image to display in Internet Explorer. The DTD is stored within Microsoft Internet Explorer and must be called from the browser:

```
<html xmlns:v="urn:schemas-microsoft-com:vml"
xmlns:o="urn:schemas-microsoft-com:office:office"
xmlns:w="urn:schemas-microsoft-com:office:word"
xmlns="http://www.w3.org/TR/REC-html40">
<head>
<link rel=File-List href="./Document2_files/filelist.xml">
<style>
v\:* {behavior:url(#default#VML);}
o\:* {behavior:url(#default#VML);}
w\:* {behavior:url(#default#VML);}
.shape {behavior:url(#default#VML);}
</style>
</head>
<body lang=EN-US style='tab-interval:.5in'>
<div class=Section1>
<v:shapetype id="_x0000_t96" coordsize="21600,21600"
 o:spt="96" adj="17520"
path="m10800,0qx0,10800,10800,21600,21600,10800,10800,0xem7340,6445qx
6215,7570,7340,8695,8465,7570,7340,6445xnfem14260,6445qx13135,7570,14
260,8695,15385,7570,14260,6445xnfem4960@0c8853@3,12747@3,16640@0nfe">
 <v:formulas>
  <v:f eqn="sum 33030 0 #0"/>
  <v:f eqn="prod #0 4 3"/>
  <v:f eqn="prod @0 1 3"/>
  <v:f eqn="sum @1 0 @2"/>
 </v:formulas>
 <v:path o:extrusionok="f" gradientshapeok="t" o:connecttype="custom"

o:connectlocs="10800,0;3163,3163;0,10800;3163,18437;10800,21600;18437,
```

www.it-ebooks.info

```
18437;21600,10800;18437,3163"
  textboxrect="3163,3163,18437,18437"/>
 <v:handles>
  <v:h position="center,#0" yrange="15510,17520"/>
 </v:handles>
<o:complex v:ext="view"/>
</v:shapetype><v:shape id="_x0000_s1026" type="#_x0000_t96"
style='position:absolute;
 margin-left:112.05pt;margin-top:36.2pt;width:1in;height:1in;z-index:1'/>
<table cellpadding=0 cellspacing=0 align=left>
 <tr>
  <td width=149 height=48></td>
</tr>
<tr>
 <td></td>
 <td><img width=97 height=97 src="./Document2_files/image001.gif"
v:shapes="_x0000_s1026"></td>
 </tr>
</table>
</span>
</body>
</html>
```

Web browsers do not recognize the VML tags unless they have the DTD for VML installed. The tags can only be displayed correctly with the inclusion of `xmlns:v="urn:schemas-microsoft-com:vml"` in the HTML tag. The tags are Microsoft specific and so only display correctly in Internet Explorer. Beware of this when saving a Word document as a Web page.

DTDs provide programmers the flexibility to create their own tags and, through linking to the DTD, share the result with other users. This enables e-commerce between companies to greatly extend. No longer are expensive EDI solutions required to exchange information. The content in an XML document can be easily and efficiently shared between disparate systems.



Further information on XML can be found at `www.xml.org`, `www.biztalk.org`, and `www.w3.org`. Each site provides a wealth of information on the latest DTDs, XML vocabularies, and supporting third-party products. The XML revolution is coming, and it will be here sooner rather than later.

Transitioning to XHTML 1.0

December 19, 2000 marked the day when HTML made a bold move. Tim Berners-Lee, author of the World Wide Web and director of the World Wide Web Consortium (`www.w3.org`), announced his support for XHTML 1.0, the eXtensible HyperText Markup Language. XHTML addresses the essential need of many developers: the creation of an HTML standard that is XML based. XHTML is the first major revision of HTML since 1997.

www.it-ebooks.info

Making HTML XML compliant

To make HTML XML compliant and modular, a number of syntactical rules must be adhered to, the same as for XML documents, as mentioned at the start of the chapter. The most important rule to follow is to properly manage tags. Browsers have very loosely supported strict HTML. The proper closure of tags such as the <H1> or the <P> tag is not always needed. The following script shows an HTML example that will be presented accurately in any current Web browser:

```
<html>
<body bgcolor="#FFFFFF" text="#000000">
<p><font face="Arial" size="4">The following text will be displayed
correctly in an HTML Web Browser.
<p>Even though some of the tags are not correctly closed.<b>Ok
```

Strict XHTML would not display this script. To begin with, the and tags are not correctly closed with a and . The strict nature of tag management within XML requires that all tags be correctly closed. In addition, the <font> is not supported in XHTML. Only cascading style sheets can be used to format the display of XHTML. The final statement that is missing from this document is the XML declaration at the top of the document identifying and defining the type of XML document and the location of the DTD file to translate the XML tags.

To have the previous code work correctly as XHTML, it has to resemble the following:

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>XHTML Page</title>
<style type="text/css" xml:space="preserve">
 body {
  background-color: #FFFFFF;
  color: #000000;
 }
 P {
  Font-Family: Arial;
  Font-Size: 14pt;
 }
</style>
</head>
<body>
<p>The following text will be displayed correctly in an XHTML
Web Browser.</p><p>As all of the tags are correctly closed.<b/>Ok</p>
</html>
```

Several new features are added to the document. The first immediate change is the addition of the XML declaration at the top of the document. The insertion <?xml version="1.0"?> declares that the following document is an XML document and should be treated with the rules used with any XML document.

www.it-ebooks.info

Following the XML declaration is a DOCTYPE declaration, which declares the DTD that is used. As the strict translation of XHTML, the DOCTYPE is "html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN".

There are three types of XHTML: Strict, Transitional, and Frames. XHTML-Strict is the implementation of XHTML as defined in the complete XHTML 1.0 standard. XHTML-Transitional is a definition of XHTML that enables HTML 4.0 tags to be used within the syntax, enabling a transition from one language to the next. XHTML-Frames is a version of the language that enables Netscape Frame elements to be used within XHTML-formatted Web pages.

The line http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd directs the browser to download the appropriate DTD file to translate the document.

The XHMTL namespace is defined within the <HTML> tag, identifying where the namespace is located on the Internet.

Formatting can't be done with the <font> and <body> tags in XHTML. A CSS replaces the tags in the first version of the document. The <body> background color and default text color are identified as white (#FFFFFF) and black (#000000), respectively. The presentation style for the <P> tag is defined as font Arial and size 14pt.

The rest of the document is presented as it would be in HTML, only without the formatting tags.

These are but a few of the number of changes that must be made to a document for it to be XHTML compliant. Many tags that have been used by HTML designers have been removed. As has already been noted, the <font> tag must be replaced with CSS. The same is true for the tag. This tag must now be a CSS style. The italics tag, <I>, must be replaced with the equivalent <EM> tag, and the bold tag <B> must be replaced with <STRONG>.

Note that the <b> tag now needs to be replaced by <b/>. In fact, it is recommended that you use <b /> in order not to confuse some old browsers.

A complete description of how to make a document XHTML compliant is detailed at the W3C's Web site (www.w3.org/MarkUp). The site contains comprehensive documentation on the new standard, including links to tools that convert HTML 4.0 pages to XHTML.

The W3C provides a set of tools to prepare Web pages for XHTML. The tools can be located at the W3C HTML Validation Service (http://validator.w3.org/) site. One particular tool, HTML Tidy, is very useful in converting poorly written HTML documents, such as those generated by Microsoft Word, and making them compliant with XHTML standards.

Contrasting XHTML with HTML 4.0

HTML 4.0 bears many similarities to XHTML. Both languages require the use of CSS for formatting, and both require the closure of tags for paragraphs. In many ways,

www.it-ebooks.info

XHMTL is a reformulation of HTML 4.0 in order to follow the rules of XML. By reproducing HTML 4.0 as an XML language, many of the extensions that are added to XHTML can be done successfully in a modular format.

XHTML 1.0 enables the massive release of new Internet accessing programs, such as PDAs, game consoles, and interactive TV boxes, which can be kept current with the latest Web page standards. All that is needed is a browser that understands XML and DTDs. A designer can still use text-editing tools to develop Web pages.

With the rapidly changing environment of the Internet, the inclusion of a modular and extensible language, such as XHTML, is a natural fit.

Facing browser compatibility problems

When working with new and emerging technologies, the designer and programmer are always faced with this challenging question: Should I use a new feature in the latest browser even though many customers may not have that browser, or should I design for the lowest-common browser?

A good way to answer this question is to examine the Web server log files for your site. Each Web server generates text files, called logs, that record information about each and every visitor to your site, including the type of Web browser your visitors are using. Programs such as WebTrends interpret the data in log files and provide visual usage graphs.

W3Schools (`www.w3schools.com/browsers/browsers_stats.asp`) reports that Microsoft Internet Explorer 6 is dominating the browser landscape with 73 percent penetration. Internet Explorer version 5 and Mozilla are joint second both with 10 percent usage. Opera and Netscape Navigator 4.0 and above are joint fourth each with 2 percent. With this information in hand, it can be clearly surmised that a Web site built for Internet Explorer 5+ reaches 83 percent of the potential audience; 17 percent will not be able to view the pages.

Perhaps, a more precise phrasing of the earlier question would be to ask: Is it worth developing a site for all browsers? Does the final 17 percent carry enough value to have the site designed for all browsers? One fact to take into account is that you can never know what new browsers will appear in the future and what their acceptance will be, so it makes sense to adhere to the W3C standards when designing Web pages in order to reach the largest possible audience.

An easier task is designing Web sites for intranets. Many companies have standardized, using a single Web browser for the entire company. Whether the browser is Netscape Navigator 2 or Internet Explorer 6 does not matter. Knowing the program that is used to present the material on the Web page increases the number of tools a developer can leverage. For instance, if Internet Explorer 6 is being used on an entire intranet, a designer knows to embed fonts for page formatting into a CSS, resulting in a 100 percent guarantee that the page using the embedded font displays correctly, unlike a 73 percent guarantee on the Internet.

www.it-ebooks.info

When developing for the Internet, a developer must always be aware which browsers are most commonly being used. Knowing which browser your audience is using enables you to make an educated guess at how to build the pages on your site.

Summary

Your Web site needs updating in two ways. First, the content must be kept fresh. Second, the technology must keep pace with new developments. This chapter introduced you to the following concepts and topics:

- XML (eXtensible Markup Language) is the latest evolution in Web page creation. It allows for the creation of extended element definitions.
- DTDs (Document Type Definitions) is used to define the structure of an XML document.
- XSL (eXtensible Stylesheet Language) is used to transform XML to another format.

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

CHAPTER 30

Making Your Site Mobile with WAP/WML

In This Chapter

What is WML?

Interacting with users

Menus in WML

Differences between HTML lists and WML lists

Transforming XHTML into WML



As it stands right now, there are more mobile computing devices in use than desktop ones. And the trend is that the growth in the number of portable devices will increase even more rapidly. With the implementation of the Internet Protocol Version 6, approximately 2^128 or 340,282,366,920,938,463,463,374,607,431,768,211,456 devices will be able to be directly connected to the Internet.

Many cell phones and almost all personal digital assistants (PDAs) already have the capability to connect to Internet. And one of the problems faced by today's Web developers is that they are used to designing pages for desktop computers with screen sizes of 15 inches and up. The question that faces them is how to view the same information in a cell phone with a screen that could be little more than 1 inch in width. This is where Wireless Markup Language (WML) comes in.

What Is WML?

WML is similar to XHTML. Think of it as a subset of XHTML but with a few additions. WML is the file format used by most mobile phone browsers. Although the tags themselves are simpler, in some ways WML has more features than HTML. For example, with input text boxes in WML you can supply the format of the data required and the browser automatically validates the data before sending the form. In HTML you could achieve the same, but in a much more complicated fashion, with JavaScript. Many people speak of WAP-enabled phones and WAP pages. Really the page is a WML page that you view in a WML browser on a WAP-enabled phone.

WAP is the wireless application protocol. WAP is to mobile phones what the HyperText Transfer Protocol (http) is to

www.it-ebooks.info

desktop computer and laptops. It is the link between your mobile phone and Internet, and your Internet service provider (ISP) in this scenario is the phone company. The ISP acts as a proxy or gateway retrieving the WML pages you request and then sends them via WAP to your phone. Any Web site can have WML pages without requiring any extra software.

The scripting language for mobile phones is called WMLScript. WMLScript is similar in function and syntax to JavaScript, which can be used on a standard Web page. However, as we have already mentioned, the most common uses of JavaScript, such as validating forms, is already incorporated into WML, so WMLScript doesn't play as big a role in the design of WML pages as JavaScript in HTML pages.

There are several interesting WAP sites on the Internet with in-depth information about WAP and WML including:

- www.wapforums.com
- www.wap.com
- www.w3schools.org
- www.nokia.com

Now that you understand some of the basic terminology, we can turn to the design of WML pages.

Dealing with WML

WML is designed for devices with limited memory, a limited bandwidth wireless connection to the Internet, and a low screen resolution, which could be 320 × 240 pixels.

In an attempt to overcome these problems the designers of WML decided to break away from the way HTML works where each HTML file represents a page on the screen. In WML one file can contain several pages. The main file is called a deck and the pages inside are called cards.

The basic structure of a WML file is as follows:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
  <card title="Welcome to WML">
         <p>
                <em>Emphasised Text!</em><br/>
                <i>Italic Text!</i><br/>
                <b>Bold Text!</b><br/>
                <strong>Strong Text!</strong><br/>
                <u>Underlined Text!</u><br/>
                <small>Small Text!</small><br/>
           </p>
  </card>
</wml>
```

www.it-ebooks.info

Note that this is an XML file, so the `<br>` tag, which has no content, is specified as `<br/>`.



See Chapter 29 for details of the rules for XML.

To view the file you have several choices. One is to put the file on any Web server and connect to it with a WAP-enabled mobile phone with Internet access. You could also use a browser that emulates a mobile phone and provides WML support—many are available on the Internet. Alternatively, you can use a normal desktop browser that supports WML. Opera is one of the few desktop browsers that is WML compatible—a free version is available from `www.opera.com`. Figure 30-1 shows how Opera displays the previous WML file.

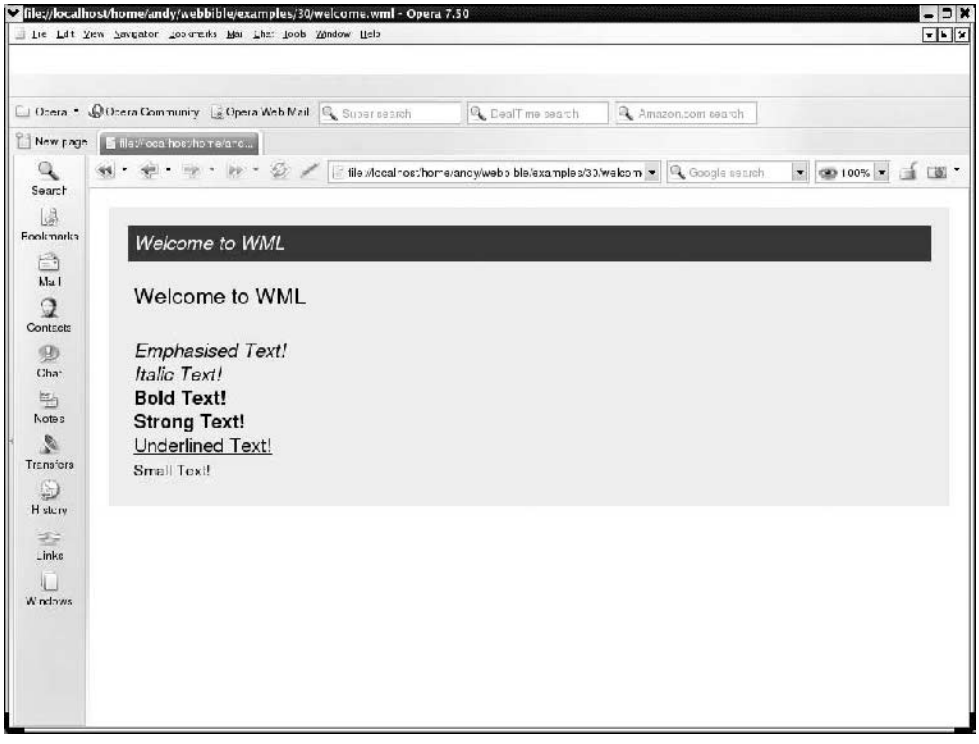


Figure 30-1: A WMLdocument viewed in the Opera Browser.

Let's go through the WML file step by step:

```
<?xml version="1.0"?>
```

This is the declaration that specifies that the file is XML.

```
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
```

www.it-ebooks.info

This is the declaration of the DTD for WML 1.1.

```
<wml>
```

The `<wml>` tag is the equivalent of the `<HTML>` tag for an HTML file. It is the root element for a WML document.

```
<card title="Welcome to WML">
```

A `card` element is equivalent to an HTML page. WML browsers only show one `card` element at a time.

The following tags are identical to their HTML equivalents with the exception of the `<small>` tag, which has the effect of displaying the text in a smaller font.

```
        <p>

              <em>Emphasised Text!</em><br/>
              <i>Italic Text!</i><br/>
              <b>Bold Text!</b><br/>
              <strong>Strong Text!</strong><br/>
              <u>Underlined Text!</u><br/>
              <small>Small Text!</small><br/>
        </p>
  </card>
</wml>
```



See Chapter 3 for more information on the basic HTML elements.

WML tags

Most of the tags in WML are very similar to the tags in HTML. The major difference is that the number of attributes that they have is vastly reduced. Table 30-1 shows the set of WML tags in version 1.1.

Table 30-1

WML Tag	*Definition*
a	Anchor or link*
anchor	Anchor or link (see a)
access	Used to define permissions
b	Bold*
big	Larger font*
br	Line break*
card	A unit of user interaction (a page)

www.it-ebooks.info

WML Tag	*Definition*
`do`	A button for navigation or submitting a form
`em`	Emphasis*
`fieldset`	Groups a number of input controls together*
`go`	Used to display a given card
`head`	Head*
`i`	Italic*
`img`	Image*
`input`	Input control*
`meta`	Meta tag*
`noop`	No operation (used in event handlers)
`p`	Paragraph*
`postfield`	Used to specify a parameter and value to send to the server
`pre`	Preformatted text*
`prev`	Displays the previous page equivalent to `history.go(-1)` in JavaScript
`onevent`	Specifies an event handler
`optgroup`	Group options in sub menu*
`option`	Option in a select menu*
`refresh`	Refreshes the page applying variable
`select`	Select menu*
`small`	Small text*
`strong`	Bold*
`table`	Table*
`template`	A template for multiple cards
`timer`	Used to create a timer
`td`	Table data*
`tr`	Table row*
`u`	Underline*
`setvar`	Used to set a variable to a given value
`wml`	The root element of a WML page

*Similar to the `<HTML>` tag with the same name

www.it-ebooks.info

Shuffling the cards

We have now seen one card in a deck, but there are many more cards available. Of course, we need to provide some means of navigating from one card to the others. One possibility is with anchors.

Notice in the following WML file that each card has an `id` attribute. The relative URL for a card in WML is `#id` where `id` refers to the `id` specified in the card you want to show.

The file `shuffle.wml` follows:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
        <card id="one" title="Welcome to WML">
                <p><big>A Big Welcome to WML</big></p>
                <a href="#two">Next</a>
        </card>
        <card id="two" title="Welcome to WML">
                <p>A Medium Welcome to WML</p>
                <a href="#one">Prev</a>
                <a href="#three">Next</a>
        </card>
        <card id="three" title="Welcome to WML">
                <p><small>A Small Welcome to WML</small></p>
                <a href="#two">Prev</a>
        </card>
</wml>
```

Figure 30-2 shows the previous WML file.

Remember that although there can be several cards in one deck, 52 is going a bit too far.

To go from one card to another you would use the URL of the deck (the WML file) plus # and the `id` of the card. You can of course jump from one deck to another. For example, if we want to show the second card in the previous file from another WML file in the same directory, we would use the relative URL: "`shuffle.wml#two`".

```
<a href="shuffle.wml#two">Jump from one deck to another</a>
```



See Chapter 3 for more information on the basic HTML elements.

If you prefer buttons to hyperlinks, the last code changes to:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
```

www.it-ebooks.info

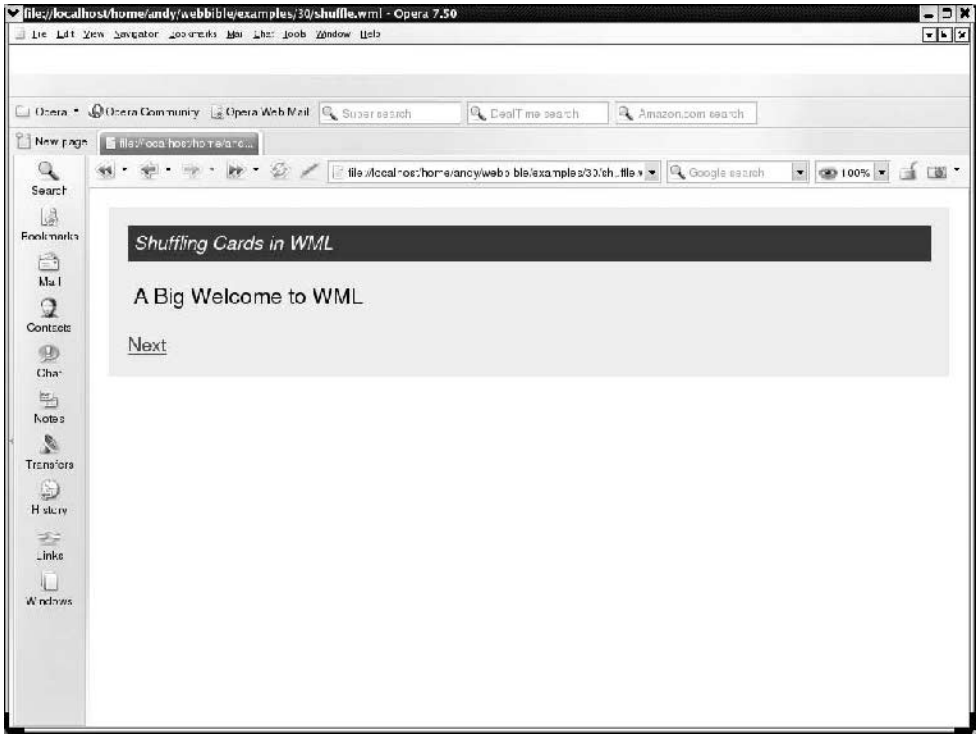


Figure 30-2: A WML document composed of several cards with hyperlinks.

```
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
        <card id="one" title="Shuffling Cards in WML">
                <p><big>A Big Welcome to WML</big></p>
                <do type="accept" label="Next">
                        <go href="#two"/>
                </do>
        </card>
        <card id="two" title="Shuffling Cards in WML">
                <p>A Medium Welcome to WML</p>
                <do type="accept" label="Prev">
                        <go href="#one"/>
                </do>
                <do type="accept" label="Next">
                        <go href="#three"/>
                </do>
        </card>
        <card id="three" title="Shuffling Cards in WML">
                <p><small>A Small Welcome to WML</small></p>
                <do type="accept" label="Prev">
                        <go href="#two"/>
                </do>
        </card>
</wml>
```

www.it-ebooks.info

Figure 30-3 shows the second card in the Opera browser.

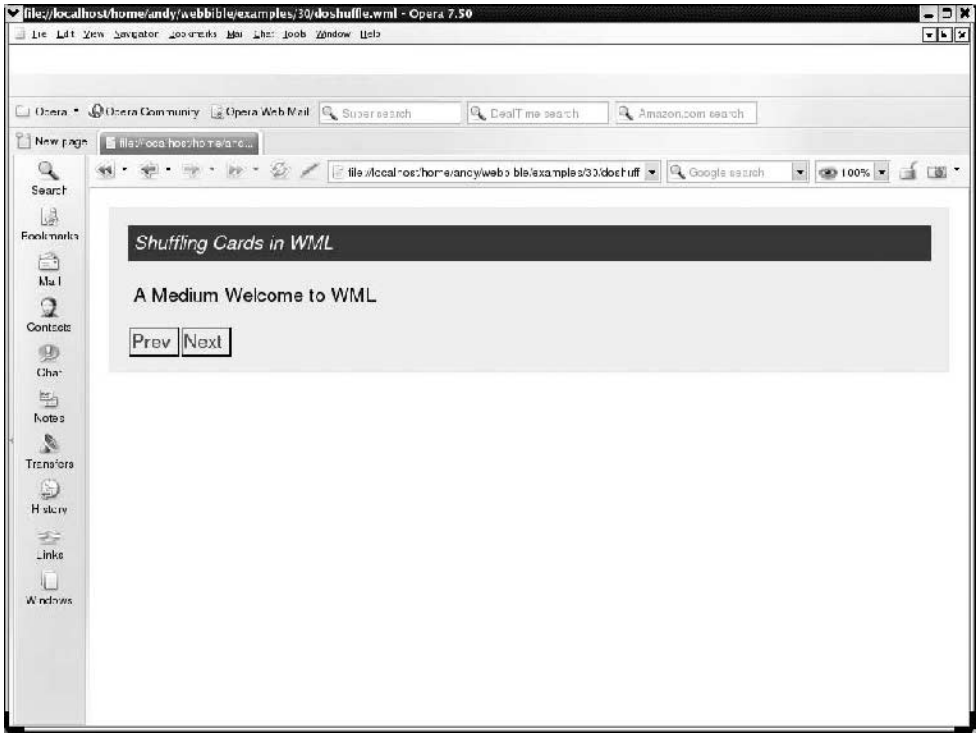


Figure 30-3: Here is the second of three cards with navigation buttons.

The equivalent in HTML would be to use:

```
<input type="button" onclick="window.location='next.html'" value="Next">
```



See Chapter 15 for information on buttons and Chapter 16 for details on JavaScript.

Interacting with Users

Up to now we have seen the way to present information to the user, but most sites want to do more than that. After all if someone wanted to navigate the Web, they would hardly choose a screen 1 inch by 1 inch to do so. The latest trend is mobile computing, where you can access the same sort of services available on the Web anytime, anyplace using a PDA, mobile phone, or an even smaller device. For example, many banks provide services such as consulting the balance or making a

www.it-ebooks.info

transaction. There is no reason to stop anyone from providing services for mobile devices. However, to do so you obviously need to solicit information from your users, even if it is just their login name and password.

Forms in WML

We have already seen in Chapter 15 how to use HTML forms to get input from users. In WML the form elements are very similar. However, in WML there is no `<form>` tag—all of the controls are directly in the card as if the card is already a form. WML has one-line text fields, multiline text areas, and single-selection and multiselection lists. The processing of the form values on the server is much the same as for HTML forms.

The main difference between WML and HTML is that there are no option button or check box input types. The equivalent is to use single-selection lists instead of option buttons and multiselection lists instead of check boxes. How a browser displays them depends on the browser implementation. It may display the options as a list or as option buttons or check boxes.

Here is an example of one-line input boxes:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
        <card id="one" title="User Interaction in WML">
                <p>Please enter your name</p>
                <input type="text" name="name"/>
                <do type="accept" label="Next">
                        <go href="#two"/>
                </do>
        </card>
        <card id="two" title="User Interaction in WML">
                <p>Hello $name</p>
                <do type="accept" label="Prev">
                        <go href="#one"/>
                </do>
        </card>
</wml>
```

Figures 30-4 and 30-5 show how forms and variables function in WML.

Specifying the format

One advantage of WML is that there are attributes for the `<input>` tag that specify the type of value a user must enter into a field before advancing to the next card or whether a field can be left blank. To specify what format is required, you use the `format` attribute.

www.it-ebooks.info

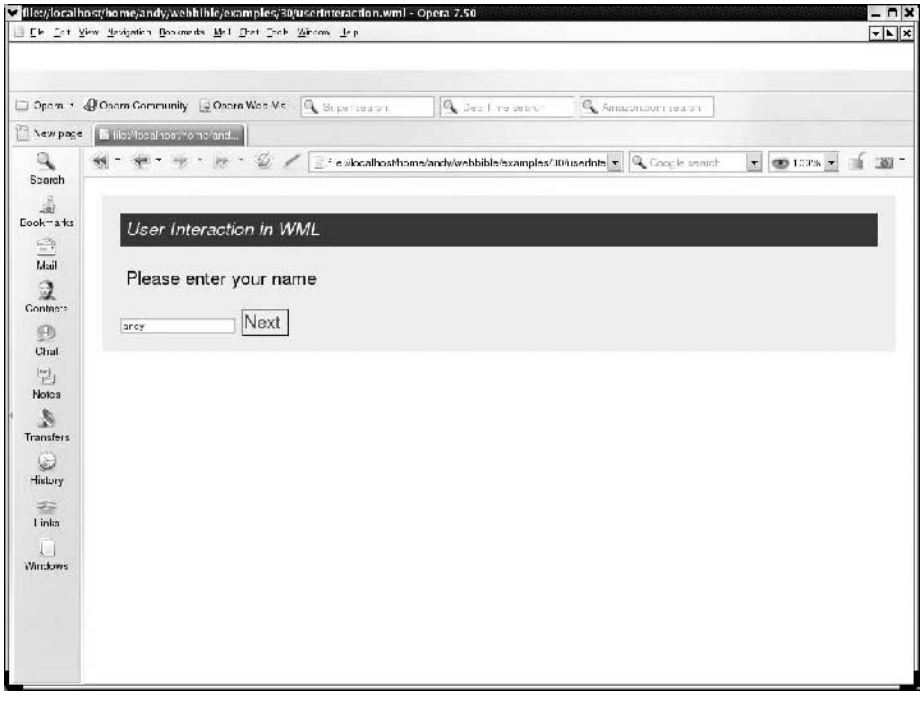


Figure 30-4: A WML document showing the form to enter your name.

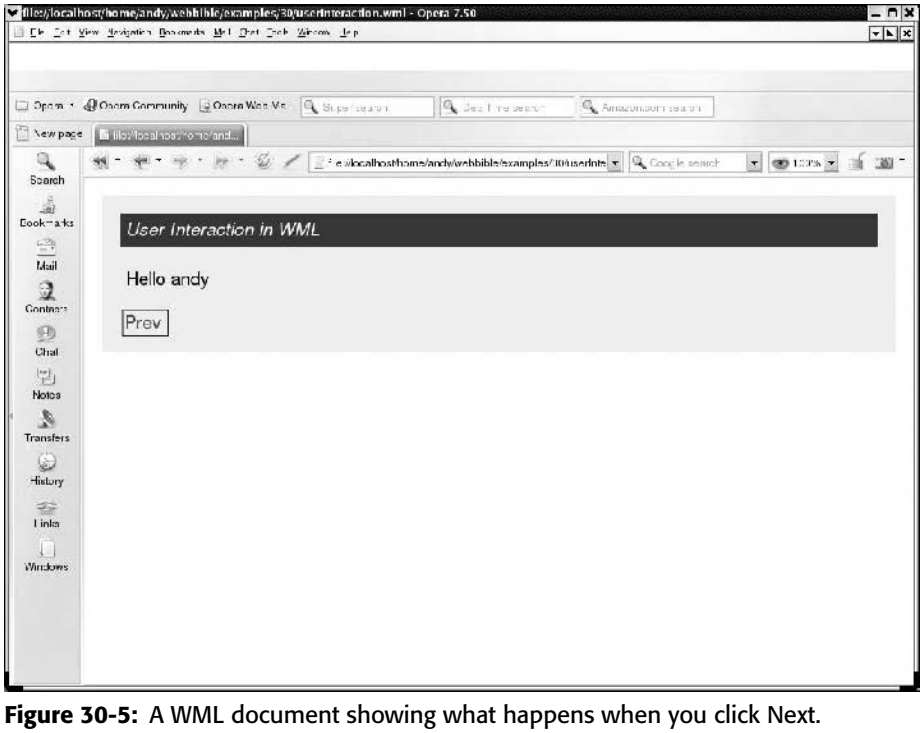


Figure 30-5: A WML document showing what happens when you click Next.

www.it-ebooks.info

Specifying `format="*M"` means any number of any character. This is the default format. The asterisk represents zero or more times. If you want to specify one or more times, you could use "`M*M`". This would mean one character plus zero or more characters, which adds up to one or more characters.

To restrict the input to only alphabetic characters you would use `format="*A"` for uppercase and `format="*a"` for lowercase. If you wanted to only allow numerical characters (0–9) then you would use `format="*N"`.

You can specify how many digits or characters you want to accept by either repeating a format code a certain number of times or by putting a digit before the previous codes. For example, to specify a maximum of five digits you can either use `format="5N"` or `format="NNNNN"`.

Specifying required values

`Emptyok` is the attribute for the input box that specifies, not surprisingly, whether the box can be left empty or not. The `emptyok` attribute works in conjunction with the `format` attribute to specify what values are acceptable.

The format mask by itself can be used to specify a required value. As mentioned earlier, "`M*M`" means one or more characters, so its obviously a required field. However, in this case you could accomplish the same result by just saying `emptyok="false"`. Remember that the default format value is "`*M`"; in other words, if the format mask says zero or more characters *and* `emptyok="false"`, meaning it cannot be zero characters, then it has to be one or more "M*M".

The default is `emptyok="true"` unless the format specifies otherwise. However, you can override what the format specification requires by explicitly including the `emptyok` attribute.

Although this approach is a little complicated at first sight, it is a great improvement on HTML where you have to use a scripting language to validate every input field.



See Chapter 15 for information on text boxes and Chapter 16 on JavaScript.

Menus a la card

It's an incredibly laborious task to enter text with a mobile phone. For those of you not familiar with mobile phones, you have to press a button anywhere from one to four times just to type in one letter. So when designing a WML site you should try to avoid using text boxes. One way of doing this is to use menus. As mentioned earlier, with WML you can specify single-selection and multiple-selection lists. It's almost the same as HTML—the basics are very similar to the HTML lists seen in Chapter 15.

www.it-ebooks.info

The following is a WML page showing a single-choice menu:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
  <card title="Menus in WML">
        <p><big>How about a menu</big></p>
        <select name="number">
                    <option value="1">One</option>
                    <option value="2">Two</option>
                    <option value="3">Three</option>
                    <option value="4">Four</option>
        </select>
        <do type="accept" label="ok">
              <go href="#next"/>
        </do>
 </card>
 <card id="next" title="Menus Choice">
        You chose $number
 </card>
</wml>
```

Figures 30-6 and 30-7 show the interaction of the two cards in this WML file.

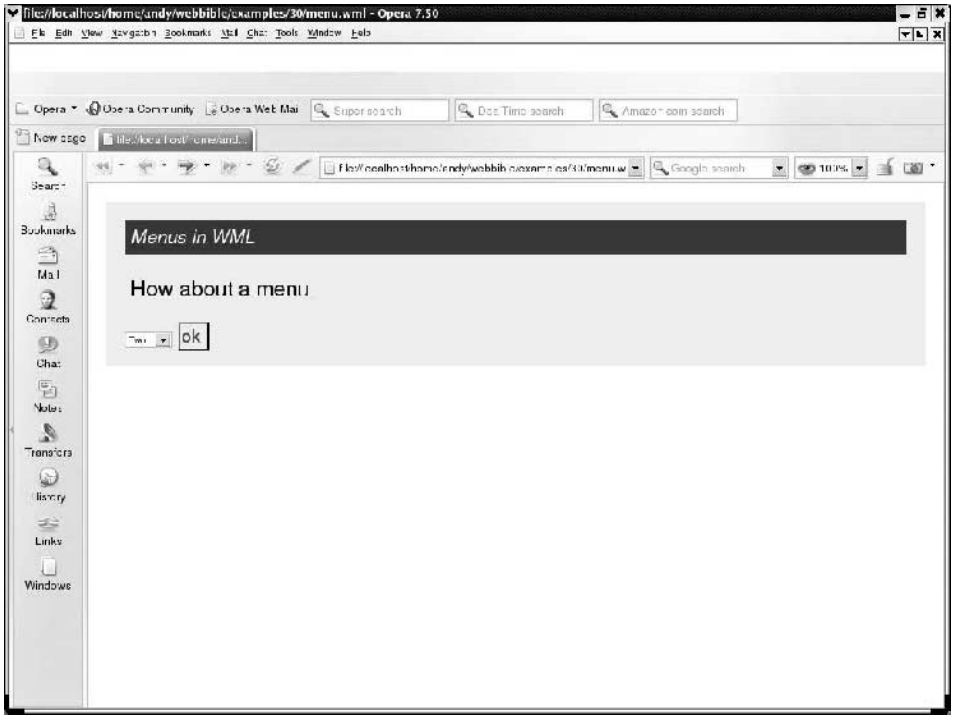


Figure 30-6: Here is a single-choice menu in WML. In some browsers this is presented as a set of option buttons.

www.it-ebooks.info

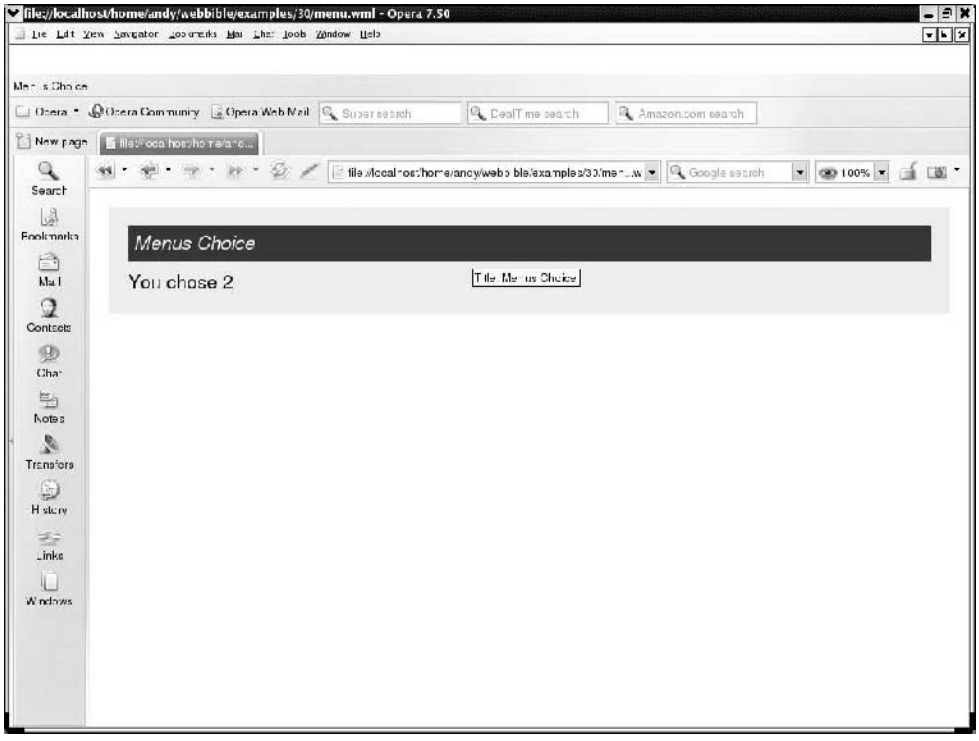


Figure 30-7: This is what you get after clicking OK in Figure 30-6.

Multiple-choice menus

For multiple-choice menus you have to specify that the attribute `multiple` is `true`. After the user selects from the list, the value of the variable number will be a semicolon-separated list of the chosen options.

Here is an example of a multiple-choice menu:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
  <card title="Menus in WML">
         <p><big>How about a multiple choice menu</big></p>
         <select name="number" multiple="true">
                     <option value="1">One</option>
                     <option value="2">Two</option>
                     <option value="3">Three</option>
                     <option value="4">Four</option>
          </select>
          <do type="accept" label="ok">
                 <go href="#next"/>
```

www.it-ebooks.info

```
        </do>
  </card>
  <card id="next" title="Menus Choice">
        You chose $number
  </card>
</wml>
```

Figure 30-8 shows the multiple-choice menu in action.



Figure 30-8: Here is an example of a multiple-choice menu. In some browsers it is presented as a set of check boxes.

Differences between HTML lists and WML lists

The main difference between HTML lists and WML lists is that WML provides extra attributes for select tags: `value`, `iname`, and `ivalue`.

In HTML if you want to specify that a certain option is selected you have to specify the selected attribute in the option or options. However, in WML you just specify the variable name with the `name` attribute. If the variable is set, the corresponding option is automatically selected. If the variable is not set, then the default value is specified by the `value` attribute.

www.it-ebooks.info

In WML you can also work with the indixes instead of the values. The attribute `iname` specifies the name of a variable that holds the value of the index. If the variable is undefined, then `ivalue` specifies the selected index or indixes.

This may not seem like a huge benefit, but when you get down to server-side programming you will appreciate that WML is much easier for this purpose.



See Chapter 15 for details on getting input with HTML forms.

Transforming XHTML into WML

You've probably been wondering what all of the fuss about XHTML and XSL was for. After reading this section, you'll understand why.

At the beginning of the chapter we talked about the challenge that Web developers face to design sites that can be viewed by both a desktop machine and a mobile phone. Of course, you would not be able to see the same pages on both devices, but the content of each page should be essentially the same.

One solution is to have the content in one file and separate templates that create the actual pages for each type of browser. For example, one template for a WML browser on cell phone with a minimal WML design, a second for an XHTML browser on a PDA only using basic XHTML, a third for a desktop machine with a resolution of 800 × 600, and yet another for a high-resolution graphical workstation with a broadband connection to the Internet. All this and more can be accomplished with XSL.

To demonstrate the principle, we are going to transform an example XHTML file into a WML file using a fairly simple XSL style sheet.

The XHTML file is:

```
<html>
<head>
<title>An example page</title>
</head>
<body>
<h2>This is an example page</h2>
<p>This is a paragraph</p>
<ul>
<li>List Item 1</li>
<li>List Item 2</li>
<li>List Item 3</li>
<li>List Item 4</li>
</ul>
<b>Bold</b>
<hr/>
<em>Emphasised</em>
</body>
</html>
```

www.it-ebooks.info

Figure 30-9 shows the XHTML file.

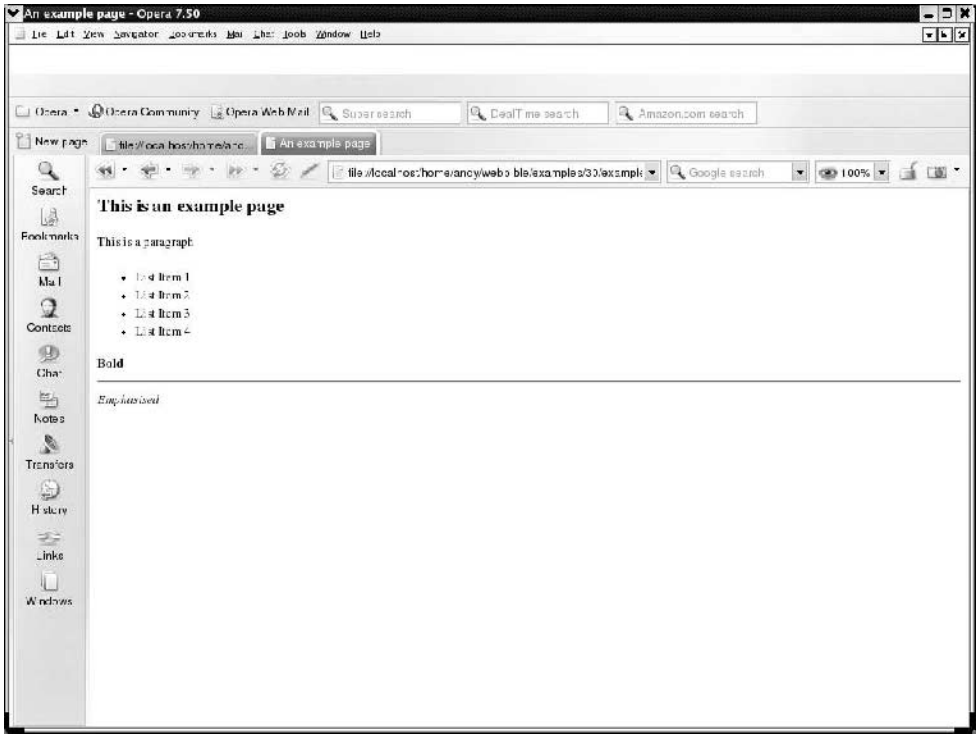


Figure 30-9: This is the XHTML file that will be transformed.

The XSL file is:

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="xml"/>

<!-- replace html with wml -->
<xsl:template match="html">
<wml>
<xsl:apply-templates match="@*|*|text()"/>
</wml>
</xsl:template>

<!-- copy all html elements complete with attributes and content.
Except for the templates after this one -->
<xsl:template match="*"> <!-- head|meta -->
<xsl:copy>
<xsl:apply-templates match="@*|*|text()"/>
</xsl:copy>
</xsl:template>
```

www.it-ebooks.info

```
<!-- replace headers with big text -->
<xsl:template match="h1|h2|h3|h4|h5|h6|h7|h8|h9">
<big>
<xsl:value-of select="."/>
</big>
</xsl:template>

<!-- copy the following elements without copying attributes -->
<xsl:template match="table|tr|td">
<xsl:copy>
<xsl:apply-templates/>
</xsl:copy>
</xsl:template>

<!-- change th to td -->
<xsl:template match="th">
<td class="th">
<xsl:apply-templates/>
</td>
</xsl:template>

<!-- change body to card -->
<xsl:template match="body">
<card title="{/html/head/title}">
<xsl:apply-templates/>
</card>
</xsl:template>

<!-- replace the following elements with their content and a line break -->
<xsl:template match="li|dt|hr">
<xsl:apply-templates/>
<br/>
</xsl:template>

<!-- replace the following elements with a paragraph -->
<xsl:template match="div|ul|ol|dl|blockquote|form">
<p>
<xsl:apply-templates/>
</p>
</xsl:template>

<!-- ignore the following elements -->
<xsl:template match="title">
</xsl:template>

</xsl:stylesheet>
```

The resulting WML file after being processed by a program to apply XSLT transforms is:

```
<?xml version="1.0"?>
<wml>
<head>

</head>
```

www.it-ebooks.info

```
<card title="An example page">
<big>This is an example page</big>
<p>This is a paragraph</p>
<p>
List Item 1<br/>
List Item 2<br/>
List Item 3<br/>
List Item 4<br/>
</p>
<b>Bold</b>
<br/>
<em>Emphasised</em>
</card>
</wml>
```

Figure 30-10 shows the resulting WML file.

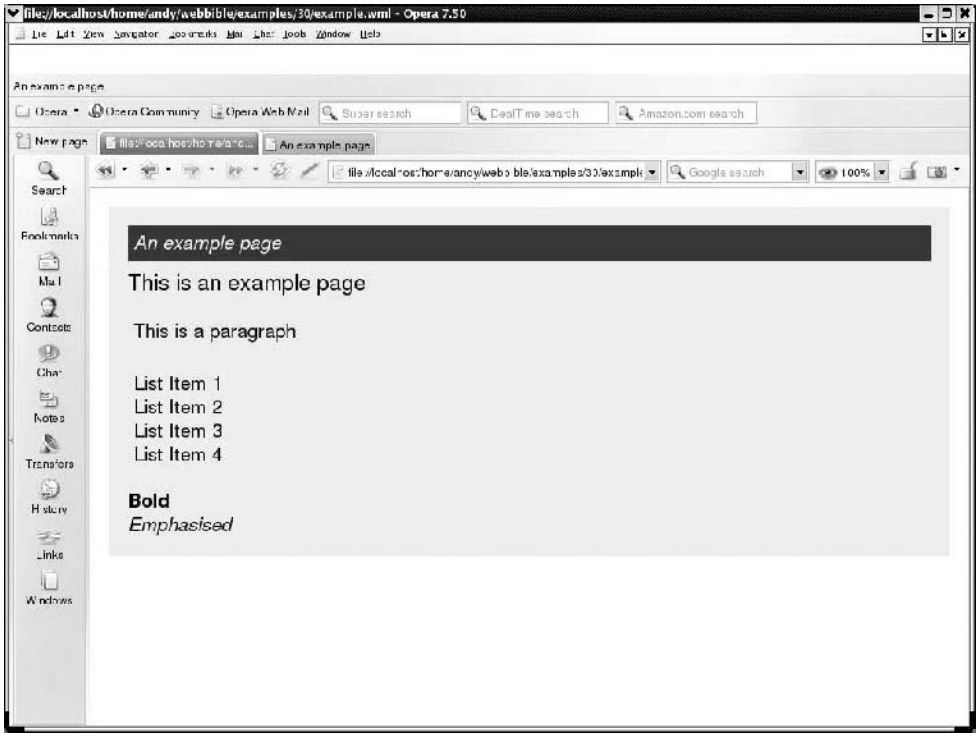


Figure 30-10: This is the result of our transformation.

Explaining how this transformation works is beyond the scope of this book. For more information on XSL, see the XSL specification `www.w3c.org`. A quick search on the Internet also returns a long list of tutorials; for a good tutorial go to `www.w3schools.org`.

www.it-ebooks.info



See Chapter 29 for more information on XML and XSLT transformations.

Summary

WML is very similar to XHTML but with the introduction of decks and cards. This chapter introduced you to the following concepts and topics:

- How to format text in WML
- How to use WML hyperlinks
- User interaction with text boxes
- Single and multiple-choice menus
- How to transform XHTML into WML

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

P A R T

VII

Appendixes



In This Part

Appendix A
HTML 4.01 Specification

Appendix B
XHTML 1.0, Second Edition Specification

Appendix C
XML 1.0, Third Edition Specification

Appendix D
WML 2.0 Specification

Appendix E
JavaScript Reference



www.it-ebooks.info

www.it-ebooks.info



HTML 4.01 Specification

HTML was the breakthrough that made the World Wide Web possible. The capability to link one file to another brought the early promise of hypertext—a nonlinear form of publishing—into reality. Beyond the simple but vital fact of linking, HTML's array of elements and their attributes made it possible to control the appearance of a document when displayed in a Web browser.

The language has gone through many changes over the years, both in concept and execution. The final version, 4.01, consists of the elements and attributes in this appendix.



See Chapter 3 for information on HTML fundamentals. Also see Appendix B for the XHTML specification.

HTML 4.01 Specification

The original of the HTML 4.01 specification can be found at `www.w3.org/TR/html401/`.

"Copyright © 1997–1999 World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved. (`www.w3.org/Consortium/Legal/`)"

The status of this document is a W3C Recommendation.

www.it-ebooks.info

Index of Elements

Legend: Optional, Forbidden, Empty, Deprecated, Loose DTD, Frameset DTD						
Name	***Start Tag***	***End Tag***	***Empty***	***Depr.***	***DTD***	***Description***
A						Anchor
ABBR						Abbreviated form (for example, WWW, HTTP, and so on)
ACRONYM						Acronym information
ADDRESS						Information on author
APPLET				D	L	Java applet
AREA		F	E			Client-side image map area
B						Bold text style
BASE		F	E			Document base URI
BASEFONT		F	E	D	L	Base font size
BDO						I18N BiDi over-ride
BIG						Large text style
BLOCKQUOTE						Long quotation
BODY	O	O				Document body
BR		F	E			Forced line break
BUTTON						Push button
CAPTION						Table caption
CENTER				D	L	Shorthand for DIV `align=center`
CITE						Citation
CODE						Computer code fragment
COL		F	E			Table column
COLGROUP		O				Table column group
DD		O				Definition description
DEL						Deleted text
DFN						Instance definition
DIR				D	L	Directory list

www.it-ebooks.info

Name	*Start Tag*	*End Tag*	*Empty*	*Depr.*	*DTD*	*Description*
DIV						Generic language or style container
DL						Definition list
DT		O				Definition term
EM						Emphasis
FIELDSET						Form control group
FONT				D	L	Local change to font
FORM						Interactive form
FRAME		F	E		F	Subwindow
FRAMESET					F	Window subdivision
H1						Heading
H2						Heading
H3						Heading
H4						Heading
H5						Heading
H6						Heading
HEAD	O	O				Document head
HR		F	E			Horizontal rule
HTML	O	O				Document root element
I						Italic text style
IFRAME					L	Inline subwindow
IMG		F	E			Embedded image
INPUT		F	E			Form control
INS						Inserted text
ISINDEX		F	E	D	L	Single line prompt
KBD						Text to be entered by the user
LABEL						Form field label text
LEGEND						Fieldset legend
LI		O				List item
LINK		F	E			Media-independent link
MAP						Client-side image map

Continued

www.it-ebooks.info

(continued)

Name	*Start Tag*	*End Tag*	*Empty*	*Depr.*	*DTD*	*Description*
MENU				D	L	Menu list
META		F	E			Generic meta-information
NOFRAMES					F	Alternate content container for non-frame–based rendering
NOSCRIPT						Alternate content container for non-script–based rendering
OBJECT						Generic embedded object
OL						Ordered list
OPTGROUP						Option group
OPTION		O				Selectable choice
P		O				Paragraph
PARAM		F	E			Named property value
PRE						Preformatted text
Q						Short inline quotation
S				D	L	Strikethrough text style
SAMP						Sample program output, scripts, and so on
SCRIPT						Script statements
SELECT						Option selector
SMALL						Small text style
SPAN						Generic language or style container
STRIKE				D	L	Strikethrough text
STRONG						Strong emphasis
STYLE						Style info
SUB						Subscript
SUP						Superscript
TABLE						Table element
TBODY	O	O				Table body
TD		O				Table data cell

www.it-ebooks.info

Name	Start Tag	End Tag	Empty	Depr.	DTD	Description
TEXTAREA						Multiline text field
TFOOT		O				Table footer
TH		O				Table header cell
THEAD		O				Table header
TITLE						Document title
TR		O				Table row
TT						Teletype or monospaced text style
U				D	L	Underlined text style
UL						Unordered list
VAR						Instance of a variable or program argument

Index of Attributes

Legend: Deprecated, Loose DTD, Frameset DTD						
Name	**Related Elements**	**Type**	**Default**	**Depr.**	**DTD**	**Comment**
abbr	TD, TH	%Text;	#IMPLIED			Abbreviation for header cell
accept-charset	FORM	%Charsets;	#IMPLIED			List of supported charsets
accept	FORM, INPUT	%ContentTypes;	#IMPLIED			List of MIME types for file upload
accesskey	A, AREA, BUTTON, INPUT, LABEL, LEGEND, TEXTAREA	%Character;	#IMPLIED			Accessibility key character
action	FORM	%URI;	#REQUIRED			Server-side form handler

Continued

www.it-ebooks.info

(continued)

Name	*Related Elements*	*Type*	*Default*	*Depr.*	*DTD*	*Comment*
align	CAPTION	%CAlign;	#IMPLIED	D	L	Relative to table
align	APPLET, IFRAME, IMG, INPUT, OBJECT	%IAlign;	#IMPLIED	D	L	Vertical or horizontal alignment
align	LEGEND	%LAlign;	#IMPLIED	D	L	Relative to fieldset
align	TABLE	%TAlign;	#IMPLIED	D	L	Table position relative to window
align	HR	(left | center | right)	#IMPLIED	D	L	
align	DIV, H1, H2, H3, H4, H5, H6, P	(left | center | right | justify)	#IMPLIED	D	L	Align, text alignment
align	COL, COLGROUP, TBODY, TD, TFOOT, TH, THEAD, TR	(left | center | right | justify | char)	#IMPLIED			
alink	BODY	%Color;	#IMPLIED	D	L	Color of selected links
alt	APPLET	%Text;	#IMPLIED	D	L	Short description
alt	AREA, IMG	%Text;	#REQUIRED			Short description
alt	INPUT	CDATA	#IMPLIED			Short description
archive	APPLET	CDATA	#IMPLIED	D	L	Comma-separated archive list
archive	OBJECT	CDATA	#IMPLIED			Space-separated list of URIs
axis	TD, TH	CDATA	#IMPLIED			Comma-separated list of related headers
background	BODY	%URI;	#IMPLIED	D	L	Texture tile for document background
bgcolor	TABLE	%Color;	#IMPLIED	D	L	Background color for cells

www.it-ebooks.info

Name	Related Elements	Type	Default	Depr.	DTD	Comment
bgcolor	TR	%Color;	#IMPLIED	D	L	Background color for rows
bgcolor	TD, TH	%Color;	#IMPLIED	D	L	Cell background color
bgcolor	BODY	%Color;	#IMPLIED	D	L	Document background color
border	TABLE	%Pixels;	#IMPLIED			Controls frame width around table
border	IMG, OBJECT	%Pixels;	#IMPLIED	D	L	Link border width
cellpadding	TABLE	%Length;	#IMPLIED			Spacing within cells
cellspacing	TABLE	%Length;	#IMPLIED			Spacing between cells
char	COL, COLGROUP, TBODY, TD, TFOOT, TH, THEAD, TR	%Character;	#IMPLIED			Slignment char, e.g. char=':'
charoff	COL, COLGROUP, TBODY, TD, TFOOT, TH, THEAD, TR	%Length;	#IMPLIED			Offset for alignment char
charset	A, LINK, SCRIPT	%Charset;	#IMPLIED			Char encoding of linked resource
checked	INPUT	(checked)	#IMPLIED			For option buttons and check boxes
cite	BLOCKQUOTE, Q	%URI;	#IMPLIED			URI for source document or msg
cite	DEL, INS	%URI;	#IMPLIED			Info on reason for change
class	All elements but BASE, BASEFONT, HEAD, HTML, META, PARAM, SCRIPT, STYLE, TITLE	CDATA	#IMPLIED			Space-separated list of classes

Continued

www.it-ebooks.info

(continued)

Name	Related Elements	Type	Default	Depr.	DTD	Comment
classid	OBJECT	%URI;	#IMPLIED			Identifies an implementation
clear	BR	(left | all | right | none)	none	D	L	Control of text flow
code	APPLET	CDATA	#IMPLIED	D	L	Applet class file
codebase	OBJECT	%URI;	#IMPLIED			Base URI for classid, data, archive
codebase	APPLET	%URI;	#IMPLIED	D	L	Optional base URI for applet
codetype	OBJECT	%ContentType;	#IMPLIED			Content type for code
color	BASEFONT, FONT	%Color;	#IMPLIED	D	L	Text color
cols	FRAMESET	%MultiLengths;	#IMPLIED		F	List of lengths, default: 100% (1 col)
cols	TEXTAREA	NUMBER	#REQUIRED			Columns
colspan	TD, TH	NUMBER	1			Number of cols spanned by cell
compact	DIR, DL, MENU, OL, UL	(compact)	#IMPLIED	D	L	Reduced interitem spacing
content	META	CDATA	#REQUIRED			Associated information
coords	AREA	%Coords;	#IMPLIED			Comma-separated list of lengths
coords	A	%Coords;	#IMPLIED			For use with client-side image maps
data	OBJECT	%URI;	#IMPLIED			Reference to object data
datetime	DEL, INS	%Datetime;	#IMPLIED			Date and time of change

www.it-ebooks.info

Name	*Related Elements*	*Type*	*Default*	*Depr.*	*DTD*	*Comment*
declare	OBJECT	(declare)	#IMPLIED			Declare but don't instantiate flag
defer	SCRIPT	(defer)	#IMPLIED			UA may defer execution of script
dir	All elements but APPLET, BASE, BASEFONT, BDO, BR, FRAME, FRAMESET, IFRAME, PARAM, SCRIPT	(ltr | rtl)	#IMPLIED			Direction for weak or neutral text
dir	BDO	(ltr | rtl)	#REQUIRED			Directionality
disabled	BUTTON, INPUT, OPTGROUP, OPTION, SELECT, TEXTAREA	(disabled)	#IMPLIED			Unavailable in this context
enctype	FORM	%Content-Type;	"application/x-www-form-urlencoded"			
face	BASEFONT, FONT	CDATA	#IMPLIED	D	L	Comma-separated list of font names
for	LABEL	IDREF	#IMPLIED			Matches field ID value
frame	TABLE	%TFrame;	#IMPLIED			Which parts of a frame to render
frame-border	FRAME, IFRAME	(1 | 0)	1		F	Request frame borders?
headers	TD, TH	IDREFS	#IMPLIED			List of id's for header cells
height	IFRAME	%Length;	#IMPLIED		L	Frame height
height	TD, TH	%Length;	#IMPLIED	D	L	Height for cell

Continued

www.it-ebooks.info

(continued)

Name	*Related Elements*	*Type*	*Default*	*Depr.*	*DTD*	*Comment*
height	IMG, OBJECT	%Length;	#IMPLIED			Override height
height	APPLET	%Length;	#REQUIRED	D	L	Initial height
href	A, AREA, LINK	%URI;	#IMPLIED			URI for linked resource
href	BASE	%URI;	#IMPLIED			URI that acts as base URI
hreflang	A, LINK	%Language-Code;	#IMPLIED			Language code
hspace	APPLET, IMG, OBJECT	%Pixels;	#IMPLIED	D	L	Horizontal gutter
http-equiv	META	NAME	#IMPLIED			HTTP response header name
id	All elements but BASE, HEAD, HTML, META, SCRIPT, STYLE, TITLE	ID	#IMPLIED			Document-wide unique id
ismap	IMG, INPUT	(ismap)	#IMPLIED			Use server-side image map
label	OPTION	%Text;	#IMPLIED			For use in hierarchical menus
label	OPTGROUP	%Text;	#REQUIRED			For use in hierarchical menus
lang	All elements but APPLET, BASE, BASEFONT, BR, FRAME, FRAMESET, IFRAME, PARAM, SCRIPT	%Language-Code;	#IMPLIED			Language code
language	SCRIPT	CDATA	#IMPLIED	D	L	Predefined script language name
link	BODY	%Color;	#IMPLIED	D	L	Color of links
longdesc	IMG	%URI;	#IMPLIED			Link to long description (complements alt)

www.it-ebooks.info

Name	*Related Elements*	*Type*	*Default*	*Depr.*	*DTD*	*Comment*
longdesc	FRAME, IFRAME	%URI;	#IMPLIED		F	Link to long description (complements title)
margin-height	FRAME, IFRAME	%Pixels;	#IMPLIED		F	Margin height in pixels
margin-width	FRAME, IFRAME	%Pixels;	#IMPLIED		F	Margin widths in pixels
max-length	INPUT	NUMBER	#IMPLIED			Max chars for text fields
media	STYLE	%MediaDesc;	#IMPLIED			Designed for use with these media
media	LINK	%MediaDesc;	#IMPLIED			For rendering on these media
method	FORM	(GET | POST)	GET			HTTP method used to submit the form
multiple	SELECT	(multiple)	#IMPLIED			Default is single selection
name	BUTTON, TEXTAREA	CDATA	#IMPLIED			
name	APPLET	CDATA	#IMPLIED	D	L	Allows applets to find each other
name	SELECT	CDATA	#IMPLIED			Field name
name	FORM	CDATA	#IMPLIED			Name of form for scripting
name	FRAME, IFRAME	CDATA	#IMPLIED		F	Name of frame for targeting
name	IMG	CDATA	#IMPLIED			Name of image for scripting
name	A	CDATA	#IMPLIED			Named link end
name	INPUT, OBJECT	CDATA	#IMPLIED			Submit as part of form
name	MAP	CDATA	#REQUIRED			For reference by usemap
name	PARAM	CDATA	#REQUIRED			Property name
name	META	NAME	#IMPLIED			Meta-information name

Continued

www.it-ebooks.info

(continued)

Name	*Related Elements*	*Type*	*Default*	*Depr.*	*DTD*	*Comment*
nohref	AREA	(nohref)	#IMPLIED			This region has no action
noresize	FRAME	(noresize)	#IMPLIED		F	Allow users to resize frames?
noshade	HR	(noshade)	#IMPLIED	D	L	
nowrap	TD, TH	(nowrap)	#IMPLIED	D	L	Suppress word wrap
object	APPLET	CDATA	#IMPLIED	D	L	Serialized applet file
onblur	A, AREA, BUTTON, INPUT, LABEL, SELECT, TEXTAREA	%Script;	#IMPLIED			The element lost the focus
onchange	INPUT, SELECT, TEXTAREA	%Script;	#IMPLIED			The element value was changed
onclick	All elements but APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED			A pointer button was clicked
ondbl click	All elements but APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED			A pointer button was double-clicked
onfocus	A, AREA, BUTTON, INPUT, LABEL, SELECT, TEXTAREA	%Script;	#IMPLIED			The element got the focus

www.it-ebooks.info

Name	*Related Elements*	*Type*	*Default*	*Depr.*	*DTD*	*Comment*
onkey down	All elements but APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED			A key was pressed down
onkey press	All elements but APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED			A key was pressed and released
onkeyup	All elements but APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED			A key was released
onload	FRAMESET	%Script;	#IMPLIED		F	All the frames have been loaded
onload	BODY	%Script;	#IMPLIED			The document has been loaded
onmouse down	All elements but APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED			A pointer button was pressed down

Continued

www.it-ebooks.info

(continued)

Name	*Related Elements*	*Type*	*Default*	*Depr.*	*DTD*	*Comment*
onmouse-move	All elements but APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED			A pointer was moved within
onmouse-out	All elements but APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED			A pointer was moved away
onmouse-over	All elements but APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED			A pointer was moved onto
onmouse-up	All elements but APPLET, BASE, BASEFONT, BDO, BR, FONT, FRAME, FRAMESET, HEAD, HTML, IFRAME, ISINDEX, META, PARAM, SCRIPT, STYLE, TITLE	%Script;	#IMPLIED			A pointer button was released
onreset	FORM	%Script;	#IMPLIED			The form was reset
onselect	INPUT, TEXTAREA	%Script;	#IMPLIED			Some text was selected

www.it-ebooks.info

Name	*Related Elements*	*Type*	*Default*	*Depr.*	*DTD*	*Comment*
onsubmit	FORM	%Script;	#IMPLIED			The form was submitted
onunload	FRAMESET	%Script;	#IMPLIED		F	All the frames have been removed
onunload	BODY	%Script;	#IMPLIED			The document has been removed
profile	HEAD	%URI;	#IMPLIED			Named dictionary of meta info
prompt	ISINDEX	%Text;	#IMPLIED	D	L	Prompt message
readonly	TEXTAREA	(readonly)	#IMPLIED			
readonly	INPUT	(readonly)	#IMPLIED			For text and passwd
rel	A, LINK	%LinkTypes;	#IMPLIED			Forward link types
rev	A, LINK	%LinkTypes;	#IMPLIED			Reverse link types
rows	FRAMESET	%MultiLengths;	#IMPLIED		F	List of lengths, default: 100% (1 row)
rows	TEXTAREA	NUMBER	#REQUIRED			
rowspan	TD, TH	NUMBER	1			Number of rows spanned by cell
rules	TABLE	%TRules;	#IMPLIED			Rulings between rows and cols
scheme	META	CDATA	#IMPLIED			Select form of content
scope	TD, TH	%Scope;	#IMPLIED			Scope covered by header cells
scrolling	FRAME, IFRAME	(yes | no | auto)	auto		F	Scrollbar or none
selected	OPTION	(selected)	#IMPLIED			

Continued

www.it-ebooks.info

(continued)

Name	*Related Elements*	*Type*	*Default*	*Depr.*	*DTD*	*Comment*
shape	AREA	%Shape;	rect			Controls interpretation of coords
shape	A	%Shape;	rect			For use with client-side image maps
size	HR	%Pixels;	#IMPLIED	D	L	
size	FONT	CDATA	#IMPLIED	D	L	[+|-]nn, e.g. `size="+1"`, `size="4"`
size	INPUT	CDATA	#IMPLIED			Specific to each type of field
size	BASEFONT	CDATA	#REQUIRED	D	L	Base font size for FONT elements
size	SELECT	NUMBER	#IMPLIED			Rows visible
span	COL	NUMBER	1			COL attributes affect N columns
span	COLGROUP	NUMBER	1			Default number of columns in group
src	SCRIPT	%URI;	#IMPLIED			URI for an external script
src	INPUT	%URI;	#IMPLIED			For fields with images
src	FRAME, IFRAME	%URI;	#IMPLIED		F	Source of frame content
src	IMG	%URI;	#REQUIRED			URI of image to embed
standby	OBJECT	%Text;	#IMPLIED			Message to show while loading
start	OL	NUMBER	#IMPLIED	D	L	Starting sequence number

www.it-ebooks.info

Name	*Related Elements*	*Type*	*Default*	*Depr.*	*DTD*	*Comment*
style	All elements but BASE, BASEFONT, HEAD, HTML, META, PARAM, SCRIPT, STYLE, TITLE	%Style Sheet;	#IMPLIED			Associated style info
summary	TABLE	%Text;	#IMPLIED			Purpose or structure for speech output
tabindex	A, AREA, BUTTON, INPUT, OBJECT, SELECT, TEXTAREA	NUMBER	#IMPLIED			Position in tabbing order
target	A, AREA, BASE, FORM, LINK	%Frame Target;	#IMPLIED		L	Render in this frame
text	BODY	%Color;	#IMPLIED	D	L	Document text color
title	All elements but BASE, BASEFONT, HEAD, HTML, META, PARAM, SCRIPT, TITLE	%Text;	#IMPLIED			Advisory title
type	A, LINK	%Content Type;	#IMPLIED			Advisory content type
type	OBJECT	%Content Type;	#IMPLIED			Content type for data
type	PARAM	%Content Type;	#IMPLIED			Content type for value when valuetype=ref
type	SCRIPT	%Content Type;	#REQUIRED			Content type of script language
type	STYLE	%Content Type;	#REQUIRED			Content type of style language
type	INPUT	%Input Type;	TEXT			What kind of widget is needed
type	LI	%LIStyle;	#IMPLIED	D	L	List item style
type	OL	%OLStyle;	#IMPLIED	D	L	Numbering style
type	UL	%ULStyle;	#IMPLIED	D	L	Bullet style

Continued

www.it-ebooks.info

(continued)

Name	*Related Elements*	*Type*	*Default*	*Depr.*	*DTD*	*Comment*
type	BUTTON	(button | submit | reset)	submit			For use as form button
usemap	IMG, INPUT, OBJECT	%URI;	#IMPLIED			Use client-side Image map
valign	COL, COLGROUP, TBODY, TD, TFOOT, TH, THEAD, TR	(top | middle | bottom | baseline)	#IMPLIED			Vertical alignment in cells
value	INPUT	CDATA	#IMPLIED			Specify for radio buttons and check boxes
value	OPTION	CDATA	#IMPLIED			Defaults to element content
value	PARAM	CDATA	#IMPLIED			Property value
value	BUTTON	CDATA	#IMPLIED			Sent to server when submitted
value	LI	NUMBER	#IMPLIED	D	L	Reset sequence number
valuetype	PARAM	(DATA | REF | OBJECT)	DATA			How to interpret value
version	HTML	CDATA	%HTML .Version;	D	L	Constant
vlink	BODY	%Color;	#IMPLIED	D	L	Color of visited links
vspace	APPLET, IMG, OBJECT	%Pixels;	#IMPLIED	D	L	Vertical gutter
width	HR	%Length;	#IMPLIED	D	L	
width	IFRAME	%Length;	#IMPLIED		L	Frame width
width	IMG, OBJECT	%Length;	#IMPLIED			Override width
width	TABLE	%Length;	#IMPLIED			Table width

www.it-ebooks.info

Name	*Related Elements*	*Type*	*Default*	*Depr.*	*DTD*	*Comment*
width	TD, TH	%Length;	#IMPLIED	D	L	Width for cell
width	APPLET	%Length;	#REQUIRED	D	L	Initial width
width	COL	%MultiLength;	#IMPLIED			Column width specification
width	COLGROUP	%MultiLength;	#IMPLIED			Default width for enclosed COLs
width	PRE	NUMBER	#IMPLIED	D	L	

www.it-ebooks.info

www.it-ebooks.info

APPENDIX B



XHTML 1.0, Second Edition Specification

XHTML, the eXtensible HyperText Markup Language, is an outgrowth of HTML, the Hypertext Markup Language. In XHTML, HTML is made compatible with the principles underlying XML, the eXtensible Markup Language.



See Chapter 29 for information on XML and XHTML. Also see Appendix C for the XML specification.

This appendix contains the official World Wide Web Consortium specification for XHTML 1.0, Second Edition.

XHTML™ 1.0 The Extensible HyperText Markup Language (Second Edition)

A Reformulation of HTML 4 in XML 1.0

W3C Recommendation 26 January 2000, revised 1 August 2002

This version: `www.w3.org/TR/2002/REC-xhtml1-20020801`

Latest version: `www.w3.org/TR/xhtml1`

Previous version: `www.w3.org/TR/2000/REC-xhtml1-20000126`

Diff-marked version: `www.w3.org/TR/2002/REC-xhtml1-20020801/xhtml1-diff.html`

Authors:

See acknowledgments.

www.it-ebooks.info

Please refer to the **errata** for this document, which may include some normative corrections. See also **translations**.

This document is also available in these non-normative formats: Multi-part XHTML file, PostScript version, PDF version, ZIP archive, and Gzip'd TAR archive.

Copyright ©2002 W3C® (MIT, INRIA, Keio), All Rights Reserved. W3C liability, trademark, document use and software licensing rules apply.

Abstract

This specification defines the Second Edition of XHTML 1.0, a reformulation of HTML 4 as an XML 1.0 application, and three DTDs corresponding to the ones defined by HTML 4. The semantics of the elements and their attributes are defined in the W3C Recommendation for HTML 4. These semantics provide the foundation for future extensibility of XHTML. Compatibility with existing HTML user agents is possible by following a small set of guidelines.

Status of this document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. The latest status of this document series is maintained at the W3C.

This document is the second edition of the XHTML 1.0 specification incorporating the errata changes as of 1 August 2002. Changes between this version and the previous Recommendation are illustrated in a diff-marked version.

This second edition is *not* a new version of XHTML 1.0 (first published 26 January 2000). The changes in this document reflect corrections applied as a result of comments submitted by the community and as a result of ongoing work within the HTML Working Group. There are no substantive changes in this document - only the integration of various errata.

The list of known errors in this specification is available at `http://www.w3.org/2002/08/REC-xhtml1-20020801-errata`.

Please report errors in this document to www-html-editor@w3.org (archive). Public discussion on HTML features takes place on the mailing list www-html@w3.org (archive).

This document has been produced as part of the W3C HTML Activity. The goals of the HTML Working Group *(members only)* are discussed in the HTML Working Group charter.

At the time of publication, the working group believed there were zero patent disclosures relevant to this specification. A current list of patent disclosures relevant to this specification may be found on the Working Group's patent disclosure page.

www.it-ebooks.info

A list of current W3C Recommendations and other technical documents can be found at `http://www.w3.org/TR`.

Quick Table of Contents

1. What is XHTML?
2. Definitions
3. Normative Definition of XHTML 1.0
4. Differences with HTML 4
5. Compatibility Issues
   - A. DTDs
   - B. Element Prohibitions
   - C. HTML Compatibility Guidelines
   - D. Acknowledgements
   - E. References

Full Table of Contents

1. WHAT IS XHTML? 705
   - 1.1. WHAT IS HTML 4? 705
   - 1.2. WHAT IS XML? 706
   - 1.3. WHY THE NEED FOR XHTML? 706
2. DEFINITIONS 707
   - 2.1. TERMINOLOGY 707
   - 2.2. GENERAL TERMS 708
3. NORMATIVE DEFINITION OF XHTML 1.0 709
   - 3.1. DOCUMENT CONFORMANCE 709
     - 3.1.1. Strictly Conforming Documents 709
     - 3.1.2. Using XHTML with other namespaces 710
   - 3.2. USER AGENT CONFORMANCE 711
4. DIFFERENCES WITH HTML 4 712
   - 4.1. DOCUMENTS MUST BE WELL-FORMED 712
   - 4.2. ELEMENT AND ATTRIBUTE NAMES MUST BE IN LOWER CASE 713
   - 4.3. FOR NON-EMPTY ELEMENTS, END TAGS ARE REQUIRED 713
   - 4.4. ATTRIBUTE VALUES MUST ALWAYS BE QUOTED 713
   - 4.5. ATTRIBUTE MINIMIZATION 713

www.it-ebooks.info

4.6. EMPTY ELEMENTS 714
4.7. WHITE SPACE HANDLING IN ATTRIBUTE VALUES 714
4.8. SCRIPT AND STYLE ELEMENTS 714
4.9. SGML EXCLUSIONS 714
4.10. THE ELEMENTS WITH 'ID' AND 'NAME' ATTRIBUTES 715
4.11. ATTRIBUTES WITH PRE-DEFINED VALUE SETS 715
4.12. ENTITY REFERENCES AS HEX VALUES 715
5. COMPATIBILITY ISSUES 715
5.1. INTERNET MEDIA TYPE 715
A. DTDS 716
A.1. DOCUMENT TYPE DEFINITIONS 716
A.1.1. XHTML-1.0-Strict 716
A.1.2. XHTML-1.0-Transitional 716
A.1.3. XHTML-1.0-Frameset 716
A.2. ENTITY SETS 716
A.2.1. Latin-1 characters 716
A.2.2. Special characters 716
A.2.3. Symbols 717
B. ELEMENT PROHIBITIONS 717
C. HTML COMPATIBILITY GUIDELINES 717
C.1. PROCESSING INSTRUCTIONS AND THE XML DECLARATION 717
C.2. EMPTY ELEMENTS 718
C.3. ELEMENT MINIMIZATION AND EMPTY ELEMENT CONTENT 718
C.4. EMBEDDED STYLE SHEETS AND SCRIPTS 718
C.5. LINE BREAKS WITHIN ATTRIBUTE VALUES 718
C.6. ISINDEX 718
C.7. THE LANG AND XML:LANG ATTRIBUTES 718
C.8. FRAGMENT IDENTIFIERS 718
C.9. CHARACTER ENCODING 719
C.10. BOOLEAN ATTRIBUTES 719
C.11. DOCUMENT OBJECT MODEL AND XHTML 720
C.12. USING AMPERSANDS IN ATTRIBUTE VALUES (AND ELSEWHERE) 720
C.13. CASCADING STYLE SHEETS (CSS) AND XHTML 720

www.it-ebooks.info

C.14. REFERENCING STYLE ELEMENTS WHEN SERVING AS XML 721
C.15. WHITE SPACE CHARACTERS IN HTML VS. XML 722
C.16. THE NAMED CHARACTER REFERENCE &APOS; 722
D. ACKNOWLEDGEMENTS 722
E. REFERENCES 723

1. What is XHTML?

This section is informative.

XHTML is a family of current and future document types and modules that reproduce, subset, and extend HTML 4 [HTML4]. XHTML family document types are XML based, and ultimately are designed to work in conjunction with XML-based user agents. The details of this family and its evolution are discussed in more detail in [XHTMLMOD].

XHTML 1.0 (this specification) is the first document type in the XHTML family. It is a reformulation of the three HTML 4 document types as applications of XML 1.0 [XML]. It is intended to be used as a language for content that is both XML-conforming and, if some simple guidelines are followed, operates in HTML 4 conforming user agents. Developers who migrate their content to XHTML 1.0 will realize the following benefits:

- XHTML documents are XML conforming. As such, they are readily viewed, edited, and validated with standard XML tools.
- XHTML documents can be written to operate as well or better than they did before in existing HTML 4-conforming user agents as well as in new, XHTML 1.0 conforming user agents.
- XHTML documents can utilize applications (e.g. scripts and applets) that rely upon either the HTML Document Object Model or the XML Document Object Model [DOM].
- As the XHTML family evolves, documents conforming to XHTML 1.0 will be more likely to interoperate within and among various XHTML environments.

The XHTML family is the next step in the evolution of the Internet. By migrating to XHTML today, content developers can enter the XML world with all of its attendant benefits, while still remaining confident in their content's backward and future compatibility.

1.1. What is HTML 4?

HTML 4 [HTML4] is an SGML (Standard Generalized Markup Language) application conforming to International Standard ISO 8879, and is widely regarded as the standard publishing language of the World Wide Web.

SGML is a language for describing markup languages, particularly those used in electronic document exchange, document management, and document publishing. HTML is an example of a language defined in SGML.

www.it-ebooks.info

SGML has been around since the middle 1980's and has remained quite stable. Much of this stability stems from the fact that the language is both feature-rich and flexible. This flexibility, however, comes at a price, and that price is a level of complexity that has inhibited its adoption in a diversity of environments, including the World Wide Web.

HTML, as originally conceived, was to be a language for the exchange of scientific and other technical documents, suitable for use by non-document specialists. HTML addressed the problem of SGML complexity by specifying a small set of structural and semantic tags suitable for authoring relatively simple documents. In addition to simplifying the document structure, HTML added support for hypertext. Multimedia capabilities were added later.

In a remarkably short space of time, HTML became wildly popular and rapidly outgrew its original purpose. Since HTML's inception, there has been rapid invention of new elements for use within HTML (as a standard) and for adapting HTML to vertical, highly specialized, markets. This plethora of new elements has led to interoperability problems for documents across different platforms.

1.2. What is XML?

XML™ is the shorthand name for Extensible Markup Language [XML].

XML was conceived as a means of regaining the power and flexibility of SGML without most of its complexity. Although a restricted form of SGML, XML nonetheless preserves most of SGML's power and richness, and yet still retains all of SGML's commonly used features.

While retaining these beneficial features, XML removes many of the more complex features of SGML that make the authoring and design of suitable software both difficult and costly.

1.3. Why the need for XHTML?

The benefits of migrating to XHTML 1.0 are described above. Some of the benefits of migrating to XHTML in general are:

- Document developers and user agent designers are constantly discovering new ways to express their ideas through new markup. In XML, it is relatively easy to introduce new elements or additional element attributes. The XHTML family is designed to accommodate these extensions through XHTML modules and techniques for developing new XHTML-conforming modules (described in the XHTML Modularization specification). These modules will permit the combination of existing and new feature sets when developing content and when designing new user agents.
- Alternate ways of accessing the Internet are constantly being introduced. The XHTML family is designed with general user agent interoperability in mind. Through a new user agent and document profiling mechanism, servers, proxies, and user agents will be able to perform best effort content

www.it-ebooks.info

transformation. Ultimately, it will be possible to develop XHTML-conforming content that is usable by any XHTML-conforming user agent.

2. Definitions

This section is normative.

2.1. Terminology

The following terms are used in this specification. These terms extend the definitions in [RFC2119] in ways based upon similar definitions in ISO/IEC 9945-1:1990 [POSIX.1]:

May

With respect to implementations, the word "may" is to be interpreted as an optional feature that is not required in this specification but can be provided. With respect to Document Conformance, the word "may" means that the optional feature must not be used. The term "optional" has the same definition as "may".

Must

In this specification, the word "must" is to be interpreted as a mandatory requirement on the implementation or on Strictly Conforming XHTML Documents, depending upon the context. The term "shall" has the same definition as "must".

Optional

See "May".

Reserved

A value or behavior is unspecified, but it is not allowed to be used by Conforming Documents nor to be supported by Conforming User Agents.

Shall

See "Must".

Should

With respect to implementations, the word "should" is to be interpreted as an implementation recommendation, but not a requirement. With respect to documents, the word "should" is to be interpreted as recommended programming practice for documents and a requirement for Strictly Conforming XHTML Documents.

Supported

Certain facilities in this specification are optional. If a facility is supported, it behaves as specified by this specification.

Unspecified

When a value or behavior is unspecified, the specification defines no portability requirements for a facility on an implementation even when faced with a document

www.it-ebooks.info

that uses the facility. A document that requires specific behavior in such an instance, rather than tolerating any behavior when using that facility, is not a Strictly Conforming XHTML Document.

2.2. General Terms

Attribute

An attribute is a parameter to an element declared in the DTD. An attribute's type and value range, including a possible default value, are defined in the DTD.

DTD

A DTD, or document type definition, is a collection of XML markup declarations that, as a collection, defines the legal structure, elements, and attributes that are available for use in a document that complies to the DTD.

Document

A document is a stream of data that, after being combined with any other streams it references, is structured such that it holds information contained within elements that are organized as defined in the associated DTD. See Document Conformance for more information.

Element

An element is a document structuring unit declared in the DTD. The element's content model is defined in the DTD, and additional semantics may be defined in the prose description of the element.

Facilities

Facilities are elements, attributes, and the semantics associated with those elements and attributes.

Implementation

See User Agent.

Parsing

Parsing is the act whereby a document is scanned, and the information contained within the document is filtered into the context of the elements in which the information is structured.

Rendering

Rendering is the act whereby the information in a document is presented. This presentation is done in the form most appropriate to the environment (e.g. aurally, visually, in print).

User Agent

A user agent is a system that processes XHTML documents in accordance with this specification. See User Agent Conformance for more information.

www.it-ebooks.info

Validation

Validation is a process whereby documents are verified against the associated DTD, ensuring that the structure, use of elements, and use of attributes are consistent with the definitions in the DTD.

Well-formed

A document is well-formed when it is structured according to the rules defined in Section 2.1 of the XML 1.0 Recommendation [XML].

3. Normative Definition of XHTML 1.0

This section is normative.

3.1. Document Conformance

This version of XHTML provides a definition of strictly conforming XHTML 1.0 documents, which are restricted to elements and attributes from the XML and XHTML 1.0 namespaces. See Section 3.1.2 for information on using XHTML with other namespaces, for instance, to include metadata expressed in RDF within XHTML documents.

3.1.1. Strictly Conforming Documents

A Strictly Conforming XHTML Document is an XML document that requires only the facilities described as mandatory in this specification. Such a document must meet all of the following criteria:

1. It must conform to the constraints expressed in one of the three DTDs found in DTDs and in Appendix B.
2. The root element of the document must be `html`.
3. The root element of the document must contain an `xmlns` declaration for the XHTML namespace [XMLNS]. The namespace for XHTML is defined to be `http://www.w3.org/1999/xhtml`. An example root element might look like:
4. `<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">`
5. There must be a DOCTYPE declaration in the document prior to the root element. The public identifier included in the DOCTYPE declaration must reference one of the three DTDs found in DTDs using the respective Formal Public Identifier. The system identifier may be changed to reflect local system conventions.
6. `<!DOCTYPE html`
7. `PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"`
8. `"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">`
9. `<!DOCTYPE html`

www.it-ebooks.info

10. PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
11. "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
12. <!DOCTYPE html
13. PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
14. "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
15. The DTD subset must not be used to override any parameter entities in the DTD.

An XML declaration is not required in all XML documents; however XHTML document authors are strongly encouraged to use XML declarations in all their documents. Such a declaration is required when the character encoding of the document is other than the default UTF-8 or UTF-16 and no encoding was determined by a higher-level protocol. Here is an example of an XHTML document. In this example, the XML declaration is included.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html
     PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head>
    <title>Virtual Library</title>
  </head>
  <body>
    <p>Moved to <a href="http://example.org/">example.org</a>.</p>
  </body>
</html>
```

3.1.2. Using XHTML with other namespaces

The XHTML namespace may be used with other XML namespaces as per [XMLNS], although such documents are not strictly conforming XHTML 1.0 documents as defined above. Work by W3C is addressing ways to specify conformance for documents involving multiple namespaces. For an example, see [XHTML+MathML].

The following example shows the way in which XHTML 1.0 could be used in conjunction with the MathML Recommendation:

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
  <head>
    <title>A Math Example</title>
  </head>
  <body>
    <p>The following is MathML markup:</p>
    <math xmlns="http://www.w3.org/1998/Math/MathML">
      <apply> <log/>
        <logbase>
          <cn> 3 </cn>
```

www.it-ebooks.info

```
        </logbase>
        <ci> x </ci>
      </apply>
    </math>
  </body>
</html>
```

The following example shows the way in which XHTML 1.0 markup could be incorporated into another XML namespace:

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- initially, the default namespace is "books" -->
<book xmlns='urn:loc.gov:books'
    xmlns:isbn='urn:ISBN:0-395-36341-6' xml:lang="en" lang="en">
  <title>Cheaper by the Dozen</title>
  <isbn:number>1568491379</isbn:number>
  <notes>
    <!-- make HTML the default namespace for a hypertext commentary -
->
    <p xmlns='http://www.w3.org/1999/xhtml'>
        This is also available <a
href="http://www.w3.org/">online</a>.
    </p>
  </notes>
</book>
```

3.2. User Agent Conformance

A conforming user agent must meet all of the following criteria:

1. In order to be consistent with the XML 1.0 Recommendation [XML], the user agent must parse and evaluate an XHTML document for well-formedness. If the user agent claims to be a validating user agent, it must also validate documents against their referenced DTDs according to [XML].
2. When the user agent claims to support facilities defined within this specification or required by this specification through normative reference, it must do so in ways consistent with the facilities' definition.
3. When a user agent processes an XHTML document as generic XML, it shall only recognize attributes of type `ID` (i.e. the `id` attribute on most XHTML elements) as fragment identifiers.
4. If a user agent encounters an element it does not recognize, it must process the element's content.
5. If a user agent encounters an attribute it does not recognize, it must ignore the entire attribute specification (i.e., the attribute and its value).
6. If a user agent encounters an attribute value it does not recognize, it must use the default attribute value.
7. If it encounters an entity reference (other than one of the entities defined in this recommendation or in the XML recommendation) for which the user agent has processed no declaration (which could happen if the declaration is in the

www.it-ebooks.info

external subset which the user agent hasn't read), the entity reference should be processed as the characters (starting with the ampersand and ending with the semi-colon) that make up the entity reference.

8. When processing content, user agents that encounter characters or character entity references that are recognized but not renderable may substitute another rendering that gives the same meaning, or must display the document in such a way that it is obvious to the user that normal rendering has not taken place.
9. White space is handled according to the following rules. The following characters are defined in [XML] white space characters:

    - SPACE ( )
    - HORIZONTAL TABULATION (	)
    - CARRIAGE RETURN (
)
    - LINE FEED (
)

    The XML processor normalizes different systems' line end codes into one single LINE FEED character, that is passed up to the application.

    The user agent must use the definition from CSS for processing whitespace characters [CSS2]. *Note that the CSS2 recommendation does not explicitly address the issue of whitespace handling in non-Latin character sets. This will be addressed in a future version of CSS, at which time this reference will be updated.*

Note that in order to produce a Canonical XHTML document, the rules above must be applied and the rules in [XMLC14N] must also be applied to the document.

4. Differences with HTML 4

This section is informative.

Due to the fact that XHTML is an XML application, certain practices that were perfectly legal in SGML-based HTML 4 [HTML4] must be changed.

4.1. Documents must be well-formed

Well-formedness is a new concept introduced by [XML]. Essentially this means that all elements must either have closing tags or be written in a special form (as described below), and that all the elements must nest properly.

Although overlapping is illegal in SGML, it is widely tolerated in existing browsers.

CORRECT: nested elements.

```
<p>here is an emphasized <em>paragraph</em>.</p>
```

www.it-ebooks.info

INCORRECT: overlapping elements

```
<p>here is an emphasized <em>paragraph.</p></em>
```

4.2. Element and attribute names must be in lower case

XHTML documents must use lower case for all HTML element and attribute names. This difference is necessary because XML is case-sensitive e.g. and <LI> are different tags.

4.3. For non-empty elements, end tags are required

In SGML-based HTML 4 certain elements were permitted to omit the end tag; with the elements that followed implying closure. XML does not allow end tags to be omitted. All elements other than those declared in the DTD as `EMPTY` must have an end tag. Elements that are declared in the DTD as `EMPTY` can have an end tag *or* can use empty element shorthand (see Empty Elements).

CORRECT: terminated elements

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

INCORRECT: unterminated elements

```
<p>here is a paragraph.<p>here is another paragraph.
```

4.4. Attribute values must always be quoted

All attribute values must be quoted, even those which appear to be numeric.

CORRECT: quoted attribute values

```
<td rowspan="3">
```

INCORRECT: unquoted attribute values

```
<td rowspan=3>
```

4.5. Attribute Minimization

XML does not support attribute minimization. Attribute-value pairs must be written in full. Attribute names such as `compact` and `checked` cannot occur in elements without their value being specified.

CORRECT: unmimimized attributes

```
<dl compact="compact">
```

INCORRECT: mimimized attributes

```
<dl compact>
```

www.it-ebooks.info

4.6. Empty Elements

Empty elements must either have an end tag or the start tag must end with />. For instance, <br/> or <hr></hr>. See HTML Compatibility Guidelines for information on ways to ensure this is backward compatible with HTML 4 user agents.

CORRECT: terminated empty elements

```
<br/><hr/>
```

INCORRECT: unterminated empty elements

```
<br><hr>
```

4.7. White Space handling in attribute values

When user agents process attributes, they do so according to Section 3.3.3 of [XML]:

- ✦ Strip leading and trailing white space.
- ✦ Map sequences of one or more white space characters (including line breaks) to a single inter-word space.

4.8. Script and Style elements

In XHTML, the script and style elements are declared as having #PCDATA content. As a result, < and & will be treated as the start of markup, and entities such as < and & will be recognized as entity references by the XML processor to < and & respectively. Wrapping the content of the script or style element within a CDATA marked section avoids the expansion of these entities.

```
<script type="text/javascript">
<![CDATA[
... unescaped script content ...
]]>
</script>
```

CDATA sections are recognized by the XML processor and appear as nodes in the Document Object Model, see Section 1.3 of the DOM Level 1 Recommendation [DOM].

An alternative is to use external script and style documents.

4.9. SGML exclusions

SGML gives the writer of a DTD the ability to exclude specific elements from being contained within an element. Such prohibitions (called "exclusions") are not possible in XML.

For example, the HTML 4 Strict DTD forbids the nesting of an 'a' element within another 'a' element to any descendant depth. It is not possible to spell out such prohibitions in XML. Even though these prohibitions cannot be defined in the DTD,

www.it-ebooks.info

certain elements should not be nested. A summary of such elements and the elements that should not be nested in them is found in the normative Element Prohibitions.

4.10. The elements with 'id' and 'name' attributes

HTML 4 defined the `name` attribute for the elements `a`, `applet`, `form`, `frame`, `iframe`, `img`, and `map`. HTML 4 also introduced the `id` attribute. Both of these attributes are designed to be used as fragment identifiers.

In XML, fragment identifiers are of type `ID`, and there can only be a single attribute of type `ID` per element. Therefore, in XHTML 1.0 the `id` attribute is defined to be of type `ID`. In order to ensure that XHTML 1.0 documents are well-structured XML documents, XHTML 1.0 documents MUST use the `id` attribute when defining fragment identifiers on the elements listed above. See the HTML Compatibility Guidelines for information on ensuring such anchors are backward compatible when serving XHTML documents as media type `text/html`.

Note that in XHTML 1.0, the `name` attribute of these elements is formally deprecated, and will be removed in a subsequent version of XHTML.

4.11. Attributes with pre-defined value sets

HTML 4 and XHTML both have some attributes that have pre-defined and limited sets of values (e.g. the `type` attribute of the `input` element). In SGML and XML, these are called *enumerated attributes*. Under HTML 4, the interpretation of these values was *case-insensitive*, so a value of `TEXT` was equivalent to a value of `text`. Under XML, the interpretation of these values is *case-sensitive*, and in XHTML 1 all of these values are defined in lower-case.

4.12. Entity references as hex values

SGML and XML both permit references to characters by using hexadecimal values. In SGML these references could be made using either &#Xnn; or &#xnn;. In XML documents, you must use the lower-case version (i.e. &#xnn;)

5. Compatibility Issues

This section is normative.

Although there is no requirement for XHTML 1.0 documents to be compatible with existing user agents, in practice this is easy to accomplish. Guidelines for creating compatible documents can be found in Appendix C.

5.1. Internet Media Type

XHTML Documents which follow the guidelines set forth in Appendix C, "HTML Compatibility Guidelines" may be labeled with the Internet Media Type "text/html" [RFC2854], as they are compatible with most HTML browsers. Those documents, and any other document conforming to this specification, may also be labeled with

www.it-ebooks.info

the Internet Media Type "application/xhtml+xml" as defined in [RFC3236]. For further information on using media types with XHTML, see the informative note [XHTMLMIME].

A. DTDs

This appendix is normative.

These DTDs and entity sets form a normative part of this specification. The complete set of DTD files together with an XML declaration and SGML Open Catalog is included in the zip file and the gzip'd tar file for this specification. Users looking for local copies of the DTDs to work with should download and use those archives rather than using the specific DTDs referenced below.

A.1. Document Type Definitions

These DTDs approximate the HTML 4 DTDs. The W3C recommends that you use the authoritative versions of these DTDs at their defined SYSTEM identifiers when validating content. If you need to use these DTDs locally you should download one of the archives of this version. For completeness, the normative versions of the DTDs are included here:

A.1.1. XHTML-1.0-Strict

The file `DTD/xhtml1-strict.dtd` is a normative part of this specification. The annotated contents of this file are available in this separate section for completeness.

A.1.2. XHTML-1.0-Transitional

The file `DTD/xhtml1-transitional.dtd` is a normative part of this specification. The annotated contents of this file are available in this separate section for completeness.

A.1.3. XHTML-1.0-Frameset

The file `DTD/xhtml1-frameset.dtd` is a normative part of this specification. The annotated contents of this file are available in this separate section for completeness.

A.2. Entity Sets

The XHTML entity sets are the same as for HTML 4, but have been modified to be valid XML 1.0 entity declarations. Note the entity for the Euro currency sign (`€` or `€` or `€`) is defined as part of the special characters.

A.2.1. Latin-1 characters

The file `DTD/xhtml-lat1.ent` is a normative part of this specification. The annotated contents of this file are available in this separate section for completeness.

www.it-ebooks.info

A.2.2. Special characters

The file `DTD/xhtml-special.ent` is a normative part of this specification. The annotated contents of this file are available in this separate section for completeness.

A.2.3. Symbols

The file `DTD/xhtml-symbol.ent` is a normative part of this specification. The annotated contents of this file are available in this separate section for completeness.

B. Element Prohibitions

This appendix is normative.

The following elements have prohibitions on which elements they can contain (see SGML Exclusions). This prohibition applies to all depths of nesting, i.e. it contains all the descendant elements.

`a`

must not contain other `a` elements.

`pre`

must not contain the `img`, `object`, `big`, `small`, `sub`, or `sup` elements.

`button`

must not contain the `input`, `select`, `textarea`, `label`, `button`, `form`, `fieldset`, `iframe` or `isindex` elements.

`label`

must not contain other `label` elements.

`form`

must not contain other `form` elements.

C. HTML Compatibility Guidelines

This appendix is informative.

This appendix summarizes design guidelines for authors who wish their XHTML documents to render on existing HTML user agents. *Note that this recommendation does not define how HTML conforming user agents should process HTML documents. Nor does it define the meaning of the Internet Media Type* `text/html`. *For these definitions, see [HTML4] and [RFC2854] respectively.*

C.1. Processing Instructions and the XML Declaration

Be aware that processing instructions are rendered on some user agents. Also, some user agents interpret the XML declaration to mean that the document is

www.it-ebooks.info

unrecognized XML rather than HTML, and therefore may not render the document as expected. For compatibility with these types of legacy browsers, you may want to avoid using processing instructions and XML declarations. Remember, however, that when the XML declaration is not included in a document, the document can only use the default character encodings UTF-8 or UTF-16.

C.2. Empty Elements

Include a space before the trailing / and > of empty elements, e.g. `<br />`, `<hr />` and `<img src="karen.jpg" alt="Karen" />`. Also, use the minimized tag syntax for empty elements, e.g. `<br />`, as the alternative syntax `<br></br>` allowed by XML gives uncertain results in many existing user agents.

C.3. Element Minimization and Empty Element Content

Given an empty instance of an element whose content model is not `EMPTY` (for example, an empty title or paragraph) do not use the minimized form (e.g. use `<p></p>` and not `<p />`).

C.4. Embedded Style Sheets and Scripts

Use external style sheets if your style sheet uses < or & or]]> or --. Use external scripts if your script uses < or & or]]> or --. Note that XML parsers are permitted to silently remove the contents of comments. Therefore, the historical practice of "hiding" scripts and style sheets within "comments" to make the documents backward compatible is likely to not work as expected in XML-based user agents.

C.5. Line Breaks within Attribute Values

Avoid line breaks and multiple white space characters within attribute values. These are handled inconsistently by user agents.

C.6. Isindex

Don't include more than one `isindex` element in the document `head`. The `isindex` element is deprecated in favor of the `input` element.

C.7. The lang and xml:lang Attributes

Use both the `lang` and `xml:lang` attributes when specifying the language of an element. The value of the `xml:lang` attribute takes precedence.

C.8. Fragment Identifiers

In XML, URI-references [RFC2396] that end with fragment identifiers of the form "#foo" do not refer to elements with an attribute `name="foo"`; rather, they refer to elements with an attribute defined to be of type `ID`, e.g., the `id` attribute in HTML 4. Many existing HTML clients don't support the use of `ID`-type attributes in this way, so identical values may be supplied for both of these attributes to ensure maximum forward and backward compatibility (e.g., `<a id="foo" name="foo">...</a>`).

www.it-ebooks.info

Further, since the set of legal values for attributes of type `ID` is much smaller than for those of type `CDATA`, the type of the `name` attribute has been changed to `NMTOKEN`. This attribute is constrained such that it can only have the same values as type `ID`, or as the `Name` production in XML 1.0 Section 2.3, production 5. Unfortunately, this constraint cannot be expressed in the XHTML 1.0 DTDs. Because of this change, care must be taken when converting existing HTML documents. The values of these attributes must be unique within the document, valid, and any references to these fragment identifiers (both internal and external) must be updated should the values be changed during conversion.

Note that the collection of legal values in XML 1.0 Section 2.3, production 5 is much larger than that permitted to be used in the `ID` and `NAME` types defined in HTML 4. When defining fragment identifiers to be backward-compatible, only strings matching the pattern `[A-Za-z][A-Za-z0-9:_.-]*` should be used. See Section 6.2 of [HTML4] for more information.

Finally, note that XHTML 1.0 has deprecated the `name` attribute of the `a`, `applet`, `form`, `frame`, `iframe`, `img`, and `map` elements, and it will be removed from XHTML in subsequent versions.

C.9. Character Encoding

Historically, the character encoding of an HTML document is either specified by a web server via the charset parameter of the HTTP Content-Type header, or via a `meta` element in the document itself. In an XML document, the character encoding of the document is specified on the XML declaration (e.g., `<?xml version="1.0" encoding="EUC-JP"?>`). In order to portably present documents with specific character encodings, the best approach is to ensure that the web server provides the correct headers. If this is not possible, a document that wants to set its character encoding explicitly must include both the XML declaration an encoding declaration and a `meta` http-equiv statement (e.g., `<meta http-equiv="Content-type" content="text/html; charset=EUC-JP" />`). In XHTML-conforming user agents, the value of the encoding declaration of the XML declaration takes precedence.

Note: be aware that if a document must include the character encoding declaration in a meta http-equiv statement, that document may always be interpreted by HTTP servers and/or user agents as being of the internet media type defined in that statement. If a document is to be served as multiple media types, the HTTP server must be used to set the encoding of the document.

C.10. Boolean Attributes

Some HTML user agents are unable to interpret boolean attributes when these appear in their full (non-minimized) form, as required by XML 1.0. Note this problem doesn't affect user agents compliant with HTML 4. The following attributes are involved: `compact`, `nowrap`, `ismap`, `declare`, `noshade`, `checked`, `disabled`, `readonly`, `multiple`, `selected`, `noresize`, `defer`.

www.it-ebooks.info

C.11. Document Object Model and XHTML

The Document Object Model level 1 Recommendation [DOM] defines document object model interfaces for XML and HTML 4. The HTML 4 document object model specifies that HTML element and attribute names are returned in upper-case. The XML document object model specifies that element and attribute names are returned in the case they are specified. In XHTML 1.0, elements and attributes are specified in lower-case. This apparent difference can be addressed in two ways:

1. User agents that access XHTML documents served as Internet media type `text/html` via the DOM can use the HTML DOM, and can rely upon element and attribute names being returned in upper-case from those interfaces.
2. User agents that access XHTML documents served as Internet media types `text/xml`, `application/xml`, or `application/xhtml+xml` can also use the XML DOM. Elements and attributes will be returned in lower-case. Also, some XHTML elements may or may not appear in the object tree because they are optional in the content model (e.g. the `tbody` element within `table`). This occurs because in HTML 4 some elements were permitted to be minimized such that their start and end tags are both omitted (an SGML feature). This is not possible in XML. Rather than require document authors to insert extraneous elements, XHTML has made the elements optional. User agents need to adapt to this accordingly. For further information on this topic, see [DOM2]

C.12. Using Ampersands in Attribute Values (and Elsewhere)

In both SGML and XML, the ampersand character ("&") declares the beginning of an entity reference (e.g., ® for the registered trademark symbol "®"). Unfortunately, many HTML user agents have silently ignored incorrect usage of the ampersand character in HTML documents - treating ampersands that do not look like entity references as literal ampersands. XML-based user agents will not tolerate this incorrect usage, and any document that uses an ampersand incorrectly will not be "valid", and consequently will not conform to this specification. In order to ensure that documents are compatible with historical HTML user agents and XML-based user agents, ampersands used in a document that are to be treated as literal characters must be expressed themselves as an entity reference (e.g. "`&`"). For example, when the `href` attribute of the `a` element refers to a CGI script that takes parameters, it must be expressed as `http://my.site.dom/cgi-bin/myscript.pl?class=guest&name=user` rather than as `http://my.site.dom/cgi-bin/myscript.pl?class=guest & name=user`.

C.13. Cascading Style Sheets (CSS) and XHTML

The Cascading Style Sheets level 2 Recommendation [CSS2] defines style properties which are applied to the parse tree of the HTML or XML documents. Differences in parsing will produce different visual or aural results, depending on the selectors used. The following hints will reduce this effect for documents which are served without modification as both media types:

www.it-ebooks.info

1. CSS style sheets for XHTML should use lower case element and attribute names.
2. In tables, the tbody element will be inferred by the parser of an HTML user agent, but not by the parser of an XML user agent. Therefore you should always explicitly add a tbody element if it is referred to in a CSS selector.
3. Within the XHTML namespace, user agents are expected to recognize the "id" attribute as an attribute of type ID. Therefore, style sheets should be able to continue using the shorthand "#" selector syntax even if the user agent does not read the DTD.
4. Within the XHTML namespace, user agents are expected to recognize the "class" attribute. Therefore, style sheets should be able to continue using the shorthand "." selector syntax.
5. CSS defines different conformance rules for HTML and XML documents; be aware that the HTML rules apply to XHTML documents delivered as HTML and the XML rules apply to XHTML documents delivered as XML.

C.14. Referencing Style Elements when serving as XML

In HTML 4 and XHTML, the `style` element can be used to define document-internal style rules. In XML, an XML stylesheet declaration is used to define style rules. In order to be compatible with this convention, `style` elements should have their fragment identifier set using the `id` attribute, and an XML stylesheet declaration should reference this fragment. For example:

```
<?xml-stylesheet href="http://www.w3.org/StyleSheets/TR/W3C-REC.css"
type="text/css"?>
<?xml-stylesheet href="#internalStyle" type="text/css"?>
<!DOCTYPE html
     PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
<title>An internal stylesheet example</title>
<style type="text/css" id="internalStyle">
  code {
    color: green;
    font-family: monospace;
    font-weight: bold;
  }
</style>
</head>
<body>
<p>
  This is text that uses our
  <code>internal stylesheet</code>.
</p>
</body>
</html>
```

www.it-ebooks.info

C.15. White Space Characters in HTML vs. XML

Some characters that are legal in HTML documents, are illegal in XML document. For example, in HTML, the Formfeed character (U+000C) is treated as white space, in XHTML, due to XML's definition of characters, it is illegal.

C.16. The Named Character Reference '

The named character reference `'` (the apostrophe, U+0027) was introduced in XML 1.0 but does not appear in HTML. Authors should therefore use `'` instead of `'` to work as expected in HTML 4 user agents.

D. Acknowledgements

This appendix is informative.

This specification was written with the participation of the members of the W3C HTML Working Group.

At publication of the second edition, the membership was:

Steven Pemberton, CWI/W3C (HTML Working Group Chair)

Daniel Austin, Grainger

Jonny Axelsson, Opera Software

Tantek Çelik, Microsoft

Doug Dominiak, Openwave Systems

Herman Elenbaas, Philips Electronics

Beth Epperson, Netscape/AOL

Masayasu Ishikawa, W3C (HTML Activity Lead)

Shin'ichi Matsui, Panasonic

Shane McCarron, Applied Testing and Technology

Ann Navarro, WebGeek, Inc.

Subramanian Peruvemba, Oracle

Rob Relyea, Microsoft

Sebastian Schnitzenbaumer, SAP

Peter Stark, Sony Ericsson

At publication of the first edition, the membership was:

Steven Pemberton, CWI (HTML Working Group Chair)

Murray Altheim, Sun Microsystems

Daniel Austin, AskJeeves (CNET: The Computer Network through July 1999)

www.it-ebooks.info

Frank Boumphrey, HTML Writers Guild

John Burger, Mitre

Andrew W. Donoho, IBM

Sam Dooley, IBM

Klaus Hofrichter, GMD

Philipp Hoschka, W3C

Masayasu Ishikawa, W3C

Warner ten Kate, Philips Electronics

Peter King, Phone.com

Paula Klante, JetForm

Shin'ichi Matsui, Panasonic (W3C visiting engineer through September 1999)

Shane McCarron, Applied Testing and Technology (The Open Group through August 1999)

Ann Navarro, HTML Writers Guild

Zach Nies, Quark

Dave Raggett, W3C/HP (HTML Activity Lead)

Patrick Schmitz, Microsoft

Sebastian Schnitzenbaumer, Stack Overflow

Peter Stark, Phone.com

Chris Wilson, Microsoft

Ted Wugofski, Gateway 2000

Dan Zigmond, WebTV Networks

E. References

This appendix is informative.

[CSS2]
"Cascading Style Sheets, level 2 (CSS2) Specification", B. Bos, H. W. Lie, C. Lilley, I. Jacobs, 12 May 1998.
Latest version available at: `www.w3.org/TR/REC-CSS2`

[DOM]
"Document Object Model (DOM) Level 1 Specification", Lauren Wood et al., 1 October 1998.
Latest version available at: `www.w3.org/TR/REC-DOM-Level-1`

www.it-ebooks.info

[DOM2]

"Document Object Model (DOM) Level 2 Core Specification", A. Le Hors, et al., 13 November 2000.
Latest version available at: www.w3.org/TR/DOM-Level-2-Core

[HTML]

"HTML 4.01 Specification", D. Raggett, A. Le Hors, I. Jacobs, 24 December 1999.
Latest version available at: www.w3.org/TR/html401

[POSIX.1]

"ISO/IEC 9945-1:1990 Information Technology - Portable Operating System Interface (POSIX) - Part 1: System Application Program Interface (API) [C Language]", Institute of Electrical and Electronics Engineers, Inc, 1990.

[RFC2045]

"Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", N. Freed and N. Borenstein, November 1996. Note that this RFC obsoletes RFC1521, RFC1522, and RFC1590.

[RFC2046]

"RFC2046: Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", N. Freed and N. Borenstein, November 1996.
Available at www.ietf.org/rfc/rfc2046.txt. Note that this RFC obsoletes RFC1521, RFC1522, and RFC1590.

[RFC2119]

"RFC2119: Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997.
Available at: www.ietf.org/rfc/rfc2119.txt

[RFC2376]

"RFC2376: XML Media Types", E. Whitehead, M. Murata, July 1998.
This document is obsoleted by [RFC3023].
Available at: www.ietf.org/rfc/rfc2376.txt

[RFC2396]

"RFC2396: Uniform Resource Identifiers (URI): Generic Syntax", T. Berners-Lee, R. Fielding,
L. Masinter, August 1998.
This document updates RFC1738 and RFC1808.
Available at: www.ietf.org/rfc/rfc2396.txt

[RFC2854]

"RFC2854: The text/html Media Type", D. Conolly, L. Masinter, June 2000.
Available at: www.ietf.org/rfc/rfc2854.txt

www.it-ebooks.info

[RFC3023]

"RFC3023: XML Media Types", M. Murata, S. St.Laurent, D. Kohn, January 2001.
This document obsoletes [RFC2376].
Available at: `www.ietf.org/rfc/rfc3023.txt`

[RFC3066]

"Tags for the Identification of Languages", H. Alvestrand, January 2001.
Available at: `www.ietf.org/rfc/rfc3066.txt`

[RFC3236]

"The 'application/xhtml+xml' Media Type", M. Baker, P. Stark, January 2002.
Available at: `www.ietf.org/rfc/rfc3236.txt`

[XHTML+MathML]

"XHTML plus Math 1.1 DTD", "A.2 MathML as a DTD Module", Mathematical Markup Language (MathML) Version 2.0. Available at: `www.w3.org/TR/MathML2/dtd/xhtml-math11-f.dtd`

[XHTMLMIME]

"XHTML Media Types", Masayasu Ishikawa, 1 August 2002.
Latest version available at: `www.w3.org/TR/xhtml-media-types`

[XHTMLMOD]

"Modularization of XHTML", M. Altheim et al., 10 April 2001.
Latest version available at: `http://www.w3.org/TR/xhtml-modularization`

[XML]

"Extensible Markup Language (XML) 1.0 Specification (Second Edition)", T. Bray, J. Paoli, C. M. Sperberg-McQueen, E. Maler, 6 October 2000.
Latest version available at: `www.w3.org/TR/REC-xml`

[XMLNS]

"Namespaces in XML", T. Bray, D. Hollander, A. Layman, 14 January 1999.
XML namespaces provide a simple method for qualifying names used in XML documents by associating them with namespaces identified by URI.
Latest version available at: `www.w3.org/TR/REC-xml-names`

[XMLC14N]

"Canonical XML Version 1.0", J. Boyer, 15 March 2001.
This document describes a method for generating a physical representation, the canonical form, of an XML document.
Latest version available at: `www.w3.org/TR/xml-c14n`

✦ ✦ ✦

www.it-ebooks.info

www.it-ebooks.info

APPENDIX C

XML 1.0, Third Edition Specification

XML, the eXtensible Markup Language, is an enhancement of HTML's capabilities. XML allows many new abilities.



See Chapter 29 for information on XML. Also see Appendix A for the HTML specification.

This appendix contains the official World Wide Web Consortium specification for XML 1.0, Third Edition.

Extensible Markup Language (XML) 1.0 (Third Edition)

W3C Recommendation 04 February 2004

This version: www.w3.org/TR/2004/REC-xml-20040204

Latest version: www.w3.org/TR/REC-xml

Previous version: www.w3.org/TR/2003/PER-xml-20031030

Editors:

Tim Bray, Textuality and Netscape <tbray@textuality.com>

Jean Paoli, Microsoft <jeanpa@microsoft.com>

C. M. Sperberg-McQueen, W3C <cmsmcq@w3.org>

www.it-ebooks.info

Eve Maler, Sun Microsystems, Inc. <eve.maler@east.sun.com> - Second Edition

François Yergeau <francois@yergeau.com> - Third Edition

Please refer to the **errata** for this document, which may include some normative corrections.

This document is also available in these non-normative formats: XML and XHTML with color-coded revision indicators.

See also **translations**.

Copyright © 2004 W3C® (MIT, ERCIM, Keio), All Rights Reserved. W3C liability, trademark, document use and software licensing rules apply.

Abstract

The Extensible Markup Language (XML) is a subset of SGML that is completely described in this document. Its goal is to enable generic SGML to be served, received, and processed on the Web in the way that is now possible with HTML. XML has been designed for ease of implementation and for interoperability with both SGML and HTML.

Status of this Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at www.w3.org/TR/.

This document is a Recommendation of the W3C. It has been reviewed by W3C Members and other interested parties, and has been endorsed by the Director as a W3C Recommendation. It is a stable document and may be used as reference material or cited as a normative reference from another document. W3C's role in making the Recommendation is to draw attention to the specification and to promote its widespread deployment. This enhances the functionality and interoperability of the Web.

This document specifies a syntax created by subsetting an existing, widely used international text processing standard (Standard Generalized Markup Language, ISO 8879:1986(E) as amended and corrected) for use on the World Wide Web. It is a product of the XML Core Working Group as part of the XML Activity. The English version of this specification is the only normative version. However, for translations of this document, see www.w3.org/2003/03/Translations/byTechnology?technology=REC-xml.

This third edition is *not* a new version of XML. As a convenience to readers, it incorporates the changes dictated by the accumulated errata (available at www.w3.org/XML/xml-V10-2e-errata) to the Second Edition of XML 1.0, dated

www.it-ebooks.info

6 October 2000. In addition, markup has been introduced on a significant portion of the prescriptions of the specification, clarifying when prescriptive keywords such as MUST, SHOULD and MAY are used in the formal sense defined in [IETF RFC 2119]. For the convenience of readers, an XHTML version with color-coded revision indicators is also provided; this version highlights each change due to an erratum published in the errata list, together with a link to the particular erratum in that list. Most of the errata in the list provide a rationale for the change.

An implementation report is available at `www.w3.org/XML/2003/09/xml10-3e-implementation.html`.

Documentation of intellectual property possibly relevant to this recommendation may be found at the Working Group's public IPR disclosure page.

Please report errors in this document to xml-editor@w3.org; archives are available. The errata list for this third edition is available at `www.w3.org/XML/xml-V10-3e-errata`.

A Test Suite is maintained to help assessing conformance to this specification.

Table of Contents

1. INTRODUCTION 731
1.1. ORIGIN AND GOALS 732
1.1. TERMINOLOGY 732
2. DOCUMENTS 734
2.1. WELL-FORMED XML DOCUMENTS 734
2.2. CHARACTERS 734
2.3. COMMON SYNTACTIC CONSTRUCTS 735
2.4. CHARACTER DATA AND MARKUP 737
2.5. COMMENTS 738
2.6. PROCESSING INSTRUCTIONS 738
2.7. CDATA SECTIONS 739
2.8. PROLOG AND DOCUMENT TYPE DECLARATION 739
2.9. STANDALONE DOCUMENT DECLARATION 742
2.10. WHITE SPACE HANDLING 744
2.11. END-OF-LINE HANDLING 744
2.12. LANGUAGE IDENTIFICATION 744
3. LOGICAL STRUCTURES 746
3.1. START-TAGS, END-TAGS, AND EMPTY-ELEMENT TAGS 747

www.it-ebooks.info

3.2. ELEMENT TYPE DECLARATIONS 748
3.2.1. Element Content 749
3.2.2. Mixed Content 750
3.3. ATTRIBUTE-LIST DECLARATIONS 751
3.3.1. Attribute Types 751
3.3.2. Attribute Defaults 754
3.3.3. Attribute-Value Normalization 755
3.4. CONDITIONAL SECTIONS 756
4. PHYSICAL STRUCTURES 757
4.1. CHARACTER AND ENTITY REFERENCES 758
4.2. ENTITY DECLARATIONS 760
4.2.1. Internal Entities 760
4.2.2. External Entities 761
4.3. PARSED ENTITIES 762
4.3.1. The Text Declaration 762
4.3.2. Well-Formed Parsed Entities 763
4.3.3. Character Encoding in Entities 763
4.4. XML PROCESSOR TREATMENT OF ENTITIES AND REFERENCES 764
4.4.1. Not Recognized 765
4.4.2. Included 766
4.4.3. Included If Validating 766
4.4.4. Forbidden 766
4.4.5. Included in Literal 766
4.4.6. Notify 767
4.4.7. Bypassed 767
4.4.8. Included as PE 767
4.4.9. Error 767
4.5. CONSTRUCTION OF ENTITY REPLACEMENT TEXT 767
4.6. PREDEFINED ENTITIES 768
4.7. NOTATION DECLARATIONS 768
4.8. DOCUMENT ENTITY 769
5. CONFORMANCE 769
5.1. VALIDATING AND NON-VALIDATING PROCESSORS 769

www.it-ebooks.info

5.2. USING XML PROCESSORS 770
6. NOTATION 770

Appendices

A. REFERENCES 772
A.1. NORMATIVE REFERENCES 772
A.2. OTHER REFERENCES 774
B. CHARACTER CLASSES 775
C. XML AND SGML (NON-NORMATIVE) 779
D. EXPANSION OF ENTITY AND CHARACTER REFERENCES (NON-NORMATIVE) 779
E. DETERMINISTIC CONTENT MODELS (NON-NORMATIVE) 780
F. AUTODETECTION OF CHARACTER ENCODINGS (NON-NORMATIVE) 781
F.1. DETECTION WITHOUT EXTERNAL ENCODING INFORMATION 781
F.2. PRIORITIES IN THE PRESENCE OF EXTERNAL ENCODING INFORMATION 783
G. W3C XML WORKING GROUP (NON-NORMATIVE) 783
H. W3C XML CORE WORKING GROUP (NON-NORMATIVE) 784
I. PRODUCTION NOTES (NON-NORMATIVE) 785

1 Introduction

Extensible Markup Language, abbreviated XML, describes a class of data objects called XML documents and partially describes the behavior of computer programs which process them. XML is an application profile or restricted form of SGML, the Standard Generalized Markup Language [ISO 8879]. By construction, XML documents are conforming SGML documents.

XML documents are made up of storage units called entities, which contain either parsed or unparsed data. Parsed data is made up of characters, some of which form character data, and some of which form markup. Markup encodes a description of the document's storage layout and logical structure. XML provides a mechanism to impose constraints on the storage layout and logical structure.

[Definition: A software module called an **XML processor** is used to read XML documents and provide access to their content and structure.] [Definition: It is assumed that an XML processor is doing its work on behalf of another module, called the **application**.] This specification describes the required behavior of an XML processor in terms of how it must read XML data and the information it must provide to the application.

www.it-ebooks.info

1.1 Origin and Goals

XML was developed by an XML Working Group (originally known as the SGML Editorial Review Board) formed under the auspices of the World Wide Web Consortium (W3C) in 1996. It was chaired by Jon Bosak of Sun Microsystems with the active participation of an XML Special Interest Group (previously known as the SGML Working Group) also organized by the W3C. The membership of the XML Working Group is given in an appendix. Dan Connolly served as the Working Group's contact with the W3C.

The design goals for XML are:

1. XML shall be straightforwardly usable over the Internet.
2. XML shall support a wide variety of applications.
3. XML shall be compatible with SGML.
4. It shall be easy to write programs which process XML documents.
5. The number of optional features in XML is to be kept to the absolute minimum, ideally zero.
6. XML documents should be human-legible and reasonably clear.
7. The XML design should be prepared quickly.
8. The design of XML shall be formal and concise.
9. XML documents shall be easy to create.
10. Terseness in XML markup is of minimal importance.

This specification, together with associated standards (Unicode [Unicode] and ISO/IEC 10646 [ISO/IEC 10646] for characters, Internet RFC 3066 [IETF RFC 3066] for language identification tags, ISO 639 [ISO 639] for language name codes, and ISO 3166 [ISO 3166] for country name codes), provides all the information necessary to understand XML Version 1.0 and construct computer programs to process it.

This version of the XML specification may be distributed freely, as long as all text and legal notices remain intact.

1.2 Terminology

The terminology used to describe XML documents is defined in the body of this specification. The key words MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL, when EMPHASIZED, are to be interpreted as described in [IETF RFC 2119]. In addition, the terms defined in the following list are used in building those definitions and in describing the actions of an XML processor:

error

[Definition: A violation of the rules of this specification; results are undefined. Unless otherwise specified, failure to observe a prescription of this specification indicated by one of the keywords MUST, REQUIRED, MUST NOT, SHALL and SHALL NOT is an error. Conforming software MAY detect and report an error and MAY recover from it.]

www.it-ebooks.info

fatal error

[Definition: An error which a conforming XML processor MUST detect and report to the application. After encountering a fatal error, the processor MAY continue processing the data to search for further errors and MAY report such errors to the application. In order to support correction of errors, the processor MAY make unprocessed data from the document (with intermingled character data and markup) available to the application. Once a fatal error is detected, however, the processor MUST NOT continue normal processing (i.e., it MUST NOT continue to pass character data and information about the document's logical structure to the application in the normal way).]

at user option

[Definition: Conforming software MAY or MUST (depending on the modal verb in the sentence) behave as described; if it does, it MUST provide users a means to enable or disable the behavior described.]

validity constraint

[Definition: A rule which applies to all valid XML documents. Violations of validity constraints are errors; they MUST, at user option, be reported by validating XML processors.]

well-formedness constraint

[Definition: A rule which applies to all well-formed XML documents. Violations of well-formedness constraints are fatal errors.]

match

[Definition: (Of strings or names:) Two strings or names being compared MUST be identical. Characters with multiple possible representations in ISO/IEC 10646 (e.g. characters with both precomposed and base+diacritic forms) match only if they have the same representation in both strings. No case folding is performed. (Of strings and rules in the grammar:) A string matches a grammatical production if it belongs to the language generated by that production. (Of content and content models:) An element matches its declaration when it conforms in the fashion described in the constraint **[VC: Element Valid]**.]

for compatibility

[Definition: Marks a sentence describing a feature of XML included solely to ensure that XML remains compatible with SGML.]

for interoperability

[Definition: Marks a sentence describing a non-binding recommendation included to increase the chances that XML documents can be processed by the existing installed base of SGML processors which predate the WebSGML Adaptations Annex to ISO 8879.]

www.it-ebooks.info

2 Documents

[Definition: A data object is an **XML document** if it is well-formed, as defined in this specification. A well-formed XML document MAY in addition be valid if it meets certain further constraints.]

Each XML document has both a logical and a physical structure. Physically, the document is composed of units called entities. An entity MAY refer to other entities to cause their inclusion in the document. A document begins in a "root" or document entity. Logically, the document is composed of declarations, elements, comments, character references, and processing instructions, all of which are indicated in the document by explicit markup. The logical and physical structures MUST nest properly, as described in **4.3.2 Well-Formed Parsed Entities**.

2.1 Well-Formed XML Documents

[Definition: A textual object is a **well-formed** XML document if:]

Taken as a whole, it matches the production labeled document.

It meets all the well-formedness constraints given in this specification.

Each of the parsed entities which is referenced directly or indirectly within the document is well-formed.

Document

[1] `document ::= prolog element Misc*`

Matching the document production implies that:

1. It contains one or more elements.
2. [Definition: There is exactly one element, called the **root**, or document element, no part of which appears in the content of any other element.] For all other elements, if the start-tag is in the content of another element, the end-tag is in the content of the same element. More simply stated, the elements, delimited by start- and end-tags, nest properly within each other.

[Definition: As a consequence of this, for each non-root element `C` in the document, there is one other element `P` in the document such that `C` is in the content of `P`, but is not in the content of any other element that is in the content of `P`. `P` is referred to as the **parent** of `C`, and `C` as a **child** of `P`.]

2.2 Characters

[Definition: A parsed entity contains **text**, a sequence of characters, which may represent markup or character data.] [Definition: A **character** is an atomic unit of text as specified by ISO/IEC 10646:2000 [ISO/IEC 10646]. Legal characters are tab, carriage return, line feed, and the legal characters of Unicode and ISO/IEC 10646. The versions of these standards cited in **A.1 Normative References** were current at the

www.it-ebooks.info

time this document was prepared. New characters may be added to these standards by amendments or new editions. Consequently, XML processors MUST accept any character in the range specified for Char.]

Character Range

```
[2]    Char    ::=    #x9 | #xA | #xD |
                      [#x20-#xD7FF] |
                      [#xE000-#xFFFD] |
                      [#x10000-#x10FFFF]      /* any Unicode
                                                character, excluding
                                                the surrogate blocks,
                                                FFFE, and FFFF. */
```

The mechanism for encoding character code points into bit patterns MAY vary from entity to entity. All XML processors MUST accept the UTF-8 and UTF-16 encodings of Unicode 3.1 [Unicode3]; the mechanisms for signaling which of the two is in use, or for bringing other encodings into play, are discussed later, in **4.3.3 Character Encoding in Entities**.

Note:

Document authors are encouraged to avoid "compatibility characters", as defined in section 6.8 of [Unicode] (see also D21 in section 3.6 of [Unicode3]). The characters defined in the following ranges are also discouraged. They are either control characters or permanently undefined Unicode characters:

```
[#x7F-#x84], [#x86-#x9F], [#xFDD0-#xFDDF],
[#1FFFE-#x1FFFF], [#2FFFE-#x2FFFF], [#3FFFE-#x3FFFF],
[#4FFFE-#x4FFFF], [#5FFFE-#x5FFFF], [#6FFFE-#x6FFFF],
[#7FFFE-#x7FFFF], [#8FFFE-#x8FFFF], [#9FFFE-#x9FFFF],
[#AFFFE-#xAFFFF], [#BFFFE-#xBFFFF], [#CFFFE-#xCFFFF],
[#DFFFE-#xDFFFF], [#EFFFE-#xEFFFF], [#FFFFE-#xFFFFF],
[#10FFFE-#x10FFFF].
```

2.3 Common Syntactic Constructs

This section defines some symbols used widely in the grammar.

S (white space) consists of one or more space (#x20) characters, carriage returns, line feeds, or tabs.

White Space

```
[3]    S    ::=    (#x20 | #x9 | #xD | #xA)+
```

Note:

The presence of #xD in the above production is maintained purely for backward compatibility with the First Edition. As explained in **2.11 End-of-Line Handling**, all #xD characters literally present in an XML document are either removed or replaced by #xA characters before any other processing is done. The only way to get a #xD character to match this production is to use a character reference in an entity value literal.

www.it-ebooks.info

Characters are classified for convenience as letters, digits, or other characters. A letter consists of an alphabetic or syllabic base character or an ideographic character. Full definitions of the specific characters in each class are given in B Character Classes.

[Definition: A Name is a token beginning with a letter or one of a few punctuation characters, and continuing with letters, digits, hyphens, underscores, colons, or full stops, together known as name characters.] Names beginning with the string "xml", or with any string which would match ((‘X’|‘x’) (‘M’|‘m’) (‘L’|‘l’)), are reserved for standardization in this or future versions of this specification.

Note:

The Namespaces in XML Recommendation [XML Names] assigns a meaning to names containing colon characters. Therefore, authors should not use the colon in XML names except for namespace purposes, but XML processors must accept the colon as a name character.

An Nmtoken (name token) is any mixture of name characters.

Names and Tokens

```
[4]   NameChar    ::=   Letter | Digit | '.'
                        | '-' | '_' | ':' |
                        CombiningChar |
                        Extender
[5]   Name        ::=   (Letter | '_' | ':')
                        (NameChar)*
[6]   Names       ::=   Name (#x20 Name)*
[7]   Nmtoken     ::=   (NameChar)+
[8]   Nmtokens    ::=   Nmtoken (#x20
                        Nmtoken)*
```

Note:

The Names and Nmtokens productions are used to define the validity of tokenized attribute values after normalization (see **3.3.1 Attribute Types**).

Literal data is any quoted string not containing the quotation mark used as a delimiter for that string. Literals are used for specifying the content of internal entities (EntityValue), the values of attributes (AttValue), and external identifiers (SystemLiteral). Note that a SystemLiteral can be parsed without scanning for markup.

Literals

```
[9]   EntityValue    ::=   '"' ([^%&"] |
                           PEReference |
                           Reference)* '"'

                     |     "'" ([^%&'] |
                           PEReference |
                           Reference)* "'"
```

www.it-ebooks.info

[10]	AttValue	::=	'"' ([^<&"] | Reference)* '"' | "'" ([^<&'] | Reference)* "'"
[11]	SystemLiteral	::=	('"' [^"]* '"') | ("'" [^']* "'")
[12]	PubidLiteral	::=	'"' PubidChar* '"' | "'" (PubidChar - "'")* "'"
[13]	PubidChar	::=	#x20 | #xD | #xA | [a-zA-Z0-9] | [-'()+,./:=?;!*#@$_%]

Note:

Although the EntityValue production allows the definition of a general entity consisting of a single explicit < in the literal (e.g.,<!ENTITY mylt "<">), it is strongly advised to avoid this practice since any reference to that entity will cause a well-formedness error.

2.4 Character Data and Markup

Text consists of intermingled character data and markup. [Definition: **Markup** takes the form of start-tags, end-tags, empty-element tags, entity references, character references, comments, CDATA section delimiters, document type declarations, processing instructions, XML declarations, text declarations, and any white space that is at the top level of the document entity (that is, outside the document element and not inside any other markup).]

[Definition: All text that is not markup constitutes the **character data** of the document.]

The ampersand character (&) and the left angle bracket (<) MUST NOT appear in their literal form, except when used as markup delimiters, or within a comment, a processing instruction, or a CDATA section. If they are needed elsewhere, they MUST be escaped using either numeric character references or the strings "&" and "<" respectively. The right angle bracket (>) MAY be represented using the string ">", and MUST, for compatibility, be escaped using either ">" or a character reference when it appears in the string "]]>" in content, when that string is not marking the end of a CDATA section.

In the content of elements, character data is any string of characters which does not contain the start-delimiter of any markup and does not include the CDATA-section-close delimiter, "]]>". In a CDATA section, character data is any string of characters not including the CDATA-section-close delimiter, "]]>".

www.it-ebooks.info

To allow attribute values to contain both single and double quotes, the apostrophe or single-quote character (') MAY be represented as "'", and the double-quote character (") as "'"".

Character Data

[14] CharData ::= [^<&]* - ([^<&]* ']]>' [^<&]*)

2.5 Comments

[Definition: **Comments** MAY appear anywhere in a document outside other markup; in addition, they MAY appear within the document type declaration at places allowed by the grammar. They are not part of the document's character data; an XML processor MAY, but need not, make it possible for an application to retrieve the text of comments. For compatibility, the string "--" (double-hyphen) MUST NOT occur within comments.] Parameter entity references MUST NOT be recognized within comments.

Comments

[15] Comment ::= '<!--' ((Char - '-') | ('-' (Char - '-')))* '->'

An example of a comment:

```
<!-- declarations for <head> & <body> -->
```

Note that the grammar does not allow a comment ending in --->. The following example is *not* well-formed.

```
<!-- B+, B, or B--->
```

2.6 Processing Instructions

[Definition: **Processing instructions** (PIs) allow documents to contain instructions for applications.]

Processing Instructions

[16] PI ::= '<?' PITarget (S (Char* - (Char* '?>' Char*)))? '?>'

[17] PITarget ::= Name - (('X' | 'x') ('M' | 'm') ('L' | 'l'))

www.it-ebooks.info

PIs are not part of the document's character data, but MUST be passed through to the application. The PI begins with a target (PITarget) used to identify the application to which the instruction is directed. The target names "XML", "xml", and so on are reserved for standardization in this or future versions of this specification. The XML Notation mechanism MAY be used for formal declaration of PI targets. Parameter entity references MUST NOT be recognized within processing instructions.

2.7 CDATA Sections

[Definition: CDATA sections MAY occur anywhere character data may occur; they are used to escape blocks of text containing characters which would otherwise be recognized as markup. CDATA sections begin with the string "<![CDATA[" and end with the string "]]>":]

CDATA Sections

[18]	CDSect	::=	CDStart CData CDEnd
[19]	CDStart	::=	'<![CDATA['
[20]	CData	::=	(Char* - (Char* ']]>' Char*))
[21]	CDEnd	::=	']]>'

Within a CDATA section, only the CDEnd string is recognized as markup, so that left angle brackets and ampersands may occur in their literal form; they need not (and cannot) be escaped using "<" and "&". CDATA sections cannot nest.

An example of a CDATA section, in which "<greeting>"and "</greeting>" are recognized as character data, not markup:

```
<![CDATA[<greeting>Hello, world!</greeting>]]>
```

2.8 Prolog and Document Type Declaration

[Definition: XML documents SHOULD begin with an **XML declaration** which specifies the version of XML being used.] For example, the following is a complete XML document, well-formed but not valid:

```
<?xml version="1.0"?>
<greeting>Hello, world!</greeting>
```

and so is this:

```
<greeting>Hello, world!</greeting>
```

www.it-ebooks.info

The function of the markup in an XML document is to describe its storage and logical structure and to associate attribute name-value pairs with its logical structures. XML provides a mechanism, the document type declaration, to define constraints on the logical structure and to support the use of predefined storage units. [Definition: An XML document is **valid** if it has an associated document type declaration and if the document complies with the constraints expressed in it.]

The document type declaration MUST appear before the first element in the document.

Prolog

[22]	prolog	::=	XMLDecl? Misc* (doctypedecl Misc*)?
[23]	XMLDecl	::=	'<?xml' VersionInfo EncodingDecl? SDDecl? S? '?>'
[24]	VersionInfo	::=	S 'version' Eq ("'" VersionNum "'" | '"' VersionNum '"')
[25]	Eq	::=	S? '=' S?
[26]	VersionNum	::=	'1.0'
[27]	Misc	::=	Comment | PI | S

[Definition: The XML **document type declaration** contains or points to markup declarations that provide a grammar for a class of documents. This grammar is known as a document type definition, or **DTD**. The document type declaration can point to an external subset (a special kind of external entity) containing markup declarations, or can contain the markup declarations directly in an internal subset, or can do both. The DTD for a document consists of both subsets taken together.]

[Definition: A **markup declaration** is an element type declaration, an attribute-list declaration, an entity declaration, or a notation declaration.] These declarations MAY be contained in whole or in part within parameter entities, as described in the well-formedness and validity constraints below. For further information, see **4 Physical Structures**.

Document Type Definition

[28]	doctypedecl	::=	'<!DOCTYPE' S Name (S ExternalID)? S? ('[' intSubset ']' S?)? '>'	[VC: Root Element Type] [WFC: External Subset]

www.it-ebooks.info

[28a]	DeclSep	::=	PEReference | S	[WFC: PE Between Declarations]
[28b]	intSubset	::=	(markupdecl | DeclSep)*	
[29]	markupdecl	::=	elementdecl | AttlistDecl | EntityDecl | NotationDecl | PI | Comment	[VC: Proper Declaration/PE Nesting] [WFC: PEs in Internal Subset]

Note that it is possible to construct a well-formed document containing a doctypedecl that neither points to an external subset nor contains an internal subset.

The markup declarations MAY be made up in whole or in part of the replacement text of parameter entities. The productions later in this specification for individual nonterminals (elementdecl, AttlistDecl, and so on) describe the declarations *after* all the parameter entities have been included.

Parameter entity references are recognized anywhere in the DTD (internal and external subsets and external parameter entities), except in literals, processing instructions, comments, and the contents of ignored conditional sections (see **3.4 Conditional Sections**). They are also recognized in entity value literals. The use of parameter entities in the internal subset is restricted as described below.

Validity constraint: Root Element Type
The Name in the document type declaration MUST match the element type of the root element.

Validity constraint: Proper Declaration/PE Nesting
Parameter-entity replacement text MUST be properly nested with markup declarations. That is to say, if either the first character or the last character of a markup declaration (markupdecl above) is contained in the replacement text for a parameter-entity reference, both MUST be contained in the same replacement text.

Well-formedness constraint: PEs in Internal Subset
In the internal DTD subset, parameter-entity references MUST NOT occur within markup declarations; they MAY occur where markup declarations can occur. (This does not apply to references that occur in external parameter entities or to the external subset.)

Well-formedness constraint: External Subset
The external subset, if any, MUST match the production for extSubset.

Well-formedness constraint: PE Between Declarations
The replacement text of a parameter entity reference in a DeclSep MUST match the production extSubsetDecl.

www.it-ebooks.info

Like the internal subset, the external subset and any external parameter entities referenced in a DeclSep MUST consist of a series of complete markup declarations of the types allowed by the non-terminal symbol markupdecl, interspersed with white space or parameter-entity references. However, portions of the contents of the external subset or of these external parameter entities MAY conditionally be ignored by using the conditional section construct; this is not allowed in the internal subset but is allowed in external parameter entities referenced in the internal subset.

External Subset

[30]	extSubset	::=	TextDecl? extSubsetDecl
[31]	extSubsetDecl	::=	(markupdecl | conditionalSect | DeclSep)*

The external subset and external parameter entities also differ from the internal subset in that in them, parameter-entity references are permitted *within* markup declarations, not only *between* markup declarations.

An example of an XML document with a document type declaration:

```
<?xml version="1.0"?>
<!DOCTYPE greeting SYSTEM "hello.dtd">
<greeting>Hello, world!</greeting>
```

The system identifier "`hello.dtd`" gives the address (a URI reference) of a DTD for the document.

The declarations can also be given locally, as in this example:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE greeting [
  <!ELEMENT greeting (#PCDATA)>
]>
<greeting>Hello, world!</greeting>
```

If both the external and internal subsets are used, the internal subset MUST be considered to occur before the external subset. This has the effect that entity and attribute-list declarations in the internal subset take precedence over those in the external subset.

2.9 Standalone Document Declaration

Markup declarations can affect the content of the document, as passed from an XML processor to an application; examples are attribute defaults and entity declarations. The standalone document declaration, which MAY appear as a component of the XML declaration, signals whether or not there are such declarations which appear external to the document entity or in parameter entities. [Definition: An **external**

www.it-ebooks.info

markup declaration is defined as a markup declaration occurring in the external subset or in a parameter entity (external or internal, the latter being included because non-validating processors are not required to read them).]

Standalone Document Declaration

[32]	SDDecl	::=	S 'standalone' Eq (("'" ('yes' | 'no') "'") | ('"' ('yes' | 'no') '"'))	[VC: Standalone Document Declaration]

In a standalone document declaration, the value "yes" indicates that there are no external markup declarations which affect the information passed from the XML processor to the application. The value "no" indicates that there are or may be such external markup declarations. Note that the standalone document declaration only denotes the presence of external *declarations*; the presence, in a document, of references to external *entities*, when those entities are internally declared, does not change its standalone status.

If there are no external markup declarations, the standalone document declaration has no meaning. If there are external markup declarations but there is no standalone document declaration, the value "no" is assumed.

Any XML document for which `standalone="no"` holds can be converted algorithmically to a standalone document, which may be desirable for some network delivery applications.

Validity constraint: Standalone Document Declaration

The standalone document declaration MUST have the value "no" if any external markup declarations contain declarations of:

- attributes with default values, if elements to which these attributes apply appear in the document without specifications of values for these attributes, or
- entities (other than `amp`, `lt`, `gt`, `apos`, `quot`), if references to those entities appear in the document, or
- attributes with tokenized types, where the attribute appears in the document with a value such that *normalization* will produce a different value from that which would be produced in the absence of the declaration, or
- element types with element content, if white space occurs directly within any instance of those types.

An example XML declaration with a standalone document declaration:

```
<?xml version="1.0" standalone='yes'?>
```

www.it-ebooks.info

2.10 White Space Handling

In editing XML documents, it is often convenient to use "white space" (spaces, tabs, and blank lines) to set apart the markup for greater readability. Such white space is typically not intended for inclusion in the delivered version of the document. On the other hand, "significant" white space that should be preserved in the delivered version is common, for example in poetry and source code.

An XML processor MUST always pass all characters in a document that are not markup through to the application. A validating XML processor MUST also inform the application which of these characters constitute white space appearing in element content.

A special attribute named `xml:space` MAY be attached to an element to signal an intention that in that element, white space should be preserved by applications. In valid documents, this attribute, like any other, MUST be declared if it is used. When declared, it MUST be given as an enumerated type whose values are one or both of "default" and "preserve". For example:

```
<!ATTLIST poem xml:space (default | preserve) 'preserve'>

<!ATTLIST pre xml:space (preserve) #FIXED 'preserve'>
```

The value "default" signals that applications' default white-space processing modes are acceptable for this element; the value "preserve" indicates the intent that applications preserve all the white space. This declared intent is considered to apply to all elements within the content of the element where it is specified, unless overridden with another instance of the `xml:space` attribute. This specification does not give meaning to any value of `xml:space` other than "default" and "preserve". It is an error for other values to be specified; the XML processor MAY report the error or MAY recover by ignoring the attribute specification or by reporting the (erroneous) value to the application. Applications may ignore or reject erroneous values.

The root element of any document is considered to have signaled no intentions as regards application space handling, unless it provides a value for this attribute or the attribute is declared with a default value.

2.11 End-of-Line Handling

XML parsed entities are often stored in computer files which, for editing convenience, are organized into lines. These lines are typically separated by some combination of the characters CARRIAGE RETURN (#xD) and LINE FEED (#xA).

To simplify the tasks of applications, the XML processor MUST behave as if it normalized all line breaks in external parsed entities (including the document entity) on input, before parsing, by translating both the two-character sequence #xD #xA and any #xD that is not followed by #xA to a single #xA character.

2.12 Language Identification

In document processing, it is often useful to identify the natural or formal language in which the content is written. A special attribute named `xml:lang` MAY be inserted

www.it-ebooks.info

in documents to specify the language used in the contents and attribute values of any element in an XML document. In valid documents, this attribute, like any other, MUST be declared if it is used. The values of the attribute are language identifiers as defined by [IETF RFC 3066], *Tags for the Identification of Languages*, or its successor; in addition, the empty string MAY be specified.

(Productions 33 through 38 have been removed.)

For example:

```
<p xml:lang="en">The quick brown fox jumps over the lazy dog.</p>
<p xml:lang="en-GB">What colour is it?</p>
<p xml:lang="en-US">What color is it?</p>
<sp who="Faust" desc='leise' xml:lang="de">
  <l>Habe nun, ach! Philosophie,</l>
  <l>Juristerei, und Medizin</l>
  <l>und leider auch Theologie</l>
  <l>durchaus studiert mit heißem Bemüh'n.</l>
</sp>
```

The intent declared with `xml:lang` is considered to apply to all attributes and content of the element where it is specified, unless overridden with an instance of `xml:lang` on another element within that content. In particular, the empty value of `xml:lang` is used on an element B to override a specification of `xml:lang` on an enclosing element A, without specifying another language. Within B, it is considered that there is no language information available, just as if `xml:lang` had not been specified on B or any of its ancestors.

Note:

Language information may also be provided by external transport protocols (e.g. HTTP or MIME). When available, this information may be used by XML applications, but the more local information provided by `xml:lang` should be considered to override it.

A simple declaration for `xml:lang` might take the form

```
xml:lang CDATA #IMPLIED
```

but specific default values MAY also be given, if appropriate. In a collection of French poems for English students, with glosses and notes in English, the `xml:lang` attribute might be declared this way:

```
<!ATTLIST poem   xml:lang CDATA 'fr'>
<!ATTLIST gloss  xml:lang CDATA 'en'>
<!ATTLIST note   xml:lang CDATA 'en'>
```

www.it-ebooks.info

3 Logical Structures

[Definition: Each XML document contains one or more **elements**, the boundaries of which are either delimited by start-tags and end-tags, or, for empty elements, by an empty-element tag. Each element has a type, identified by name, sometimes called its "generic identifier" (GI), and MAY have a set of attribute specifications.] Each attribute specification has a name and a value.

Element

[39] element ::= EmptyElemTag
| STag content ETag [WFC: Element Type Match] [VC: Element Valid]

This specification does not constrain the semantics, use, or (beyond syntax) names of the element types and attributes, except that names beginning with a match to `((‘X’|‘x’)(‘M’|‘m’)(‘L’|‘l’))` are reserved for standardization in this or future versions of this specification.

Well-formedness constraint: Element Type Match

The Name in an element's end-tag MUST match the element type in the start-tag.

Validity constraint: Element Valid

An element is valid if there is a declaration matching elementdecl where the Name matches the element type, and one of the following holds:

1. The declaration matches **EMPTY** and the element has no content (not even entity references, comments, PIs or white space).
2. The declaration matches children and the sequence of child elements belongs to the language generated by the regular expression in the content model, with optional white space, comments and PIs (i.e. markup matching production [27] Misc) between the start-tag and the first child element, between child elements, or between the last child element and the end-tag. Note that a CDATA section containing only white space or a reference to an entity whose replacement text is character references expanding to white space do not match the nonterminal S, and hence cannot appear in these positions; however, a reference to an internal entity with a literal value consisting of character references expanding to white space does match S, since its replacement text is the white space resulting from expansion of the character references.
3. The declaration matches Mixed and the content (after replacing any entity references with their replacement text) consists of character data, comments, PIs and child elements whose types match names in the content model.
4. The declaration matches **ANY**, and the content (after replacing any entity references with their replacement text) consists of character data and child elements whose types have been declared.

www.it-ebooks.info

3.1 Start-Tags, End-Tags, and Empty-Element Tags

[Definition: The beginning of every non-empty XML element is marked by a **start-tag**.]

Start-tag

[40]	STag	::=	'<' Name (S Attribute)* S? '>'	[WFC: Unique Att Spec]
[41]	Attribute	::=	Name Eq AttValue	[VC: Attribute Value Type]
				[WFC: No External Entity References]
				[WFC: No < in Attribute Values]

The Name in the start- and end-tags gives the element's **type**. [Definition: The Name-AttValue pairs are referred to as the **attribute specifications** of the element], [Definition: with the Name in each pair referred to as the **attribute name**] and [Definition: the content of the AttValue (the text between the ' or " delimiters) as the **attribute value**.] Note that the order of attribute specifications in a start-tag or empty-element tag is not significant.

Well-formedness constraint: Unique Att Spec

An attribute name MUST NOT appear more than once in the same start-tag or empty-element tag.

Validity constraint: Attribute Value Type

The attribute MUST have been declared; the value MUST be of the type declared for it. (For attribute types, see 3.3 Attribute-List Declarations.)

Well-formedness constraint: No External Entity References

Attribute values MUST NOT contain direct or indirect entity references to external entities.

Well-formedness constraint: No < in Attribute Values

The replacement text of any entity referred to directly or indirectly in an attribute value MUST NOT contain a <.

An example of a start-tag:

```
<termdef id="dt-dog" term="dog">
```

[Definition: The end of every element that begins with a start-tag MUST be marked by an **end-tag** containing a name that echoes the element's type as given in the start-tag:]

www.it-ebooks.info

End-tag

[42] ETag ::= '</' Name S? '>'

An example of an end-tag:

```
</termdef>
```

[Definition: The text between the start-tag and end-tag is called the element's **content**:]

Content of Elements

[43] content ::= CharData? ((element | Reference | CDSect | PI | Comment) CharData?)*

[Definition: An element with no content is said to be **empty**.] The representation of an empty element is either a start-tag immediately followed by an end-tag, or an empty-element tag. [Definition: An **empty-element tag** takes a special form:]

Tags for Empty Elements

[44] EmptyElemTag ::= '<' Name (S Attribute)* S? '/>' [WFC: Unique Att Spec]

Empty-element tags MAY be used for any element which has no content, whether or not it is declared using the keyword **EMPTY**. For interoperability, the empty-element tag SHOULD be used, and SHOULD only be used, for elements which are declared EMPTY.

Examples of empty elements:

```
<IMG align="left"
 src="http://www.w3.org/Icons/WWW/w3c_home" />
<br></br>
<br/>
```

3.2 Element Type Declarations

The element structure of an XML document MAY, for validation purposes, be constrained using element type and attribute-list declarations. An element type declaration constrains the element's content.

Element type declarations often constrain which element types can appear as children of the element. At user option, an XML processor MAY issue a warning when a declaration mentions an element type for which no declaration is provided, but this is not an error.

[Definition: An **element type declaration** takes the form:]

www.it-ebooks.info

Element Type Declaration

[45]	elementdecl	::=	'<!ELEMENT' S Name S contentspec S? '>'	[VC: Unique Element Type Declaration]
[46]	contentspec	::=	'EMPTY' | 'ANY' | Mixed | children	

where the Name gives the element type being declared.

Validity constraint: Unique Element Type Declaration

An element type MUST NOT be declared more than once.

Examples of element type declarations:

```
<!ELEMENT br EMPTY>
<!ELEMENT p (#PCDATA|emph)* >
<!ELEMENT %name.para; %content.para; >
<!ELEMENT container ANY>
```

3.2.1 Element Content

[Definition: An element type has **element content** when elements of that type MUST contain only child elements (no character data), optionally separated by white space (characters matching the nonterminal S).] [Definition: In this case, the constraint includes a **content model**, a simple grammar governing the allowed types of the child elements and the order in which they are allowed to appear.] The grammar is built on content particles (cps), which consist of names, choice lists of content particles, or sequence lists of content particles:

Element-content Models

[47]	children	::=	(choice | seq) ('?' | '*' | '+')?	
[48]	cp	::=	(Name | choice | seq) ('?' | '*' | '+')?	
[49]	choice	::=	'(' S? cp (S? '|' S? cp) + S? ')'	[VC: Proper Group/PE Nesting]
[50]	seq	::=	'(' S? cp (S? ',' S? cp) * S? ')'	[VC: Proper Group/PE Nesting]

where each Name is the type of an element which MAY appear as a child. Any content particle in a choice list MAY appear in the element content at the location where the choice list appears in the grammar; content particles occurring in a sequence list MUST each appear in the element content in the order given in the list. The optional character following a name or list governs whether the element or the content particles in the list may occur one or more (+), zero or more (*), or zero or one times (?). The absence of such an operator means that the element or content

www.it-ebooks.info

particle MUST appear exactly once. This syntax and meaning are identical to those used in the productions in this specification.

The content of an element matches a content model if and only if it is possible to trace out a path through the content model, obeying the sequence, choice, and repetition operators and matching each element in the content against an element type in the content model. For compatibility, it is an error if the content model allows an element to match more than one occurrence of an element type in the content model. For more information, see **E Deterministic Content Models**.

Validity constraint: Proper Group/PE Nesting

Parameter-entity replacement text MUST be properly nested with parenthesized groups. That is to say, if either of the opening or closing parentheses in a choice, seq, or Mixed construct is contained in the replacement text for a parameter entity, both MUST be contained in the same replacement text.

For interoperability, if a parameter-entity reference appears in a choice, seq, or Mixed construct, its replacement text SHOULD contain at least one non-blank character, and neither the first nor last non-blank character of the replacement text SHOULD be a connector (| or ,).

Examples of element-content models:

```
<!ELEMENT spec (front, body, back?)>
<!ELEMENT div1 (head, (p | list | note)*, div2*)>
<!ELEMENT dictionary-body (%div.mix; | %dict.mix;)*>
```

3.2.2 Mixed Content

[Definition: An element type has **mixed content** when elements of that type MAY contain character data, optionally interspersed with child elements.] In this case, the types of the child elements MAY be constrained, but not their order or their number of occurrences:

Mixed-content Declaration

```
[51]  Mixed  ::=  '(' S? '#PCDATA' (S? '|' S? Name)* S? ')*'
                  |'(' S? '#PCDATA' S? ')'
```

[VC: Proper Group/PE Nesting]
[VC: No Duplicate Types]

where the Names give the types of elements that may appear as children. The keyword **#PCDATA** derives historically from the term "parsed character data."

Validity constraint: No Duplicate Types

The same name MUST NOT appear more than once in a single mixed-content declaration.

www.it-ebooks.info

Examples of mixed content declarations:

```
<!ELEMENT p (#PCDATA|a|ul|b|i|em)*>
<!ELEMENT p (#PCDATA | %font; | %phrase; |
%special; | %form;)* >
<!ELEMENT b (#PCDATA)>
```

3.3 Attribute-List Declarations

Attributes are used to associate name-value pairs with elements. Attribute specifications MUST NOT appear outside of start-tags and empty-element tags; thus, the productions used to recognize them appear in **3.1 Start-Tags, End-Tags, and Empty-Element Tags**. Attribute-list declarations MAY be used:

- ✦ To define the set of attributes pertaining to a given element type.
- ✦ To establish type constraints for these attributes.
- ✦ To provide default values for attributes.

[Definition: **Attribute-list declarations** specify the name, data type, and default value (if any) of each attribute associated with a given element type:]

Attribute-list Declaration

[52]	AttlistDecl	::=	'<!ATTLIST' S Name AttDef* S? '>'
[53]	AttDef	::=	S Name S AttType S DefaultDecl

The Name in the AttlistDecl rule is the type of an element. At user option, an XML processor MAY issue a warning if attributes are declared for an element type not itself declared, but this is not an error. The Name in the AttDef rule is the name of the attribute.

When more than one AttlistDecl is provided for a given element type, the contents of all those provided are merged. When more than one definition is provided for the same attribute of a given element type, the first declaration is binding and later declarations are ignored. For interoperability, writers of DTDs MAY choose to provide at most one attribute-list declaration for a given element type, at most one attribute definition for a given attribute name in an attribute-list declaration, and at least one attribute definition in each attribute-list declaration. For interoperability, an XML processor MAY at user option issue a warning when more than one attribute-list declaration is provided for a given element type, or more than one attribute definition is provided for a given attribute, but this is not an error.

3.3.1 Attribute Types

XML attribute types are of three kinds: a string type, a set of tokenized types, and enumerated types. The string type may take any literal string as a value; the tokenized types have varying lexical and semantic constraints. The validity constraints noted in the grammar are applied after the attribute value has been normalized as described in **3.3.3 Attribute-Value Normalization**.

www.it-ebooks.info

Attribute Types

```
[54]  AttType         ::=  StringType |
                           TokenizedType |
                           EnumeratedType
[55]  StringType      ::=  'CDATA'
[56]  TokenizedType   ::=  'ID'                [VC: ID]
                                               [VC: One ID per Element Type]
                                               [VC: ID Attribute Default]
                      |'IDREF'                 [VC: IDREF]
                      |'IDREFS'                [VC: IDREF]
                      |'ENTITY'                [VC: Entity Name]
                      |'ENTITIES'              [VC: Entity Name]
                      |'NMTOKEN'               [VC: Name Token]
                      |'NMTOKENS'              [VC: Name Token]
```

Validity constraint: ID

Values of type ID MUST match the Name production. A name MUST NOT appear more than once in an XML document as a value of this type; i.e., ID values MUST uniquely identify the elements which bear them.

Validity constraint: One ID per Element Type

An element type MUST NOT have more than one ID attribute specified.

Validity constraint: ID Attribute Default

An ID attribute MUST have a declared default of #IMPLIED or #REQUIRED.

Validity constraint: IDREF

Values of type IDREF MUST match the Name production, and values of type IDREFS MUST match Names; each Name MUST match the value of an ID attribute on some element in the XML document; i.e. IDREF values MUST match the value of some ID attribute.

Validity constraint: Entity Name

Values of type ENTITY MUST match the Name production, values of type ENTITIES MUST match Names; each Name MUST match the name of an unparsed entity declared in the DTD.

www.it-ebooks.info

Validity constraint: Name Token

Values of type NMTOKEN MUST match the Nmtoken production; values of type NMTOKENS MUST match Nmtokens.

[Definition: Enumerated attributes MUST take one of a list of values provided in the declaration]. There are two kinds of enumerated types:

Enumerated Attribute Types

[57]	EnumeratedType	::=	NotationType | Enumeration	
[58]	NotationType	::=	'NOTATION' S '(' S? Name (S? '|' S? Name)* S? ')'	[VC: Notation Attributes] [VC: One Notation Per Element Type] [VC: No Notation on Empty Element] [VC: No Duplicate Tokens]
[59]	Enumeration	::=	'(' S? Nmtoken (S? '|' S? Nmtoken)* S? ')'	[VC: Enumeration] [VC: No Duplicate Tokens]

A **NOTATION** attribute identifies a notation, declared in the DTD with associated system and/or public identifiers, to be used in interpreting the element to which the attribute is attached.

Validity constraint: Notation Attributes

Values of this type MUST match one of the *notation* names included in the declaration; all notation names in the declaration MUST be declared.

Validity constraint: One Notation Per Element Type

An element type MUST NOT have more than one **NOTATION** attribute specified.

Validity constraint: No Notation on Empty Element

For compatibility, an attribute of type **NOTATION** MUST NOT be declared on an element declared **EMPTY**.

Validity constraint: No Duplicate Tokens

The notation names in a single NotationType attribute declaration, as well as the NmTokens in a single Enumeration attribute declaration, MUST all be distinct.

www.it-ebooks.info

Validity constraint: Enumeration

Values of this type MUST match one of the Nmtoken tokens in the declaration.

For interoperability, the same Nmtoken SHOULD NOT occur more than once in the enumerated attribute types of a single element type.

3.3.2 Attribute Defaults

An attribute declaration provides information on whether the attribute's presence is REQUIRED, and if not, how an XML processor is to react if a declared attribute is absent in a document.

Attribute Defaults

[60] `DefaultDecl` ::= `'#REQUIRED' | '#IMPLIED' | (('#FIXED' S)? AttValue)`

[VC: Required Attribute]

[VC: Attribute Default Value Syntactically Correct]

[WFC: No < in Attribute Values]

[VC: Fixed Attribute Default]

In an attribute declaration, **#REQUIRED** means that the attribute MUST always be provided, **#IMPLIED** that no default value is provided. [Definition: If the declaration is neither **#REQUIRED** nor **#IMPLIED**, then the AttValue value contains the declared **default** value; the **#FIXED** keyword states that the attribute MUST always have the default value. When an XML processor encounters an element without a specification for an attribute for which it has read a default value declaration, it MUST report the attribute with the declared default value to the application.]

Validity constraint: Required Attribute

If the default declaration is the keyword **#REQUIRED**, then the attribute MUST be specified for all elements of the type in the attribute-list declaration.

Validity constraint: Attribute Default Value Syntactically Correct

The declared default value MUST meet the syntactic constraints of the declared attribute type.

Note that only the syntactic constraints of the type are required here; other constraints (e.g. that the value be the name of a declared unparsed entity, for an attribute of type ENTITY) may come into play if the declared default value is actually used (an element without a specification for this attribute occurs).

www.it-ebooks.info

Validity constraint: Fixed Attribute Default

If an attribute has a default value declared with the **#FIXED** keyword, instances of that attribute MUST match the default value.

Examples of attribute-list declarations:

```
<!ATTLIST termdef
          id      ID       #REQUIRED
          name    CDATA    #IMPLIED>
<!ATTLIST list
          type    (bullets |ordered|glossary) "ordered">
<!ATTLIST form
          method  CDATA    #FIXED "POST">
```

3.3.3 Attribute-Value Normalization

Before the value of an attribute is passed to the application or checked for validity, the XML processor MUST normalize the attribute value by applying the algorithm below, or by using some other method such that the value passed to the application is the same as that produced by the algorithm.

1. All line breaks MUST have been normalized on input to #xA as described in **2.11 End-of-Line Handling**, so the rest of this algorithm operates on text normalized in this way.
2. Begin with a normalized value consisting of the empty string.
3. For each character, entity reference, or character reference in the unnormalized attribute value, beginning with the first and continuing to the last, do the following:
   - For a character reference, append the referenced character to the normalized value.
   - For an entity reference, recursively apply step 3 of this algorithm to the replacement text of the entity.
   - For a white space character (#x20, #xD, #xA, #x9), append a space character (#x20) to the normalized value.
   - For another character, append the character to the normalized value.

If the attribute type is not CDATA, then the XML processor MUST further process the normalized attribute value by discarding any leading and trailing space (#x20) characters, and by replacing sequences of space (#x20) characters by a single space (#x20) character.

Note that if the unnormalized attribute value contains a character reference to a white space character other than space (#x20), the normalized value contains the referenced character itself (#xD, #xA or #x9). This contrasts with the case where the unnormalized value contains a white space character (not a reference), which is replaced with a space character (#x20) in the normalized value and also contrasts

www.it-ebooks.info

with the case where the unnormalized value contains an entity reference whose replacement text contains a white space character; being recursively processed, the white space character is replaced with a space character (#x20) in the normalized value.

All attributes for which no declaration has been read SHOULD be treated by a non-validating processor as if declared **CDATA**.

It is an error if an attribute value contains a reference to an entity for which no declaration has been read.

Following are examples of attribute normalization. Given the following declarations:

```
<!ENTITY d “&#xD;”>
<!ENTITY a “&#xA;”>
<!ENTITY da “&#xD;&#xA;”>
```

the attribute specifications in the left column below would be normalized to the character sequences of the middle column if the attribute a is declared **NMTOKENS** and to those of the right columns if a is declared **CDATA**.

Attribute specification	*a is NMTOKENS*	*a is CDATA*
a=“xyz”	x y z	#x20 #x20 x y z
a=“&d;&d;A&a; &a;B&da;”	A #x20 B	#x20 #x20 A #x20 #x20 #x20 B #x20 #x20
a= “

A

B
” | #xD #xD A #xA #xA B #xD #xA | #xD #xD A #xA #xA B #xD #xA |

Note that the last example is invalid (but well-formed) if a is declared to be of type **NMTOKENS**.

3.4 Conditional Sections

[Definition: **Conditional sections** are portions of the document type declaration external subset or of external parameter entities which are included in, or excluded from, the logical structure of the DTD based on the keyword which governs them.]

Conditional Section

[61] conditionalSect ::= includeSect | ignoreSect

[62] includeSect ::= ‘<![’ S? ‘INCLUDE’ S? ‘[’ extSubsetDecl ‘]]>’ [VC: Proper Conditional Section/PE Nesting]

[63] ignoreSect ::= ‘<![’ S? ‘IGNORE’ S? ‘[’ ignoreSectContents* ‘]]>’ [VC: Proper Conditional Section/PE Nesting]

www.it-ebooks.info

```
[64]  ignoreSectContents      ::=   Ignore (‘<![’
                                    ignoreSectContents ‘]]>’
                                    Ignore)*
[65]  Ignore                  ::=   Char* - (Char* (‘<![’ |
                                    ‘]]>’) Char*)
```

Validity constraint: Proper Conditional Section/PE Nesting

If any of the “<![”, “[”, or “]]>” of a conditional section is contained in the replacement text for a parameter-entity reference, all of them MUST be contained in the same replacement text.

Like the internal and external DTD subsets, a conditional section may contain one or more complete declarations, comments, processing instructions, or nested conditional sections, intermingled with white space.

If the keyword of the conditional section is INCLUDE, then the contents of the conditional section MUST be considered part of the DTD. If the keyword of the conditional section is IGNORE, then the contents of the conditional section MUST be considered as not logically part of the DTD. If a conditional section with a keyword of INCLUDE occurs within a larger conditional section with a keyword of IGNORE, both the outer and the inner conditional sections MUST be ignored. The contents of an ignored conditional section MUST be parsed by ignoring all characters after the “[” following the keyword, except conditional section starts “<![” and ends “]]>”, until the matching conditional section end is found. Parameter entity references MUST NOT be recognized in this process.

If the keyword of the conditional section is a parameter-entity reference, the parameter entity MUST be replaced by its content before the processor decides whether to include or ignore the conditional section.

An example:

```
<!ENTITY % draft ‘INCLUDE’ >
<!ENTITY % final ‘IGNORE’ >

<![%draft;[
<!ELEMENT book (comments*, title, body, supplements?)>
]]>
<![%final;[
<!ELEMENT book (title, body, supplements?)>
]]>
```

4 Physical Structures

[Definition: An XML document may consist of one or many storage units. These are called **entities**; they all have **content** and are all (except for the document entity and the external DTD subset) identified by entity **name**.] Each XML document has one

www.it-ebooks.info

entity called the document entity, which serves as the starting point for the XML processor and may contain the whole document.

Entities may be either parsed or unparsed. [Definition: The contents of a **parsed entity** are referred to as its replacement text; this text is considered an integral part of the document.]

[Definition: An **unparsed entity** is a resource whose contents may or may not be text, and if text, may be other than XML. Each unparsed entity has an associated notation, identified by name. Beyond a requirement that an XML processor make the identifiers for the entity and notation available to the application, XML places no constraints on the contents of unparsed entities.]

Parsed entities are invoked by name using entity references; unparsed entities by name, given in the value of **ENTITY** or **ENTITIES** attributes.

[Definition: **General entities** are entities for use within the document content. In this specification, general entities are sometimes referred to with the unqualified term *entity* when this leads to no ambiguity.] [Definition: **Parameter entities** are parsed entities for use within the DTD.] These two types of entities use different forms of reference and are recognized in different contexts. Furthermore, they occupy different namespaces; a parameter entity and a general entity with the same name are two distinct entities.

4.1 Character and Entity References

[Definition: A **character reference** refers to a specific character in the ISO/IEC 10646 character set, for example one not directly accessible from available input devices.]

Character Reference

[66] `CharRef ::= '&#' [0-9]+ ';'`
`| '&#x' [0-9a-fA-F]+ ';'` [WFC: Legal Character]

Well-formedness constraint: Legal Character

Characters referred to using character references MUST match the production for Char.

If the character reference begins with "&#x", the digits and letters up to the terminating ; provide a hexadecimal representation of the character's code point in ISO/IEC 10646. If it begins just with "&#", the digits up to the terminating ; provide a decimal representation of the character's code point.

[Definition: An entity reference refers to the content of a named entity.] [Definition: References to parsed general entities use ampersand (&) and semicolon (;) as delimiters.] [Definition: Parameter-entity references use percent-sign (%) and semicolon (;) as delimiters.]

www.it-ebooks.info

Entity Reference

[67]	Reference	::=	EntityRef | CharRef	
[68]	EntityRef	::=	'&' Name ';'	[WFC: Entity Declared] [VC: Entity Declared] [WFC: Parsed Entity] [WFC: No Recursion]
[69]	PEReference	::=	'%' Name ';'	[VC: Entity Declared] [WFC: No Recursion] [WFC: In DTD]

Well-formedness constraint: Entity Declared

In a document without any DTD, a document with only an internal DTD subset which contains no parameter entity references, or a document with "`standalone='yes'`", for an entity reference that does not occur within the external subset or a parameter entity, the Name given in the entity reference MUST match that in an *entity declaration* that does not occur within the external subset or a parameter entity, except that well-formed documents need not declare any of the following entities: `amp`, `lt`, `gt`, `apos`, `quot`. The declaration of a general entity MUST precede any reference to it which appears in a default value in an attribute-list declaration.

Note that non-validating processors are *not obligated to* to read and process entity declarations occurring in parameter entities or in the external subset; for such documents, the rule that an entity must be declared is a well-formedness constraint only if *standalone='yes'*.

Validity constraint: Entity Declared

In a document with an external subset or external parameter entities with "`standalone='no'`", the Name given in the entity reference MUST match that in an *entity declaration*. For interoperability, valid documents SHOULD declare the entities `amp`, `lt`, `gt`, `apos`, `quot`, in the form specified in **4.6 Predefined Entities**. The declaration of a parameter entity MUST precede any reference to it. Similarly, the declaration of a general entity MUST precede any attribute-list declaration containing a default value with a direct or indirect reference to that general entity.

Well-formedness constraint: Parsed Entity

An entity reference MUST NOT contain the name of an unparsed entity. Unparsed entities may be referred to only in attribute values declared to be of type ENTITY or ENTITIES.

www.it-ebooks.info

Well-formedness constraint: No Recursion

A parsed entity MUST NOT contain a recursive reference to itself, either directly or indirectly.

Well-formedness constraint: In DTD

Parameter-entity references MUST NOT appear outside the DTD.

Examples of character and entity references:

```
Type <key>less-than</key> (&#x3C;) to save options.
This document was prepared on &docdate; and
is classified &security-level;.
```

Example of a parameter-entity reference:

```
<!-- declare the parameter entity "ISOLat2"... -->
<!ENTITY % ISOLat2
         SYSTEM "http://www.xml.com/iso/isolat2-xml.entities" >
<!-- ... now reference it. -->
%ISOLat2;
```

4.2 Entity Declarations

[Definition: Entities are declared thus:]

Entity Declaration

[70]	EntityDecl	::=	GEDecl | PEDecl
[71]	GEDecl	::=	'<!ENTITY' S Name S EntityDef S? '>'
[72]	PEDecl	::=	'<!ENTITY' S '%' S Name S PEDef S? '>'
[73]	EntityDef	::=	EntityValue| (ExternalID NDataDecl?)
[74]	PEDef	::=	EntityValue | ExternalID

The Name identifies the entity in an entity reference or, in the case of an unparsed entity, in the value of an **ENTITY** or **ENTITIES** attribute. If the same entity is declared more than once, the first declaration encountered is binding; at user option, an XML processor MAY issue a warning if entities are declared multiple times.

4.2.1 Internal Entities

[Definition: If the entity definition is an EntityValue, the defined entity is called an **internal entity**. There is no separate physical storage object, and the content of the entity is given in the declaration.] Note that some processing of entity and character

www.it-ebooks.info

references in the literal entity value may be required to produce the correct replacement text: see **4.5 Construction of Internal Entity Replacement Text**.

An internal entity is a parsed entity.

Example of an internal entity declaration:

```
<!ENTITY Pub-Status "This is a pre-release of the
specification.">
```

4.2.2 External Entities

[Definition: If the entity is not internal, it is an **external entity**, declared as follows:]

External Entity Declaration

[75]	ExternalID	::=	'SYSTEM' S SystemLiteral |'PUBLIC' S PubidLiteral S SystemLiteral	
[76]	NDataDecl	::=	S 'NDATA' S Name	[VC: Notation Declared]

If the NDataDecl is present, this is a general unparsed entity; otherwise it is a parsed entity.

Validity constraint: Notation Declared

The Name MUST match the declared name of a notation.

[Definition: The SystemLiteral is called the entity's system identifier. It is meant to be converted to a URI reference (as defined in [IETF RFC 2396], updated by [IETF RFC 2732]), as part of the process of dereferencing it to obtain input for the XML processor to construct the entity's replacement text.] It is an error for a fragment identifier (beginning with a # character) to be part of a system identifier. Unless otherwise provided by information outside the scope of this specification (e.g. a special XML element type defined by a particular DTD, or a processing instruction defined by a particular application specification), relative URIs are relative to the location of the resource within which the entity declaration occurs. This is defined to be the external entity containing the '<' which starts the declaration, at the point when it is parsed as a declaration. A URI might thus be relative to the document entity, to the entity containing the external DTD subset, or to some other external parameter entity. Attempts to retrieve the resource identified by a URI MAY be redirected at the parser level (for example, in an entity resolver) or below (at the protocol level, for example, via an HTTP `Location`: header). In the absence of additional information outside the scope of this specification within the resource, the base URI of a resource is always the URI of the actual resource returned. In other words, it is the URI of the resource retrieved after all redirection has occurred.

www.it-ebooks.info

System identifiers (and other XML strings meant to be used as URI references) MAY contain characters that, according to [IETF RFC 2396] and [IETF RFC 2732], must be escaped before a URI can be used to retrieve the referenced resource. The characters to be escaped are the control characters #x0 to #x1F and #x7F (most of which cannot appear in XML), space #x20, the delimiters '<' #x3C, '>' #x3E and "" #x22, the *unwise* characters '{' #x7B, '}' #x7D, '|' #x7C, '\' #x5C, '^' #x5E and '`' #x60, as well as all characters above #x7F. Since escaping is not always a fully reversible process, it MUST be performed only when absolutely necessary and as late as possible in a processing chain. In particular, neither the process of converting a relative URI to an absolute one nor the process of passing a URI reference to a process or software component responsible for dereferencing it SHOULD trigger escaping. When escaping does occur, it MUST be performed as follows:

1. Each character to be escaped is represented in UTF-8 [Unicode3] as one or more bytes.
2. The resulting bytes are escaped with the URI escaping mechanism (that is, converted to %*HH*, where HH is the hexadecimal notation of the byte value).
3. The original character is replaced by the resulting character sequence.

[Definition: In addition to a system identifier, an external identifier MAY include a **public identifier**.] An XML processor attempting to retrieve the entity's content MAY use any combination of the public and system identifiers as well as additional information outside the scope of this specification to try to generate an alternative URI reference. If the processor is unable to do so, it MUST use the URI reference specified in the system literal. Before a match is attempted, all strings of white space in the public identifier MUST be normalized to single space characters (#x20), and leading and trailing white space MUST be removed.

Examples of external entity declarations:

```
<!ENTITY open-hatch
         SYSTEM
"http://www.textuality.com/boilerplate/OpenHatch.xml">
<!ENTITY open-hatch
         PUBLIC "-//Textuality//TEXT Standard open-hatch
boilerplate//EN"
         "http://www.textuality.com/boilerplate/OpenHatch.xml">
<!ENTITY hatch-pic
         SYSTEM "../grafix/OpenHatch.gif"
         NDATA gif >
```

4.3 Parsed Entities

4.3.1 The Text Declaration

External parsed entities SHOULD each begin with a **text declaration**.

Text Declaration

[77] TextDecl ::= '<?xml' VersionInfo? EncodingDecl S? '?>'

www.it-ebooks.info

The text declaration MUST be provided literally, not by reference to a parsed entity. The text declaration MUST NOT appear at any position other than the beginning of an external parsed entity. The text declaration in an external parsed entity is not considered part of its replacement text.

4.3.2 Well-Formed Parsed Entities

The document entity is well-formed if it matches the production labeled document. An external general parsed entity is well-formed if it matches the production labeled extParsedEnt. All external parameter entities are well-formed by definition.

Well-Formed External Parsed Entity

```
[78]    extParsedEnt    ::=    TextDecl? content
```

An internal general parsed entity is well-formed if its replacement text matches the production labeled content. All internal parameter entities are well-formed by definition.

A consequence of well-formedness in general entities is that the logical and physical structures in an XML document are properly nested; no start-tag, end-tag, empty-element tag, element, comment, processing instruction, character reference, or entity reference can begin in one entity and end in another.

4.3.3 Character Encoding in Entities

Each external parsed entity in an XML document MAY use a different encoding for its characters. All XML processors MUST be able to read entities in both the UTF-8 and UTF-16 encodings. The terms "UTF-8" and "UTF-16" in this specification do not apply to character encodings with any other labels, even if the encodings or labels are very similar to UTF-8 or UTF-16.

Entities encoded in UTF-16 MUST and entities encoded in UTF-8 MAY begin with the Byte Order Mark described by Annex H of [ISO/IEC 10646:2000], section 2.4 of [Unicode], and section 2.7 of [Unicode3] (the ZERO WIDTH NO-BREAK SPACE character, #xFEFF). This is an encoding signature, not part of either the markup or the character data of the XML document. XML processors MUST be able to use this character to differentiate between UTF-8 and UTF-16 encoded documents.

Although an XML processor is required to read only entities in the UTF-8 and UTF-16 encodings, it is recognized that other encodings are used around the world, and it may be desired for XML processors to read entities that use them. In the absence of external character encoding information (such as MIME headers), parsed entities which are stored in an encoding other than UTF-8 or UTF-16 MUST begin with a text declaration (see **4.3.1 The Text Declaration**) containing an encoding declaration:

Encoding Declaration

```
[80]    EncodingDecl    ::=    S 'encoding' Eq ('"'
                               EncName '"' | "'"
                               EncName "'" )
```

www.it-ebooks.info

```
[81]  EncName  ::=  [A-Za-z]                    /* Encoding name contains
                    ([A-Za-z0-9._] | '-')*      only Latin characters */
```

In the document entity, the encoding declaration is part of the XML declaration. The EncName is the name of the encoding used.

In an encoding declaration, the values "UTF-8", "UTF-16", "ISO-10646-UCS-2", and "ISO-10646-UCS-4" SHOULD be used for the various encodings and transformations of Unicode / ISO/IEC 10646, the values "ISO-8859-1", "ISO-8859-2",... "ISO-8859-*n*" (where *n* is the part number) SHOULD be used for the parts of ISO 8859, and the values "ISO-2022-JP", "Shift_JIS", and "EUC-JP" SHOULD be used for the various encoded forms of JIS X-0208-1997. It is RECOMMENDED that character encodings registered (as *charsets*) with the Internet Assigned Numbers Authority [IANA-CHARSETS], other than those just listed, be referred to using their registered names; other encodings SHOULD use names starting with an "x-" prefix. XML processors SHOULD match character encoding names in a case-insensitive way and SHOULD either interpret an IANA-registered name as the encoding registered at IANA for that name or treat it as unknown (processors are, of course, not required to support all IANA-registered encodings).

In the absence of information provided by an external transport protocol (e.g. HTTP or MIME), it is a fatal error for an entity including an encoding declaration to be presented to the XML processor in an encoding other than that named in the declaration, or for an entity which begins with neither a Byte Order Mark nor an encoding declaration to use an encoding other than UTF-8. Note that since ASCII is a subset of UTF-8, ordinary ASCII entities do not strictly need an encoding declaration.

It is a fatal error for a TextDecl to occur other than at the beginning of an external entity.

It is a fatal error when an XML processor encounters an entity with an encoding that it is unable to process. It is a fatal error if an XML entity is determined (via default, encoding declaration, or higher-level protocol) to be in a certain encoding but contains byte sequences that are not legal in that encoding. Specifically, it is a fatal error if an entity encoded in UTF-8 contains any irregular code unit sequences, as defined in Unicode 3.1 [Unicode3]. Unless an encoding is determined by a higher-level protocol, it is also a fatal error if an XML entity contains no encoding declaration and its content is not legal UTF-8 or UTF-16.

Examples of text declarations containing encoding declarations:

```
<?xml encoding='UTF-8'?>
<?xml encoding='EUC-JP'?>
```

4.4 XML Processor Treatment of Entities and References

The table below summarizes the contexts in which character references, entity references, and invocations of unparsed entities might appear and the REQUIRED behavior of an XML processor in each case. The labels in the leftmost column describe the recognition context:

www.it-ebooks.info

Reference in Content

as a reference anywhere after the start-tag and before the end-tag of an element; corresponds to the nonterminal content.

Reference in Attribute Value

as a reference within either the value of an attribute in a start-tag, or a default value in an attribute declaration; corresponds to the nonterminal AttValue.

Occurs as Attribute Value

as a Name, not a reference, appearing either as the value of an attribute which has been declared as type **ENTITY**, or as one of the space-separated tokens in the value of an attribute which has been declared as type **ENTITIES**.

Reference in Entity Value

as a reference within a parameter or internal entity's literal entity value in the entity's declaration; corresponds to the nonterminal EntityValue.

Reference in DTD

as a reference within either the internal or external subsets of the DTD, but outside of an EntityValue, AttValue, PI, Comment, SystemLiteral, PubidLiteral, or the contents of an ignored conditional section (see **3.4 Conditional Sections**).

	Entity Type				Character
	Parameter	Internal General	External Parsed General	Unparsed	
Reference in Content	*Not recognized*	*Included*	*Included if validating*	*Forbidden*	*Included*
Reference in Attribute Value	*Not recognized*	*Included in literal*	*Forbidden*	*Forbidden*	*Included*
Occurs as Attribute Value	*Not recognized*	*Forbidden*	*Forbidden*	*Notify*	*Not recognized*
Reference in EntityValue	*Included in literal*	*Bypassed*	*Bypassed*	*Error*	*Included*
Reference in DTD	*Included as PE*	*Forbidden*	*Forbidden*	*Forbidden*	*Forbidden*

4.4.1 Not Recognized

Outside the DTD, the % character has no special significance; thus, what would be parameter entity references in the DTD are not recognized as markup in content. Similarly, the names of unparsed entities are not recognized except when they appear in the value of an appropriately declared attribute.

www.it-ebooks.info

4.4.2 Included

[Definition: An entity is **included** when its replacement text is retrieved and processed, in place of the reference itself, as though it were part of the document at the location the reference was recognized.] The replacement text MAY contain both character data and (except for parameter entities) markup, which MUST be recognized in the usual way. (The string "`AT&T;`" expands to "`AT&T;`" and the remaining ampersand is not recognized as an entity-reference delimiter.) A character reference is **included** when the indicated character is processed in place of the reference itself.

4.4.3 Included If Validating

When an XML processor recognizes a reference to a parsed entity, in order to validate the document, the processor MUST include its replacement text. If the entity is external, and the processor is not attempting to validate the XML document, the processor MAY, but need not, include the entity's replacement text. If a non-validating processor does not include the replacement text, it MUST inform the application that it recognized, but did not read, the entity.

This rule is based on the recognition that the automatic inclusion provided by the SGML and XML entity mechanism, primarily designed to support modularity in authoring, is not necessarily appropriate for other applications, in particular document browsing. Browsers, for example, when encountering an external parsed entity reference, might choose to provide a visual indication of the entity's presence and retrieve it for display only on demand.

4.4.4 Forbidden

The following are forbidden, and constitute fatal errors:

- ✦ the appearance of a reference to an unparsed entity, except in the EntityValue in an entity declaration.
- ✦ the appearance of any character or general-entity reference in the DTD except within an EntityValue or AttValue.
- ✦ a reference to an external entity in an attribute value.

4.4.5 Included in Literal

When an entity reference appears in an attribute value, or a parameter entity reference appears in a literal entity value, its replacement text MUST be processed in place of the reference itself as though it were part of the document at the location the reference was recognized, except that a single or double quote character in the replacement text MUST always be treated as a normal data character and MUST NOT terminate the literal. For example, this is well-formed:

```
<!ENTITY % YN '"Yes"' >
<!ENTITY WhatHeSaid "He said %YN;" >
```

www.it-ebooks.info

while this is not:

```
<!ENTITY EndAttr "27'" >
<element attribute='a-&EndAttr;>
```

4.4.6 Notify

When the name of an unparsed entity appears as a token in the value of an attribute of declared type **ENTITY** or **ENTITIES**, a validating processor MUST inform the application of the system and public (if any) identifiers for both the entity and its associated notation.

4.4.7 Bypassed

When a general entity reference appears in the EntityValue in an entity declaration, it MUST be bypassed and left as is.

4.4.8 Included as PE

Just as with external parsed entities, parameter entities need only be *included if validating*. When a parameter-entity reference is recognized in the DTD and included, its replacement text MUST be enlarged by the attachment of one leading and one following space (#x20) character; the intent is to constrain the replacement text of parameter entities to contain an integral number of grammatical tokens in the DTD. This behavior MUST NOT apply to parameter entity references within entity values; these are described in **4.4.5 Included in Literal**.

4.4.9 Error

It is an error for a reference to an unparsed entity to appear in the EntityValue in an entity declaration.

4.5 Construction of Entity Replacement Text

In discussing the treatment of entities, it is useful to distinguish two forms of the entity's value. [Definition: For an internal entity, the **literal entity value** is the quoted string actually present in the entity declaration, corresponding to the non-terminal EntityValue.] [Definition: For an external entity, the **literal entity value** is the exact text contained in the entity.] [Definition: For an internal entity, the **replacement text** is the content of the entity, after replacement of character references and parameter-entity references.] [Definition: For an external entity, the **replacement text** is the content of the entity, after stripping the text declaration (leaving any surrounding whitespace) if there is one but without any replacement of character references or parameter-entity references.]

The literal entity value as given in an internal entity declaration (EntityValue) MAY contain character, parameter-entity, and general-entity references. Such references MUST be contained entirely within the literal entity value. The actual replacement text that is included (or included in literal) as described above MUST contain the *replacement text* of any parameter entities referred to, and MUST contain the character referred to, in place of any character references in the literal entity value;

www.it-ebooks.info

however, general-entity references MUST be left as-is, unexpanded. For example, given the following declarations:

```
<!ENTITY % pub     "&#xc9;ditions Gallimard" >
<!ENTITY   rights  "All rights reserved" >
<!ENTITY   book    "La Peste: Albert Camus,
&#xA9; 1947 %pub;. &rights;" >
```

then the replacement text for the entity "book" is:

```
La Peste: Albert Camus,
© 1947 Éditions Gallimard. &rights;
```

The general-entity reference "&rights;" would be expanded should the reference "&book;" appear in the document's content or an attribute value.

These simple rules may have complex interactions; for a detailed discussion of a difficult example, see **D Expansion of Entity and Character References**.

4.6 Predefined Entities

[Definition: Entity and character references MAY both be used to **escape** the left angle bracket, ampersand, and other delimiters. A set of general entities (amp, lt, gt, apos, quot) is specified for this purpose. Numeric character references MAY also be used; they are expanded immediately when recognized and MUST be treated as character data, so the numeric character references "<" and "&" MAY be used to escape < and & when they occur in character data.]

All XML processors MUST recognize these entities whether they are declared or not. For interoperability, valid XML documents SHOULD declare these entities, like any others, before using them. If the entities lt or amp are declared, they MUST be declared as internal entities whose replacement text is a character reference to the respective character (less-than sign or ampersand) being escaped; the double escaping is REQUIRED for these entities so that references to them produce a well-formed result. If the entities gt, apos, or quot are declared, they MUST be declared as internal entities whose replacement text is the single character being escaped (or a character reference to that character; the double escaping here is OPTIONAL but harmless). For example:

```
<!ENTITY lt     "&#38;#60;">
<!ENTITY gt     "&#62;">
<!ENTITY amp    "&#38;#38;">
<!ENTITY apos   "&#39;">
<!ENTITY quot   "&#34;">
```

4.7 Notation Declarations

[Definition: **Notations** identify by name the format of unparsed entities, the format of elements which bear a notation attribute, or the application to which a processing instruction is addressed.]

www.it-ebooks.info

[Definition: **Notation declarations** provide a name for the notation, for use in entity and attribute-list declarations and in attribute specifications, and an external identifier for the notation which may allow an XML processor or its client application to locate a helper application capable of processing data in the given notation.]

Notation Declarations

[82]	NotationDecl	::=	'<!NOTATION' S Name S (ExternalID | PublicID) S? '>'	[VC: Unique Notation Name]
[83]	PublicID	::=	'PUBLIC' S PubidLiteral	

Validity constraint: Unique Notation Name

A given Name MUST NOT be declared in more than one notation declaration.

XML processors MUST provide applications with the name and external identifier(s) of any notation declared and referred to in an attribute value, attribute definition, or entity declaration. They MAY additionally resolve the external identifier into the system identifier, file name, or other information needed to allow the application to call a processor for data in the notation described. (It is not an error, however, for XML documents to declare and refer to notations for which notation-specific applications are not available on the system where the XML processor or application is running.)

4.8 Document Entity

[Definition: The **document entity** serves as the root of the entity tree and a starting-point for an XML processor.] This specification does not specify how the document entity is to be located by an XML processor; unlike other entities, the document entity has no name and might well appear on a processor input stream without any identification at all.

5 Conformance

5.1 Validating and Non-Validating Processors

Conforming XML processors fall into two classes: validating and non-validating.

Validating and non-validating processors alike MUST report violations of this specification's well-formedness constraints in the content of the document entity and any other parsed entities that they read.

[Definition: **Validating processors** MUST, at user option, report violations of the constraints expressed by the declarations in the DTD, and failures to fulfill the validity constraints given in this specification.] To accomplish this, validating XML processors MUST read and process the entire DTD and all external parsed entities referenced in the document.

Non-validating processors are REQUIRED to check only the document entity, including the entire internal DTD subset, for well-formedness. [Definition: While they

www.it-ebooks.info

are not required to check the document for validity, they are REQUIRED to **process** all the declarations they read in the internal DTD subset and in any parameter entity that they read, up to the first reference to a parameter entity that they do *not* read; that is to say, they MUST use the information in those declarations to *normalize* attribute values, *include* the replacement text of internal entities, and supply *default attribute values*.] Except when `standalone="yes"`, they MUST NOT process entity declarations or attribute-list declarations encountered after a reference to a parameter entity that is not read, since the entity may have contained overriding declarations; when `standalone="yes"`, processors MUST process these declarations.

Note that when processing invalid documents with a non-validating processor the application may not be presented with consistent information. For example, several requirements for uniqueness within the document may not be met, including more than one element with the same id, duplicate declarations of elements or notations with the same name, etc. In these cases the behavior of the parser with respect to reporting such information to the application is undefined.

5.2 Using XML Processors

The behavior of a validating XML processor is highly predictable; it must read every piece of a document and report all well-formedness and validity violations. Less is required of a non-validating processor; it need not read any part of the document other than the document entity. This has two effects that may be important to users of XML processors:

- Certain well-formedness errors, specifically those that require reading external entities, may fail to be detected by a non-validating processor. Examples include the constraints entitled *Entity Declared*, *Parsed Entity*, and *No Recursion*, as well as some of the cases described as forbidden in **4.4 XML Processor Treatment of Entities and References**.
- The information passed from the processor to the application may vary, depending on whether the processor reads parameter and external entities. For example, a non-validating processor may fail to *normalize* attribute values, *include* the replacement text of internal entities, or supply *default attribute values*, where doing so depends on having read declarations in external or parameter entities.

For maximum reliability in interoperating between different XML processors, applications which use non-validating processors SHOULD NOT rely on any behaviors not required of such processors. Applications which require DTD facilities not related to validation (such as the declaration of default attributes and internal entities that are or may be specified in external entities) SHOULD use validating XML processors.

6 Notation

The formal grammar of XML is given in this specification using a simple Extended Backus-Naur Form (EBNF) notation. Each rule in the grammar defines one symbol, in the form

www.it-ebooks.info

```
symbol ::= expression
```

Symbols are written with an initial capital letter if they are the start symbol of a regular language, otherwise with an initial lowercase letter. Literal strings are quoted.

Within the expression on the right-hand side of a rule, the following expressions are used to match strings of one or more characters:

`#xN`

where `N` is a hexadecimal integer, the expression matches the character whose number (code point) in ISO/IEC 10646 is `N`. The number of leading zeros in the `#xN` form is insignificant.

`[a-zA-Z],[#xN-#xN]`

matches any Char with a value in the range(s) indicated (inclusive).

`[abc], [#xN#xN#xN]`

matches any Char with a value among the characters enumerated. Enumerations and ranges can be mixed in one set of brackets.

`[^a-z], [^#xN-#xN]`

matches any Char with a value *outside* the range indicated.

`[^abc], [^#xN#xN#xN]`

matches any Char with a value not among the characters given. Enumerations and ranges of forbidden values can be mixed in one set of brackets.

`"string"`

matches a literal string matching that given inside the double quotes.

`'string'`

matches a literal string matching that given inside the single quotes.

These symbols may be combined to match more complex patterns as follows, where `A` and `B` represent simple expressions:

`( expression )`

expression is treated as a unit and may be combined as described in this list.

`A?`

matches `A` or nothing; optional `A`.

www.it-ebooks.info

`A B`

matches `A` followed by `B`. This operator has higher precedence than alternation; thus `A B | C D` is identical to `(A B) | (C D)`.

`A | B`

matches A or B.

`A - B`

matches any string that matches `A` but does not match `B`.

`A+`

matches one or more occurrences of `A`. Concatenation has higher precedence than alternation; thus `A+ | B+` is identical to `(A+) | (B+)`.

`A*`

matches zero or more occurrences of `A`. Concatenation has higher precedence than alternation; thus `A* | B*` is identical to `(A*) | (B*)`.

Other notations used in the productions are:

`/* ... */`

comment.

`[ wfc: ... ]`

well-formedness constraint; this identifies by name a constraint on well-formed documents associated with a production.

`[ vc: ... ]`

validity constraint; this identifies by name a constraint on valid documents associated with a production.

A References

A.1 Normative References

IANA-CHARSETS

(Internet Assigned Numbers Authority) *Official Names for Character Sets*, ed. Keld Simonsen et al. (See `http://www.iana.org/assignments/character-sets`.)

www.it-ebooks.info

IETF RFC 2119

IETF (Internet Engineering Task Force). *RFC 2119: Key words for use in RFCs to Indicate Requirement Levels*. Scott Bradner, 1997. (See `http://www.ietf.org/rfc/rfc2119.txt`.)

IETF RFC 2396

IETF (Internet Engineering Task Force). *RFC 2396: Uniform Resource Identifiers (URI): Generic Syntax*. T. Berners-Lee, R. Fielding, L. Masinter. 1998. (See `http://www.ietf.org/rfc/rfc2396.txt`.)

IETF RFC 2732

IETF (Internet Engineering Task Force). *RFC 2732: Format for Literal IPv6 Addresses in URL's*. R. Hinden, B. Carpenter, L. Masinter. 1999. (See `http://www.ietf.org/ rfc/rfc2732.txt`.)

IETF RFC 3066

IETF (Internet Engineering Task Force). *RFC 3066: Tags for the Identification of Languages*, ed. H. Alvestrand. 2001. (See `http://www.ietf.org/rfc/rfc3066.txt`.)

ISO/IEC 10646

ISO (International Organization for Standardization). *ISO/IEC 10646-1:2000. Information technology — Universal Multiple-Octet Coded Character Set (UCS)— Part 1: Architecture and Basic Multilingual Plane* and *ISO/IEC 10646-2:2001. Information technology — Universal Multiple-Octet Coded Character Set (UCS) — Part 2: Supplementary Planes*, as, from time to time, amended, replaced by a new edition or expanded by the addition of new parts. [Geneva]: International Organization for Standardization. (See `http://www.iso.ch` for the latest version.)

ISO/IEC 10646:2000

ISO (International Organization for Standardization). *ISO/IEC 10646-1:2000. Information technology — Universal Multiple-Octet Coded Character Set (UCS) — Part 1: Architecture and Basic Multilingual Plane.* [Geneva]: International Organization for Standardization, 2000.

Unicode

The Unicode Consortium. *The Unicode Standard, Version 2.0.* Reading, Mass.: Addison-Wesley Developers Press, 1996.

Unicode3

The Unicode Consortium. *The Unicode Standard, Version 3.2*, defined by: *The Unicode Standard, Version 3.0* (Reading, MA, Addison-Wesley, 2000.

www.it-ebooks.info

ISBN 0-201-61633-5), as amended by the *Unicode Standard Annex #27: Unicode 3.1* (http://www.unicode .org/reports/tr27) and the *Unicode Standard Annex #28: Unicode 3.2* (http://www.unicode.org/reports/tr28).

A.2 Other References

Aho/Ullman

Aho, Alfred V., Ravi Sethi, and Jeffrey D. Ullman. *Compilers: Principles, Techniques, and Tools*. Reading: Addison-Wesley, 1986, rpt. corr. 1988.

Brüggemann-Klein

Brüggemann-Klein, Anne. *Formal Models in Document Processing*. Habilitationsschrift. Faculty of Mathematics at the University of Freiburg, 1993. (See ftp://ftp.informatik.uni-freiburg.de/documents/papers/brueggem/habil.ps.)

Brüggemann-Klein and Wood

Brüggemann-Klein, Anne, and Derick Wood. *Deterministic Regular Languages*. Universität Freiburg, Institut für Informatik, Bericht 38, Oktober 1991. Extended abstract in A. Finkel, M. Jantzen, Hrsg., STACS 1992, S. 173-184. Springer-Verlag, Berlin 1992. Lecture Notes in Computer Science 577. Full version titled *One-Unambiguous Regular Languages* in Information and Computation 140 (2): 229-253, February 1998.

Clark

James Clark. *Comparison of SGML and XML*. (See http://www.w3.org/TR/NOTE-sgml-xml-971215.)

IANA-LANGCODES

(Internet Assigned Numbers Authority) *Registry of Language Tags*, ed. Keld Simonsen et al. (See http://www.iana.org/assignments/language-tags.)

IETF RFC 2141

IETF (Internet Engineering Task Force). *RFC 2141: URN Syntax*, ed. R. Moats. 1997. (See http://www.ietf.org/rfc/rfc2141.txt.)

IETF RFC 3023

IETF (Internet Engineering Task Force). *RFC 3023: XML Media Types*. eds. M. Murata, S. St.Laurent, D. Kohn. 2001. (See http://www.ietf.org/rfc/rfc3023.txt.)

www.it-ebooks.info

IETF RFC 2781

IETF (Internet Engineering Task Force). *RFC 2781: UTF-16, an encoding of ISO 10646*, ed. P. Hoffman, F. Yergeau. 2000. (See http://www.ietf.org/rfc/rfc2781.txt.)

ISO 639

(International Organization for Standardization). *ISO 639:1988 (E). Code for the representation of names of languages.* [Geneva]: International Organization for Standardization, 1988.

ISO 3166

(International Organization for Standardization). *ISO 3166-1:1997 (E). Codes for the representation of names of countries and their subdivisions — Part 1: Country codes* [Geneva]: International Organization for Standardization, 1997.

ISO 8879

ISO (International Organization for Standardization). *ISO 8879:1986(E). Information processing—Text and Office Systems—Standard Generalized Markup Language (SGML).* First edition—1986-10-15. [Geneva]: International Organization for Standardization, 1986.

ISO/IEC 10744

ISO (International Organization for Standardization). *ISO/IEC 10744-1992 (E). Information technology—Hypermedia/Time-based Structuring Language (HyTime).* [Geneva]: International Organization for Standardization, 1992. *Extended Facilities Annexe.* [Geneva]: International Organization for Standardization, 1996.

WEBSGML

ISO (International Organization for Standardization). *ISO 8879:1986 TC2. Information technology — Document Description and Processing Languages*. [Geneva]: International Organization for Standardization, 1998. (See http://www.sgmlsource.com/8879/n0029.htm.)

XML Names

Tim Bray, Dave Hollander, and Andrew Layman, editors. *Namespaces in XML*. Textuality, Hewlett-Packard, and Microsoft. World Wide Web Consortium, 1999. (See http://www.w3.org/TR/REC-xml-names/.)

B Character Classes

Following the characteristics defined in the Unicode standard, characters are classed as base characters (among others, these contain the alphabetic characters of the

www.it-ebooks.info

Latin alphabet), ideographic characters, and combining characters (among others, this class contains most diacritics). Digits and extenders are also distinguished.

Characters

```
[84]    Letter          ::=     BaseChar | Ideographic

[85]    BaseChar        ::=     [#x0041-#x005A] | [#x0061-#x007A]
                                | [#x00C0-#x00D6] | [#x00D8-#x00F6]
                                | [#x00F8-#x00FF] | [#x0100-#x0131]
                                | [#x0134-#x013E] | [#x0141-#x0148]
                                | [#x014A-#x017E] | [#x0180-#x01C3]
                                | [#x01CD-#x01F0] | [#x01F4-#x01F5]
                                | [#x01FA-#x0217] | [#x0250-#x02A8]
                                | [#x02BB-#x02C1] | #x0386 | [#x0388-#x038A]
                                | #x038C | [#x038E-#x03A1] | [#x03A3-#x03CE]
                                | [#x03D0-#x03D6] | #x03DA | #x03DC | #x03DE
                                | #x03E0 | [#x03E2-#x03F3] | [#x0401-#x040C]
                                | [#x040E-#x044F] | [#x0451-#x045C]
                                | [#x045E-#x0481] | [#x0490-#x04C4]
                                | [#x04C7-#x04C8] | [#x04CB-#x04CC]
                                | [#x04D0-#x04EB] | [#x04EE-#x04F5]
                                | [#x04F8-#x04F9] | [#x0531-#x0556] | #x0559
                                | [#x0561-#x0586] | [#x05D0-#x05EA]
                                | [#x05F0-#x05F2] | [#x0621-#x063A]
                                | [#x0641-#x064A] | [#x0671-#x06B7]
                                | [#x06BA-#x06BE] | [#x06C0-#x06CE]
                                | [#x06D0-#x06D3] | #x06D5 | [#x06E5-#x06E6]
                                | [#x0905-#x0939] | #x093D | [#x0958-#x0961]
                                | [#x0985-#x098C] | [#x098F-#x0990]
                                | [#x0993-#x09A8] | [#x09AA-#x09B0] | #x09B2
                                | [#x09B6-#x09B9] | [#x09DC-#x09DD]
                                | [#x09DF-#x09E1] | [#x09F0-#x09F1]
                                | [#x0A05-#x0A0A] | [#x0A0F-#x0A10]
                                | [#x0A13-#x0A28] | [#x0A2A-#x0A30]
                                | [#x0A32-#x0A33] | [#x0A35-#x0A36]
                                | [#x0A38-#x0A39] | [#x0A59-#x0A5C] | #x0A5E
                                | [#x0A72-#x0A74] | [#x0A85-#x0A8B] | #x0A8D
                                | [#x0A8F-#x0A91] | [#x0A93-#x0AA8]
                                | [#x0AAA-#x0AB0] | [#x0AB2-#x0AB3]
                                | [#x0AB5-#x0AB9] | #x0ABD | #x0AE0
                                | [#x0B05-#x0B0C] | [#x0B0F-#x0B10]
                                | [#x0B13-#x0B28] | [#x0B2A-#x0B30]
                                | [#x0B32-#x0B33] | [#x0B36-#x0B39] | #x0B3D
                                | [#x0B5C-#x0B5D] | [#x0B5F-#x0B61]
                                | [#x0B85-#x0B8A] | [#x0B8E-#x0B90]
                                | [#x0B92-#x0B95] | [#x0B99-#x0B9A] | #x0B9C
                                | [#x0B9E-#x0B9F] | [#x0BA3-#x0BA4]
                                | [#x0BA8-#x0BAA] | [#x0BAE-#x0BB5]
                                | [#x0BB7-#x0BB9] | [#x0C05-#x0C0C]
                                | [#x0C0E-#x0C10] | [#x0C12-#x0C28]
                                | [#x0C2A-#x0C33] | [#x0C35-#x0C39]
                                | [#x0C60-#x0C61] | [#x0C85-#x0C8C]
                                | [#x0C8E-#x0C90] | [#x0C92-#x0CA8]
                                | [#x0CAA-#x0CB3] | [#x0CB5-#x0CB9] | #x0CDE
```

www.it-ebooks.info

```
                                    | [#x0CE0-#x0CE1] | [#x0D05-#x0D0C]
                                    | [#x0D0E-#x0D10] | [#x0D12-#x0D28]
                                    | [#x0D2A-#x0D39] | [#x0D60-#x0D61]
                                    | [#x0E01-#x0E2E] | #x0E30 | [#x0E32-#x0E33]
                                    | [#x0E40-#x0E45] | [#x0E81-#x0E82] | #x0E84
                                    | [#x0E87-#x0E88] | #x0E8A | #x0E8D
                                    | [#x0E94-#x0E97] | [#x0E99-#x0E9F]
                                    | [#x0EA1-#x0EA3] | #x0EA5 | #x0EA7
                                    | [#x0EAA-#x0EAB] | [#x0EAD-#x0EAE] | #x0EB0
                                    | [#x0EB2-#x0EB3] | #x0EBD | [#x0EC0-#x0EC4]
                                    | [#x0F40-#x0F47] | [#x0F49-#x0F69]
                                    | [#x10A0-#x10C5] | [#x10D0-#x10F6] | #x1100
                                    | [#x1102-#x1103] | [#x1105-#x1107] | #x1109
                                    | [#x110B-#x110C] | [#x110E-#x1112] | #x113C
                                    | #x113E | #x1140 | #x114C | #x114E | #x1150
                                    | [#x1154-#x1155] | #x1159 | [#x115F-#x1161]
                                    | #x1163 | #x1165 | #x1167 | #x1169
                                    | [#x116D-#x116E] | [#x1172-#x1173] | #x1175
                                    | #x119E | #x11A8 | #x11AB | [#x11AE-#x11AF]
                                    | [#x11B7-#x11B8] | #x11BA | [#x11BC-#x11C2]
                                    | #x11EB | #x11F0 | #x11F9 | [#x1E00-#x1E9B]
                                    | [#x1EA0-#x1EF9] | [#x1F00-#x1F15]
                                    | [#x1F18-#x1F1D] | [#x1F20-#x1F45]
                                    | [#x1F48-#x1F4D] | [#x1F50-#x1F57] | #x1F59
                                    | #x1F5B | #x1F5D | [#x1F5F-#x1F7D]
                                    | [#x1F80-#x1FB4] | [#x1FB6-#x1FBC] | #x1FBE
                                    | [#x1FC2-#x1FC4] | [#x1FC6-#x1FCC]
                                    | [#x1FD0-#x1FD3] | [#x1FD6-#x1FDB]
                                    | [#x1FE0-#x1FEC] | [#x1FF2-#x1FF4]
                                    | [#x1FF6-#x1FFC] | #x2126 | [#x212A-#x212B]
                                    | #x212E | [#x2180-#x2182] | [#x3041-#x3094]
                                    | [#x30A1-#x30FA] | [#x3105-#x312C]
                                    | [#xAC00-#xD7A3]

[86]    Ideographic     ::=     [#x4E00-#x9FA5] | #x3007 | [#x3021-#x3029]

[87]    CombiningChar   ::=     [#x0300-#x0345] | [#x0360-#x0361]
                                    | [#x0483-#x0486] | [#x0591-#x05A1]
                                    | [#x05A3-#x05B9] | [#x05BB-#x05BD] | #x05BF
                                    | [#x05C1-#x05C2] | #x05C4 | [#x064B-#x0652]
                                    | #x0670 | [#x06D6-#x06DC] | [#x06DD-#x06DF]
                                    | [#x06E0-#x06E4] | [#x06E7-#x06E8]
                                    | [#x06EA-#x06ED] | [#x0901-#x0903] | #x093C
                                    | [#x093E-#x094C] | #x094D | [#x0951-#x0954]
                                    | [#x0962-#x0963] | [#x0981-#x0983] | #x09BC
                                    | #x09BE | #x09BF | [#x09C0-#x09C4]
                                    | [#x09C7-#x09C8] | [#x09CB-#x09CD] | #x09D7
                                    | [#x09E2-#x09E3] | #x0A02 | #x0A3C | #x0A3E
                                    | #x0A3F | [#x0A40-#x0A42] | [#x0A47-#x0A48]
                                    | [#x0A4B-#x0A4D] | [#x0A70-#x0A71]
                                    | [#x0A81-#x0A83] | #x0ABC | [#x0ABE-#x0AC5]
                                    | [#x0AC7-#x0AC9] | [#x0ACB-#x0ACD]
                                    | [#x0B01-#x0B03] | #x0B3C | [#x0B3E-#x0B43]
                                    | [#x0B47-#x0B48] | [#x0B4B-#x0B4D]
                                    | [#x0B56-#x0B57] | [#x0B82-#x0B83]
```

www.it-ebooks.info

```
                                    | [#x0BBE-#x0BC2] | [#x0BC6-#x0BC8]
                                    | [#x0BCA-#x0BCD] | #x0BD7 | [#x0C01-#x0C03]
                                    | [#x0C3E-#x0C44] | [#x0C46-#x0C48]
                                    | [#x0C4A-#x0C4D] | [#x0C55-#x0C56]
                                    | [#x0C82-#x0C83] | [#x0CBE-#x0CC4]
                                    | [#x0CC6-#x0CC8] | [#x0CCA-#x0CCD]
                                    | [#x0CD5-#x0CD6] | [#x0D02-#x0D03]
                                    | [#x0D3E-#x0D43] | [#x0D46-#x0D48]
                                    | [#x0D4A-#x0D4D] | #x0D57 | #x0E31
                                    | [#x0E34-#x0E3A] | [#x0E47-#x0E4E] | #x0EB1
                                    | [#x0EB4-#x0EB9] | [#x0EBB-#x0EBC]
                                    | [#x0EC8-#x0ECD] | [#x0F18-#x0F19] | #x0F35
                                    | #x0F37 | #x0F39 | #x0F3E | #x0F3F
                                    | [#x0F71-#x0F84] | [#x0F86-#x0F8B]
                                    | [#x0F90-#x0F95] | #x0F97 | [#x0F99-#x0FAD]
                                    | [#x0FB1-#x0FB7] | #x0FB9 | [#x20D0-#x20DC]
                                    | #x20E1 | [#x302A-#x302F] | #x3099 | #x309A
[88] Digit      ::=                 [#x0030-#x0039] | [#x0660-#x0669]
                                    | [#x06F0-#x06F9] | [#x0966-#x096F]
                                    | [#x09E6-#x09EF] | [#x0A66-#x0A6F]
                                    | [#x0AE6-#x0AEF] | [#x0B66-#x0B6F]
                                    | [#x0BE7-#x0BEF] | [#x0C66-#x0C6F]
                                    | [#x0CE6-#x0CEF] | [#x0D66-#x0D6F]
                                    | [#x0E50-#x0E59] | [#x0ED0-#x0ED9]
                                    | [#x0F20-#x0F29]
[89] Extender   ::=                 #x00B7 | #x02D0 | #x02D1 | #x0387 | #x0640
                                    | #x0E46 | #x0EC6 | #x3005 | [#x3031-#x3035]
                                    | [#x309D-#x309E] | [#x30FC-#x30FE]
```

The character classes defined here can be derived from the Unicode 2.0 character database as follows:

- Name start characters must have one of the categories Ll, Lu, Lo, Lt, Nl.
- Name characters other than Name-start characters must have one of the categories Mc, Me, Mn, Lm, or Nd.
- Characters in the compatibility area (i.e. with character code greater than #xF900 and less than #xFFFE) are not allowed in XML names.
- Characters which have a font or compatibility decomposition (i.e. those with a "compatibility formatting tag" in field 5 of the database – marked by field 5 beginning with a "<") are not allowed.
- The following characters are treated as name-start characters rather than name characters, because the property file classifies them as Alphabetic: [#x02BB-#x02C1], #x0559, #x06E5, #x06E6.
- Characters #x20DD-#x20E0 are excluded (in accordance with Unicode 2.0, section 5.14).
- Character #x00B7 is classified as an extender, because the property list so identifies it.

www.it-ebooks.info

- Character #x0387 is added as a name character, because #x00B7 is its canonical equivalent.
- Characters ':' and '_' are allowed as name-start characters.
- Characters '-' and '.' are allowed as name characters.

C XML and SGML (Non-Normative)

XML is designed to be a subset of SGML, in that every XML document should also be a conforming SGML document. For a detailed comparison of the additional restrictions that XML places on documents beyond those of SGML, see [Clark].

D Expansion of Entity and Character References (Non-Normative)

This appendix contains some examples illustrating the sequence of entity- and character-reference recognition and expansion, as specified in **4.4 XML Processor Treatment of Entities and References**.

If the DTD contains the declaration

```
<!ENTITY example "<p>An ampersand (&#38;#38;) may be escaped
numerically (&#38;#38;#38;) or with a general entity
(&amp;amp;).</p>" >
```

then the XML processor will recognize the character references when it parses the entity declaration, and resolve them before storing the following string as the value of the entity "`example`":

```
<p>An ampersand (&#38;) may be escaped
numerically (&#38;#38;) or with a general entity
(&amp;amp;).</p>
```

A reference in the document to "`&example;`" will cause the text to be reparsed, at which time the start- and end-tags of the `p` element will be recognized and the three references will be recognized and expanded, resulting in a `p` element with the following content (all data, no delimiters or markup):

```
An ampersand (&) may be escaped
numerically (&#38;) or with a general entity
(&amp;).
```

A more complex example will illustrate the rules and their effects fully. In the following example, the line numbers are solely for reference.

www.it-ebooks.info

```
1 <?xml version='1.0'?>
2 <!DOCTYPE test [
3 <!ELEMENT test (#PCDATA) >
4 <!ENTITY % xx '&#37;zz;'>
5 <!ENTITY % zz '&#60;!ENTITY tricky "error-prone" >' >
6 %xx;
7 ]>
8 <test>This sample shows a &tricky; method.</test>
```

This produces the following:

- in line 4, the reference to character 37 is expanded immediately, and the parameter entity "xx" is stored in the symbol table with the value "%zz;". Since the replacement text is not rescanned, the reference to parameter entity "zz" is not recognized. (And it would be an error if it were, since "zz" is not yet declared.)
- in line 5, the character reference "<" is expanded immediately and the parameter entity "zz" is stored with the replacement text "<!ENTITY tricky "error-prone" >", which is a well-formed entity declaration.
- in line 6, the reference to "xx" is recognized, and the replacement text of "xx" (namely "%zz;") is parsed. The reference to "zz" is recognized in its turn, and its replacement text ("<!ENTITY tricky "error-prone" >") is parsed. The general entity "tricky" has now been declared, with the replacement text "error-prone".
- in line 8, the reference to the general entity "tricky" is recognized, and it is expanded, so the full content of the test element is the self-describing (and ungrammatical) string *This sample shows a error-prone method.*

E Deterministic Content Models (Non-Normative)

As noted in **3.2.1 Element Content**, it is required that content models in element type declarations be deterministic. This requirement is for compatibility with SGML (which calls deterministic content models "unambiguous"); XML processors built using SGML systems may flag non-deterministic content models as errors.

For example, the content model ((b, c) | (b, d)) is non-deterministic, because given an initial b the XML processor cannot know which b in the model is being matched without looking ahead to see which element follows the b. In this case, the two references to b can be collapsed into a single reference, making the model read (b, (c | d)). An initial b now clearly matches only a single name in the content model. The processor doesn't need to look ahead to see what follows; either c or d would be accepted.

More formally: a finite state automaton may be constructed from the content model using the standard algorithms, e.g. algorithm 3.5 in section 3.9 of Aho, Sethi, and Ullman [Aho/Ullman]. In many such algorithms, a follow set is constructed for each position in the regular expression (i.e., each leaf node in the syntax tree for the

www.it-ebooks.info

regular expression); if any position has a follow set in which more than one following position is labeled with the same element type name, then the content model is in error and may be reported as an error.

Algorithms exist which allow many but not all non-deterministic content models to be reduced automatically to equivalent deterministic models; see Brüggemann-Klein 1991 [Brüggemann-Klein].

F Autodetection of Character Encodings (Non-Normative)

The XML encoding declaration functions as an internal label on each entity, indicating which character encoding is in use. Before an XML processor can read the internal label, however, it apparently has to know what character encoding is in use—which is what the internal label is trying to indicate. In the general case, this is a hopeless situation. It is not entirely hopeless in XML, however, because XML limits the general case in two ways: each implementation is assumed to support only a finite set of character encodings, and the XML encoding declaration is restricted in position and content in order to make it feasible to autodetect the character encoding in use in each entity in normal cases. Also, in many cases other sources of information are available in addition to the XML data stream itself. Two cases may be distinguished, depending on whether the XML entity is presented to the processor without, or with, any accompanying (external) information. We consider the first case first.

F.1 Detection Without External Encoding Information

Because each XML entity not accompanied by external encoding information and not in UTF-8 or UTF-16 encoding must begin with an XML encoding declaration, in which the first characters must be '<?xml', any conforming processor can detect, after two to four octets of input, which of the following cases apply. In reading this list, it may help to know that in UCS-4, '<' is "#x0000003C" and '?' is "#x0000003F", and the Byte Order Mark required of UTF-16 data streams is "#xFEFF". The notation ## is used to denote any byte value except that two consecutive ##s cannot be both 00.

With a Byte Order Mark:

00 00 FE FF	UCS-4, big-endian machine (1234 order)
FF FE 00 00	UCS-4, little-endian machine (4321 order)
00 00 FF FE	UCS-4, unusual octet order (2143)
FE FF 00 00	UCS-4, unusual octet order (3412)
FE FF ## ## ## ##	UTF-16, big-endian
FF FE ## ## ## ##	UTF-16, little-endian
EF BB BF	UTF-8

www.it-ebooks.info

Without a Byte Order Mark:

00 00 00 3C 3C 00 00 00 00 00 3C 00 00 3C 00 00	UCS-4 or other encoding with a 32-bit code unit and ASCII characters encoded as ASCII values, in respectively big-endian (1234), little-endian (4321) and two unusual byte orders (2143 and 3412). The encoding declaration must be read to determine which of UCS-4 or other supported 32-bit encodings applies.
00 3C 00 3F	UTF-16BE or big-endian ISO-10646-UCS-2 or other encoding with a 16-bit code unit in big-endian order and ASCII characters encoded as ASCII values (the encoding declaration must be read to determine which)
3C 00 3F 00	UTF-16LE or little-endian ISO-10646-UCS-2 or other encoding with a 16-bit code unit in little-endian order and ASCII characters encoded as ASCII values (the encoding declaration must be read to determine which)
3C 3F 78 6D	UTF-8, ISO 646, ASCII, some part of ISO 8859, Shift-JIS, EUC, or any other 7-bit, 8-bit, or mixed-width encoding which ensures that the characters of ASCII have their normal positions, width, and values; the actual encoding declaration must be read to detect which of these applies, but since all of these encodings use the same bit patterns for the relevant ASCII characters, the encoding declaration itself may be read reliably
4C 6F A7 94	EBCDIC (in some flavor; the full encoding declaration must be read to tell which code page is in use)
Other	UTF-8 without an encoding declaration, or else the data stream is mislabeled (lacking a required encoding declaration), corrupt, fragmentary, or enclosed in a wrapper of some kind

Note:

In cases above which do not require reading the encoding declaration to determine the encoding, section 4.3.3 still requires that the encoding declaration, if present, be read and that the encoding name be checked to match the actual encoding of the entity. Also, it is possible that new character encodings will be invented that will make it necessary to use the encoding declaration to determine the encoding, in cases where this is not required at present.

This level of autodetection is enough to read the XML encoding declaration and parse the character-encoding identifier, which is still necessary to distinguish the individual members of each family of encodings (e.g. to tell UTF-8 from 8859, and the

www.it-ebooks.info

parts of 8859 from each other, or to distinguish the specific EBCDIC code page in use, and so on).

Because the contents of the encoding declaration are restricted to characters from the ASCII repertoire (however encoded), a processor can reliably read the entire encoding declaration as soon as it has detected which family of encodings is in use. Since in practice, all widely used character encodings fall into one of the categories above, the XML encoding declaration allows reasonably reliable in-band labeling of character encodings, even when external sources of information at the operating-system or transport-protocol level are unreliable. Character encodings such as UTF-7 that make overloaded usage of ASCII-valued bytes may fail to be reliably detected.

Once the processor has detected the character encoding in use, it can act appropriately, whether by invoking a separate input routine for each case, or by calling the proper conversion function on each character of input.

Like any self-labeling system, the XML encoding declaration will not work if any software changes the entity's character set or encoding without updating the encoding declaration. Implementors of character-encoding routines should be careful to ensure the accuracy of the internal and external information used to label the entity.

F.2 Priorities in the Presence of External Encoding Information

The second possible case occurs when the XML entity is accompanied by encoding information, as in some file systems and some network protocols. When multiple sources of information are available, their relative priority and the preferred method of handling conflict should be specified as part of the higher-level protocol used to deliver XML. In particular, please refer to [IETF RFC 3023] or its successor, which defines the `text/xml` and `application/xml` MIME types and provides some useful guidance. In the interests of interoperability, however, the following rule is recommended.

- If an XML entity is in a file, the Byte-Order Mark and encoding declaration are used (if present) to determine the character encoding.

G W3C XML Working Group (Non-Normative)

This specification was prepared and approved for publication by the W3C XML Working Group (WG). WG approval of this specification does not necessarily imply that all WG members voted for its approval. The current and former participants of the XML WG are:

- Jon Bosak, Sun (*Chair*)
- James Clark (*Technical Lead*)
- Tim Bray, Textuality and Netscape (*XML Co-editor*)

www.it-ebooks.info

- Jean Paoli, Microsoft (*XML Co-editor*)
- C. M. Sperberg-McQueen, U. of Ill. (*XML Co-editor*)
- Dan Connolly, W3C (*W3C Liaison*)
- Paula Angerstein, Texcel
- Steve DeRose, INSO
- Dave Hollander, HP
- Eliot Kimber, ISOGEN
- Eve Maler, ArborText
- Tom Magliery, NCSA
- Murray Maloney, SoftQuad, Grif SA, Muzmo and Veo Systems
- MURATA Makoto (FAMILY Given), Fuji Xerox Information Systems
- Joel Nava, Adobe
- Conleth O'Connell, Vignette
- Peter Sharpe, SoftQuad
- John Tigue, DataChannel

H W3C XML Core Working Group (Non-Normative)

The third edition of this specification was prepared by the W3C XML Core Working Group (WG). The participants in the WG at the time of publication of this edition were:

- Leonid Arbouzov, Sun Microsystems
- Mary Brady
- John Cowan
- John Evdemon, Microsoft
- Andrew Fang, Arbortext
- Paul Grosso, Arbortext (*Co-Chair*)
- Arnaud Le Hors, IBM
- Dmitry Lenkov, Oracle
- Anjana Manian, Oracle
- Glenn Marcy, IBM
- Jonathan Marsh, Microsoft
- Sandra Martinez, NIST
- Liam Quin, W3C (*Staff Contact*)

www.it-ebooks.info

- Lew Shannon
- Richard Tobin, University of Edinburgh
- Daniel Veillard
- Norman Walsh, Sun Microsystems (*Co-Chair*)
- François Yergeau (*Third Edition Editor*)

I Production Notes (Non-Normative)

This Third Edition was encoded in a slightly modified version of the XMLspec DTD, v2.5. The XHTML versions were produced with a combination of the xmlspec.xsl, diffspec.xsl, and REC-xml-3e.xsl XSLT stylesheets.

www.it-ebooks.info

www.it-ebooks.info

APPENDIX D

WML 2.0 Specification

The Wireless Markup Language specification describes an extension of XHTML targeted toward mobile devices. This appendix contains the official WAP Forum™ specification for WML 2.0.

Wireless Markup Language Version 2.0

Version 11-Sep-2001

Wireless Application Protocol

WAP-238-WML-20010911-a

A list of errata and updates to this document is available from the WAP Forum™ Web site, `www.wapforum.org/`, in the form of SIN documents, which are subject to revision or removal without notice.

© 2001, Wireless Application Protocol Forum, Ltd.

All rights reserved

© 2001, Wireless Application Protocol Forum, Ltd. All rights reserved.

Terms and conditions of use are available from the WAP Forum™ Web site at `www.wapforum.org/what/copyright.htm`.

You may use this document or any part of the document for internal or educational purposes only, provided you do not modify, edit or take out of context the information in this document in any manner. You may not use this document in any other manner without the prior written permission of the WAP Forum™. The WAP Forum authorises you to copy this document, provided that you retain all copyright and other

www.it-ebooks.info

proprietary notices contained in the original materials on any copies of the materials and that you comply strictly with these terms. This copyright permission does not constitute an endorsement of the products or services offered by you.

The WAP Forum™ assumes no responsibility for errors or omissions in this document. In no event shall the WAP Forum be liable for any special, indirect or consequential damages or any damages whatsoever arising out of or in connection with the use of this information.

WAP Forum™ members have agreed to use reasonable endeavors to disclose in a timely manner to the WAP Forum the existence of all intellectual property rights (IPR's) essential to the present document. The members do not have an obligation to conduct IPR searches. This information is publicly available to members and non-members of the WAP Forum and may be found on the "WAP IPR Declarations" list at `www.wapforum.org/what/ipr.htm`. Essential IPR is available for license on the basis set out in the schedule to the WAP Forum Application Form.

No representations or warranties (whether express or implied) are made by the WAP Forum™ or any WAP Forum member or its affiliates regarding any of the IPR's represented on this list, including but not limited to the accuracy, completeness, validity or relevance of the information or whether or not such rights are essential or non-essential.

This document is available online in PDF format at `www.wapforum.org/`.

Known problems associated with this document are published at `www.wapforum.org/`.

Comments regarding this document can be submitted to the WAP Forum™ in the manner published at `www.wapforum.org/`.

Document History

WAP-238-WML-20010911-a Current

WAP-238_100-WML-20010911-a SIN

Contents

1. SCOPE 791
2. REFERENCES 792
2.1. NORMATIVE REFERENCES 792
2.2. INFORMATIVE RREREFERENCES 793
3. TERMINOLOGY AND CONVENTIONS 794
3.1. CONVENTIONS 794
3.2. DEFINITIONS 795
3.3. ABBREVIATIONS 796

www.it-ebooks.info

4. INTRODUCTION (INFORMATIVE) 796
4.1. BACKGROUND 796
4.2. THE XHTML MOBILE PROFILE DOCUMENT TYPE 797
4.3. THE WML2 DOCUMENT TYPE 797
4.4. CONFORMANCE GUIDELINES 798
5. USER AGENT BEHAVIOUR (NORMATIVE) 799
5.1. USER AGENT BEHAVIOUR AND FEATURES IN WAE 799
5.2. USER AGENT CONTEXT 799
5.2.1. Variables 799
5.2.2. Navigation History 799
5.3. NAVIGATION REFERENCE PROCESSING MODEL 800
5.3.1. The Go Task 801
5.3.2. The Prev Task 802
5.3.3. The Noop Task 803
5.3.4. The Refresh Task 803
5.3.5. Task Execution Failure 803
5.4. FORM PROCESSING REFERENCE MODEL 804
5.4.1. Overview of Form Processing Model 804
5.4.2. Form Initialisation 805
5.4.3. Form Interaction 812
5.4.4. Committing Form Data 815
5.4.5. Form Submission 815
5.5. ATTRIBUTE EXPRESSION SYNTAX 815
5.5.1. Attribute Expression Syntax Processing 816
5.6. WML2 EVENT MODEL 821
5.6.1. WML Intrinsic Events 822
5.6.2. WML Extrinsic Events 822
5.6.3. Event Bindings 822
5.7. IDENTIFICATION OF DOCUMENT TYPES 826
5.8. COMMON USER AGENT BEHAVIOUR DEPENDING ON THE TYPE OF ELEMENTS 827
5.8.1. Activation of Elements using Access Keys 827
5.9. THE BACK KEY IN WML2 827
5.10. NAVIGATION USER INTERFACE USING THE `WML:DO` ELEMENT 827

www.it-ebooks.info

5.10.1. Processing the `type` Attribute 828
5.10.2. Overriding the BACK Key Using the `wml:do` Element 829
5.11. TIMER PROCESSING 829
5.12. ACCEPTANCE OF XHTML 830
5.13. USER AGENT CONFORMANCE RULES 830
6. WML2 MARKUP ELEMENTS AND ATTRIBUTES (NORMATIVE) 832
6.1. XHTML BASIC AND EXTENSIONS 832
6.2. THE STRUCTURE MODULE 832
6.2.1. The `body` Element 833
6.2.2. The `html` Element 833
6.2.3. The wml:card Element 833
6.3. TEXT MODULE 833
6.3.1. The `p` Element 833
6.4. HYPERTEXT MODULE 834
6.5. FORMS MODULE 834
6.5.1. The `select` Element 834
6.5.2. Text Control Format Attributes 834
6.5.3. Form Control Initialisation Attribute 837
6.5.4. Activation and Event Handler Attributes 837
6.6. TABLES MODULE 837
6.6.1. The `table` Element 838
6.7. LISTS MODULE 838
6.8. IMAGE MODULE 838
6.8.1. The `img` Element 839
6.9. METAINFORMATION MODULE 839
6.9.1. The `meta` Element 839
6.10. LINK MODULE 840
6.11. BASE MODULE 840
6.12. OBJECT MODULE 840
6.13. STYLE SHEET MODULE 840
6.14. PRESENTATION MODULE 841
6.15. EVENTS MODULE 841
6.15.1. The `wml:onevent` Element 841

www.it-ebooks.info

6.16. CONTEXT AND NAVIGATION MODULE 841
6.16.1. The wml:anchor Element 841
6.16.2. The wml:access Element 842
6.16.3. The wml:do Element 843
6.16.4. The wml:go Element 843
6.16.5. The wml:noop Element 846
6.16.6. The wml:prev Element 846
6.16.7. The wml:refresh Element 846
6.16.8. The wml:postfield Element 846
6.16.9. The wml:setvar Element 847
6.16.10. The wml:getvar Element 847
6.16.11. The wml:timer Element 848
6.17. DOCUMENT CONFORMANCE 848
7. USE OF STYLE SHEETS WITH WML2 (NORMATIVE) 849
7.1. ADDING STYLE TO WML2 DOCUMENTS 849
7.2. THE DEFAULT STYLE SHEET FOR WML2 850
APPENDIX A. THE DTD FOR WML2 (NORMATIVE) 850
APPENDIX B. THE WML2 DEFAULT STYLE SHEET (INFORMATIVE) 851
APPENDIX C. THE WML2 ELEMENTS (INFORMATIVE) 851
APPENDIX D. STATIC CONFORMANCE REQUIREMENTS (NORMATIVE) 859
APPENDIX E. CHANGE HISTORY (INFORMATIVE) 864

1. Scope

Wireless Application Protocol (WAP) is a result of continuous work to define an industry-wide specification for developing applications that operate over wireless communication networks. The scope for the WAP Forum is to define a set of specifications to be used by service applications. The wireless market is growing very quickly and reaching new customers and services. To enable operators and manufacturers to meet the challenges in advanced services, differentiation, and fast/flexible service creation, WAP defines a set of protocols in transport, session, and application layers. For additional information on the WAP architecture, refer to [WAPARCH].

This specification defines the Wireless Markup Language version 2.0 (WML2), a language that extends XHTML with the unique semantics of WML1 [WML1].

www.it-ebooks.info

WML2 is used for backwards compatibility only. It is not intended for content authoring. WAP2 content is created with XHTML Mobile Profile [XHTMLMP].

The complete behaviour for a WAE user agent is not described in this specification, only the WML2 specific behaviour. In order to build a WAE compliant user agent, this specification must be read in conjunction with the Wireless Application Environment Specification [WAE].

2. References

2.1. Normative References

[CREQ] "Specification of WAP Conformance Requirements", WAP Forum™ , WAP-221-CREQ-20000915-a.

URL: http://www.wapforum.org/

[HTML4] "HTML 4.01 Specification", W3C Recommendation 24 December 1999.

URL: http://www.w3.org/TR/1999/REC-html401-19991224, D. Raggett et al., December 1999.

[RFC2045] "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", N. Freed, et al., November 1996.

URL: http://www.ietf.org/rfc/rfc2045.txt

[RFC2119] "Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997.

URL: http://www.ietf.org/rfc/rfc2119.txt

[RFC2616] "Hypertext Transfer Protocol - HTTP/1.1", R. Fielding et al., June 1999.

URL: http://www.ietf.org/rfc/rfc2616.txt

[RFC2388] "Returning Values from Forms: multipart/form-data" L. Masinter, August 1998.

URL: http://www.ietf.org/rfc/rfc2388.txt

[RFC2396] "Uniform Resource Identifiers (URI): Generic Syntax", T. Berners-Lee et al., August 1998.

URL: http://www.ietf.org/rfc/rfc2396.txt

[RFC3066] "Tags for the Identification of Languages", H. Alvestrand, January 2001.

URL: http://www.ietf.org/rfc/rfc3066.txt

www.it-ebooks.info

[UNICODE] "The Unicode Standard: Version 2.0", The Unicode Consortium, Addison-Wesley Developers Press, 1996.

URL: http://www.unicode.org/

[WCSS] "WAP CSS Specification", WAP Forum⇔, WAP-239-WCSS.

URL: http//www.wapforum.org/

[XHTMLBasic] "XHTML™ Basic 1.0", W3C Recommendation 19 December 2000, M. Ishikawa et al.

URL: http://www.w3.org/TR/2000/REC-xhtml-basic-20001219

[XHTMLMod] "Modularization of XHTML™ ", W3C Recommendation 10 April 2001, M. Altheim et al.

URL: http://www.w3.org/TR/2001/REC-xhtml-modularization-20010410

[XHTMLMP] "XHTML Mobile Profile", WAP Forum⇔. WAP-277-XHTMLMP.

URL: http://www.wapforum.org/

[XML] "Extensible Markup Language (XML) 1.0 (Second Edition) ",

W3C Recommendation 6 October 2000, T. Bray et al.

URL: http://www.w3.org/TR/2000/REC-xml-20001006

[XMLN] "Namespaces in XML", 14 January 1999, W3C Recommendation 14 January 1999, T. Bray et al.

URL: http://www.w3.org/TR/1999/REC-xml-names-19990114

2.2. Informative References

[CACHE] "WAP Caching Model Specification", WAP Forum⇔, WAP-120-CachingMod-19990211-a.

URL: http://www.wapforum.org/

[CSSMP] "CSS Mobile Profile 1.0", W3C Working Draft, 29 January 2001, T. Wugofski et al.

URL: http://www.w3.org/TR/2001/WD-css-mobile-20010129

[CSS2] "Cascading Style Sheets, level 2", B. Bos et. al., 12 May 1998.

URL: http://www.w3.org/TR/1998/REC-CSS2-19980512.

www.it-ebooks.info

[DOM2EVENT] "Document Object Model (DOM) Level 2 Events Specification Version 1", W3C

Recommendation 13 November 2000, T. Pixley.

URL: http://www.w3.org/TR/2000/REC-DOM-Level-2-Events-20001113

[ISO8879] "Information Processing - Text and Office Systems - Standard Generalised Markup Language (SGML) ", ISO 8879:1986.

[PICTO] "WAP Pictogram", WAP Forum™ , WAP-213-WAPInterPic.

URL: http://www.wapforum.org/

[WAE] "Wireless Application Environment Specification", WAP Forum™ , WAP-236-WAE.

URL: http://www.wapforum.org/

[WAPARCH] "WAP Architecture", WAP Forum™ , WAP-210-WAPArch-20010712-a.

URL: http://www.wapforum.org/URL:http//www.wapforum.org/

[WML1] "Wireless Markup Language Version 1.3", WAP Forum™ , WAP-191-WML-20000219-a.

URL: http://www.wapforum.org/

[WMLScript] "WMLScript Language Specification" , WAP Forum™ , WAP-193-WMLScript-20000324-a.

URL: http://www.wapforum.org/

[WTA] "Wireless Telephony Application Specification ", WAP Forum™ , WAP-266-WTA.

URL: http://www.wapforum.org/

[XHTML] "XHTML™ 1.1 - Module-based XHTML", W3C Recommendation 31 May 2001, S. Pemberton et al.

URL: http://www.w3.org/TR/2001/REC-xhtml11-20010531

3. Terminology and Conventions

3.1. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

www.it-ebooks.info

All sections and appendixes, except “Scope” and “Introduction”, are normative, unless they are explicitly indicated to be informative.

3.2. Definitions

Author - an author is a person or program that writes or generates WML, or other content.

Card - a single WML unit of navigation and user interface. May contain information to present to the user, instructions for gathering user input, etc.

Character Encoding – when used as a verb, character encoding refers to conversion between sequence of characters and a sequence of bytes. When used as a noun, character encoding refers to a method for converting a sequence of bytes to a sequence of characters. Typically, WML document character encoding is captured in transport headers attributes (e.g., Content-Type’s “charset” parameter) or the XML declaration defined by [XML].

Client - a device (or application) that initiates a request for connection with a server.

Content - subject matter (data) stored or generated at an origin server. Content is typically displayed or interpreted by a user agent in response to a user request.

Device - a network entity that is capable of sending and receiving packets of information and has a unique device address. A device can act as both a client and a server within a given context or across multiple contexts. For example, a device can service a number of clients (as a server) while being a client to another server.

Man-Machine Interface - a synonym for user interface.

Resource - a network data object or service that can be identified by a URL. Resources may be available in multiple representations (e.g., multiple languages, data formats, size and resolutions) or vary in other ways.

Server - a device (or application) that passively waits for connection requests from one or more clients. Once a connection is established, it is used to deliver a Resource to the client.

SGML - the Standard Generalised Markup Language (defined in [ISO8879]) is a general-purpose language for domainspecific markup languages.

User - a user is a person who interacts with a user agent to view, hear or otherwise use a resource.

User Agent - a user agent is any software or device that interprets WML, WMLScript, WTAI or other resources. This may include textual browsers, voice browsers, search engines, etc.

www.it-ebooks.info

XML - the Extensible Markup Language is a World Wide Web Consortium (W3C) standard for Internet markup languages, of which WML is one such language. XML is a restricted subset of SGML.

3.3. Abbreviations

CSS	Cascading Style Sheet
DTD	Document Type Definition
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IDL	Interface Definition Language
RFC	Request For Comments
SGML	Standard Generalised Markup Language
UI	User Interface
URI	Uniform Resource Identifier
W3C	World Wide Web Consortium
WAE	Wireless Application Environment
WAP	Wireless Application Protocol
WML	Wireless Markup Language
WSP	Wireless Session Protocol
XHTML	Extensible HyperText Markup Language
XML	Extensible Markup Language

4. Introduction (Informative)

This specification describes the processing of the WML2 document type by the WAE user agent. This section describes the WML2 document type, and its relationship to the XHTML Mobile Profile [XHTMLMP] document type.

4.1. Background

The primary requirements for the WAP2 service/application authoring environment are twofold:

- ✦ Backward compatibility with WAP 1 services
- ✦ Convergence with existing and evolving Internet standards, in particular the W3C specifications XHTML Basic and CSS Mobile Profile

www.it-ebooks.info

The goal of WML2 is therefore to create a document type that extends the syntax and semantics of XHTML Basic [XHTMLBasic] and CSS Mobile Profile (convergence) with the unique semantics of WML1 (backward compatibility).

The design guidelines for WML2 were:

- ✦ Defer to XHTML in the case of duplicated semantics (elements, attributes, and attribute values).
- ✦ Remove WML elements, attributes, and attribute values when they can be expressed in XHTML and CSS.
- ✦ Include WML1 elements and attributes when WML1 features cannot be expressed in XHTML and CSS. These elements are included using the WML namespace, identified by the "wml:" prefix.
- ✦ Deprecate elements and attributes where the W3C has also deprecated them.

The result is a document type with an XHTML core and WAP extensions. The core document type, known as XHTML Mobile Profile, can be used to author content convergent with W3C specifications. The WML2 document type (XHTML core plus WAP extensions) can be used to deliver WML1 content to WAP 2 clients, achieving backward compatibility with WML1 in a manner that is transparent to the end user. The structure and relationship of the document types allows an efficient implementation of a user agent that supports both types.

4.2. The XHTML Mobile Profile Document Type

The XHTML Mobile Profile document type is a strict superset of the XHTML Basic document type. XHTML Mobile Profile is used as an authoring language for WAP 2 services/applications. This document type achieves convergence with XHTML for WAP 2 services/applications.

The XHTML Mobile Profile document type is defined by [XHTMLMP].

The WML2 document type extends the XHTML Mobile Profile document type. The WML2 document type is detailed in section 4.3.

4.3. The WML2 Document Type

The WML2 document type is a strict superset of the XHTML Mobile Profile document type. It adds extensions to XHTML Mobile Profile for WML1 compatibility, creating a document type that is used for achieving backward compatibility with services/applications written using WML1.

The WML2 document type defines the following elements and attributes.

In particular, it includes the WAP extension elements, those elements brought from WML1 because the equivalent capabilities are not found in XHTML:

www.it-ebooks.info

Element	
wml:access	wml:onevent
wml:anchor	wml:postfield
wml:card	wml:prev
wml:do	wml:refresh
wml:getvar	wml:setvar
wml:go	wml:timer
wml:noop	

The WML2 document type also adds to to the XHTML elements specified the following attributes:

Element	Attributes Added
body	wml:onenterbackward, wml:onenterforward, wml:ontimer, wml:newcontext
html	wml:onenterforward, wml:onenterbackward, wml:ontimer, wml:use-xml-fragments
img	wml:localsrc
input	wml:emptyok, wml:format, wml:name
meta	wml:forua
option	wml:onpick
p	wml:mode
select	wml:iname, wml:ivalue, wml:name, wml:value
textarea	wml:emptyok, wml:format, wml:name

The WML2 document type includes the `u` element for backward compatibility.

4.4 Conformance Guidelines

The specification adopts existing W3C specifications [XHTMLBasic] and [XHTMLMod], but does not itself specify conformance requirements for [XHTML].

Conformance to this specification is defined in the Static Conformance Requirements (SCR) table in Appendix D.

* MERGEFORMAT * MERGEFORMAT * MERGEFORMAT * MERGEFORMAT * MERGEFORMAT *

MERGEFORMAT * MERGEFORMAT * MERGEFORMAT * MERGEFORMAT

www.it-ebooks.info

5. User Agent Behaviour (Normative)

5.1. User Agent Behaviour and Features in WAE

The specification covers the elements of the language (see section 6), the use of style with WML (see section 7) and other aspects of user agent behaviour necessary to implement WML as part of the Wireless Application Environment.

However WML2 depends on user agent behaviour and features described outside of this specification. To fully understand, implement and use WML2, this specification must be considered in conjunction with the Wireless Application Environment specification [WAE] and other specifications referenced in both this specification and [WAE].

5.2. User Agent Context

The *user agent context* consists of variables and navigation history.

5.2.1. Variables

A *WML variable* is a name-value pair, where the value is any string in the document character set.

This section describes WML variables in the user agent context. The user agent MUST implement WML variables.

WML variables can be used in the place of strings and are substituted at run-time with their current value.

A variable is said to be *set* if it has a value not equal to the empty string. A variable is *not set* if it has a value equal to the empty string, or is undefined in the current user agent context.

5.2.2. Navigation History

The *navigation history* is a logical stack of the resources in the navigational path the user traversed to arrive at the current resource.

This section describes the navigation history maintained by the user agent. The user agent MUST implement the navigation history model and support all the operations on it defined below.

WML2 includes a simple navigational history model that allows the author to manage backward navigation in a convenient and efficient manner. The user agent history is modelled as a stack of entries that represent the resources in the navigational path the user traversed to arrive at the current location. The stack is configured temporally, such that the newest entry is at the top of the stack and the oldest entry is at the bottom of the stack. Three operations may be performed on the history stack:

www.it-ebooks.info

- ✦ Reset - the history stack may be reset to a state where it only contains the current entry. See the `wml:newcontext` attribute(section 6.2.1) and `newcontext` attribute (section 6.2.3) for more information.
- ✦ Push - a new entry is pushed onto the history stack as an effect of forward navigation.
- ✦ Pop - the current entry (top of the stack) is popped as a result of backward navigation.

As each body/card is accessed via an explicitly specifiedURI, (e.g., via the `href` attribute in `wml:go` element,) an entry for the body/card is added to the history stack even if it is identical to the most recent entry. At a minimum, each entry must record the resource request information that comprises the absolute URI of the body/card, the method (get or post) used to access the body/card, the value of any postfields, any XHTML form data name-value pairs, and any request headers. Document content is not stored in the history. Variable references are never stored in the history. Any variable references in the resource request information must be replaced with the current value of the variables before the entry is added to the stack.

The user agent MUST return the user to the previous body/card specified in the history if a prev task is executed (see section 5.3.2). The execution of prev task pops the current entry from the history stack.

If the body/card is part of a document that was originally fetched using an HTTP post method, and the user agent MUST re-fetch the document to establish the body/card, then the user agent MUST reissue any post data associated with fetching the original document. The post data values must be the same values used to fetch the original document.

Refer to section 5.3.2 for more information on the semantics of the prev task. See [CACHE] for more information on caching semantics.

5.3. Navigation Reference Processing Model

A *task* is a navigational directive executed by the user agent in response to an event such as the activation of a `wml:do` element or the expiration of a timer. The *navigation reference processing model* defines the navigational tasks and the operations those tasks perform on the user agent context.

The user agent MUST implement all the tasks defined in this section. In addition, the user agent MUST implement the task failure behaviour.

The following tasks are defined in this section:

- ✦ The go task
- ✦ The prev task

www.it-ebooks.info

- ✦ The noop task
- ✦ The refresh task

5.3.1. The Go Task

The user agent MUST perform the following steps to execute a go task:

1. If the originating task contains `wml:setvar` elements, the variable name and value in each `wml:setvar` element is converted into a simple string by substituting all referenced variables. The resulting collection of variable names and values is stored in temporary memory for later processing. See section 5.5.1.1 for more information on variable substitution.
2. The target URI is identified and fetched by the user agent. The `href` attribute value is first converted into a simple string by substituting all referenced variables.
3. The access control parameters for the fetched document are processed as specified in section 6.16.
4. The destination is located using the fragment identifier specified in the URI.
    a. If a fragment identifier is not specified as part of the URI, the first body/card in the document is the destination.
    b. If the `wml:use-xml-fragments` attribute on the `html` element is `true` (default state)
        i. If a fragment identifier is specified as part of the URI and the document contains a matching element (`id` attribute of the element is identical to the fragment identifier), the enclosing body/card is the destination. If the matching element is a `body` or `wml:card` that body/card is the destination. If the destination is within the current body/card, the user agent should ensure the matching element is visible to the user.
        ii. If a matching element cannot be found, the first segment in the document is the destination.
    c. If the `wml:use-xml-fragments` attribute on the `html` element is `false`
        i. If a fragment identifier is specified as part of the URI and the document contains a matching body/card (id attribute of the body/card is identical to the fragment identifier), that body/card is the destination.
        ii. If a matching body/card cannot be found, the first body/card in the document is the destination.
5. The variable assignments resulting from the processing done in step #1 (the `wml:setvar` element) are applied to the current user agent context.
6. If the destination contains a `newcontext` or `wml:newcontext` attribute, a new user agent context is created as described in sections 6.2.1 and 6.2.3.

www.it-ebooks.info

7. Create a stack entry. If the destination contains any forms, initialise each form according to section 5.4.2 Form Initialisation, then store all locally-scoped form control variables in the entry.
8. An entry referring to the destination is pushed onto the history stack.
9. If the destination specifies an enterforward intrinsic event binding, the task associated with the event binding is executed and processing stops. See section 5.6.1 for more information.
10. If the destination contains a `wml:timer` element, the timer is started as specified in section 6.16.11.
11. The destination is displayed using the current variable state and processing stops.

5.3.2. The Prev Task

The user agent MUST perform the following steps to execute a prev task:

1. If the originating task contains `wml:setvar` elements, the variable name and value in each `wml:setvar` element is converted into a simple string by substituting all referenced variables. The resulting collection of variable names and values is stored in temporary memory for later processing. See section 5.5.1.1 for more information on variable substitution.
2. The history stack is popped. The target URI is the top of the history stack. The target URI is identified and fetched by the user agent. The processing that occurs if there is no previous entry in the history stack is implementation dependent.
3. The destination is located using the fragment identifier specified in the URI.
    a. If a fragment identifier is not specified as part of the URI, the first body/card in the document is the destination.
    b. If the `wml:use-xml-fragments` attribute on the `html` element is `true`
        i. If a fragment identifier is specified as part of the URI and the document contains a matching element (`id` attribute of the element is identical to the fragment identifier), the enclosing body/card is the destination. If the matching element is a `body` or `wml:card` , that body/card is the destination. If the destination is within the current body/card, the user agent should ensure the matching element is visible to the user.
        ii. If a matching element cannot be found, the first body/card in the document is the destination.
    c. If the `wml:use-xml-fragments` attribute on the `html` element is `false`
        i. If a fragment identifier is specified as part of the URI and the document contains a matching body/card (`id` attribute of the body/card is identical to the fragment identifier), that body/card is the destination.
        ii. If a matching body/card cannot be found, the first body/card in the document is the destination.

www.it-ebooks.info

4. The variable assignments resulting from the processing done in step #1 (the `wml:setvar` element) are applied to the current user agent context.
5. If the destination contains any locally scoped form data, it is refreshed with the form data stored in the entry at the top of the history stack.
6. If the destination specifies an enterbackward intrinsic event binding, the task associated with the event binding is executed and processing stops. See section 5.6.1 for more information.
7. If the destination contains a `wml:timer` element, the timer is started as specified in section 6.16.
8. The destination is displayed using the current variable state and processing stops.

5.3.3. The Noop Task

No processing is done for a noop task.

5.3.4. The Refresh Task

The user agent MUST perform the following steps to execute a refresh task:

1. For each `wml:setvar` element, the variable name and value in each `wml:setvar` element is converted into a simple string by substituting all referenced variables. See section 5.5.1.1 for more information on variable substitution.
2. The variable assignments resulting from the processing done in step #1 (the `wml:setvar` element) are applied to the current user agent context.
3. If the destination contains any locally scoped form data, it is refreshed with the form data stored in the entry at the top of the history stack.
4. If the current body/card contains a `wml:timer` element, the timer is started as specified in section 6.16.11.
5. The current body/card is re-displayed using the current variable state and processing stops.

User-visible effects of the state changes (e.g., a change in the screen display resulting from the changes of referenced variables) SHALL occur during the processing of the refresh task. A refresh and its user-visible effects SHALL occur even if the elements have no `wml:setvar` elements, given that context may change by other means (e.g., timer).

5.3.5. Task Execution Failure

If a task fails to fetch its target resource or the access control restrictions prevent a successful inter-body/card transition, the user agent MUST notify the user and take the following actions:

- ✦ The invoking body/card remains the current body/card.

www.it-ebooks.info

- ✦ No changes are made to the browser context, including any pending variable assignments or newcontext processing.
- ✦ No intrinsic event bindings are executed.

5.4. Form Processing Reference Model

A *form* is a part of a WML2 document consisting of a container element enclosing structured content and one or more of the elements `input`, `select`, `textarea`. These elements are called form control elements. Form control elements may be placed within a `form` element, or they may be placed directly within a `wml:card` element.

Throughout this specification, unless stated otherwise, *form* refers to either a `form` element containing form control elements, or a `wml:card` element containing form control elements.

This section describes the processing of forms.

The user agent MUST implement the processing of forms.

5.4.1. Overview of Form Processing Model

The form control elements are those from the Basic Forms Module of XHTML. To allow authors to group logically related content and controls, the `optgroup` and `fieldset` elements from the XHTML Forms Module (a superset of Basic Forms) are also included.

The WML2 Form Processing Model describes the life cycle of a form. The stages are:

1. Initialisation
   (See 5.4.2 for how the user agent must process a form before displaying a body/card containing that form.)
2. Interaction
   (See 5.4.3 for how the user agent must handle user interactions with the form.)
3. Committing Data
   (See 5.4.4 for how the user agent must commit the form data to the user agent context before leaving a form.)
4. Submission
   (See 5.4.5 for how the user agent must submit a form.)

In WML2, each form control is associated with a *form control variable.* A form control variable is a WML variable. The control's value is stored in this variable for later processing.

For compatibility with XHTML forms, form control variables have scope. The declaration of the form control variable determines its scope. There are two types of scope: global and local (form scope). Locally scoped variables are only visible within the form in which they are declared. Globally scoped variables are visible throughout the user agent context. See section 5.5.1.4 for details on variable scoping.

www.it-ebooks.info

5.4.2. Form Initialisation

The form initialisation phase of the WML2 Form Processing Model determines the initial values of the form's controls, that is, the values they have upon initial display of the document containing the form. After the controls' initial values are determined, they are committed to form control variables. These initial values are also saved for use during form reset.

Prior to displaying a `wml:card` or a `body` element containing a form, the user agent MUST initialise each form in the following manner:

- The user agent MUST initialise each control within the form according to section 5.4.2.1, Form Control Initialisation.
- The user agent MUST process the controls in the order in which they appear in the form container element.
- The user agent MUST store the initial value of every control, for use during form reset.

The user agent MUST perform form initialisation when a body or card is reached through either forward or backward navigation or refresh. The user agent MUST perform form initialisation at the appropriate step in the reference processing model. See section 5.3 for details.

5.4.2.1. Form Control Initialisation

The user agent initialises each form control within the form as it prepares the form for display. Initialisation of a form control means determining its initial value, and storing that value in the form control variable.

Prior to form display, the user agent MUST initialise each form control in the following manner:

- The user agent MUST process text input controls according to section 5.4.2.2. Text controls are `input` elements with `type="text"` and `type="password"` and `textarea` elements.
- The user agent MUST process menu controls according to section 5.4.2.3. Menu controls are `select` elements together with their `option` children.
- The user agent MUST process checkbox controls according to section 5.4.2.4. Checkbox controls are `input` elements with `type="checkbox"`.
- The user agent MUST process radio button controls according to section 5.4.2.5. Radio button controls are `input` elements with `type="radio"`.
- The user agent MUST process submit button controls according to section 5.4.2.6. Submit button controls are `input` elements with `type="submit"`.
- The user agent MUST process reset button controls according to section 5.4.2.7. Reset button controls are `input` elements with `type="reset"`.
- The user agent MUST process hidden controls according to section 5.4.2.8. Hidden controls are `input` elements with `type="hidden"`.

www.it-ebooks.info

5.4.2.2. Text Input Control Initialisation

The user agent MUST initialise a text input control in the following manner:

- If the `wml:name` attribute is assigned, and the control is an `input` element, set the initial value of the control as follows:
  1. The `wml:name` attribute names the form control variable. If the value of the form control variable conforms to the input mask, as defined by the `wml:format` and `wml:emptyok` attributes or their CSS equivalents, the initial value of the control is the value of the control variable. See section 6.5.2 for definition of input mask and details of `wml:format` and `wml:emptyok` attributes.
  2. If the value of the form control variable does not conform to the input mask, it is ignored and the initial value of the control is determined by the `value` attribute:
     a. If the value of the `value` attribute conforms to the input mask, the initial value of the control is that attribute value.
     b. If the value of the `value` attribute does not conform to the input mask, the `value` attribute is ignored and the initial value is the empty string.
     c. If the `value` attribute is not specified, the initial value of the control is the empty string.
  3. Set the control variable equal to the initial value of the control.
  4. Pre-load the initial value into the form control.
- If the `wml:name` attribute is assigned, and the control is a `textarea` element, set the initial value of the control as follows:
  1. The `wml:name` attribute names the form control variable. If the value of the form control variable conforms to the input mask, as defined by the `wml:format` and `wml:emptyok` attributes or their CSS equivalents, the initial value of the control is the value of the control variable.
  2. If the value of the form control variable does not conform to the input mask, it is ignored and the initial value of the control is determined by the `textarea` element:
     a. If the content of the `textarea` element conforms to the input mask, the initial value of the control is the content of the element.
     b. If the content of the `textarea` element does not conform to the input mask, the content of the element is ignored and the initial value is the empty string.
     c. If the `textarea` element is an empty element, the initial value of the control is the empty string.
  3. Set the control variable equal to the initial value of the control.
  4. Pre-load the initial value into the form control.

www.it-ebooks.info

- If the `name` attribute is assigned, and the control is an `input` element, set the initial value of the control as follows:

    1. The initial value of the control is determined by the `value` attribute and the input mask, as defined by the `wml:format` and `wml:emptyok` attributes or their CSS equivalents:

        a. If no input mask is specified, the initial value of the control is the value of the `value` attribute.
        If the `value` attribute is not specified, the initial value of the control is the empty string

        b. If the input mask is specified and the value of the `value` attribute conforms to the input mask, the initial value of the form control is that value.

        c. If the input mask is specified and the `value` of the value attribute does not conform to the input mask, the initial value of the control is the empty string.

    2. Set the control variable equal to the initial value. (The name of the control variable is the value of the `name` attribute.)

    3. Pre-load the initial value into the form control.

- If the `name` attribute is assigned, and the control is a `textarea` element, set the initial value of the control as follows:

    1. The initial value of the control is determined by the content of the `textarea` element and the input mask, as defined by the `wml:format` and `wml:emptyok` attributes or their CSS equivalents:

        a. If no input mask is specified, the initial value of the control is the content of the element. If the element is an empty element, the initial value of the control is the empty string.

        b. If the input mask is specified and the content of the element conforms to the input mask, the initial value of the control is the content of the element.

        c. If the input mask is specified and the content of the element does not conform to the input mask, the initial value of the control is the empty string.

    2. Set the control variable equal to the initial value. (The name of the control variable is the value of the `name` attribute.)

    3. Pre-load the initial value into the form control.

- If both the `wml:name` attribute and the `name` attribute are assigned, `wml:name` takes precedence, and the `name` attribute MUST be ignored.

- If neither the `wml:name` attribute nor the `name` attribute is assigned:

    1. The control does not have a corresponding form control variable. This control cannot be *successful* as defined by [HTML4].

www.it-ebooks.info

2. The initial value of the control is the value of the `value` attribute, except when the form control element is `textarea`. For `textarea`, the initial value is the content of the element. If the `value` attribute is not assigned, or the form control has no content, then the initial value is the empty string.
3. Pre-load the initial value into the form control.

5.4.2.3. Menu Control Initialisation

The pre-selection of option elements includes an operation named *validate*. This operates on a value, and determines if that value is a legal option index (or indices for a multiple-selection menu). The operation consists of the following steps:

1. Remove all non-integer indices from the value.
2. Remove all out-of-range indices from the value, where out-of-range is defined as any index with a value greater than the number of options in the menu or with a value less than one.
3. Remove duplicate indices.

Note that an invalid index will result in an *empty* value.

The user agent MUST initialise a menu control in the following manner:

- ✦ If any of the `wml:name`, `wml:iname`, `wml:value` or `wml:ivalue` attributes is assigned, the `selected` attribute of the `option` element is ignored. Set the initial value of the control as follows:
- ✦ If the control only permits a single selection:
    1. Determine the default option index:
        a. If the `wml:iname` attribute is specified, it names the variable to use to determine the default option index. The default index is the value of named variable.
        b. If the default option index has not yet been determined and the `wml:ivalue` attribute is specified, the default option index is the value of the `wml:ivalue` attribute.
        c. If the default option index has not yet been determined and the `wml:name` attribute is specified, it names the variable to use to determine the default option index. The default index is the index of the option whose `value` attribute equals the value of the named variable.
        d. If the default option index has not yet been determined and the `wml:value` attribute is specified, the default option index is the index of the option whose `value` attribute equals the value of the `wml:value` attribute.
        e. If no option has been pre-selected, the default option index is one.

www.it-ebooks.info

2. Set the control variable(s):

   a. If the `wml:iname` attribute is specified, set the named variable with the default option index.

   b. If the `wml:name` attribute is specified, set the named variable with the value of the `value` attribute on the `option` element at the default option index. If the `value` attribute is not specified, set the variable to the empty string.

3. Pre-select the option at the default option index.

✦ If the control permits multiple selections:

1. Determine the default option indices:

   a. If the `wml:iname` attribute is specified, it names the variable used to determine the default option indices. The value of the named variable is a semicolon-delimited list of the indices of the pre-selected `option` elements.

   b. If the default option indices have not yet been determined and the `wml:ivalue` attribute is specified, its value is used to determine the default option indices. The attribute value is a semicolon-delimited list of the indices of the pre-selected `option` elements.

   c. If the default option indices have not yet been determined and the `wml:name` attribute is specified, it names the variable used to determine the default option indices. The value of the named variable is a semicolon-delimited list of the values of the pre-selected `option` elements' `value` attributes. The default indices are the corresponding indices.

   d. If the default option indices have not yet been determined and the `wml:value` attribute is specified, its value is used to determine the default option indices. The attribute value is a semicolon-delimited list of the values of the pre-selected `option` elements' `value` attributes.

      The default indices are the corresponding indices.

2. Set the control variable(s):

   a. If the `wml:iname` attribute is specified, set the named variable with the default option indices.
   If there are no default options, set the variable to the empty string.

   b. If the `wml:name` attribute is specified, set the named variable to a semicolon-delimited list of the following values: for each index in the default indices greater than zero, the value of the `value` attribute on the `option` element at the index. If there are no default options, or if all selected `option` elements contain an empty `value` attribute, set the variable to the empty string.

www.it-ebooks.info

3. Pre-select options specified by the default option indices:

   a. Deselect all options.

   b. For each index greater than zero, select the option specified by the index.

- If the `name` attribute is assigned, it names the control variable. Set the initial value of the control as follows:
- If the control only permits a single selection:

  1. If exactly one `option` has the `selected` attribute assigned, that option is the pre-selected option. If more than one `option` has the `selected` attribute assigned, the first such option is the pre-selected option.
  2. Set the control variable equal to the value of the pre-selected `option` element's `value` attribute. If the `value` attribute is not specified, set the variable to the contents of the `option` element. If the element is empty, set the variable to the empty string.
  3. Select the menu item corresponding to the pre-selected option.

- If the control permits multiple selections:

  1. The pre-selected options are those option elements with their `selected` attribute assigned. If no `option` has the `selected` attribute assigned, there are no pre-selected options.
  2. Set the control variable equal to a string consisting of the semicolon-delimited list of the values of the preselected `option` elements' `value` attributes. If the `value` attribute of any pre-selected `option` is not specified, substitute the contents of the element, but ignore the option if the element is empty. If there are no pre-selected options, set the control variable to the empty string.
  3. Select the menu items corresponding to the pre-selected options.

- Assigning any of `wml:name`, `wml:iname`, `wml:value` or `wml:ivalue` takes precedence over `name`; the `name` attribute and `option` selections expressed via the `selected` attribute MUST be ignored whenever any of `wml:name`, `wml:iname`, `wml:value` or `wml:ivalue` is assigned.

5.4.2.4. Checkbox Control Initialisation

The user agent MUST initialise a checkbox control in the following manner:

- Initialise the control as specified for a multiple-selection select, where each checkbox control that shares the same control name is mapped to a subordinate `option` element. As part of that mapping, the `checked` attribute is mapped to the `option` element's `selected` attribute. See section 5.4.2.3.

www.it-ebooks.info

5.4.2.5. Radio Button Control Initialisation

The user agent MUST initialise a radio button control in the following manner:

- Initialise the control as specified for a single-selection select, where each radio button control that shares the same control name is mapped to a subordinate `option` element. As part of that mapping, the `checked` attribute is mapped to the `option` element's `selected` attribute. See section 5.4.2.3.

5.4.2.6. Submit Button Control Initialisation

The user agent MUST initialise a submit button control in the following manner:

- The name of the control variable is the value of the `name` attribute. Set the control variable equal to the `value` of the value attribute.

5.4.2.7. Reset Button Control Initialisation

The user agent MUST initialise a reset button control in the following manner:

- No initialisation is performed.

5.4.2.8. Hidden Control Initialisation

The user agent MUST initialise a hidden control in the following manner:

- If the `wml:name` attribute is assigned:
  1. The `wml:name` attribute names the control variable. The initial value of the control is the value of the control variable. If the control variable is not set, the initial value of the control is the value of the `value` attribute. If the `value` attribute is not specified, the initial value is the empty string.
  2. Set the control variable equal to the initial value of the control.
- If the `name` attribute is assigned:
  1. The name of the control variable is the value of the `name` attribute. The initial value of the control is the value of the `value` attribute. If the `value` attribute is not specified, the initial value is the empty string.
  2. Set the control variable equal to the initial value of the control.
- If both are assigned:
  1. The `wml:name` attribute takes precedence, and the `name` attribute MUST be ignored.
- If neither attribute is assigned:
  1. The control does not have a corresponding form control variable. The control cannot be *successful* as defined by [HTML4].
     a. The name of the control variable is the value of the `name` attribute. Set the control variable to the value of the `value` attribute.

www.it-ebooks.info

5.4.3. Form Interaction

The interaction phase of the WML2 Form Processing Model determines user agent behaviour as the user interacts with form controls within a form. The interaction phase updates the control's current value based on user interactions. If user interaction with a form control modifies its current value, the user agent performs an update to the value presented to the user as well as to the form control variable.

Once a form has been displayed, the user agent MUST handle user interaction with the individual controls within the form according to section 5.4.3.1, Form Control Interaction. The user agent MUST continue processing user interaction until the execution of the next task.

The user agent MAY re-display the current body or card when form control variables are updated, but authors must not rely on this behaviour.

5.4.3.1. Form Control Interaction

If user interaction with a form control modifies the control's current value, the user agent MUST handle that interaction as follows:

- The user agent MUST process text input controls according to section 5.4.3.2. Text controls are input elements with `type="text"` and `type="password"` and the `textarea` element.
- The user agent MUST process menu controls according to section 5.4.3.3. Menu controls are `select` elements together with their `option` children.
- The user agent MUST process checkbox controls according to section 5.4.3.4. Checkbox controls are `input` elements with `type="checkbox"`.
- The user agent MUST process radio button controls according to section 5.4.3.5. Radio button controls are `input` elements with `type="radio"`.
- The user agent MUST process submit button controls according to section 5.4.3.6. Submit button controls are `input` elements with `type= "submit"`.
- The user agent MUST process reset button controls according to section 5.4.3.7. Reset button controls are `input` elements with `type="reset"`.
- The user agent MUST process hidden controls according to section 5.4.3.8. Hidden controls are `input` elements with `type="hidden"`.

5.4.3.2. Text Input Control Interaction

The user agent MUST handle interaction with a text input control in the following manner:

- When the user attempts to commit input, the user agent MUST validate the input against the input mask, as defined by the `wml:format` and `wml:emptyok` attributes or their CSS equivalents:
  1. If no input mask is specified, the user agent MUST set the control variable to the control's current value.
  2. If the control's current value conforms to the input mask, the user agent MUST set the control variable to that value.

www.it-ebooks.info

3. If the control's current value does not conform to the input mask, the user agent MUST NOT commit that input and MUST notify the user that the input was rejected and allow the user to resubmit new input. In this case, the control variable MUST NOT be modified.

✦ The user agent MAY validate user input at any other appropriate time, e.g., when control focus is lost or with each character entered.

5.4.3.3. Menu Control Interaction

Interaction with a menu control is the selection or de-selection of an `option` element. The user agent MUST handle this interaction by updating the control variable(s) in the following manner:

✦ If the `wml:name` or `wml:iname` attributes were used to name the control variable(s):

1. If the control only permits a single selection:

   a. If the `wml:name` attribute is specified, set the named variable to the value of the `value` attribute on the selected `option` element. If the `value` attribute is not specified, set the variable to the empty string.

   b. If the `wml:iname` attribute is specified, set the named variable to the index of the selected option.

      Index numbering begins at one and increases monotonically.

2. If the control permits multiple selections:

   a. If the `wml:name` attribute is specified, set the named variable to a string consisting of the semicolondelimited list of the values of the selected `option` elements' `value` attributes. If any `value` attribute is not specified or is empty, ignore that option. If there are no selected options, or if all selected `option` elements contain an empty `value` attribute, set the control variable to the empty string.

   b. If the `wml:iname` attribute is specified, set the named variable to a string consisting of the semicolondelimited list of the indices of the selected `option` elements. If there are no selected options, set the control variable to the empty string.

✦ If the `name` attribute was used to name the control variable:

1. If the control only permits a single selection:

   a. Set the control variable equal to the value of the selected `option` element's `value` attribute. If the `value` attribute is not specified, set the variable to the contents of the selected `option` element, or to the empty string if the element is empty.

2. If the control permits multiple selections:

   a. Set the control variable equal to a string consisting of the semicolon-delimited list of the values of the selected `option` elements' `value` attributes. If the `value` attribute of any selected

www.it-ebooks.info

option is not specified, substitute the contents of the element, but ignore the option if the element is empty. If there are no selected options, set the control variable to the empty string.

Controls that permit multiple selections may result in a control variable with a value that is a semicolon-delimited list (e.g., "dog;cat"). This is not an ordered list and the user agent MAY construct the list in any order that is convenient.

Authors must not rely on a particular value ordering. The user agent MUST ensure that the wml:iname variable value contains no duplicate index values. The wml:name variable value MUST contain duplicate values in the situation where multiple selected option elements have the same value. The wml:name result MUST NOT contain empty values (e.g., "cat;;dog" is illegal).

However, in all cases, the user agent MUST NOT display user-visible effects as a result of updating wml:name and wml:iname variables, except when there is an explicit refresh task (see section 5.3). The user agent MUST update wml:name and wml:iname variables (if specified) for each select element in the card before each and every task invocation.

5.4.3.4. Checkbox Control Interaction

The user agent MUST handle interaction with a checkbox control in the following manner:

- ✦ Handle the control as specified for a multiple-selection select, where each checkbox control that shares the same control name is mapped to a subordinate option element. As part of that mapping, the checked attribute is mapped to the option element's selected attribute. See section 5.4.3.3.

5.4.3.5. Radio Button Control Interaction

The user agent MUST handle interaction with a radio button control in the following manner:

- ✦ Handle the control as specified for a single-selection select, where each radio button control that shares the same control name is mapped to a subordinate option element. As part of that mapping, the checked attribute is mapped to the option element's selected attribute. See section 5.4.3.3.

5.4.3.6. Submit Button Control Interaction

The user agent MUST handle interaction with a submit button control in the following manner:

- ✦ Submit the form as specified in section 5.4.5, Form Submission.

5.4.3.7. Reset Button Control Interaction

The user agent MUST handle interaction with a reset button control in the following manner:

www.it-ebooks.info

1. Set the current value of each control within the current form to its initial value, as in [HTML4]
2. Update each form control variable to reflect the new current value of the control.

5.4.3.8. Hidden Control Interaction

By definition, no interaction with hidden controls is possible.

5.4.4. Committing Form Data

Before navigation away from a body or card containing a form, the user agent MUST commit the current value of each form control to its form control variable, as specified in section 5.4.3, Form Interaction. This applies to both form submission and the execution of any task.

5.4.5. Form Submission

The form submission phase of the WML2 Form Processing Model determines how the form data set is submitted to the server for processing. This section describes submission of forms whose container element is the `form` element.

Submission of forms whose container element is the `wml:card` element is accomplished with a go task. See section 6.16.4 for more information. The functionality of the go task is a superset of XHTML form submission. To submit a form, the user agent must logically map form submission into the execution of a go task. It is not necessary for the user agent to physically construct a go task element.

The user agent MUST submit a form for processing according to the following steps:

1. Construct the form data set by executing the steps of [HTML4] section 17.13.3, Processing form data. The user agent MUST process all successful controls, including those bound to global variables.
2. Construct a go task. Map the `form` element's `action` attribute to the go task's `href` attribute. Map the `form` element's `method` and `enctype` attributes to the go task's `method` and `enctype` attributes, respectively.
3. For each name-value pair in the form data set, insert a `postfield` element as a child of the go task. Map the control name to the `postfield` element's `name` attribute and the control value to its `value` attribute.
4. Execute the go task.

5.5. Attribute Expression Syntax

The attribute expression syntax is used in parameterisation of the values of certain types of attributes, which are interpreted as expressions and evaluated at run-time by the user agent.

With the attribute expression syntax, all WML content can be parameterised, allowing the author a great deal of flexibility in creating documents with improved

www.it-ebooks.info

caching behaviour and better perceived interactivity. The values of certain types of attributes are interpreted as expressions and evaluated at run-time. The attribute expression syntax in conjunction with the `wml:getvar` element (see section 6.16.10) provides authors a simple yet powerful mechanism to parameterise content.

Only the values of attributes of type `CDATA` are interpreted as expressions. This includes attributes with the XHTML derived attribute types Character, Charset, Charsets, Color, ContentType, ContentTypes, Datetime, Length, MediaDesc, MultiLength, Number, Pixels, Script, Text, URI and URIs [XHTMLMod]. The result of evaluating an expression is always a string, in the character set of the document. The document that results from evaluating all expressions within a source document's `CDATA` attributes and processing all `wml:getvar` elements is called the *presentation document*.

For a document whose type is XHTML Mobile Profile, the user agent MUST process the document assuming no attribute expression syntax.

For a document whose type is WML2, the user agent MUST process the document according to the attribute expression syntax.

For identification of the document type, see section 5.7.

If the type of a document cannot be determined, the user agent MAY process the document assuming no attribute expression syntax. In this case, the user agent MAY display an appropriate warning message to the user.

5.5.1. Attribute Expression Syntax Processing

The values of variables can be substituted into certain attribute values in WML elements, using a variable reference.

When a variable reference is resolved, it results in the substitution of the variable's value. Substitution does not affect the current value of the variable and is defined as a string substitution operation. If an undefined variable is referenced, it results in the substitution of the empty string. The `wml:getvar` element can be used to substitute the values of variables into the text (`#PCDATA`) of a document (see section 6.16.10 for more information).

The source document is processed according to the variable expression syntax as follows:

For each CDATA attribute type defined in section 5.5, evaluate the value of the attribute as an expression in the attribute expression syntax. Each expression is evaluated from left to right. Evaluation of an expression consists of the following steps:

- ✦ Initially set the result string to the empty string.
- ✦ For each literal character, append it to the result string.

www.it-ebooks.info

- ✦ For each valid variable reference, append the value of the variable to the result string. Variable references are described in section 5.5.1.1.
- ✦ For each literal dollar sign escape sequence, append the '$' character to the result string. The dollar sign escape sequence is described in section 5.5.1.3.

These operations are described in the following sections.

5.5.1.1. Variable References

WML variable names consist of a US-ASCII letter or underscore followed by zero or more letters, digits or underscores. Any other characters are illegal. Variable names are case sensitive.

The following description of the variable reference syntax uses the Extended Backus-Naur Form (EBNF) notation established in [XML].

```
varref ::= ( "$" varname ) | ( "$(" varname ( conv )? ")" )
varname ::= ( "_" | alpha ) ( "_" | alpha | digit )*
conv ::= ":" ( "escape" | "noesc" | "unesc" )
alpha ::= [a-zA-Z]
digit ::= [0-9]
```

Variable references may use parentheses for clarity. Parentheses are required anywhere the end of a variable name cannot be inferred from the surrounding context.

For example:

```
This is a $(var).
```

In this example, a reference to the variable "direction" requires parentheses:

```
You should navigate $(direction)ward.
```

In the next example, the variable name "var" can be inferred, so parentheses are not needed:

```
Here's another reference to $var.
```

Other legal variable forms:

```
$_X $X32 $Test_9A
```

The value of variables can be converted into a different form as they are substituted. A conversion can be specified in the variable reference following the colon. Table 5-1 summarises the current conversions.

www.it-ebooks.info

Table 5-1
Conversions

	Conversion Effect
noesc	No change to the value of the variable.
escape	URI-escape the value of the variable.
unesc	URI-unescape the value of the variable.

The use of a conversion during variable substitution does not affect the actual value of the variable. URI-escaping is detailed in [RFC2396]. All lexically sensitive characters defined in WML must be escaped, including all characters not in the `unreserved` set specified by [RFC2396].

If no conversion is specified, the variable is substituted using the conversion format appropriate for the context. All attributes defined as type URI or URIs [XHTMLMod], default to `escape` conversion. For attributes with other types, no conversion is done. Specifying the `noesc` conversion disables context sensitive escaping of a variable.

For example:

```
This is an escaped $(var:escape).
This is an unescaped $(var:unesc).
```

5.5.1.2. Parsing the Variable Reference Syntax

The variable reference syntax (e.g., `$X`) is parsed after all XML parsing is complete. In XML terminology, variable substitution is done after the *XML processor* has parsed the document and provided the resulting parsed form to the *XML application*. In the context of this specification, the WML parser and user agent is the XML application.

This implies that all variable reference syntax is parsed *after* the XML constructs, such as tags and entities, have been parsed. In the context of variable parsing, all XML syntax has a higher precedence than the variable reference syntax, e.g., entity substitution occurs before the variable reference syntax is parsed. The following examples are identical references to the variable named X:

```
$X
&#x24;X
$&#x58;
&#36;&#x58;
```

5.5.1.3. The Dollar-sign Character

A side effect of the parsing rules is that the literal dollar sign must be encoded with a pair of dollar sign entities in any attribute values. A single dollar-sign entity, even specified as `$`, results in a variable substitution.

www.it-ebooks.info

In order to include a '$' character in a WML document, it must be explicitly escaped. This can be accomplished with the following syntax:

```
$$
```

Two dollar signs in a row are replaced with a single '$' character. For example:

```
This is a $$ character.
```

This would be evaluated as:

```
This is a $ character.
```

To include the '$' character in URI-escaped strings, specify it with the URI-escaped form:

```
%24
```

5.5.1.4. Variable Scoping

One way that variables are used in WML is as *form control variables*. A form control variable is a variable that stores the current value of a form control. For compatibility with XHTML forms, form control variables have scope. The declaration of the form control variable determines its scope.

There are two types of scope: global and local (form scope). Locally-scoped variables are only visible within the form in which they are declared. Globally-scoped variables are visible throughout the current document and in all documents processed in the current context.

5.5.1.4.1. Scoped Variable Declarations

Locally-scoped (form-scoped) variables are declared by use of the `name` attribute on a form control element. Globally-scoped variables are declared by use of the `wml:name` attribute on a form control element. A form can contain any mix of locally- and globally-scoped form control variables.

For example, the following form declares a locally-scoped variable `x` and a globally-scoped variable `y`:

```
<form action="...">
<p>
<input name="x" value="foo"/> <!-- locally-scoped var x -->
<input wml:name="y"/> <!-- globally-scoped var y -->
</p>
</form>
```

Form control elements may be placed outside of a `form` element. If such a control names a control variable with the `wml:name` attribute, a globally-scoped form control variable is declared. If such a form control names a control variable with the `name` attribute, the control does not map to a variable (for compatibility with XHTML form semantics).

www.it-ebooks.info

Note: Variables declared (set) with the `wml:setvar` element are always globally-scoped, regardless of the positioning of the `wml:setvar` element relative to a form element.

5.5.1.4.2. Scoped Variable References

Due to form control variable scoping, variable references are evaluated according to their position relative to the `form` element. Within a `form` element, any locally-scoped variable will hide a globally-scoped one with the same name. If none is present, a reference binds to the globally-scoped variable. Outside of a `form` element (either within a `body` or `wml:card` element), a reference always binds to a globally-scoped variable.

For example, assuming the global variable `x` is set:

```
<wml:card>
<wml:getvar name="x"/> <!-- refers to global variable x -->
<form action="...">
<p>
<input name="x"/>
<wml:getvar name="x"/> <!-- refers to local variable x -->
<wml:getvar name="y"/> <!-- reference not resolved -->
</p>
</form>
</wml:card>
```

If a variable reference names a form control variable, the user agent MUST evaluate the reference as follows:

- If the reference is located within a `form` element, the reference evaluates to the value of the locally-scoped variable with the specified name. If there is no such local variable set, the reference evaluates to the value of the global variable with the specified name. If there is no such global variable set, the reference evaluates to the empty string.
- If the reference is located outside of a `form` element, the reference evaluates to the value of the global variable with the specified name. If there is no such global variable set, the reference evaluates to the empty string.

5.5.1.5. Validation

Within the attribute expression syntax, any string following a single dollar sign ('$') must be treated as a variable reference and validated, unless it is part of an escaped literal dollar sign sequence according to section 5.5.1.3. Each reference must use proper variable name syntax, according to section 5.5.1.1. Specifying any conversion other than those specified in section 5.5.1.1 results in an invalid variable reference.

Values of attributes not evaluated as part of the attribute expression syntax must use an escaped literal dollar sign to prevent the creation of an otherwise valid variable reference.

www.it-ebooks.info

The document is in error if any variable reference uses invalid syntax or is placed in an invalid location.

An example of an invalid variable reference:

```
<!-- bad variable syntax -->
<img src="$?"/>
```

An example of escaped dollar sign in an attribute of type `CDATA`:

```
<!-- Dollar sign escaped in title attribute -->
<a href="/next" title="Win $$1,000,000!">
```

5.5.1.6. Event Parameters

For WML2, the variable reference syntax add the ability to reference ordered local event parameters, such as those used in [WTA], or event attributes as defined by IDL order in [DOM2EVENT]. An event parameter reference is a variable name consisting of one or more digits, such as `$1` or `$(2:esc)`.

All event parameters are empty except during the presentation evaluation of event listeners, such as tasks in a `wml:do` or `wml:onevent` element. The definition of event parameters is part of the event definition; by default, an event has no parameters.

For variable expressions which comply with the syntactic requirements of section 5.5.1.1 except that the variable name is a positive integer rather than an alphanumeric string, the WML Variable attribute expression language SHALL evaluate variable references as defined in section 5.5.1.1, except that the value of the variable SHALL be defined by the event whose binding the presentation renderer is currently evaluating. If an event defines no parameters, or the current presentation of the document is not evaluating an event listener, the value of the expression SHALL be the empty string.

5.5.1.7. Re-generation of Presentation Document

Before executing a task (see section 5.3) or processing an event binding (see section 5.6), the user agent MUST either re-generate the presentation document, or perform an operation that is indistinguishable from it. This ensures that the evaluation of all expressions reflects the most current user agent context.

5.6. WML2 Event Model

An *event* is a signal to the user agent that some action has taken place. Several WML elements are capable of generating events when the user interacts with them. In addition, the user agent generates events while executing tasks or upon timer expiration.

The following event types are defined in WML:

- *Intrinsic event*: an event generated by the user agent
- *Extrinsic event*: an event sent to the user agent by some external agent

www.it-ebooks.info

5.6.1. WML Intrinsic Events

Intrinsic events indicate state transitions inside the user agent.

Each event has a target element. The target element is the element within the document to which the event applies. For user interface elements, the target is the element itself. For events generated by the user agent, the target is the current `body` or `wml:card` element. The target element determines where event bindings should be placed.

WML defines the following intrinsic event types (Table 5-2).

Table 5-2
WML Intrinsic Events

Event Type	*Target Element*	*Description*
timer	body or wml:card	The timer event occurs when a timer expires. Timers are specified using the wml:timer element (see section 6.16).
enterforward	body or wml:card	The enterforward event occurs when the user agent enters a body/card via a go task or any method with identical semantics. This includes body/card entry caused by a script function or user-agent-specific mechanisms, such as a means to directly enter and navigate to a URI.
enterbackward	body or wml:card	The enterbackward event occurs when the user agent enters a body/card via a prev task or any method with identical semantics. In other words, the enterbackward event occurs when the user causes the user agent to navigate into a body/card by using a URI retrieved from the history stack. This includes navigation caused by a script function or user-agent specific mechanisms.
pick	option	The pick event occurs when the user selects or deselects this item.

5.6.2. WML Extrinsic Events

This specification does not specify any classes of extrinsic events. One example of a WML extrinsic event class may be WTA events [WTA].

5.6.3. Event Bindings

An *event binding* specifies a task to be executed when an event occurs.

The user agent MUST implement bindings of typed and untyped events.

www.it-ebooks.info

5.6.3.1. Bindings for Typed Events

The specification of an event binding for a typed event takes one of two forms: attribute syntax or element syntax. All the events in Table 5-3 are typed events.

The attribute syntax uses an attribute to specify a URI to be navigated to when the event occurs. This type of event binding is specified in a well-defined element-specific attribute and is the equivalent of a `go` task. For example:

```
<wml:card onenterforward="/url"> <p> Hello </p> </wml:card>
```

This attribute value may only specify a URI.

Table 5-3
Event Bindings for Attribute Syntax

Event Type	*Attribute Name*
timer	ontimer
enterforward	onenterforward
enterbackward	onenterbackward
pick	onpick

The element syntax is an expanded version of the previous, allowing the author more control over user agent behaviour by giving the ability to specify any task. A `wml:onevent` element is declared within a parent element, specifying the full event binding for a particular event. For example, the following is identical to the previous example:

```
<wml:card>
<wml:onevent type="enterforward"> <wml:go href="/url"/>
</wml:onevent>
<p>
Hello
</p>
</wml:card>
```

The user agent MUST treat the attribute syntax as an abbreviated form of the `wml:onevent` element where the attribute name is mapped to the `type` attribute of the `wml:onevent` element.

An event binding is scoped to the element in which it is declared, e.g., an event binding declared in a `body` or `wml:card` is local to that card or body. Any event binding declared in an element is active only within that element. If the event binding element specifies an event type which does not apply to its parent element, the user agent MUST ignore the event binding. Conflicting event bindings within an element are an error.

www.it-ebooks.info

5.6.3.2. Bindings for Untyped Events

Events targeted at `do` elements are untyped, or anonymous, and cannot be bound using the event binding forms specified in the previous section.

The `wml:do` element therefore uses an abbreviated syntax for specifying event bindings – the task element is placed directly within the `do` element. The `wml:onevent` container element is not used. For example:

```
<wml:do type="accept>
<wml:go href="url"/>
</wml:do>
```

The only way to bind a task to an event targeted at a `wml:do` element is to place the task element within the `wml:do` element as shown above.

5.6.3.3. Event Binding Scope

Event bindings may be declared within the `body` or `wml:card` elements, and may also be declared at the document level, within the `html` element:

- Card-level: the event binding may appear inside a `body` or `wml:card` element and specify event-processing behaviour for that particular card or body.
- Document-level: the event binding may appear inside the `html` element and specify event-processing behaviour for each body/card in the document. A document-level event binding is equivalent to specifying the event binding in each body/card.

A card-level event binding overrides a document-level event binding if they both specify the same event. A card-level `wml:onevent` element will override a document-level `wml:onevent` element if they both have the same `type`.

For example, in the following document the event handler in the card overrides the one specified in the document root although the two have been defined using different syntax:

```
<html>
...
<wml:onevent type="enterbackward">
<wml:go href="aaa"/>
</wml:onevent>
<wml:card onenterbackward="bbb">
<p>Hello</p>
</wml:card>
</html>
```

For a given body/card, the *active* event bindings are defined as the event bindings specified in the body/card that do not bind a `noop` task, plus any event bindings specified in the document root element not overridden in the card or binding a noop

www.it-ebooks.info

task. Overridden event bindings, event bindings defined in other cards, and event bindings that bind a noop task are considered *inactive*.

If a card-level binding overrides a document-level binding and the card-level binding specifies a noop task, the event binding for that event will be completely masked. In this situation, the card- and document-level bindings will be ignored and no side effects will occur on delivery of the event. In other words, both the card-level and the documentlevel bindings are considered inactive in such a case.

If a card-level binding or document-level binding specifies a `noop` task but does not override and is not overridden by another binding, then the binding for that event will also be masked and similarly ignored with no side effects.

In the following example, a document-level event binding indicates that a go task should execute on receipt of a timer expiration event. The first card inherits the event binding specified in the `html` element. The binding will be active in that card. The second card overrides the document-level event binding with a noop task. The event binding will therefore be inactive when the second card is displayed. The third card overrides the document-level event binding, replacing it with an alternate binding and causing the user agent to execute a different go task if the timer expiration event occurs while the third card is displayed.

```
<html>
...
<!-- Document-level event binding -->
<wml:onevent type="timer">
<wml:go href="/general"/>
</wml:onevent>
<wml:card id="first">
<!-- Document-level event binding not overridden. The event binding is active
in this card. -->
<!-- rest of card -->
...
</wml:card>
<wml:card id="second">
<!-- Document-level event binding is overridden with noop.
Binding is inactive in this card. -->
<wml:onevent type="timer">
<wml:noop/>
</wml:onevent>
<!-- rest of card -->
...
</wml:card>
<wml:card id="third">
<!-- Document-level event binding is overridden. It is replaced by a
card-level event binding. -->
<wml:onevent type="timer">
<wml:go href="/specific"/>
</wml:onevent>
<!-- rest of card -->
...
</wml:card>
</html>
```

www.it-ebooks.info

5.7. Identification of Document Types

WML2 documents are identified by the MIME media type “application/wml+xml”.

XHTML Mobile Profile documents are identified by the MIME media type “application/vnd.wap.xhtml+xml”.

XHTML Basic documents, however, do not have a unique MIME media type.

The type “application/xhtml+xml” can be used to identify documents from any of the XHTML-based markup languages, including XHTML Basic.

Section 2.1, Document Conformance, of [XHTMLBasic], gives a set of criteria that all conforming XHTML Basic documents must meet. A document that meets all these criteria is positively identified as an XHTML Basic document.

However, as such criteria require validating the document according to the DTD, they are not useful for all user agents.

For user agents that do not validate documents according to the DTD, the following criteria may help to identify a document with media type “application/xhtml+xml” as an XHTML Basic document:

An XHTML Basic document must have the string “<!DOCTYPE html” preceding the root element, indicating a DOCTYPE declaration. (See 6.17 for details about document conformance).

The DOCTYPE declaration may include the XHTML Basic Formal Public Identifier and may also include the URI of the XHTML Basic DTD as specified below:

```
<!DOCTYPE html PUBLIC “-//W3C//DTD XHTML Basic 1.0//EN”
“http://www.w3.org/TR/xhtml-basic/xhtml-basic10.dtd”>
```

However, the Formal Public Identifier is not required, and the system identifier may be modified appropriately.

There is no requirement that XHTML Basic documents be given the media type “application/xhtml+xml”; the media type “text/html” may be used instead. As there are no conformance rules for documents with type “text/html”, there is no easy way to determine which documents of type “text/html” are XHTML Basic documents, except that the document may include the document type declaration specified above.

A conforming user agent MUST accept WML2 documents identified as “application/wml+xml”.

A conforming user agent MUST accept XHTML Mobile Profile documents identified as “application/vnd.wap.xhtml+xml”. A conforming user agent SHOULD accept XHTML Mobile Profile documents identified as “application/xhtml+xml”.

www.it-ebooks.info

5.8. Common User Agent Behaviour depending on the type of Elements

5.8.1. Activation of Elements using Access Keys

When the `accesskey` attribute or `-wap-accesskey` style sheet property is used, pressing the specified access key causes the element to be activated, or brought into focus [HTML4]. The result of activation varies by element. Table 5-4 defines activation for the elements of WML2.

Table 5-4
Activation for the elements of WML2

Elements	*Activation action*
`a`	Execute the associated task
`wml:anchor`	Execute the associated task
`input`	Give focus to the form control and allow input
`textarea`	Give focus to the form control and allow input
other elements that can be activated	Implementation dependent

5.9. The BACK Key in WML2

Unless otherwise overridden, the default behaviour of the BACK key is as follows:

- ✦ When the BACK key is activated by the end user, the user agent MUST execute a prev task as defined in section 5.3.2.

Through the use of the `wml:do` element, the author may override the default behaviour of the BACK key. When the author specifies a `wml:do` element with the pre-defined role of `prev`, the default behaviour of the WML2 user agent is overridden with the task specified by the `wml:do` element. See section 6.16.3 for details of the `wml:do` element.

5.10. Navigation User Interface Using the `wml:do` Element

The `wml:do` element provides a general mechanism for the user to act upon the current body or card (i.e., the current `body` element or `wml:card` element). A task is associated with the `wml:do` element and that task is executed when the user activates the element.

www.it-ebooks.info

The presentation of the wml:do element is user agent dependent. The author must only assume that the wml:do element is mapped to a unique user interface widget that the user can activate. In addition, the author is required to specify the role of the associated task within the application, using the type attribute. The type attribute is used by the user agent in presentation of the wml:do element, and is discussed below.

When the user activates a wml:do element, the user agent MUST execute the associated task. The user agent determines what it means for a wml:do element to be activated, but it is assumed that the means of activation is appropriate to the type of user interface widget. For example, a key is activated when the end user presses (or releases) the key.

5.10.1. Processing the type Attribute

The type attribute specifies the role of the associated task within the application. The role is a logical action, such as acceptance of a choice or a request for help. It provides information to the user agent about the author's intended use of the wml:do element, to assist in mapping the task onto a physical user interface construct.

The user agent MUST use the type attribute on the wml:do element to provide a suitable mapping onto a physical user interface construct. Although the exact mapping is user agent dependent, the user agent MUST support a default presentation for all values of the type attribute.

Authors must not rely on the semantics or behaviour of an individual type attribute value, or on the mapping of a role to a particular physical construct.

All type attribute values are reserved, except for those marked as experimental or vendor-specific.

The user agent MUST accept any type value, but MAY treat any unrecognised type value as the equivalent of unknown.

Table 5-5
Predefined roles for the wml:do element

Role	*Description*
accept	Positive acknowledgement. (e.g. OK, Yes, Accept, Done, Next)
prev	Backward history navigation.
help	Request for help. May be context sensitive.
reset	Clearing or resetting state.
options	Context-sensitive request for options or additional operations.
delete	Delete item or choice.

www.it-ebooks.info

Role	*Description*
`unknown`	A generic `do` element.
`X-*, x-*`	Experimental types. This set is not reserved.
`vnd.*, VND.*`, and any combination of `[Vv][Nn][Dd].*`	Vendor-specific or user agent-specific types. This set is not reserved.

Vendors should allocate names with the format `VND.CO-TYPE`, where `CO` is a company name abbreviation and `TYPE` is the widget type. This convention is the same as for MIME media types. See [RFC2045] for more information.

5.10.2. Overriding the BACK Key Using the `wml:do` Element

The `wml:do` element can be used to override the default behaviour for the BACK key, as defined in section 5.9. For a `wml:do` element with the `type` attribute equal to `prev`, if the default presentation is selected, the user agent MUST override the default behaviour of the BACK key with the task associated with the `wml:do` element. If there is more than one `wml:do` element with the `type` attribute equal to `prev`, then the first `wml:do` element in document order overrides the BACK key.

5.11. Timer Processing

This section describes how the user agent processes a timer.

The timer SHALL be initialised and started at card or document entry and SHALL be stopped when the card or document is exited. Card or document entry is any task or user action that results in the `wml:card` element or `body` element being activated, for example, navigating into the card. Card or document exit is defined as the execution of any task.

The value of a timer SHALL decrement from the initial value, triggering the delivery of a `timer` event on transition from a value of one to zero. If the user has not exited the card at the time of timer expiration, `a timer` event SHALL be delivered to the card or the body.

The user agent SHALL interpret the timer timeout value in units of one-tenth (1/10) of a second. However, the actual resolution of the timer maintained by the user agent is implementation dependent. The interaction of the timer with the user agent's user interface and other time-based or asynchronous device functionality is also implementation dependent.

If the value of the timeout is not a positive integral number, the user agent SHALL ignore the `wml:timer` element. A timeout value of zero (0) SHALL disable the timer.

Invoking a refresh task is considered an exit. The task SHALL stop the timer, commit its value to the context, and update the user agent accordingly. Completion of the

www.it-ebooks.info

refresh task is considered an entry to the card. At that time, the timer SHALL restart.

Note: Variables may be used to simulate resuming a timer.

5.12. Acceptance of XHTML

The user agent MUST accept a valid XHTML Basic document.

The user agent MUST accept a valid XHTML Mobile Profile[XHTMLMP] document.

5.13. User Agent Conformance Rules

A conforming user agent MUST meet all of the following criteria (as defined in [XHTMLMod]):

Note: In the following text, “must” in the original text in the reference is updated with “MUST”.

1. In order to be consistent with the XML 1.0 Recommendation [XML], the user agent MUST parse and evaluate an XHTML document for well-formedness. If the user agent claims to be a validating user agent, it MUST also validate documents against their referenced DTDs according to [XML].
2. When the user agent claims to support facilities defined within this specification or required by this specification through normative reference, it MUST do so in ways consistent with the facilities’ definition.
3. When a user agent processes an XHTML document as generic [XML], it SHALL only recognise attributes of type ID (e.g., the id attribute on most XHTML elements) as fragment identifiers.
4. If a user agent encounters an element it does not recognise, it MUST continue to process the children of that element. If the content is text, the text MUST be presented to the user.
5. If a user agent encounters an attribute it does not recognise, it MUST ignore the entire attribute specification (i.e., the attribute and its value).
6. If a user agent encounters an attribute value it does not recognise, it MUST use the default attribute value.
7. If it encounters an entity reference (other than one of the predefined entities) for which the user agent has processed no declaration (which could happen if the declaration is in the external subset which the user agent has not read), the entity reference SHOULD be rendered as the characters (starting with the ampersand and ending with the semi-colon) that make up the entity reference.
8. When rendering content, a user agent that encounters characters or character entity references that are recognised but not renderable SHOULD display the document in such a way that it is obvious to the user that normal rendering has not taken place.
9. White space is handled according to the following rules. The following characters are defined in [XML] as white space characters:

www.it-ebooks.info

- SPACE ( )
- HORIZONTAL TABULATION (	)
- CARRIAGE RETURN (
)
- LINE FEED (
)

The XML processor normalises different systems' line end codes into one single LINE FEED character, that is passed up to the application.

The user agent MUST process white space characters in the data received from the XML processor as follows:

- ✦ All white space surrounding block elements SHOULD be removed.
- ✦ Comments are removed entirely and do not affect white space handling. One white space character on either side of a comment is treated as two white space characters.
- ✦ When the 'xml:space' attribute is set to 'preserve', white space characters MUST be preserved and consequently LINE FEED characters within a block MUST NOT be converted.
- ✦ When the 'xml:space' attribute is not set to 'preserve', then:
  - # Leading and trailing white space inside a block element MUST be removed.
  - # LINE FEED characters MUST be converted into one of the following characters: a SPACE character, a ZERO WIDTH SPACE character (), or no character (i.e. removed).

    The choice of the resulting character is user agent dependent and is conditioned by the script property of the characters preceding and following the LINE FEED character.
  - # A sequence of white space characters without any LINE FEED characters MUST be reduced to a single SPACE character.
  - # A sequence of white space characters with one or more LINE FEED characters MUST be reduced in the same way as a single LINE FEED character.

White space in attribute values is processed according to [XML].

Note (*informative*): In determining how to convert a LINE FEED character a user agent should consider the following cases, whereby the script of characters on either side of the LINE FEED determines the choice of the replacement. Characters of COMMON script (such as punctuation) are treated as the same as the script on the other side:

1. If the characters preceding and following the LINE FEED character belong to a script in which the SPACE character is used as a word separator, the LINE FEED character should be converted into a SPACE character. Examples of such scripts are Latin, Greek, and Cyrillic.

www.it-ebooks.info

2. If the characters preceding and following the LINE FEED character belong to an ideographic-based script or writing system in which there is no word separator, the LINE FEED should be converted into no character. Examples of such scripts or writing systems are Chinese, Japanese.
3. If the characters preceding and following the LINE FEED character belong to a non ideographic-based script in which there is no word separator, the LINE FEED should be converted into a ZERO WIDTH SPACE character () or no character. Examples of such scripts are Thai, Khmer.
4. If none of the conditions in (1) through (3) are true, the LINE FEED character should be converted into a SPACE character.

The Unicode [UNICODE] technical report TR#24 (Script Names) provides an assignment of script names to all characters.

6. WML2 Markup Elements and Attributes (Normative)

Interoperability is guaranteed when the authors observe the WML2 DTD. It is out of scope of this specification to guarantee interoperability when another DTD is supported by the user agent.

6.1. XHTML Basic and Extensions

WML2 markup elements and attributes consist of those elements and attributes from XHTML Basic [XHTMLBasic] with additional XHTML Modularization extensions and WAP extensions.

The Style Sheet and Presentation modules are defined in XHTML Modularization [XHTMLMod]. The Events and Context and Navigation Modules are WAP extensions. The WAP extensions in XHTML Modularization-based modules are defined in the following sections.

Except where specified in this document, the semantics of all the elements and attributes are defined in [HTML4].

The WML2 DTD is defined using XML Namespaces [XMLN]. The user agent MAY implement a fully namespacesaware XML processor as defined by [XMLN], but is not required to do so in order to correctly process WML2 documents.

6.2. The Structure Module

The following elements in the Structure Module are used to specify the structure of the document:

```
body html wml:card head title
```

www.it-ebooks.info

For the details of these elements, see [HTML4] unless stated otherwise in this specification.

6.2.1. The `body` Element

The `wml:newcontext` attribute specifies whether the browser context is initialised to a well-defined state when the document is loaded. If the `wml:newcontext` attribute value is "true", the user agent MUST reinitialise the browser context upon navigation to this card.

6.2.2. The `html` Element

The `xmlns:wml` attribute refers to the WML namespace:

```
http://www.wapforum.org/2001/wml .
```

The `wml:use-xml-fragments` attribute is used to specify how a fragment identifier is interpreted by the user agent. For details of use of `wml:use-xml-fragments` in the go task and the prev task, see sections 5.3.1 and 5.3.2.

6.2.3. The wml:card Element

The `wml:card` element specifies a fragment of the document body. Multiple `wml:card` elements may appear in a single document. Each `wml:card` element represents an individual presentation and/or interaction with the user.

If the `wml:card` element's newcontext attribute value is "true", the user agent MUST reinitialise the browser context upon navigation to this card.

6.3. Text Module

The following elements in the Text Module [XHTMLMod] are used to specify different types of text:

```
abbr acronym address blockquote br cite code dfn div em h1 h2 h3 h4 h5 h6
kbd p pre q samp span strong var
```

For the details of these elements, see [HTML4] unless stated otherwise in this specification.

6.3.1. The `p` Element

The following attributes have equivalents in the WCSS properties.

```
align wml:mode
```

The `align` attribute is equivalent to the `text-align` WCSS property. The `wml:mode` is equivalent to the `whitespace` WCSS property, where the attribute value "wrap" is equal to the property value "normal" and the attribute value "nowrap" is equal to the property value "nowrap". These styling attributes are deprecated and the authors are encouraged to use the WCSS style properties.

www.it-ebooks.info

6.4. Hypertext Module

The following elements in the Hypertext Module [XHTMLMod] are used to define hypertext links to other resources:

```
a
```

For the `a` element, refer to [HTML4].

6.5. Forms Module

The following elements in the Forms Module are used to specify user interaction:

```
form input select option label textarea optgroup fieldset
```

For these elements, refer to [HTML4].

The `optgroup` and `fieldset` elements are part of the XHTML Forms Module, which extends the Basic Forms Module [XHTMLMod].

6.5.1. The `select` Element

The `wml:name` attribute specifies the name of the variable to set with the result of the selection. The `wml:value` attribute specifies the default-selected `option` element. The `wml:iname` attribute specifies the name of the variable to be set with the index of the selection. The `wml:ivalue` attribute specifies the index of the default-selected `option` element. For details of setting control variables, see section 5.4.3.2.

6.5.2. Text Control Format Attributes

6.5.2.1. The `wml:format` Attribute

The `wml:format` attribute specifies an input mask for user input entries. The *input mask* is a string consisting of mask control characters and static text that is displayed in the input area. The user agent may use the input mask to facilitate accelerated data input (e.g. the user agent may change its input mode according to the format code of the current position of the input cursor). An input mask is only valid when it contains legal format codes and static text.

User agents MUST ignore invalid masks.

Character categories are as defined by [UNICODE]:

- "Letter" refers to character categories Lu, Ll, Lm, and Lo.
- "Uppercase letter" refers to character categories Lu and Lm.
- "Lowercase letter" refers to character categories Ll and Lm.
- "Numeric character" refers to the character category Nd.
- "Punctuation" refers to character categories Pc, Pd, Ps, Pe, and Po.
- "Symbol" refers to character categories Sm, Sc, Sk, and So.

www.it-ebooks.info

User agents need only be capable of displaying and accepting the subsets of the above sets that are appropriate for all languages that they support. However, the user agent MUST support ASCII graphic characters of the Unicode Basic Latin block (U+0020 – U+007E).

For a given input element, the user agent may choose to restrict the set of allowable characters to those appropriate for the current language(s).

The *current languages* are the superset of:

- ✦ the current language of the WML document, plus
- ✦ the user agent's accept language(s), plus
- ✦ the user agent's interface language.

In caseless languages, format codes distinguishing between upper and lowercase are equivalent.

The format control characters specify the data format expected to be entered by the user.

The default format is "*M". The format codes are:

A	entry of any uppercase letter, symbol, or punctuation character. Numeric characters are excluded.
a	entry of any lowercase letter, symbol, or punctuation character. Numeric characters are excluded.
N	entry of any numeric character.
n	entry of any numeric, symbol, or punctuation character.
X	entry of any uppercase letter, numeric character, symbol, or punctuation character.
x	entry of any lowercase letter, numeric character, symbol, or punctuation character.
M	entry of any character valid in the current languages, including any letter, numeric, symbol, or punctuation character. If the language supports case and the device supports both upper and lower case entry, the user agent may choose to default to uppercase entry mode but MUST allow entry of any character.
m	entry of any character valid in the current languages, including any letter, numeric, symbol, or punctuation character. If the language supports case and the device supports both upper and lower case entry, the user agent may choose to default to lowercase entry mode but MUST allow entry of any character.

Continued

www.it-ebooks.info

<lang:class>	entry of any character belonging to language-specific class. A language-specific class character is one that delimited by a pair of '<' and '>'characters in the input format. The *lang* parameter is a language identifier as defined in [RFC3066]; i.e., a primary tag optionally followed by a subtag. The *class* parameter identifies a subset of characters specific to the identified language, and is vendor-specific or user agent specific parameter. If the user agent supports the language specified by *lang*, the user agent MUST restrict input to the characters which are members of the specified *class*. Otherwise (i.e., the user agent does not support the specified language or class), the user agent MUST treat this format code as "M".
****f***	entry of any number of characters; f is one of the above format codes and specifies what kind of characters can be entered. *Note: This format may only be specified once and must appear at the end of the format string.*
nf	entry of up to *n* characters where *n* is natural number, f is one of the above format codes (other than ** f* format code) and specifies what kind of characters can be entered. *Note: This format may only be specified once and must appear at the end of the format string.*
\c	display the character, *c*, in the entry field; allows escaping of the format codes as well as introducing nonformatting characters so they can be displayed in the entry area. Escaped characters are considered part of the input's value, and MUST be preserved by the user agent. For example, the stored value of the input "12345-123" having a mask "NNNNN\-3N" is "12345-123" and not "12345123". Similarly, if the value of the variable named by the name attribute is "12345123" and the mask is "NNNNN\-3N", the user agent MUST unset the variable since it does not conform to the mask.

6.5.2.2. The `wml:emptyok` Attribute

The `wml:emptyok` attribute indicates whether this input element accepts empty input or not. If `wml:emptyok` is "true", input is not required even if the format mask would otherwise require it. If `wml:emptyok` is "false", input is required even if the format mask would otherwise not require it.

If the author does not explicitly specify the `wml:emptyok` attribute, the format attribute fully defines the input requirement. The implied value of the `wml:emptyok` attribute is "true" when the `wml:format` attribute allows empty input (i.e., the format mask is implied or a "*f" format code). Otherwise, the implied value of the attribute is "false".

www.it-ebooks.info

Whether or not input is required, any input given MUST match the format specification.

For the following input elements, input is required:

```
<input name="x" wml:format="M*M"/> <!-- implied: wml:emptyok="false" -->
<input name="x" wml:emptyok="false"/> <!-- implied: wml:format="*M" -->
<input name="x" wml:emptyok="false" wml:format="M*M"/>
<input name="x" wml:emptyok="false" wml:format="*M"/>
```

For the following input elements, input is not required:

```
<input name="x"/> <!-- implied: wml:format="*M" wml:emptyok="true" -->
<input name="x" wml:format="*M"/> <!-- implied: wml:emptyok="true" -->
<input name="x" wml:emptyok="true"/> <!-- implied: wml:format="*M" -->
<input name="x" wml:emptyok="true" wml:format="M*M"/>
<input name="x" wml:emptyok="true" wml:format="*M"/>
```

6.5.3. Form Control Initialisation Attribute

6.5.3.1. The `wml:name` Attribute

The `wml:name` attribute names the form control variable. For details about form control variables, see section 5.4.3.2.

6.5.4. Activation and Event Handler Attributes

6.5.4.1. The `accesskey` Attribute

The `accesskey` attribute names the key to activate the form control. See section 6.4 for a description.

6.5.4.2. The wml:onpick Attribute

The `wml:onpick` attribute in the `option` element specifies a URI to be navigated to when the `pick` event occurs.

The `pick` event occurs when the user selects or deselects this option. A multiple-selection option list generates a `pick` event whenever the user selects or deselects this option. A single-selection option list generates a `pick` event when the user selects this option, i.e., no event is generated for the de-selection of any previously selected option.

6.6. Tables Module

The following elements in the XHTML Tables Module are used to define tables:

```
caption table tr td th
```

For these elements, refer to [HTML4] unless otherwise specified in the following section.

www.it-ebooks.info

6.6.1. The `table` Element

The `wml:align` attribute specifies the layout of text and images within the columns of a table. A column's contents can be centre aligned, left aligned or right aligned when it is rendered to the user. The attribute value is interpreted as a non-separated list of alignment designations, one for each column. Centre alignment is specified with the value "C", left alignment is specified with the value "L", right alignment is specified with the value "R", and default alignment is specified with the value "D". Designators are applied to columns as they are defined in the content. The first designator in the list applies to the first column, the second designator to the second column, and so forth. Default alignment is applied to columns that are missing alignment designators or have unrecognised designators. All extra designators are ignored. Determining the default alignment is implementation dependent. User agents SHOULD consider the current language when determining the default alignment and the direction of the table. A user agent MAY use other algorithms to make such decisions.

The `wml:align` attribute is deprecated. Authors are encouraged to use WCSS instead.

The presentation of the table is likely to depend on the display characteristics of the device. WML does not define how a user agent renders a table. User agents may create aligned columns for each table, or it may use a single set of aligned columns for all tables in a card. User agents that choose to render a table in a traditional tabular manner should determine the width of each column from the maximum width of the text and images in that column to ensure the narrowest display width. However, the user agent may use fixed width or other appropriate layout algorithms instead.

The user agent that chooses to render tables in a traditional tabular manner MUST use a non-zero width gutter to separate each non-empty column.

6.7. Lists Module

The following elements in the XHTML Lists Module are used to define lists:

```
dt dd dl ol ul li
```

For these elements, refer to [HTML4].

The `start` attribute of the `ol` element and the `value` attribute of the `li` elements are part of the XHTML Legacy Module, which extends the Lists Module [XHTMLMod].

6.8. Image Module

The following element in the XHTML Image Module is used to provide basic image embedding:

```
img
```

For this element, refer to [HTML4] otherwise specified in the following section.

www.it-ebooks.info

6.8.1. The `img` Element

The `wml:localsrc` attribute specifies an alternative internal representation for the image. This representation is used if it exists; otherwise the image is downloaded from the URI specified in the `src` attribute, i.e., any `wml:localsrc` parameter specified takes precedence over the image specified in the `src` parameter.

An author may use the `wml:localsrc` attribute to refer to a pictogram, as specified by [PICTO].

The following attributes have equivalents in the WCSS properties:

```
align vspace hspace
```

The `align` attribute is equivalent to the `vertical-align` property in the WCSS. The vspace and hspace attribute can be mapped to the `margin` property in the CSS. These styling attributes are deprecated and the authors are encouraged to use the WCSS style properties, if possible.

6.9. Metainformation Module

The following element in the XHTML Metainformation Module [XHTMLMod] is used to describe information within the declarative portion of a document:

```
meta
```

For this element, refer to [HTML4] otherwise specified in the following section.

6.9.1. The `meta` Element

For documents retrieved via HTTP, origin servers SHOULD use the property name specified by the `http-equiv` attribute to create an HTTP response header. For use of HTTP, see [WAE].

For a WML2 document, the user agent SHALL assume that CSS (text/css) is the default style sheet language if no “Content-Style-Type” reponse header is present and no `meta` element with the `http-equiv` attribute value of “Content-Style-Type” is specified.

The equivalent `meta` element would be written:

```
<meta http-equiv=“Content-Style-Type” content=“text/css”/>
```

If the WML document is processed by an intermediate agent, any `meta` element with `wml:forua` set to “false” SHOULD be removed before the document is sent to the client. If the `wml:forua` attribute is “true” the meta data of the element SHOULD be delivered to the user agent. The method of delivery may vary. For example, http-equiv metadata may be delivered using HTTP or WSP headers.

www.it-ebooks.info

A user agent SHOULD ignore any meta element with the `wml:forua` attribute set to "false".

The `wml:forua` attribute is deprecated because WAP 2.0 Architecture cannot guarantee the semantics of this attribute.

6.10. Link Module

The following element in the XHTML Link Module [XHTMLMod] is used to define links to external resources:

```
link
```

For this element, refer to [HTML4].

The default value for `media` attribute is `all`.

6.11. Base Module

The following element in the XHTML Base Module [XHTMLMod] is used to define a base URI against which relative URIs in the document will be resolved:

```
base
```

For this element, refer to [HTML4].

6.12. Object Module

The following elements in the XHTML Object Module [XHTMLMod] are used to define general-purpose object inclusion:

```
object param
```

For these elements, refer to [HTML4].

6.13. Style Sheet Module

The following element in the XHTML Style Sheet Module [XHTMLMod] is used to declare an internal style sheet:

```
style
```

For this element, refer to [HTML4].

The default value for the `media` attribute is all.

www.it-ebooks.info

6.14. Presentation Module

The following elements are used to define simple presentation-related markup:

```
b i small big u hr
```

These elements are not defined in XHTML Basic [XHTMLBasic]. All these elements except the `u` element are defined in the XHTML Presentation Module [XHTMLMod]. The `u` element is defined in the XHTML Legacy Module [XHTMLMod].

For these elements, refer to [HTML4].

The `b`, `i`, `small`, `big` and `u` elements are deprecated. Authors are encouraged to use the equivalent WCSS style properties for achieving the same functionalities.

6.15. Events Module

6.15.1. The `wml:onevent` Element

The `wml:onevent` element binds a task to a particular event for the immediately enclosing element, e.g., specifying a `wml:onevent` element inside a `wml:card` element associates an event binding with the `wml:card` element.

The user agent MUST ignore any `wml:onevent` element specifying a type that does not correspond to a legal event for the immediately enclosing element.

The `type` attribute indicates the name of the event. In this specification, only intrinsic event types are defined. See 5.6.1 for the types of intrinsic events.

6.16. Context and Navigation Module

6.16.1. The `wml:anchor` Element

The `wml:anchor` element defines a source anchor for a hyperlink that may be activated by the user. The source anchor is identical to a source anchor created with the `a` element, except that the destination anchor for the hyperlink is specified by the task contained within the `wml:anchor` element.

When the user activates the `wml:anchor` element, the user agent MUST execute the associated task.

Source anchors may be present in any text flow, excluding the text in `option` elements (i.e., anywhere formatted text is legal, except for `option` elements). It is an error to specify more than one task element (e.g., `wml:go`, `wml:prev` or `wml:refresh`) in an `wml:anchor` element.

The `accesskey` attribute assigns an access key to an element as defined in [HTML4]. The `accesskey` attribute is equivalent to the WCSS `wap-accesskey` property. The authors are encouraged to use the WCSS style properties, if possible.

www.it-ebooks.info

6.16.2. The `wml:access` Element

The `wml:access` element specifies access control information for the entire document. It is an error for a document to contain more than one `wml:access` element. If a document does not include a `wml:access` element, access control SHALL be disabled. When access control is disabled, `wml:card` elements or `body` elements in any document can access this document.

A document's `domain` and `path` attributes specify which other documents may access it. As the user agent navigates from one document to another, it performs access control checks to determine whether the destination document allows access from the current document.

If a document has a `domain` and/or `path` attribute, the user agent MUST verify that the referring document's URI matches the values of the attributes. Matching is done as follows: the access domain is suffix-matched against the domain name portion of the referring URI and the access path is prefix matched against the path portion of the referring URI. When the verification fails, the user agent MUST deny access.

The user agent MUST perform domain suffix matching using the entire element of each sub-domain and verify an exact match of each element (e.g., `www.wapforum.org` shall match `wapforum.org`, but shall not match `forum.org`).

The user agent MUST perform path prefix matching using entire path elements and verify an exact match of each element (e.g., `/X/Y` matches `path="/X"` attribute, but does not match `path="/XZ"` attribute). When the verification fails, the user agent MUST deny access.

The domain attribute defaults to the current document's domain. The path attribute defaults to the value "/".

To simplify the development of applications that may not know the absolute path to the current document, the `path` attribute accepts relative URIs. The user agent converts the relative path to an absolute path and then performs prefix matching against the `path` attribute.

For example, given the following access control attributes:

```
domain="wapforum.org"
path="/cbb"
```

The following referring URIs would be allowed to go to the document:

```
http://wapforum.org/cbb/stocks.cgi
https://www.wapforum.org/cbb/bonds.cgi
http://www.wapforum.org/cbb/demos/alpha/packages.cgi?x=123&y=456
```

The following referring URIs would not be allowed to go to the document:

```
http://www.test.net/cbb
http://www.wapforum.org/internal/foo.wml
```

The `domain` and `path` attributes follow URI case sensitivity rules [RFC2396].

www.it-ebooks.info

6.16.3. The `wml:do` Element

The `wml:do` element provides a general mechanism for the user to act upon the current body/card (i.e., the current `body` element or `wml:card` element). A task is associated with the `wml:do` element and that task is executed when the user activates the element.

The `type` attribute specifies the role of the associated task within the application. For the details of type processing, see section 5.10.1

If the user agent is able to dynamically label the user interface widget, the `label` attribute specifies a textual string suitable for such labelling. The user agent SHOULD make a best-effort attempt to label the UI widget and SHOULD adapt the label to the constraints of the widget (e.g., truncate the string). If an element can not be dynamically labelled, this attribute may be ignored.

To work well on a variety of user agents, labels should be six characters or shorter in length.

6.16.4. The `wml:go` Element

The `href` attribute specifies a destination URI.

The `sendreferer` attribute specifies the control of the referring document information. If the `sendreferer` attribute is true, the user agent MUST specify, for the server's benefit, the URI of the document containing this task (i.e., the referring document).

The `method` attribute specifies the HTTP submission method.

The `enctype` attribute specifies the media type used to submit the parameter to the server.

```
application/x-www-form-urlencoded
multipart/form-data
application/vnd.wap.wml.form.urlencoded
```

Note: `application/vnd.wap.wml.form.urlencoded` is added for backwards compatibility.

The `accept-charset` attribute specifies the list of character encodings for data that the origin server must accept when processing input. The value of this attribute is a comma- or space-separated list of character encoding names as specified in [RFC2045] and [RFC2616].

If the `accept-charset` attribute is not specified or is the reserved string `unknown`, the user agent should use the character encoding that was used to transmit the document that contains the `wml:go` element.

The `cache-control` attribute specifies the control of cache use. If the `cache-control` attribute is present, and the value is set to "no-cache", the client MUST reload the URI from the origin server. This attribute represents the HTTP "cache-control" header. When this attribute is present, the HTTP "cache-control"

www.it-ebooks.info

header MUST be added to the request with the same value as specified in the attribute.

The `type` attribute gives an advisory hint as to the media type of the content available at the link target address. It allows user agents to opt to use a fallback mechanism rather than fetch the content if they are advised that they will get content in a media type they do not support.

Authors who use this attribute take responsibility to manage the risk that it may become inconsistent with the content available at the link target address.

The `wml:go` element declares a go task, indicating navigation to a URI.

If the `wml:go` element's `href` attribute names a WML card or document, the user agent SHALL display the destination card or document.

A `wml:go` task SHALL execute a "push" operation on the history stack.

The user agent SHALL ignore all `wml:postfield` elements in a `wml:go` element if the target of the `wml:go` element is a card contained within the current document.

The `wml:go` element may contain one or more `wml:postfield` elements. These elements specify information to be submitted to the origin server during the request. The submission of field data is performed in the following manner:

1. The field name/value pairs are identified and all variables are substituted.
2. The user agent SHOULD transcode the field names and values to the correct character set, as specified explicitly by the accept-charset or implicitly by the document encoding.
3. If the `href` attribute value is an HTTP URI, the request is performed according to the `method` and `enctype` attributes' values:

Table 6-1
Processing according to the `method` and `enctype` attributes' values

Method	*EnctypeProcess*
`get`	`application/vnd.wap.wml.form.urlencode d` The field names and values are escaped using URI-escaping and assembled into an `application/x-www-form-urlencoded` media type.
When a `wml:postfield` element is used to name a field, and the value of the `wml:postfield`'s `value` attribute references the control variable for a multiple-selection control, the field value MUST be submitted as a semicolon-separated list of values.	
For example, for a field named "x" with value "$(x) ", where "x" is the control variable with value "choice1; choice3", the submitted data will be "x=choice1; choice3".	
The submission data is appended to the query component of the URI.	

www.it-ebooks.info

Method	*EnctypeProcess*
The result MUST be a valid query component with the original query part and the postfields combined. An HTTP GET operation is performed on the resulting URI.	
`application/x-wwwform-urlencoded` The field names and values are escaped using URI-escaping and assembled into an `application/x-www-form-urlencoded` media type.	
When a `wml:postfield` element is used to name a field, and the value of the `wml:postfield`'s `value` attribute references the control variable for a multiple-selection control, the field value MUST be submitted as specified in [HTML4], where one name/value pair is included for each item in the list.	
For example, for a field named "x" with value "$(x) ", where "x" is the control variable with value "choice1; choice3", the submitted data will be "x=choice1& x=choice3".	
The submission data is appended to the query component of the URI.	
The result MUST be a valid query component with the original query part and the postfields combined. An HTTP GET operation is performed on the resulting URI.	
`multipart/form-data` Error. The user agent MUST ignore the `wml:go` element.	
`post application/vnd.wap.wml.form.urlencoded` The field names and values are escaped using URI-escaping and assembled into an `application/x-www-form-urlencoded` media type.	
When a `wml:postfield` element is used to name a field, and the value of the `wml:postfield`'s `value` attribute references the control variable for a multiple-selection control, the field value MUST be submitted as a semicolon-separated list of values.	
For example, for a field named "x" with value "$(x) ", where "x" is the control variable with value "choice1; choice3", the submitted data will be "x=choice1; choice3".	
The submission data is sent as the body of the HTTP POST request.	
The `Content-Type` header MUST include the `charset` parameter to indicate the character encoding.	
`application/x-wwwform-urlencoded` The field names and values are escaped using URI-escaping and assembled into an `application/x-www-form-urlencoded` media type.	
When a `wml:postfield` element is used to name a field, and the value of the `wml:postfield`'s `value` attribute references the control variable for a multiple-selection control, the field value MUST be submitted as specified in [HTML4], where one name/value pair is included for each item in the list.	
For example, for a field named "x" with value "$(x) ", where "x" is the control variable with value "choice1;choice3", the submitted data will be "x=choice1&x=choice3".	
The submission data is sent as the body of the HTTP POST request.	
The `Content-Type` header MUST include the `charset` parameter to indicate the character encoding when the part contains characters not in the US-ASCII character set. `multipart/form-data` The field names and values are encoded as a `multipart/formdata` media type as defined in [RFC2388].	

www.it-ebooks.info

Method	*EnctypeProcess*
The submission data is sent as the body of the HTTP POST request.	
The `Content-Type` header MUST include the `charset` parameter to indicate the character encoding when the part contains characters not in the US-ASCII character set.	
When a `wml:postfield` element is used to name a field, and the value of the `wml:postfield`'s `value` attribute references the control variable for a multiple-selection control, the field value MUST be submitted as specified in [HTML4], where one part is included for each item in the list.	

For example, for a field named "x" with value "$(x) ", where "x" is the control variable with value "choice1; choice3", the submitted data will be:
`Content-Disposition: form-data; name="x"`
`choice1`
`----------AaAaAax`
`Content-Disposition: form-data; name="x"`
`choice3`	

6.16.5. The `wml:noop` Element

The `wml:noop` element specifies that nothing should be done, i.e., "no operation". This task is used in conjunction with the `wml:onevent` element to make inactive an event handler task that would otherwise be active.

6.16.6. The `wml:prev` Element

The `wml:prev` element declares a prev task, indicating navigation to the previous URI on the history stack. The prev task performs a "pop" operation on the history stack. See section 5.3.2 for processing of the prev task.

6.16.7. The `wml:refresh` Element

The `wml:refresh` element declares a refresh task, indicating an update of the user agent context as specified by the wml:setvar elements. See section 5.3.4 for processing of the refresh task.

6.16.8. The `wml:postfield` Element

The `name` attribute specifies the field name. The `value` attribute specifies the field value.

The `wml:postfield` element is used to indicate an instance of a variable or program argument. The `wml:postfield` element specifies a field name and value for transmission to an origin server during a URI request.

The actual encoding of the name and value will depend on the method used to communicate to the origin server.

www.it-ebooks.info

6.16.9. The `wml:setvar` Element

The `name` attribute specifies the name of the variable. The `value` attribute specifies the expression to evaluate and assign the variable.

The `wml:setvar` element specifies the variable to set in the current browser context as a side effect of executing a task.

The user agent SHALL ignore the element if the `name` attribute does not evaluate to a legal variable name at runtime.

The user agent SHALL ignore the element if the `value` attribute does not evaluate to a legal expression at runtime.

6.16.10. The `wml:getvar` Element

The `wml:getvar` element is used to substitute the value of a variable into the text (#PCDATA) of a document.

The `wml:getvar` element is used in conjunction with the attribute expression syntax, which allows authors to insert variable references into attribute values. Together, they provide a mechanism by which WML documents can be parameterised. For the details of the attribute expression syntax, see section 5.5.

The required `name` attribute specifies the name of the variable whose value is the substitution text. If the variable is not set, or if the `name` attribute value does not evaluate to a legal variable name, the substitution text is the empty string.

The user agent MUST replace the element with the substitution text.

The `conversion` attribute specifies the conversion to be applied to the value of the variable. The conversions involve URI escaping and unescaping, and are identical to those specified for variable references in section 5.5.1.1. Legal values are “noesc”, “escape” and “unesc”.

Only textual information can be substituted; no substitution of elements or attributes is possible. The user agent MUST NOT attempt to process the resulting string as XML markup.

The use of the `wml:getvar` element is illustrated by the following example.

Given a variable “x” with a value “100.00”, the following markup can be used to substitute the value of “x” into a paragraph:

```
<p>Your balance is $<wml:getvar name=“x”/>.</p>
```

This markup would be displayed as:

```
Your balance is $100.00.
```

www.it-ebooks.info

6.16.11. The `wml:timer` Element

The `wml:timer` element declares a timer for a card or body, which exposes a means of processing inactivity or idle time.

The `name` attribute specifies the name of the variable to be set with the value of the timer. The `name` variable's value is used to set the timeout period upon timer initialisation. The variable named by the `name` attribute will be set with the current timer value when the card is exited or when the timer expires. For example, if the timer expires, the `name` variable is set to a value of "`0`".

The `value` attribute indicates the default value of the variable named in the `name` attribute. When the timer is initialised and the variable named in the `name` attribute is not set, the `name` variable is assigned the value specified in the `value` attribute. If the `name` variable already contains a value, the `value` attribute is ignored. If the `name` attribute is not specified, the timeout is always initialised to the value specified in the `value` attribute.

For the details of timer processing, see section 5.11.

6.17. Document Conformance

A conforming WML2 document MUST meet all of the following criteria:

- The document MUST conform to the constraints expressed in Appendix A.
- The root element of the document MUST be <html>.
- The name of the default namespace on the root element MUST be the XHTML namespace name, `http://www.w3.org/1999/xhtml`. Also on the root element, the WML namespace MUST be declared with the name, `http://www.wapforum.org/2001/wml`, and the prefix, "`wml`".

```
<html
xmlns="http://www.w3.org/1999/xhtml"
xmlns:wml="http://www.wapforum.org/2001/wml">
```

- There MUST be a DOCTYPE declaration with a public identifier in the document prior to the root element.

  The public identifier included in the DOCTYPE declaration must reference the DTD found in Appendix A using its Formal Public Identifier. The system identifier may be modified appropriately.

```
<!DOCTYPE html PUBLIC"-//WAPFORUM//DTD WML 2.0//EN"
"http://www.wapforum.org/DTD/wml20.dtd">
```

- The DTD subset MUST NOT be used to override any parameter entities in the DTD.

www.it-ebooks.info

7. Use of Style Sheets with WML2 (Normative)

Style sheets can be used to style WML2 documents. If the user agent supports styling of documents with style sheets, it MUST support the style language WAP CSS [WCSS], a subset of CSS2 with WAP-specific extensions. A user agent MAY support other style languages.

7.1. Adding Style to WML2 Documents

Style information can be associated with a document in 3 ways:

- ✦ External style sheet
- ✦ Internal style information
- ✦ Inline style information

External Style Sheets

An external style sheet can be associated with a document using a special XML processing instruction or the `link` element. The use of the XML processing instruction is specified in [WCSS].

In the following example, the XML processing instruction is used to associate the external style sheet "mobile.css":

```
<?xml-stylesheet href="mobile.css" media="handheld" type="text/css"?>
```

The use of the `link` element (section 6.10) is specified by [HTML4].

To link an external style sheet to a document using the `link` element, `ref="stylesheet"` or `ref="alternate stylesheet"` is specified. In either case, the `type` attribute specifies the style sheet language.

For `type="text/css"`, the user agent MUST process the style sheet according to the style language WAP CSS [WCSS].

In the following example, the `link` element is used to associate the external style sheet "mystyle.css":

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

www.it-ebooks.info

Internal Style Sheets

Style information can be located within the document using the `style` element. This element, like `link`, must be located in the document header.

The following shows an example of an internal style sheet:

```
<html>
<head>
<style type="text/css">
p { text-align: center; }
</style>
...
</head>
...
</html>
```

User agents that don't support style sheets, or don't support the specific style sheet language used by a `style` element,

MUST hide the content of the `style` element.

Inline Style

An author can specify style information for a single element using the `style` attribute. This is called inline style. The `style` attribute is part of the Core attribute set and is therefore available on every element in WML2. The default style language for style information in the `style` attribute is WAP CSS.

In the following example, inline styling information is applied to a specific paragraph element:

```
<p style="text-align: center">...</p>
```

Note that not all styling rules apply to all elements, and some elements are completely unaffected by styling rules. See [WCSS] for details.

7.2. The Default Style Sheet for WML2

See Appendix B for the WML2 default style sheet.

Appendix A. The DTD for WML2 (Normative)

See `http://www.wapforum.org/DTD/wml20.zip` for the zipped file including DTDs for WML2.

It includes a flat version of the DTD, "wml20-flat.dtd". It includes a DTD driver and WML specific modules, "wml20.dtd" and "wml-*.mod".

www.it-ebooks.info

See `http://www.wapforum.org/DTD/wml20-flat.dtd` for the flat version DTD for WML2.

See `http://www.wapforum.org/DTD/wml20.dtd` for the DTD driver for WML2.

Note that since we exclude all the W3C XHTML modules, in order to use the DTD driver, the user must download the W3C modules from the W3C Web site.

Appendix B. The WML2 Default Style Sheet (Informative)

```
/* A sample style sheet for WML 2.0
Modified from the HTML 4 CSS stylesheet found in the CSS 2
specification.
*/
body, card, div, p, center, hr, h1, h2, h3, h4, h5, h6,
address, blockquote, pre, ol, ul, dl, dt, dd,
form, fieldset, object
{ display: block }
li { display: list-item }
head { display: none }
table { display: table }
tr { display: table-row }
td, th { display: table-cell }
caption { display: table-caption }
th { font-weight: bolder; text-align: center }
caption { text-align: center }
h1, h2, h3, h4, h5, h6, b,
strong { font-weight: bolder }
i, cite, em, var,
address { font-style: italic }
pre, code, kbd,
pre { white-space: pre }
big { font-size: larger}
small { font-size: smaller}
hr { border: 1px inset }
ol { list-style-type: decimal }
u { text-decoration: underline }
/* end wml-default.css */
```

Appendix C. The WML2 Elements (Informative)

Note: All the REQUIRED attributes are in bold. The default attribute value of an attribute, if one exists, is in italic.

www.it-ebooks.info

Element Attributes Content

a id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), href(CDATA), charset(CDATA), type(CDATA), hreflang(NMTOKEN), rel(NMTOKENS), rev(NMTOKENS), accesskey(CDATA), tabindex(CDATA) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | img | object | wml:getvar | input | select | textarea | label)*

abbr id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

acronym id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

address id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

b id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

base href(CDATA) EMPTY

big id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

blockquote id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang (NMTOKEN), cite(CDATA) (h1 | h2 | h3 | h4 | h5 | h6 | ul | ol | dl | p | div | pre | blockquote | address | hr | table | form | fieldset | wml:do)+

bodyid(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), wml:newcontext('true' | '*false*'), wml:onenterforward(CDATA), wml:onenterbackward(CDATA), wml:ontimer(CDATA) (wml:onevent*, wml:timer?, (h1 | h2 | h3 | h4 | h5 | h6 | ul | ol | dl | p | div | pre | blockquote | address | hr | table | form | fieldset | wml:do)+)

br id(ID), class(NMTOKENS), title(CDATA), style(CDATA) EMPTY **caption** id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br |

www.it-ebooks.info

span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

cite id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

codeid(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

dd id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | h1 | h2 | h3 | h4 | h5 | h6 | ul | ol | dl | p | div | pre | blockquote | address | hr | table | form | fieldset | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

dfn id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

div id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | h1 | h2 | h3 | h4 | h5 | h6 | ul | ol | dl | p | div | pre | blockquote | address | hr | table | form | fieldset | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

dl id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (dt | dd)+

dt id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

em id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

fieldset id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | h1 | h2 | h3 | h4 | h5 | h6 | ul | ol | dl | p | div | pre

www.it-ebooks.info

| blockquote | address | hr | table | form | fieldset | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

formid(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), **action**(CDATA), method('*get*' | 'post'), enctype(CDATA '*application/x-www-form-urlencoded*') (h1 | h2 | h3 | h4 | h5 | h6 | ul | ol | dl | p | div | pre | blockquote | address | hr | table | fieldset)+

h1 id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

h2 id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

h3 id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

h4 id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

h5 id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

h6 id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

headxml:lang(NMTOKEN), profile(CDATA) ((meta | link | style | object | wml:access)*, ((title, (meta | link | style | object | wml:access)*, (base, (meta | link | style | object | wml:access)*)?) | (base, (meta | link | style | object | wml:access)*, (title, (meta | link | style | object | wml:access)*))))

www.it-ebooks.info

hr id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) EMPTY

html version('-//WAPFORUM//DTD WML 2.0//EN'), xml:lang(NMTOKEN), xmlns:wml('http://www.wapforum.org/2001/wml'), wml:onenterforward(CDATA), wml:onenterbackward(CDATA), wml:ontimer(CDATA), wml:use-xml-fragments('*true*' | 'false') (head, (wml:onevent)*, (body | (wml:card+))) **i** id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

img id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), **src**(CDATA), **alt**(CDATA), longdesc(CDATA), height(CDATA), width(CDATA), wml:localsrc(CDATA), wml:type(CDATA), vspace(CDATA '*0*'), hspace(CDATA '*0*'), align('top' | 'middle' | '*bottom*') EMPTY

input id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), type('*text*' | 'password' | 'checkbox' | 'radio' | 'submit' | 'reset' | 'hidden'), name(CDATA), value(CDATA), checked('checked'), size(CDATA), maxlength(CDATA), src(CDATA), tabindex(CDATA), accesskey(CDATA), wml:format(CDATA), wml:emptyok('true' | 'false'), wml:name(CDATA) EMPTY

kbd id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

labelid(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), for(IDREF), accesskey(CDATA) (PCDATA | input | select | textarea | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | img | object | wml:getvar)*

li id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), value(CDATA) (PCDATA | h1 | h2 | h3 | h4 | h5 | h6 | ul | ol | dl | p | div | pre | blockquote | address | hr | table | form | fieldset | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big

|small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

link id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), charset(CDATA), href(CDATA), hreflang(NMTOKEN), type(CDATA), rel(NMTOKENS), rev(NMTOKENS), media(CDATA) EMPTY

meta xml:lang(NMTOKEN), http-equiv(NMTOKEN), name(NMTOKEN), **content**(CDATA), scheme(CDATA), wml:forua('*true*' | 'false') EMPTY

www.it-ebooks.info

object id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), declare('declare'), classid(CDATA), codebase(CDATA), data(CDATA), type(CDATA), codetype(CDATA), archive(CDATA), standby(CDATA), height(CDATA), width(CDATA), name(CDATA), tabindex(CDATA) (PCDATA | h1 | h2 | h3 | h4 | h5 | h6 | ul | ol | dl | p | div | pre | blockquote | address | hr | table | form | fieldset | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label | param)*

ol id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), start(CDATA) (li)+

optgroup id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang (NMTOKEN), **label**(CDATA) (option)+

option id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), selected('selected'), value(CDATA), wml:onpick(CDATA) (PCDATA | wml:onevent | wml:getvar)*

p id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), wml:mode('wrap' | 'nowrap'), align('*left*' | 'right' | 'center') (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label | table | fieldset)*

param id(ID), **name**(CDATA), value(CDATA), valuetype('*data*' | 'ref' | 'object'), type(CDATA) EMPTY

pre id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), xml:space('preserve') (PCDATA | br | span | i | b | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | a | wml:anchor | wml:do | wml:getvar | input | select)*

q id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), cite(CDATA) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

samp id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

select id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), name(CDATA), size(CDATA), multiple('multiple'),

www.it-ebooks.info

tabindex(CDATA), wml:iname(NMTOKEN), wml:value(CDATA), wml:ivalue(CDATA), wml:name(CDATA) (optgroup | option)+

small id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

span id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

strong id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

style title(CDATA), xml:lang(NMTOKEN), **type**(CDATA), media(CDATA), xml:space('preserve') PCDATA

tableid(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), summary(CDATA), wml:columns(CDATA), wml:align(CDATA) (caption?, tr+)

td id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), abbr(CDATA), axis(CDATA), headers(IDREFS), scope('row' | 'col'), rowspan(CDATA '1'), colspan(CDATA '1'), align('left' | 'center' | 'right'), valign('top' | 'middle' | 'bottom') (PCDATA | h1 | h2 | h3 | h4 | h5 | h6 | ul | ol | dl | p | div | pre | blockquote | address | form | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

textarea id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), name(CDATA), **rows**(CDATA), **cols**(CDATA), tabindex(CDATA), accesskey(CDATA), wml:format(CDATA), wml:emptyok('true' | 'false'), wml:name(CDATA) (PCDATA | wml:getvar)*

th id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), abbr(CDATA), axis(CDATA), headers(IDREFS), scope('row' | 'col'), rowspan(CDATA '*1*'), colspan(CDATA '*1*'), align('left' | 'center' | 'right'), valign('top' | 'middle' | 'bottom') (PCDATA | h1 | h2 | h3 | h4 | h5 | h6 | ul | ol | dl | p | div | pre | blockquote | address | form | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

title xml:lang(NMTOKEN) PCDATA

www.it-ebooks.info

tr id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), align('left' | 'center' | 'right'), valign('top' | 'middle' | 'bottom') (th | td)+

u id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

ul id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (li)+

var id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | a | wml:anchor | wml:do | img | object | wml:getvar | input | select | textarea | label)*

wml:access id(ID), class(NMTOKENS), title(CDATA), style(CDATA), domain(CDATA), path(CDATA) EMPTY

wml:anchor id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang (NMTOKEN), accesskey(CDATA) (PCDATA | br | span | em | strong | dfn | code | samp | kbd | var | cite | abbr | acronym | q | i | b | big | small | u | img | object | wml:getvar | input | select | textarea | label | wml:go | wml:prev | wml:refresh)*

wml:card id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang (NMTOKEN), newcontext('true' | '*false*'), onenterforward(CDATA), onenterbackward(CDATA), ontimer(CDATA) (wml:onevent*, wml:timer?, (h1 | h2 | h3 | h4 | h5 | h6 | ul | ol | dl | p | div | pre | blockquote | address | hr | table | form | fieldset | wml:do)+)

wml:do id(ID), class(NMTOKENS), title(CDATA), style(CDATA), xml:lang(NMTOKEN), **type**(CDATA), label(CDATA) (wml:go | wml:prev | wml:noop | wml:refresh)

wml:getvar id(ID), class(NMTOKENS), title(CDATA), style(CDATA), **name**(CDATA), conversion('escape' | '*noesc*' | 'unesc') EMPTY

wml:go id(ID), class(NMTOKENS), title(CDATA), style(CDATA), **href**(CDATA), sendreferer('true' | '*false*'), type(CDATA), method('post' | '*get*'), enctype(CDATA '*application/x-wwwform-urlencoded*'), accept-charset(CDATA), cachecontrol ('no-cache') (wml:postfield | wml:setvar)*

wml:noop id(ID), class(NMTOKENS), title(CDATA), style(CDATA) EMPTY

wml:onevent id(ID), class(NMTOKENS), title(CDATA), style(CDATA), **type**(CDATA) (wml:go | wml:prev | wml:noop | wml:refresh)

www.it-ebooks.info

wml:postfield id(ID), class(NMTOKENS), title(CDATA), style(CDATA), **name**(CDATA), **value**(CDATA) EMPTY

wml:prev id(ID), class(NMTOKENS), title(CDATA), style(CDATA) (wml:setvar)*

wml:refresh id(ID), class(NMTOKENS), title(CDATA), style(CDATA) (wml:setvar)*

wml:setvar id(ID), class(NMTOKENS), title(CDATA), style(CDATA), **name**(CDATA), **value**(CDATA) EMPTY

wml:timer id(ID), class(NMTOKENS), title(CDATA), style(CDATA), name(NMTOKEN), **value**(CDATA) EMPTY

Appendix D. Static Conformance Requirements (Normative)

The notation used in this appendix is specified in [CREQ].

User Agent Context

Item	*FunctionReference*	*Status*	*Requirement*
WMLAGENTCONTEXT-C-001	WML variable context	5.2.1	M
WMLAGENTCONTEXT-C-002	A navigation history	5.2.2	M

Navigation Reference Processing Model

Item	*FunctionReference*	*Status*	*Requirement*
WML-NAVREF-C-001	The go task	5.3.1	M
WML-NAVREF-C-002	The prev task	5.3.2	M
WML-NAVREF-C-003	The refresh task	5.3.4	M
WML-NAVREF-C-004	Task execution failure	5.3.5	M

Form Processing Model

Item	*FunctionReference*	*Status*	*Requirement*
WML-FORMREF-C-001	Text input control initialisation	5.4.2.2	M
WML-FORMREF-C-002	Menu control initialisation	5.4.2.3	M

Continued

www.it-ebooks.info

Item	*FunctionReference*	*Status*	*Requirement*
WML-FORMREF-C-003	Checkbox control initialisation	5.4.2.4	M
WML-FORMREF-C-004	Radio button control initialisation	5.4.2.5	M
WML-FORMREF-C-005	Submit button control initialisation	5.4.2.6	M
WML-FORMREF-C-006	Hidden control initialisation	5.4.2.8	M
WML-FORMREF-C-007	Text input control interaction	5.4.3.2	M
WML-FORMREF-C-008	Menu input control Interaction	5.4.3.3	M
WML-FORMREF-C-009	Checkbox button control interaction	5.4.3.4	M
WML-FORMREF-C-010	Radio button control interaction	5.4.3.5	M
WML-FORMREF-C-011	Submit button control interaction	5.4.3.6	M
WML-FORMREF-C-012	Reset button control interaction	5.4.3.7	M

Item	*FunctionReference*	*Status*	*Requirement*
WML-FORMREF-C-013	Committing form data	5.4.4	M
WML-FORMREF-C-014	Form submission	5.4.5	M

Attribute Expression Syntax

Item	*FunctionReference*	*Status*	*Requirement*
WML-ATTREXPR-C-001	Attribute expression syntax for WML2	5.5	M
WML-ATTREXPR-C-002	No attribute expression syntax for XHTML Mobile Profile	5.5	M
WML-ATTREXPR-C-003	Variable reference	5.5.1.1	M
WML-ATTREXPR-C-004	Parsing the variable reference syntax	5.5.1.2	M
WML-ATTREXPR-C-005	Variable scoping	5.5.1.4	M
WML-ATTREXPR-C-006	Invalid variable references	5.5.1.5	M
WML-ATTREXPR-C-007	Event parameters	5.5.1.6	M
WML-ATTREXPR-C-008	Re-generation of presentation document	5.5.1.7	M

www.it-ebooks.info

Event Model

Item	*FunctionReference*	*Status*	*Requirement*
WML-EVENTREF-C-001	Event binding	5.6.3.1	M
WML-EVENTREF-C-002	Bindings for untyped events	5.6.3.2	M
WML-EVENTREF-C-003	Event binding scope	5.6.3.3	M

Identification of Document Types

Item	*FunctionReference*	*Status*	*Requirement*
WML-DOCTYPES-C-001	XHTML Mobile Profile document type	5.7	M
WML-DOCTYPES-C-002	WML2 document type	5.7	M

BACK key

Item	*FunctionReference*	*Status*	*Requirement*
WML-BACKKEY-C-001	BACK key	5.9	M

Timer Processing

Item	*FunctionReference*	*Status*	*Requirement*
WML-TIMERREF-C-001	Timer processing	5.11	M

Acceptance of XHTML

Item	*FunctionReference*	*Status*	*Requirement*
WML-XHTMLBASICC-001	XHTML Basic	5.12	M
WML-XHTMLBASICC-002 XHTMLMP:MCF	XHTML Mobile Profile	5.12	M

www.it-ebooks.info

User Agent Conformance Rules

Item	*FunctionReference*	*Status*	*Requirement*
WML-AGENTCONFC-001	XHTML User Agent Conformance Rules	5.13	M

Structure Module

Item	*FunctionReference*	*Status*	*Requirement*
WML-STRUCTMODC-001	Initialisation in `wml:newcontext`	6.2.1	M
WML-STRUCTMODC-002	Initialisation in `newcontext`	6.2.3	M

Forms Module

Item	*FunctionReference*	*Status*	*Requirement*
WML-FORMSMODC-001	Mask control in `wml:format`	6.5.2.1	M
WML-FORMSMODC-002	Input control in `wml:emptyok`	6.5.2.2	M

Tables Module

Item	*FunctionReference*	*Status*	*Requirement*
WML-TABLESMODC-001	Column control in `wml:columns`	6.6.1	M
WML-TABLESMODC-002	Column alignment using `wml:align`	6.6.1	O

Image Module

Item	*FunctionReference*	*Status*	*Requirement*
WML-IMAGEMOD-C-	Image control in	6.8.1	M

www.it-ebooks.info

Item	*FunctionReference*	*Status*	*Requirement*
001 `wml:localsrc`			

Metainformation Module

Item	*FunctionReference*	*Status*	*Requirement*
WML-METAMOD-C-001	CSS as the default style sheet language	6.9.1	M
WML-METAMOD-S-001	Processing the meta element with a wml:forua attribute	6.9.1	O

Events Module

Item	*FunctionReference*	*Status*	*Requirement*
WML-EVENTSMODC-001	Binding in `wml:onevent`	6.15.1	M

Context and Navigation Module

Item	*FunctionReference*	*Status*	*Requirement*
WMLCONTEXTMOD-C-001	Task activation in `wml:anchor`	6.16.1	M
WMLCONTEXTMOD-C-002	Access control in `wml:access`	6.16.2	M
WMLCONTEXTMOD-C-003	Task activation in `wml:go`	6.16.4	M
WMLCONTEXTMOD-C-004	No operation in `wml:noop`	6.16.5	M
WMLCONTEXTMOD-C-005	Backward navigation with `wml:prev`	6.16.6	M
WMLCONTEXTMOD-C-006	Display update with `wml:refresh`	6.16.7	M
WMLCONTEXTMOD-C-007	Submission of data with `wml:postfield`	6.16.8	M
WMLCONTEXTMOD-C-008	Variable assignment in `wml:setvar`	6.16.9	M
WMLCONTEXTMOD-C-09	Variable reference in `wml:getvar`	6.16.10	M
WMLCONTEXTMOD-C-010	Timer control in `wml:timer`	6.16.11	M

www.it-ebooks.info

Use of Style Sheets

Item	*FunctionReference*	*Status*	*Requirement*
WML-USESTYLE-C-001 WCSS:MCF AND WMLUSESTYLE-C-002 AND WMLUSESTYLE-C-003 AND WMLUSESTYLE-C-004	Support for WAP CSS	7	O
WML-USESTYLE-C-002	Handling of type "text/css" for external style sheet	7.1	O
WML-USESTYLE-C-003	Handling of type "text/css" for internal style sheet	7.1	O
WML-USESTYLE-C-004	Default type "text/css" for inline style rules	7.1	O

Appendix E. Change History (Informative)

Type of Change Date Section Description

Class 0 11-Sep-2001 The initial version of this document.

✦ ✦ ✦

www.it-ebooks.info



JavaScript Reference

The JavaScript programming language is one of the key tools any Web designer has to use. This appendix provides a reference to the key points of the language.

Reserved Words

Reserved words are those that you cannot use for such things as variable names or function names when writing a JavaScript program. In some cases, like `if` or `for`, the word is already in active use as a part of the JavaScript language. Others are set aside for possible future inclusion.

abstract	else	instanceof	switch
boolean	enum	int	synchronized
break	export	interface	this
byte	extends	long	throw
case	false	native	throws
catch	final	new	transient
char	finally	null	true
class	float	package	try
const	for	private	typeof
continue	function	protected	var
debugger	goto	public	void
default	if	return	volatile
delete	implements	short	while
do	import	static	with
double	in	super	

Objects

Objects are the key to JavaScript functionality. Each element on a Web page is a separate object, and each of these objects has its own characteristics in the form of properties, methods, and event handlers.

www.it-ebooks.info



See Chapter 16 for more information on using JavaScript on your Web pages.

Array

Properties: constructor, index, input, length, prototype

Methods: concat, join, pop, push, reverse, shift, slice, splice, sort, toSource, toString, unshift, valueOf

Boolean

Properties: constructor, prototype

Methods: toSource, toString, valueOf

Date

Properties: constructor, prototype

Methods: getDate, getDay, getFullYear, getHours, getMilliseconds, getMinutes, getMonth, getSeconds, getTime, getTimezoneOffset, getUTCDate, getUTCDay, getUTCFullYear, getUTCHours, getUTCMilliseconds, getUTCMinutes, getUTCMonth, getUTCSeconds, getYear, parse, setDate, setFullYear, setHours, setMilliseconds, setMinutes, setMonth, setSeconds, setTime, setUTCDate, setUTCFullYear, setUTCHours, setUTCMilliseconds, setUTCMinutes, setUTCMonth, setUTCSeconds, setYear, toGMTString, toLocaleString, toSource, toString, toUTCString, UTC, valueOf

Function

Properties: arguments, arguments.callee, arguments.caller, arguments.length, arity, constructor, length, prototype

Methods: apply, call, toSource, toString, valueOf

java

Properties: See packages

Methods: See packages

JavaArray

Properties: length

Methods: toString

www.it-ebooks.info

JavaClass

Properties: See packages

Methods: See packages

JavaObject

Properties: Determined by the Java applet

Methods: Determined by the Java applet

JavaPackage

Properties: Determined by the Java applet

Methods: Determined by the Java applet

Math

Properties: E, LN10, LN2, LOG10E, LOG2E, PI, SQRT1_2, SQRT2

Methods: abs, acos, asin, atan, atan2, ceil, cos, exp, floor, log, max, min, pow, random, round, sin, sqrt, tan

netscape

Properties: See packages

Methods: See packages

Number

Properties: constructor, MAX_VALUE, MIN_VALUE, NaN, NEGATIVE_INFINITY, POSITIVE_INFINITY, prototype

Methods: toExponential, toFixed, toPrecision, toSource, toString, valueOf

Object

Properties: constructor, prototype

Methods: eval, toSource, toString, unwatch, valueOf, watch

www.it-ebooks.info

Packages

Properties: className, java, netscape, sun

Methods: Determined by the Java applet

RegExp

Properties: constructor, global, ignoreCase, lastIndex, multiline, prototype, source

Methods: exec, test, toSource, toString

String

Properties: constructor, length, prototype

Methods: anchor, big, blink, bold, charAt, charCodeAt, concat, fixed, fontcolor, fontsize, fromCharCode, indexOf, italics, lastIndexOf, link, match, replace, search, slice, small, split, strike, sub, substr, substring, sup, toLowerCase, toSource, toString, toUpperCase, valueOf

sun

Properties: See packages

Methods: See packages

✦ ✦ ✦

www.it-ebooks.info

Glossary

acceptable use policy A policy that sets forth the types of user behavior accepted by the management of a Web site.

Acrobat See *Portable Document Format*.

Active Server Page (ASP) A Web page created with HTML and either VBScript or JScript (The Microsoft version of JavaScript). Commands on Active Server Pages are processed with the ActiveX engine. See also *HTML*, *VBScript*, *JavaScript*, and *ActiveX*.

ActiveX A system developed by Microsoft to enable a Web page using ActiveX controls to interface with the Windows operating system.

affiliate A freelance marketing agent, who receives compensation from a company for referring potential customers.

alias An e-mail address that has no mailbox of its own and simply relays messages sent to it to another address that does have a mailbox.

anonymous FTP File access using the file transfer protocol, which is open to the public and does not require a registered user name and password. The user name is given as *anonymous* and the password is either the user's e-mail address or the word *guest.*

ASP See *Active Server Page*.

attachment A file that is sent along with an e-mail message. Most e-mail clients will place an attachment into a separate directory from the one in which the e-mail messages are stored.

attribute A method of modifying a markup language element, or of supplying necessary information for its function. The IMG element's `src` attribute, for example, provides the location of the image file to be displayed.

www.it-ebooks.info

authentication Verifying the identity of a person or service, usually done through digital IDs.

autoresponder A program that automatically responds to an e-mail message by sending back a preprogrammed e-mail message of its own. Autoresponders are useful for providing price lists, newsletters, and so on.

axis In three-dimensional (3D) graphics, the three directions are each indicated by an axis. The typical coordinate system uses an x-axis that runs horizontally, a y-axis that runs vertically, and a z-axis that runs, essentially, between the front and back of the monitor.

B2B Business-to-business e-commerce.

B2C Business-to-consumer e-commerce.

bandwidth The total amount of data that a particular network connection can handle. The higher the bandwidth, the more data can be sent in the same amount of time.

banner An image with a link associated that shows on a Web page (generally at the top of the page) and contains advertising for a sponsor.

binary file A program file. When transferring files with FTP, you must be careful to send program files as binary files rather than text files because text files have a different format.

browser See *Web browser.*

carbon copy Sending an e-mail message to more than one person at a time. The person you are sending the e-mail message to is aware of this. If you want to send a carbon copy without the original recipient knowing about it, you send a blind carbon copy. The regular kind is usually abbreviated *cc* and the blind version is abbreviated *bcc.*

cascading style sheets A method of specifying the appearance of elements on a Web page. Style sheets may be either internal or external. Internal style sheets only apply to the Web page in which they are located. External style sheets are separate files that may be linked to any number of Web pages.

CGI See *Common Gateway Interface.*

character codes and entities A numerical or mnemonic code for referencing special characters—that is, those that are not found on a typical keyboard. The ñ character, for instance, is not found on most non-Spanish keyboards. It can, however, still be entered by using the character entity `ñ` or the character code `ñ` instead.

click-through The action of a site visitor in clicking on an advertising link to visit the advertiser's site.

www.it-ebooks.info

client/server A computer that provides resources to other computers over a network is said to be providing a network *service* and is referred to as a *server*. The computers that use the services of a server are called *clients*. The term *server* is also used to refer to the software that provides the service, and the software that accesses the server is also called the *client* or *client agent*.

colocation Similar to leasing a dedicated server. The server is located on a site that provides technical services to you (such as network monitoring), but you own the server instead of leasing it. You may provide your own equipment or purchase it from the colocation firm.

Common Gateway Interface Also called *CGI*. A technique for sending information gathered by a Web form and passing it to a program for processing.

configuration The process of setting options in a program. Also, the set of options chosen.

cookie A text file that a Web site places on a visitor's computer. Cookies typically contain information such as an identification code, date of last visit, visitor preferences, etc.

CSS See *cascading style sheets*.

dedicated connection A permanent, 24-hour-a-day Internet connection. A dedicated connection can be a telephone line that is used for nothing else but a modem that connects to a local *ISP,* but it is usually a high-speed connection installed for the specific purpose of Internet usage.

dedicated server A computer that is reserved for the use of a single person, company, or organization. See also *virtual server*.

default The standard or out-of-the-box software or hardware settings. In most cases, you can change the default settings to suit your own preferences.

demographics Literally, "writing about people." Information you gather from the people who visit your site. Commonly, demographics include such things as computer system used, income level, and any other information you think you require to understand your audience. Be careful about gathering data on age, race, sex, ethnicity, religious persuasion, and the like, because this can lead to discrimination charges. Make sure that if you do ask for any of this type of information, you make it clear that such disclosures are purely optional. And never make any of this information available to anyone else, except as a lump sum total. The individual input should be held absolutely confidential.

DHTML See *Dynamic HTML*.

dial-up connection A telephone line by which occasional connections are made to the Internet through a modem call to a local ISP.

digital ID An encrypted identifier that is kept on file at a central repository called a *Certificate Authority,* which certifies that you are who you say you are (just in case you were beginning to wonder).

www.it-ebooks.info

DNS See *Domain Name System.*

Document Object Model (DOM) A means of thinking of Web pages as a collection of objects in which each element is a separate object and each is subordinate to some higher object.

Domain Name System (DNS) The Domain Name System allows human-readable Internet addresses to be used in place of the numerical addresses recognized by computers. For example, the URL `http://207.46.230.229` is meaningless to the average Web surfer but, thanks to DNS, it is the same as `http://www.microsoft.com`.

download To get a file transferred from another computer to yours. Also, the file that is downloaded. See also *upload.*

Dynamic HTML The combination of HTML along with JavaScript and Cascading Style Sheets to create a Web page with dynamic elements. See also *JavaScript* and *Cascading Style Sheets*.

e-commerce Business conducted on the World Wide Web or through intranets. Although some elements of e-commerce may contain traditional bricks-and-mortar features such as mail order forms, it takes place primarily through the Internet or some similar network system.

element The entire set of information describing an instruction in a markup language such as HTML. The element consists of the start tag, the end tag, and everything in between. *Block-level* elements are those that automatically create a line break after them. *Inline* elements are those that don't create a line break after them. They're sometimes called text-level elements because many of them deal with the appearance of textual characters. See also *tag.*

e-mail Messages that are sent through a local network or the Internet from one computer to another. When sent across the Internet, e-mail is addressed to the user as `username@hostname`. The *username* part is—you guessed it—the person's user name, and the *hostname* part is the fully qualified domain name of the user's host computer. So, if you want to send an e-mail message to someone whose user name is ralph123 and who maintains an account on toymouse.com, you'd address it to `ralph123@toymouse.com` (the @ means *at*).

encryption The encoding of data (such as e-mail messages or personal information entered into online order forms) so that they cannot be read by anyone who does not have the key to the code.

Ethernet Ethernet is the most widely installed local area network (LAN) technology. Specified in a standard, IEEE 802.3, Ethernet was created by Xerox and then co-developed by Xerox, DEC, and Intel. An Ethernet LAN typically uses coaxial cable or special grades of twisted pair wires. The most commonly installed Ethernet systems are called 10BASE-T and provide transmission speeds up to 10 Mbps.

eXtensible Hypertext Markup Language (XHTML) XHTML is a reformulation of HTML 4.01 that conforms to the requirements of XML. See also *Hypertext Markup Language* and *eXtensible Markup Language.*

www.it-ebooks.info

eXtensible Markup Language (XML) A language similar to HTML. XML, however, is designed to specify the meaning of the content of a Web page while avoiding any description of its appearance. XML is meant to work in concert with CSS, which is designed to control the appearance of Web pages. See also *cascading style sheets*, *Hypertext Markup Language,* and *eXtensible Hypertext Markup Language*.

Extensible Style Language (XSL) A language for adding style sheets to XML documents. The relationship between XML and XSL is the same as between HTML and CSS. See also *eXtensible Markup Language*, *Hypertext Markup Language*, and *cascading ctyle sheets*.

feed The transmission of messages from one news server to another.

File Transfer Protocol Also called *FTP.* FTP is a method for transferring files from one computer to another across a network or the Internet.

flame A message that contains insults or inflammatory material.

flame war An extended exchange of messages containing flames. Flame wars disrupt the normal process of communication in mailing lists or newsgroups.

frames and framesets A method of displaying multiple Web pages (or other files capable of being displayed in a Web browser) on a single screen simultaneously. A frameset is the master document, which holds the frames. The frames are the individual windows in which the files are displayed.

FTP See *File Transfer Protocol.*

GIF Images that are stored in files that follow the standards of the Graphics Interchange Format. GIFs are commonly used for limited animation.

Gopher An early method, predating the World Wide Web, for locating and accessing resources on the Internet. The name comes from the fact that you would tell it what resources you wanted to see and it would "go for" them.

header The top part of an e-mail message, which contains information showing the date and time it was sent, what path it took to reach you, and lots of other information of interest only to the true e-mail maniac. Most e-mail client programs enable you to set an option that relieves you of ever having to see header information.

hit A single access of a Web page.

home page The main Web page from which all other pages on the site branch out. It is commonly called `index.html`, although this is not a requirement. In fact, any name will work, although using another one may force visitors to face a listing of available files instead of instantly viewing your home page in their Web browser, then to figure out which file among those listed is the home page. Other common names include `index.htm` and `default.htm`. The difference between the `.htm` and `.html` extensions is practically nonexistent, by the way, going back to the days when the early DOS operating system only recognized three-letter extensions. All

www.it-ebooks.info

modern operating systems are capable of recognizing longer file type extensions, and few professionals limit themselves to the old DOS conventions anymore.

horizontal rule The `HR` element, which creates a horizontal line on a Web page.

HTML See *Hypertext Markup Language.*

HTML editor Specialized text editor for writing Web pages.

HTTP See *Hypertext Transfer Protocol.*

hyperlink See *link.*

hypertext Text available on a computer that includes links to other text, images, sound, and so on.

Hypertext Markup Language (HTML) The native language of the World Wide Web which is used to construct Web pages.

Hypertext Transfer Protocol (HTTP) A system for transferring files across the World Wide Web.

IETF See *Internet Engineering Task Force.*

image The general term for any picture or artwork included on a Web page. Images are usually stored in either GIF or JPEG format.

image map An image that contains multiple links. Users viewing an image map select different links (with mouse-oriented systems) by moving the pointer to different parts of the image and clicking. The images used with image maps must be carefully designed to provide visual clues to the contained links.

inactivity timeout The period of time after which a connection will be severed automatically if no activity occurs.

Internet Short for *internetwork*, which means to link many networks. Sometimes called the *network of networks,* the Internet was created—and continues to grow—by linking computer networks so that they can all communicate with one another.

Internet access The capability to connect to the Internet. Usually provided by an Internet service provider (ISP).

Internet Engineering Task Force (IETF) The group that is responsible for keeping track of existing Internet standards and helping create new ones. Anyone can join the IETF for free and participate directly in the creation of new Internet standards. Visit the following Web site for more information: `www.ietf.org`.

Internet Presence Provider (IPP) A company that provides servers for the use of paying customers. Most IPPs are also ISPs.

Internet Relay Chat Live, online interaction between people. The interaction takes place as a kind of live e-mail, with the participants typing messages that are instantly

www.it-ebooks.info

seen by all others who are in a particular *chat room*. You first need to use a program such as mIRC or Microsoft Chat to log onto an IRC server; then pick a chat room from the list of available ones (or create your own).

Internet service provider (ISP) A company that provides Internet access, usually by dial-up connections, although many also offer high-speed dedicated connections.

Internet site A collection of computers that are all permanently connected to the Internet and that appear to the outside world to be related to one another.

Internet standards Worldwide agreements about how Internet technology should function.

Internet telephony The Internet is essentially a telephone system connecting computers. With the addition of a microphone, speakers, and a sound card to your computer, programs such as Internet Phone and FreeTel make it possible to carry on conversations over the Internet with anyone who has a similarly equipped computer anywhere in the world. Although the sound quality falls short of what you're used to on a regular telephone, unlimited worldwide telephone service for the price of a local Internet connection is hard to beat. And the sound quality is improving all the time as Internet telephony catches on. It goes without saying that the telephone companies are somewhat upset about this development.

intranet A network consisting of several computer networks linked together. Unlike the Internet, however, an intranet is private and access to it is limited to its membership. Intranets are commonly used in B2B e-commerce situations.

IP address A unique numerical identifier for Internet sites. An IP address is a set of four numbers separated by periods or dots (also called a *dotted quad*). An IP address appears as follows: 207.46.230.229

IPP See *Internet Presence Provider.*

IRC server See *Internet Relay Chat.*

ISP See *Internet service provider.*

JavaScript A relatively simple programming language designed to add interactivity and action to Web pages. JavaScript works only within Web browsers.

JPEG Also spelled *JPG.* Images that are stored in files that follow the standards of the Joint Photographic Experts Group. JPEG files are generally smaller than other file formats, but they maintain a high-quality image nonetheless.

JPG See *JPEG.*

jukebox A CD-ROM drive that holds multiple discs simultaneously.

keyword A search term supplied by a person to a search engine, which then looks for Internet resources that contain that term. Also, terms that are included on a Web page so that search engines can properly categorize that page.

www.it-ebooks.info

link Also called a *hyperlink,* a link is a connection between a Web page and another Internet resource. Usually, a link contains the URL (Internet address) of another Web page, but it can also link to other resources, such as images. By convention, links are displayed as blue, underlined text and, when activated (usually by a mouse click), cause a Web browser to load the linked resource. Images are often used as links; by convention, such image links are surrounded by a thin blue border to differentiate them from normal images. Image maps, which are images that contain more than one link, are an exception to the blue border rule.

list owner The person who manages a mailing list. Usually, the list owner is the person who conceived and launched the list. Also called *list manager* or the *moderator.* See also *moderator.*

list server A program that processes the e-mail messages that comprise a mailing list. The list server usually handles such routine administrative matters as subscribing and unsubscribing list members when they send a message containing the appropriate commands.

log file A file that records activity. A Web server log file, for instance, records the origin of anyone who looks at a Web page, the user's browser type, and so on. Log file analyzer programs such as Statbot are useful to Webmasters in figuring out peak times of usage, number of hits per day, where visitors were referred from, and so forth.

mail filtering The process of automatically taking action on e-mail messages as they come in. This allows you to set rules that, for instance, put all mail with a particular word in the subject line into a different folder, or that delete mail from a particular person before you even see it.

mail server A program that receives, stores, and forwards e-mail messages.

mailbox The storage area where your e-mail messages are kept until you retrieve them.

mailing list A discussion group, the members of which post messages that are relayed through e-mail by a list server. Mailing list membership can be either open or closed.

markup language A computer programming language that is dedicated to the layout and function of elements on screen. HTML is the best-known markup language, although there are many others for specialized purposes, such as the Astronomical Markup Language (ASML), the Bioinformatic Sequence Markup Language (BSML), and the Math Markup Language (MATHML), as well as some which hope to compete with HTML to become the new lingua franca of Web development authors. See also *eXtensible Markup Language*, *eXtensible Hypertext Markup Language*, and *Hypertext Markup Language*.

marquee A scrolling message reminiscent of those that once were common on the signs (marquees) outside theaters.

www.it-ebooks.info

message board Similar to a newsgroup, but operates on an individual Web site. Message boards, sometimes called *graffiti walls*, enable site visitors to leave messages for other visitors to read. The more sophisticated message board programs include such niceties as message threading and keyword search capabilities.

metadata Information beyond what is necessary for the display of a Web page. Common uses of metadata, for example, are the inclusion of author name and content description in the `HEAD` section of a Web page.

modem *Mo*dulator/*dem*odulator. Device that allows a computer to send signals over a telephone line.

moderator A person who controls the content of a mailing list or newsgroup. In extreme cases, this can amount to censorship, but moderate moderation simply results in the removal of extraneous material such as sales pitches for products and services that are unrelated to the topic of the mailing list or newsgroup. A moderator may also play peacemaker, attempting to keep tempers in check among the members of the mailing list or newsgroup, thus avoiding flame wars.

multimedia The use of more than one kind of medium in a single application or Web page.

name server A computer that maintains a database of domain names and IP addresses. See also *Domain Name System.*

netiquette Inter*net* et*iquette*. A series of customs that has evolved to enable people to get along while living together in cyberspace. Basically, be nice and consider how other people might be affected by your actions.

network A collection of computers and other devices (such as printers) that are connected for the purpose of sharing resources.

network adapter A computer card that enables a computer to be connected to a network.

network hub A device that is the central point for relaying signals between parts of a network.

Network News Transfer Protocol The system by which newsgroup messages are transferred between news servers.

network node Any device on a network.

network repeater A signal amplifier that enables devices on a network to be located farther apart than usual.

newsgroup A discussion group, the members of which post messages that are relayed through a news server. Newsgroups can be either local or global. Global newsgroups are open to the public.

www.it-ebooks.info

news reader A program that is used to connect to a news server. News readers enable people to read messages and download images from newsgroups.

news server A program that receives, stores, and forwards newsgroup messages. The server that is sending the message is called the *upstream* server; the one that is receiving it is called the *downstream* server.

NNTP See *Network News Transfer Protocol.*

online service Sort of a mini-Internet; services such as CompuServe, America Online, and so forth provide many of the same functions the Internet does, but on a smaller scale. In recent years, the online services lost much business to the Internet and decided to provide Internet access as a part of their services.

packet Packets are like little envelopes sent by traditional mail. Each packet contains an address, just as any paper envelope does, and some amount of information.

page generation program Software that uses a WYSIWYG, point-and-click interface for creating Web pages.

password A secret word that is used to gain computer access. Passwords are sometimes chosen, sometimes assigned. There are two schools of thought on passwords. The first is that passwords should be so difficult to imagine that no one else could possibly guess yours in a million years. These people want everyone to have a password like *109834akjd30d9dfnap33o78dsjfa23sdorf.* While it's true that it's not a good idea to use your dog's name for your password, this approach seems a bit extreme (most passwords are between 4 and 10 characters, anyway). The other school of thought is that you should be able to remember your password. If you only have one or two passwords, it's no big deal, regardless of their complexity; you'd be surprised how quickly a random series of numbers can become familiar. One technique favored by many people is to combine two unrelated but easy to remember words, putting a couple of numbers between them, as in *music42bank.* Just make sure that the terms have no connection to you. This is a fine password for a dog breeder, but a lousy one for a banker who plays jazz gigs after hours, especially if he is 42 years old.

POP See *Post Office Protocol.*

port numbers Part of the IP address, port numbers are unique for a particular IP address, meaning that two information services can't both use port 80 on network node `207.92.75.100`. But port numbers are not unique globally. The network node at IP address `207.92.75.100` can provide an information service by using port 80, and at the same time, IP address `207.92.75.101` can provide a service by using port 80.

Together, an IP address and a port number identify a unique network node and a unique information service on the network node, respectively.

PDF See *Portable Document Format.*

www.it-ebooks.info

popup A separate window in which a Web browser displays additional information not found on the Web page which the visitor originally accessed. Popup windows are used for a variety of purposes, some considered legitimate, while others are not. Popup windows, in their "best and highest usage," are useful adjuncts to a normal Web page, adding helpful information or additional capabilities. Their common usage as advertising billboards, however, has given them a bad reputation, and programs called *popup blockers*, which prevent all popup windows from materializing, are becoming increasingly popular, so savvy Web designers should look for ways to avoid using popups.

Portable Document Format A proprietary document display format developed by Adobe Systems, Inc. Although it allows people to easily convert existing documents to being viewable on the Web, it requires a separate plug-in in order to allow visitors to view those documents.

Post Office Protocol The method by which e-mail clients pick up e-mail from a mail server.

privacy policy The set of rules by which a Web site agrees to treat information provided by its visitors as confidential, or by which it states that it shares such information with other organizations.

privileges Limitations on access or the types of commands available to various people or groups. *Root privileges,* for example, are available only to the person in charge of a server because root privileges allow anyone who has them to modify the system.

protocol In information technology, a protocol (from the Greek *protocollon*, which was a leaf of paper glued to a manuscript volume, describing its contents) is the special set of rules that end points in a telecommunication connection use when they communicate. Protocols exist at several levels in a telecommunication connection. There are hardware telephone protocols. There are protocols between each of several functional layers and the corresponding layers at the other end of a communication. Both end points must recognize and observe a protocol. Protocols are often described in an industry or international standard.

On the Internet, there are the TCP/IP protocols, consisting of:

- Transmission Control Protocol (TCP), which uses a set of rules to exchange messages with other Internet points at the information packet level.
- Internet Protocol (IP), which uses a set of rules to send and receive messages at the Internet address level.
- Additional protocols that are usually packaged with TCP/IP, including the Hypertext Transfer Protocol (HTTP) and File Transfer Protocol (FTP), each with defined sets of rules to use with corresponding programs elsewhere on the Internet.

ray tracing A method of producing images that follows rays of (theoretical) light as they pass through a (hypothetical) scene and reproduces the effects that would be produced by those light rays if the scene actually existed.

www.it-ebooks.info

reciprocal link A link to a Web page that also has a link back to your own Web page. Hyperlinking is a common practice for building traffic among Web sites. If you want to establish a reciprocal link with someone else's site, put in a link to their site on your own, and then send them an e-mail message asking them to reciprocate.

registrar An organization that is authorized to register domain names.

ring A series of Web sites that are all interconnected with links that lead from one to the other until a visitor who follows all the links has gone full circle and returned to his or her starting place.

rollover A JavaScript technique in which one image is swapped for another when a mouse pointer is placed over the first image. See *JavaScript*.

search engine A program that checks through a database of terms to see which Internet resources contain them. Local search engines are a variation of this; they actually read every page in a Web site to see if they contain the search term.

Secure Sockets Layer (SSL) An encryption system originally developed by Netscape that allows visitors to enter information into online forms and submit that information without allowing anyone else to read it. SSL is commonly used in processing order forms and other online forms that contain private information.

server See c*lient/server.*

SGML See *Standard Generalized Markup Language.*

shopping cart A program that permits people visiting your site to choose which items they will purchase. The shopping cart metaphor extends to the capability to put items into the cart and remove them before making the final purchasing decision.

Simple Mail Transfer Protocol (SMTP) A common method for sending e-mail between mail servers.

SMTP See *Simple Mail Transfer Protocol.*

spam Unsolicited e-mail messages sent in bulk quantities to several addresses. Also, the same message posted to multiple newsgroups simultaneously.

SSL See *Secure Socket Layers*.

standard Generic term for just about anything that lots of people agree on. It may be a way to transfer files from one system to another one, the exact structure and function of a computer language, or a method of compressing video images. If a recognized, organized group develops it, it's called an *official standard.* If it's just something developed by the market leader, and it's used by most of the people in a particular field, it's called a *de facto standard.*

Standard Generalized Markup Language (SGML) A method for describing markup languages such as HTML.

www.it-ebooks.info

style sheets See *cascading style sheets.*

subscriber A member of a mailing list. Also called, amazingly enough, *list member.*

tag The delimiter for the beginning or ending of an element. For example, `<P>` is the start tag for the `P` element, and `</P>` is its end tag.

TCP/IP See *Transmission Control Protocol/Internet Protocol.*

telecommuting Working from home through remote computer access.

Telnet A method of remote access to a computer. Enables users to type in commands just as though they were at the terminal of the remote computer.

text editor A simple form of word processor that saves text files without formatting.

text file A file of text. (No kidding.) When transferring text files through FTP, you should be careful to send them as text files rather than as binary files because text files often need to be converted between Windows and Unix systems, which have slightly different text formats.

traffic The number of visits to an Internet site, as in "Our traffic increased by 100,000 visitors last week."

Transmission Control Protocol/Internet Protocol (TCP/IP) The standard way for data to be relayed around the world on the Internet. Many, but not all, computers on the Internet speak TCP/IP directly. Other computers speak a different networking language and rely on translation equipment to convert between TCP/IP and their native network language.

Uniform Resource Locator (URL) Each bit of information on the Internet is known as a *resource.* A URL specifies the type of software that is used to handle the resource, the computer on which the resource is located, and the name of the resource. The URL adds a specific resource name, such as a file name, to the IP address and port-number combination. The first part is called a *prefix.* The following table shows some common prefixes.

Prefix	*Information Service*	*Example URL*
`http://`	World Wide Web	`http://www.wiley.com/`
`ftp://`	File Transfer Protocol	`ftp://ftp.microsoft.com/`
`mailto:`	E-mail address	`mailto:webmaster@idgbooks.com`

Unix A common operating system on the Internet.

upload To get a file transferred from your computer to another one. Also, the file that is uploaded. See also *download.*

www.it-ebooks.info

URL See *Uniform Resource Locator.*

user name Also called *user id.* The name by which a person is identified on a particular computer system. Usually some variant on their given name.

VBScript The Visual Basic scripting language from Microsoft. As the name suggests, it is a version of the simple BASIC programming language, but geared toward Web site development rather than general software development.

virtual server A directory on a computer that simulates the existence of a separate computer. See also *dedicated server.*

Web See *World Wide Web (WWW).*

Web browser A program that displays Web pages and follows links to enable a user to traverse the World Wide Web.

Webmaster The person in control of a Web site. A Webmaster may be an individual with only a single *Web page,* or the manager of a large team of developers who create and maintain Web sites.

Web page Also called *HTML page.* A display composed with the Hypertext Markup Language that is viewable through a Web browser.

Web-safe colors A set of 216 colors that displays the same on any computer system.

Web site Any presence on the World Wide Web. Usually applied to a collection of Web pages.

Web space The amount of room for holding Web sites that is found on a drive connected to the World Wide Web.

World Wide Web (WWW) The total selection of Web sites interconnected via the Internet.

WWW See *World Wide Web*.

x-axis See *axis*.

XHTML See *eXtensible Hypertext Markup Language.*

XML See *eXtensible Markup Language.*

XSL See *eXtensible Style Language*.

y-axis See *axis*.

z-axis See *axis*.

z-order The relative “depth” of elements on a Web page.

www.it-ebooks.info

Index

A

absolute positioning, 317–319
  calculating, 435–437
absolute size, 250
absolute URL, 72
acceptable use policy, 869
access keys, 360–361
Acrobat, 869
ACS (Automated Composing System), 491, 492
ActionScript, programming with, 477–478
Active Server Page (ASP)
  defined, 869
  handling of form data by, 362
ActiveX, 869
adding
  color(s), 195–207
  pointers to navigation bars, 413–415
  Web page navigation bars, 410–412
address alias, 869
address extensions
  .biz, 80
  .com, 80
  .coop, 80
  .gov, 80
  .info, 80
  .mil, 80
  .name, 80
  .net, 80
address verification system (AVS), 575
Adobe Acrobat, 869. *See also* PDF (Portable Document Format) files
Adobe Photoshop. *See* Photoshop
advertising
  accepting on personal Web sites, 607
  affiliate marketing, 589–590
  with banners, 17, 32–35, 587–588, 607, 608
  via e-mail, 590–591
  with Google, 586, 579–585
  hover, 592
  Internet advertising agencies, 588–589
  mailing lists, 591–592
  multimedia ads, 34
  newsletters, 592, 593
  online, typical costs per month, 33
  with search engines other than Google, 585–587
  traditional vehicles for, 35
AdWords accounts (Google), 579–585, 586
affiliate, 869
affiliate marketing, 589–590
affiliate stores, 544, 545
alias, 869
All the Web (search engine), 20
AltaVista (search engine), 18, 20
ALT tag, 22–23
analogous colors, 205
analyzing Web site types
  commercial sites, 3, 8–10
  informational sites, 4–6
  organizational sites, 6–7
  personal sites, 3, 4
  political sites, 4, 8, 9
anchors, 76
animation
  hazards of, 429–430
  using with Macromedia Flash, 449–485
Animation Timeline (Macromedia Flash)
  adding actions, 478–479
  adding layers, 470, 471
  adding MP3 sound, 477
  adding objects to layers, 470, 471
  adding text, 466
  as an advertising format, 34
  Animation Timelines, 469–477
  assigning keyframes, 470
  bitmap fills, 465–466
  choosing fonts, 466–467
  converting objects to symbols, 470, 472
  creating animation, 472
  creating layer effects, 473–475
  creating objects, 454, 456–460
  creating text effects, 468–469
  Default panel, 455
  described, 449
  design panels, 455
  development panels, 456
  exporting movies, 479–481
  frames, 469–470
  gradient fills, 463–464
  grouping objects, 461–462
  importing art, 459–460
  importing sound files into, 476
  launching for the first time, 449
  layers, 451, 470, 473–475
  layout, components of, 449
  Library panel, 460

www.it-ebooks.info

Animation Timeline (Macromedia Flash) *(continued)*
Line tools, 456, 457
making shapes with tools, 458–459
modifying objects, 460–466
objects to symbols, converting, 470, 472
optimizing movies, tips for, 483–485
panels, 454–456
programming with ActionScript, 477–478
publishing movies, 481–483
rotating and skewing objects, 463
saving work, 459
scaling objects, 462–463
scenes, 450–451
selecting and deleting objects, 457–458
setting font characteristics, 467–468
setting stroke characteristics, 456–457
sound and, 476
straightening and smoothing objects, 463
stretching and distorting objects, 460–461
symbols, 470
synchronizing sound and action, 477
the Stage, 450
the Timeline, 451–452
Tools panel, 452–454
tweening, 472–473
anonymous FTP, 869
Anti-Cybersquatting Consumer Protection Act, 598
Anti-spam laws, 591
AOL's Hometown Web-building site, 518–519
applets, Java, 498–506
adding alternative content, 499
to create link lists, 423–425
defined, 498
as objects, 502–505
parameters, 498, 500, 506
sources of, 505
using, 499–502
applet parameters, Java, 498, 500, 506
applying artistic effects filters in Macromedia Fireworks, 186
arithmetic operator, 370–371
art. *See* images
artistic effects filters, 184–187
Ask Jeeves (search engine), 20, 587
ASP (Active Server Page)
defined, 869
handling of form data by, 362
assignment operator, 366–367
attachment, 869
attribute(s),
color(s),195–196
common, 40–41
core, 40
defaults and setting values, 42
defined, 869
event, 41
i18n, 41
specialized, 42
auctions online (eBay), 539–569
after the auction, 565–566
Buy Now option, 560
checks as payment option, 562–563
debit cards, 541, 571–573
eBay template, 566–567
enhancing success with, 566
establishing yourself as a reputable dealer, 564–565
fees for selling on, 552–554
modifying the eBay template, 567
monitoring your auction, 565
overview of, 549–550
payment and shipping options, 561–564
PayPal accounts, 561–562
setting a start and finish time, 565
setting up a Dutch (multiple item) auction, 560–576
setting up an account, 551–552
setting up an auction, 554–560
setting up an eBay store, 568
shipping merchandise, 563
audio, 487–494
Automated Composing System, 491, 492
buying music, 493–494
choosing a file type, 488
digital sources for, 490–494
embedding, 488–490
making your own music, 491–492
setting volume, 490
using NOEMBED, 490
using public domain music, 493
authentication, 870
authoring tools
browser capability testers, 58–59
choosing, 55–57
code validation programs, 58
demos, trying out, 56–57
Dreamweaver, 55, 61–63, 146, 174
freeware, 57
HomeSite+, 60–61
HTML editors, 51
iSpace, 55
link checkers, 58
Notepad, 60
server monitor services, 59–60
shareware, 57
text editors, 49–50

www.it-ebooks.info

types of, 49
word processors, 54–55
WYSIWYG programs, 51–54
Automated Composing System, 491, 492
autoresponders, 97, 870
AVS (address verification system), 575
axis, 870

B

background color
changing, 196–197
defined, 196
layers, 321–322
specifying for tables, 234–240
background images
adding, 163–164
sidebar, with tables, 225–227
specifying for tables, 233–234
backup navigation, 266
bandwidth
defined, 870
unlimited, 94
banners
advertising with, 32–35, 587–588, 607
defined, 870
designing, 17, 35, 588, 607, 608
exchanges, 34–35
standard size, 33
typical charges for, 33
use of, 32–35
barrier syndrome, 409
bars, navigation
adding pointers, 413–415
adding to Web page, 410–412
graphics to enhance, 412–413
horizontal vs. vertical, 417–418
in JavaScript, 415–417
bars, scroll
in frames, 263–265
as a way to control overflow, 326–327
BASE element, 77–78
BASEFONT element
coloring text, 119
comparing font size with heading size, 117
overriding default font faces, 117–119
specifying a size, 116–117
using relative size, 117
Better-Whois.com, 598–599
bidding online (eBay), 539–569
after the auction, 565–566
Buy Now option, 560
checks as payment option, 562–563
debit cards, 541, 571–573
eBay template, 566–567
enhancing success with, 566
establishing yourself as a reputable dealer, 564–565
fees for selling on, 552–554
modifying the eBay template, 567
monitoring your auction, 565
overview of, 549–550
payment and shipping options, 561–564
PayPal accounts, 561–562
setting a start and finish time, 565
setting up a Dutch (multiple item) auction, 560–576
setting up an account, 551–552
setting up an auction, 554–560
setting up an eBay store, 568
shipping merchandise, 563
binary file, 870
bitmap fills, 465–466
.biz extension, 80
BizRate evaluation system, 616–617
block-level elements, 39, 872
BLOCKQUOTE element, indenting with, 122–123
blog
defined, 507
uses for, 508
Blogback, 523
blogBuddy, 524
blogdex, 524
Blogger
adding to a blog, 514–518
Advanced Blog Setup option, 524
BlogThis! option, 518
choosing a title and blog address, 512–513
creating links in, 516
Dashboard, 517
entering initial information into, 510–512
formatting text, 515–516
history of, 509
passwords, 511
posting, 515–516
Settings area, 517
setting up a blog on, 510–514
templates, 513–514
View Blog option, 516–517
blogging
add-ons, 523–524
AOL's Hometown Web-building site, 518–519
Blogger, 509–518
blogs.com, 519
defined, 507
full integration, 524–525
integration by reference, 524

www.it-ebooks.info

blogging *(continued)*
LiveJournal, 520–521
Radio UserLand, 521–523
tools other than Blogger, 518–524
TypePad, 519–520
as a writing style, 524
blogosphere, 507, 509
blogs.com, 519–520
blurring, images, 184
boldface, 112–113
Boolean value, 368
borders
around images, 148–150
colors (frames), 262
as image links, 158–161
margins, 262–263
tables, 217–218
width between frames, 260–262
boxes, list, 346–351
allowing multiple selections, 350–351
creating, 348
purpose of, 346–347
setting menu types, 349–350
specifying values, 349
browser capability testers, 58–59
browser compatibility
with JavaScript, 400–404
testing, 58–59, 404
with XHTML 1.0, 639–640, 656–657
browser sniffing, 404
browsers, Web
compatibility. *See* browser compatibility
defined, 882
handling of images in tables, 233
help balloons, 151
sniffing, 404
support for CSS properties, 270
support for Document Object Model (DOM), 437–441
testing for use with cascading style sheets, 270
tool tips, 151–152
tracking use of, 59
uploading via HTTP, 107
using more than one to test Web projects, 54
versions, programming for, 59
visitor-customized preference settings, 618–621
Bryce, 3D imaging program, 189, 190
B2B (business-to-business) e-commerce, 870
B2C (business-to-consumer) e-commerce, 870
buttons, INPUT, 352–355
custom, 353–354
graphical, 354–355
Reset, 352–353
Submit, 352
bulleted lists, 420, 421
business online (e-commerce), 872
advertising, 579–592, 607
affiliate stores, 544, 545
B2B (business-to-business), 870
B2C (business-to-commerce), 870
chargebacks, 575
checks, 541–542, 543, 576–577
credit cards, 541, 542, 571–573, 575
credit fraud, 575–601–602
digital certificates, 536, 537, 542
e-cash, 542–544, 577
encryption, 601, 603
general stores, 530–531
Internet buying patterns, 544, 546–547
online computer stores, 532, 533
online shopping malls, 534–535, 536–538, 605
online superstores, 531–532
open source software, 538–540
PayPal accounts, 536, 561–562, 577
personal information, asking customers to provide, 603
privacy policies, 603, 604
return and refund policies, 573–575, 603, 605–606
security and, 536, 601–603
selling on eBay, 549–569
service sites, 532, 534
setup, options for, 529
shopping carts, 535–540
single product sites, 529
specialty stores, 529–530
taking payment, 540–544
Yahoo! Store, 536
BUTTON element, 356–357
buying music, 493–494
buying online (eBay), 539–569
after the auction, 565–566
Buy Now option, 560
checks as payment option, 562–563
debit cards, 541, 571–573
eBay template, 566–567
enhancing success with, 566
establishing yourself as a reputable dealer, 564–565
fees for selling on, 552–554
modifying the eBay template, 567
monitoring your auction, 565

www.it-ebooks.info

music, 493–494
overview of, 549–550
payment and shipping options, 561–564
PayPal accounts, 561–562
setting a start and finish time, 565
setting up a Dutch (multiple item) auction, 560–576
setting up an account, 551–552
setting up an auction, 554–560
setting up an eBay store, 568
shipping merchandise, 563

C

calculating absolute positioning, 435–437
capability testers for browser, 58–59
carbon copy, 870
carts, shopping, 535–540
defined, 535–536, 880
commercial software for, 538, 539
online shopping malls, 536–538
open source software for, 538, 540
security, 536
cascading style sheet properties (CSS1), 280–288
backgrounds, 280–281
borders, 281–272
clear, 282–283
color, 283
display, 283
float, 283
fonts, 283–284
height, 284
letter-spacing, 284–285
line-height, 284
list-styles, 285–286
margins, 286
padding, 286
text, 286–287
vertical-align, 287
white-space, 287
width, 288
word-spacing, 288
cascading style sheet properties (CSS2), 288–311
Azimuth, 289
backgrounds, 289–290
borders, 290–291
bottom, 292
caption-side, 293
clear, 293
clip, 293
color, 293
content, 293–294
counter-increment, 294
counter-reset, 294
cues, 294–295
cursor, 295
direction, 295
display, 295
elevation, 295–296
empty-cells, 296
float, 296
fonts, 296–297
height, 298
left, 298
letter-spacing, 298–299
line-height, 299
list-styles, 299–300
margins, 300
marker-offset, 300
marks, 300–301
max-height, 301
max-width, 301
min-height, 301
min-width, 301
orphans, 302
outlines, 302–303
overflow, 303
padding, 303
page, 303
page-breaks, 303–304
pause, 304–305
pitch, 305
play-during, 305
position, 305–306
quotes, 306
richness, 306
right, 306
size, 306
speak, 306–307
speech-rate, 307
stress, 308
table-layout, 308
text, 308–309
top, 309
unicode-bidi, 309
vertical-align, 310
visibility, 310
voice-family, 310
volume, 310
white-space, 310–311
widows, 311
width, 311
word-spacing, 311
z-index, 311
cascading style sheets (CSS)
absolute vs. relative positioning, 317–319
browser support for, 270

www.it-ebooks.info

cascading style sheets (CSS) *(continued)*
building menus with, 277–279
cascading priorities, 270–274
and character styles compared, 111
classes, 275
CSS1 properties, 280–288
CSS2 properties, 288–311
defined, 269, 870
embedded style sheet, 272
ID selectors, 275–276
linking external style sheets, 273–274
nested elements, 276–279
positioning properties, 315–316
redefining HTML elements, 274–275
scroll bars, 326–327
style attribute, 270, 271
style element, 270
style sheet-creation programs, 312
<STYLE> tag, 272
support information sites, 271
uses of, 269, 274
Web sites concerning, 313
XHTML and, 720–721
cell padding, 218–219
cells, defined, 11
cell spacing, 219–220
cell spanning, 229–232
Certificate Authority, 871
CGI (Common Gateway Interface)
defined, 871
handling of form data by, 362
changing background color, 196–197
changing color(s) with rollovers, 432–433
changing text
aligning and indenting, 119–123
aligning images with, 153–157
BASE element, 116–119
BLOCKQUOTE element, indenting with, 122–123
boldfacing, 112–113
cascading style sheets (CSS), 286–287, 308–309
CENTER element, 120
choosing character sets, 123–127
color, 119, 160, 197
character styles, 111–115
default fonts, overriding, 117–119
deprecation and, 120–122
FONT element, 116–119
formatting in Blogger, 515–516
heading elements, 43
horizontal lines to divide, 44–47
italicizing, 112
line breaks, 43–44
in Macromedia Flash, 466–469
as placeholder for missing image, 150–152
preformatting, 113–114
special characters and entities, 127–135
style, maintaining consistency in, 67
subscripts and superscripts, 114–115
white space, 47–48
word wrap around images, 155–157
character codes and entities
common HTML, 128–131
defined, 127–128, 870
HTML technical and math, 131–135
character sets
choosing, 123–127
defined, 123
Eastern hemisphere, 127
ECMA versions, ISO 8859 standards, 126
ISO 8859 family of, 124–125
ISO 10646 (Unicode), 123
Windows-specific, 127
characters, special
common HTML character entities and codes, 128–131
HTML technical and math character entities and codes, 131–135
numbering of, 127–128
character styles, 111–115
boldface, 112–113
defined, 111
italics, 112
preformatting, 113–114
proportionally spaced fonts, 113
superscripts and subscripts, 114–115
chargebacks, 575
chat room, 875
check boxes, 343–344
checks, e-commerce, 541–542, 543, 576–577
choosing authoring tools, 55–57
choosing HTML editors, 51
classes, 275
click-through, 870
click-thru rate, 33
click-thrus, 17
ClickZ Network, 546
client, 871
client agent, 871
clip art, 176
clipping, layers, 322–324
codes and entities, character
common HTML, 128–131
defined, 127–128, 870
HTML technical and math, 131–135
code validation programs, 58
colocation, 84, 88–89, 871

www.it-ebooks.info

color pickers, 200–203
color(s),
adding, 195–207
attributes,195–196
background, 196–197, 234–240, 321–322
cascading style sheets (CSS), 283, 293
changing background, 196–197
changing with rollovers, 432–433
color pickers, 200–203
complementary, 204–205
contrast and visibility, 206
coordinating color schemes, 206
customizing in Internet Explorer, 196
dithering, 200
hexadecimal (hex notation), 198, 498
fades, 445–447
foreground, 197
link, changing, 197–198
named colors, limitations of, 199–200
primary, 198
readibility and, 66
RGB combinations, 198
RGB triplets, decoding, 198–199
and site uniformity, 66
text, 119, 160
URLs for, 207
visibility and contrast, 206
warmth, 206–207
Web-safe, 200–203
color code, 198–199
color names, 198–203
in HTML, 199–200
limitations of, 199–200
color pickers, 200–203
Microsoft Developer Network Web site, 200–201
using, 200–203
VisiBone Webmaster's Color Lab, 201–202
VisionQuest Images VQ Color Site, 201, 202
colors, analogous, 205
color(s) attribute(s), 195–196
color wheel, 205, 206
.com extension, 80
comments, 375
commercial web sites, 8–10
common attributes, 40–41
Common Gateway Interface (CGI)
defined, 871
handling of form data by, 362
comparison operator, 369–370
compatibility testers, 58–59
complementary colors, 204–205
Composing System, Automated (audio), 491, 492
composing your own music, 491–492
computer animation
hazards of, 429–430
using with Macromedia Flash, 449–485
computer Animation Timeline (Macromedia Flash)
adding actions, 478–479
adding layers, 470, 471
adding MP3 sound, 477
adding objects to layers, 470, 471
adding text, 466
as an advertising format, 34
Animation Timelines, 469–477
assigning keyframes, 470
bitmap fills, 465–466
choosing fonts, 466–467
converting objects to symbols, 470, 472
creating animation, 472
creating layer effects, 473–475
creating objects, 454, 456–460
creating text effects, 468–469
Default panel, 455
described, 449
Design panels, 455
Development panels, 456
exporting movies, 479–481
frames, 469–470
gradient fills, 463–464
grouping objects, 461–462
importing art, 459–460
importing sound files into, 476
launching for the first time, 449
layers, 451, 470, 473–475
layout, components of, 449
Library panel, 460
Line tools, 456, 457
making shapes with tools, 458–459
modifying objects, 460–466
objects to symbols, converting, 470, 472
optimizing movies, tips for, 483–485
panels, 454–456
programming with ActionScript, 477–478
publishing movies, 481–483
rotating and skewing objects, 463
saving work, 459
scaling objects, 462–463
scenes, 450–451
selecting and deleting objects, 457–458
setting font characteristics, 467–468
setting stroke characteristics, 456–457
sound and, 476
straightening and smoothing objects, 463

www.it-ebooks.info

computer Animation Timeline (Macromedia Flash) *(continued)*
stretching and distorting objects, 460–461
symbols, 470
synchronizing sound and action, 477
the Stage, 450
the Timeline, 451–452
Tools panel, 452–454
tweening, 472–473
computer audio, 487–494
computer authoring tools
browser capability testers, 58–59
choosing, 55–57
code validation programs, 58
demos, trying out, 56–57
Dreamweaver, 55, 61–63, 146, 174
freeware, 57
HomeSite+, 60–61
HTML editors, 51
iSpace, 55
link checkers, 58
Notepad, 60
server monitor services, 59–60
shareware, 57
text editors, 49–50
types of, 49
word processors, 54–55
WYSIWYG programs, 51–54
computer editing tools, image
Fireworks, 174, 175, 178, 179, 183
Painter, 174, 175, 178, 180, 183
Paint Shop Pro, 174, 176, 178, 180, 183–184
Photoshop, 173, 174, 178, 179, 180, 181–182
computer graphics, positioning in
absolute vs. relative, 317–319
adding layers, 315–322
background color and transparency in, 321–322
calculating absolute, 435–437
clipping layers, 322–324
CSS properties for, 315–316
dynamic page elements, 435–447
in HTML, 316–317
layer visibility, 324
nested elements, 320–321
normal flow, 317
overflow attributes, 324–327
overlapping layers, 321
stacking layers with z-order, 321
computer graphics software, VisiBone Webmaster's Color Lab, 201
computer image(s), 874
adding to tables, 232
aligning text with, 153–157
alignment settings, 155
artistic effects filters, 184–187
background, adding, 163–164
background, in tables, 233–234
borders around, 148–150
calculating sizes for tiling, 164
clip art, 176–177
copyrighted materials, using, 142–143
cropping, 177–178
defined, 874
editing tools. *See* image editing tools
embedding special fonts, 162–163
floating, 153–155
free artwork, downloading, 139
free, off the Web, 138–141
graphics files, 137–138
height and width, changing, 146–147
horizontal and vertical spacing, 147–148
importing into Macromedia Flash, 459–460
inserting into Web pages, 145–150
Irfan View program, 176
legal concerns, 141–143
to link files, 157–161
low-resolution to speed download time, 152–153
modifying, 176–177
multiple, aligning with text, 157
original art, finding artists, 139–140
placeholders, 150–153
public domain photographs, 138–139
resizing and resampling, 178–180
rotating and flipping, 180–184
sharpening and blurring, 184
stock photographs, 140–141
swapping with rollovers, 415–417, 430–432
3D graphics programs, 187–192
thumbnail, to speed up page downloads, 161–162
trademarks, 143
word wrap around, 155–157
computer software navigation bars
adding pointers, 413–415
adding to Web page, 410–412
graphics to enhance, 412–413
horizontal vs. vertical, 417–418
in JavaScript, 415–417
computer text
aligning and indenting, 119–123
aligning images with, 153–157
BASE element, 116–119
BLOCKQUOTE element, indenting with, 122–123

www.it-ebooks.info

boldfacing, 112–113
cascading style sheets (CSS), 286–287, 308–309
CENTER element, 120
character styles, 111–115
choosing character sets, 123–127
color, 119, 160, 197
default fonts, overriding, 117–119
deprecation and, 120–122
FONT element, 116–119
formatting in Blogger, 515–516
heading elements, 43
horizontal lines to divide, 44–47
italicizing, 112
line breaks, 43–44
in Macromedia Flash, 466–469
as placeholder for missing image, 150–152
preformatting, 113–114
special characters and entities, 127–135
style, maintaining consistency in, 67
subscripts and superscripts, 114–115
white space, 47–48
word wrap around images, 155–157
computer variables
arithmetic operator, 370–371
assigning different values, 368
assignment operator, 366–367
comparison operator, 369–370
declaring, 366
defined, 366
logical operator, 371
name application, 380
naming, 366–368
scope of, 380
valid and invalid, 368
value types, 368
computer video
digital sources of, 496–498
embedding, 494–497
movie clips, 496
configuration, 871
Consumer Protection Act, Anti-Cybersquatting, 598
container relationships, 38–39, 221
content
importance of, 68
pulled from other sources, 69
updating, 69
usability testing, 70
contrast and visibility of colors, 206
controlling overflow with scroll bars, 326–327
control panels, using to set site options, 98–101
cookie, 871
cool colors, 206–207
.coop extension, 80
coordinating colors, 206
copyright
audio, 490–491
images, 142–143
core attributes, 41
Corel Painter. *See* Painter
cost of banners (typical), 33
creating banners
advertising with, 32–35, 587–588, 607
defined, 870
designing, 17, 35, 588, 607, 608
exchanges, 34–35
standard size, 33
typical charges for, 33
use of, 32–35
creating navigation bars
adding pointers, 413–415
adding to Web page, 410–412
graphics to enhance, 412–413
horizontal vs. vertical, 417–418
in JavaScript, 415–417
credit cards, 541, 542, 571–573
chargebacks, 575
fraud, 575, 601–602
crippleware, 56
cropping
images, 178
in Macromedia Fireworks, 178
CSS (cascading style sheets)
absolute vs. relative positioning, 317–319
browser support for, 270
building menus with, 277–279
cascading priorities, 270–274
and character styles compared, 111
classes, 275
CSS1 properties, 280–288
CSS2 properties, 288–311
defined, 269, 870
embedded style sheet, 272
ID selectors, 275–276
linking external style sheets, 273–274
nested elements, 276–279
positioning properties, 315–316
redefining HTML elements, 274–275
scroll bars, 326–327
style attribute, 270, 271
style element, 270
style sheet-creation programs, 312

www.it-ebooks.info

CSS (cascading style sheets) *(continued)*
<STYLE> tag, 272
support information sites, 271
uses of, 269, 274
Web sites concerning, 313
XHTML and, 720–721
CSS1 properties, 280–288
backgrounds, 280–281
borders, 281–272
clear, 282–283
color, 283
display, 283
float, 283
fonts, 283–284
height, 284
letter-spacing, 284–285
line-height, 284
list-styles, 285–286
margins, 286
padding, 286
text, 286–287
vertical-align, 287
white-space, 287
width, 288
word-spacing, 288
CSS2 properties, 288–311
Azimuth, 289
backgrounds, 289–290
borders, 290–291
bottom, 292
caption-side, 293
clear, 293
clip, 293
color, 293
content, 293–294
counter-increment, 294
counter-reset, 294
cues, 294–295
cursor, 295
direction, 295
display, 295
elevation, 295–296
empty-cells, 296
float, 296
fonts, 296–297
height, 298
left, 298
letter-spacing, 298–299
line-height, 299
list-styles, 299–300
margins, 300
marker-offset, 300
marks, 300–301
max-height, 301
max-width, 301
min-height, 301
min-width, 301
orphans, 302
outlines, 302–303
overflow, 303
padding, 303
page, 303
page-breaks, 303–304
pause, 304–305
pitch, 305
play-during, 305
position, 305–306
quotes, 306
richness, 306
right, 306
size, 306
speak, 306–307
speech-rate, 307
stress, 308
table-layout, 308
text, 308–309
top, 309
unicode-bidi, 309
vertical-align, 310
visibility, 310
voice-family, 310
volume, 310
white-space, 310–311
widows, 311
width, 311
word-spacing, 311
z-index, 311
custom buttons, 353–354

D

DAZ3D, free 3D models, 192
dead ends, 68
debit cards, 541, 571–573
dedicated connection, 871
dedicated server, 84, 88, 871
default, 871
default attribute(s), 42
default fonts, overriding, 117–119
Definition, Document Type (DTD)
overview, 652–653
in XHTML 1.0, 716–717
in XML, 642–643, 652–653
demographics, 871
demos for authoring tools, 56–57
deprecation, 120–122, 123
derivative work, 142
designing banners, 17, 35, 588, 607, 608
designing tables

www.it-ebooks.info

adding, 211–212
adding images to, 232
alignment of, 220
background colors, 234–240
background images, 233–234
borderless, 218
borders, 217–218
cell padding, 218–219
cell spacing, 219–220
defined, 211
with forms, 222–224
horizontal alignment, 220–227
preventing word wrap, 228–229
rules, 218
setting size, 211–216
with sidebar images, 225–227
spanning rows and columns, 229–232
vertical alignment, 228
design, Web site
color, 66
content, 68
functionality and, 68, 69
navigation, 67
text style, 67
DHTML. *See* Dynamic HTML
dial-up connection, 871
digital audio, sources of, 490–494
digital certificates, 536, 537, 542
digital ID, 871
directories, Web, 27–29
categorization style of, 27–29
Free-For-All sites, 29
ranking within, 27–28
registering with, 28–29
and search engines compared, 27
Web site promotion via, 17, 27–29
Yahoo!, 27–29
disabled attribute, 339
dithering, 200
DNS. *See* Domain Name Service; Domain Name System
document object, 393
Document Object Model (DOM)
browser support for, 437–441
calculating absolute position under, 433–437
defined, 872
purpose of, 393–394
Document Type Definition (DTD)
overview, 652–653
in XHTML 1.0, 716–717
in XML, 642–643, 652–653
domain addresses (URLs)
absolute, 72
addressing resources with, 72–73
color information, 207
defined, 72, 88
examples of for Web site types, 6–7, 9–10
Internet research firms, 12
music sources, 494
online shopping malls, 538
partial, 72
prefixes, 73
privacy policies, samples of, 604
relative, 73
trademark research sites, 601
uses of, 72
domain name
buying a taken name, 80
celebrity names as, 597
choosing, 79–82
country-specific, 83
disputes over, 596–598
extensions, 79–80, 599
legal concerns about, 595–596
maintaining, 598–599
multiple-year discounts, 83
registering derivatives, 599
registrar, 82–83
registration delays, 84
registration pitfalls, 83–84
researching online, 598
securing, 79–84
theft of, 93–94
trademarks as, 597
transferring, 82
Uniform Domain Name Resolution Policy (UDRP), 597–598
domain name extensions
.biz, 80
.com, 80
.coop, 80
.gov, 80
.info, 80
.mil, 80
.name, 80
.net, 80
Domain Name Service (DNS), 84
Domain Name System (DNS), 872
domain server
categories of, 84
choosing, 84–87
co-located, 84, 88–89
dedicated, 84, 88
defined, 84
monitor services, 59–60
non-virtual hosting, 87–88

www.it-ebooks.info

domain server *(continued)*
and operating system, reliable combinations, 85
price-service ratios, 89–90
server extensions, 85
space needs, determining, 87
uploading pages with File Transfer Protocol (FTP), 102–106
uploading pages with Web browsers via HTTP, 107
virtual hosting, 84, 88, 89
domain, virtual, 88
DOM (Document Object Model)
browser support for, 437–441
calculating absolute position under, 433–437
defined, 872
purpose of, 393–394
dot-com, 79–80
double slash (//) comment marker, 374–375
download, 872
downsampling images, 178
Dreamweaver, 55, 61–63, 146, 184
drop caps, 163, 164
DTD (Document Type Definition)
overview, 652–653
in XHTML 1.0, 716–717
in XML, 642–643, 652–653
Dynamic HTML (DHTML), 872. *See also* cascading style sheets (CSS); JavaScript
dynamic Web page elements
animation, hazards of, 429
browser support for, 437–441
color fades, 445–447
Document Object Model (DOM) and, 435–447
positioning, 435–448
purpose of, 429
using rollovers to change, 430–435

E

eBay, 539–569
after the auction, 565–566
Buy Now option, 560
checks as payment option, 562–563
debit cards, 541, 571–573
eBay template, 566–567
enhancing success with, 566
establishing yourself as a reputable dealer, 564–565
fees for selling on, 552–554
modifying the eBay template, 567
monitoring your auction, 565
overview of, 549–550
payment and shipping options, 561–564
PayPal accounts, 561–562
setting a start and finish time, 565
setting up an account, 551–552
setting up an auction, 554–560
setting up a Dutch (multiple item) auction, 560–576
setting up an eBay store, 568
shipping merchandise, 563
template, eBay, 566–567
e-cash, 542–544, 577
ECMAScript, 386
e-commerce, 872
advertising, 579–592, 607
affiliate stores, 544, 545
B2B (business-to-business), 870
B2C (business-to-commerce), 870
chargebacks, 575
checks, 541–542, 543, 576–577
credit cards, 541, 542, 571–573, 575
credit fraud, 575–601–602
digital certificates, 536, 537, 542
e-cash, 542–544, 577
encryption, 601, 603
general stores, 530–531
Internet buying patterns, 544, 546–547
online computer stores, 532, 533
online shopping malls, 534–535, 536–538, 605
online superstores, 531–532
open source software, 538–540
PayPal accounts, 536, 561–562, 577
personal information, asking customers to provide, 603
privacy policies, 603, 604
return and refund policies, 573–575, 603, 605–606
security and, 536, 601–603
selling on eBay, 549–569
service sites, 532, 534
setup, options for, 529
shopping carts, 535–540
single product sites, 529
specialty stores, 529–530
taking payment, 540–544
Yahoo! Store, 536
editing tools, image
Fireworks, 174, 175, 178, 179, 183
Paint Shop Pro, 174, 176, 178, 180, 183–184
Painter, 174, 175, 178, 180, 183
Photoshop, 173, 174, 178, 179, 180, 181–182
.edu extension, 80
effects filters, artistic, 184–187
element, BASE, 77–78
element, BASEFONT

www.it-ebooks.info

coloring text, 119
comparing font size with heading size, 117
overriding default font faces, 117–119
specifying a size, 116–117
using relative size, 117
element, FONT
coloring text, 119
comparing font size with heading size, 117
overriding default fond faces, 117–119
specifying a size, 116–117
using relative size, 117
elements
block-level, 39–40
container relationships, 38–39
defined, 38, 872
defining with start and end tags, 38
inline, 40
names, 38
as objects, 393–394
elements, dynamic page
animation, hazards of, 429
browser support for, 437–441
color fades, 445–447
Document Object Model (DOM) and, 435–447
positioning, 435–448
purpose of, 429
using rollovers to change, 430–435
element, SELECT, 346–351
to create link menus, 425–426
saving space in forms with, 346–351
tab order in forms, 360
e-mail
advertising via, 590–591
for communicating with customers, 31
defined, 872
extra accounts, 95
formatting, plain text vs. HTML, 32
locating addresses, 31–32, 591–592
mailing lists, 591–592
sending with mailto links, 77
spam, 17, 31, 590, 629, 880
unsubscribing option, 32
username, 872
Web site promotion via, 17
embedded style sheet, 171
embedding computer audio, 488–490
encryption, 601, 603, 872
engines, search
about, 487
All the Web, 20
AltaVista, 18, 20
ALT tag, 22–23
Ask Jeeves, 20, 587
blocking, 26–27
commonalities of, 19–20
defined, 18, 880
keeping a site searchable, 21–23
logs, 20
META tag, 22
optimizing personal Web sites for, 17, 18–19
portals, 25
purpose of, 4, 18–19
ranking on, 24
Robots and Spiders, 19, 20, 21
submitting a site, 20–21
TITLE tag, 21–22
and Web directories compared, 17
enhancing navigation bars with graphics, 412–413
entities and codes, character
common HTML, 128–131
defined, 127–128, 870
HTML technical and math, 131–135
equality operator. *See* comparison operator
Ethernet, 872
European Computer Manufactuers Association (ECMA), 125, 386
event attribute(s), 41
event attributes, 41
event binding, 822–825
event handlers
alphabetical list of, 386–387
assignment of, 371–372
in JavaScript, 386–387
events, 386
defined, 821
extrinsic, 821, 822
intrinsic, 821, 822
in JavaScript, 387
Load, 387
Unload, 387
exchanging banners, 34–35
Excite (search engine), 18, 20, 25
eXtensible Hypertext Markup Language. *See* XHTML
eXtensible Markup Language. *See* XML
eXtensible Style Language. *See* XSL
extensions
.biz, 80
.com, 80
.coop, 80
.gov, 80
.info, 80
.mil, 80
.name, 80
.net, 80

www.it-ebooks.info

external style sheets, 273–274
extrinsic event, 821, 822–825
e-zines, 629–630

F

feed, 873
file transfer
  with File Transfer Protocol (FTP), 102–107, 873
  using Web browsers via HTTP, 107
File Transfer Protocol (FTP), 102–107, 873
file type, choosing (for audio), 488
filters, artistic effects, 184–187
Fireworks (Macromedia Fireworks)
  applying artistic effects filters, 186
  cropping, 178
  described, 174–175
  image enlargement in, 147
  resizing and resampling, 179
  rotating and flipping, 183
  sharpening and blurring, 184
flame, 873
Flash (Macromedia) Animation Timeline
  adding actions, 478–479
  adding layers, 470, 471
  adding MP3 sound, 477
  adding objects to layers, 470, 471
  adding text, 466
  as an advertising format, 34
  Animation Timelines, 469–477
  assigning keyframes, 470
  bitmap fills, 465–466
  choosing fonts, 466–467
  converting objects to symbols, 470, 472
  creating animation, 472
  creating layer effects, 473–475
  creating objects, 454, 456–460
  creating text effects, 468–469
  Default panel, 455
  described, 449
  Design panels, 455
  Development panels, 456
  exporting movies, 479–481
  frames, 469–470
  gradient fills, 463–464
  grouping objects, 461–462
  importing art, 459–460
  importing sound files into, 476
  launching for the first time, 449
  layout, components of, 449
  layers, 451, 470, 473–475
  Library panel, 460
  Line tools, 456, 457
  making shapes with tools, 458–459
  modifying objects, 460–466
  objects to symbols, converting, 470, 472
  optimizing movies, tips for, 483–485
  panels, 454–456
  programming with ActionScript, 477–478
  publishing movies, 481–483
  rotating and skewing objects, 463
  saving work, 459
  scaling objects, 462–463
  scenes, 450–451
  selecting and deleting objects, 457–458
  setting font characteristics, 467–468
  setting stroke characteristics, 456–457
  sound and, 476
  straightening and smoothing objects, 463
  stretching and distorting objects, 460–461
  symbols, 470
  synchronizing sound and action, 477
  the Stage, 450
  the Timeline, 451–452
  Tools panel, 452–454
  tweening, 472–473
(Flash) Macromedia Flash
  adding actions, 478–479
  adding layers, 470, 471
  adding MP3 sound, 477
  adding objects to layers, 470, 471
  adding text, 466
  as an advertising format, 34
  Animation Timelines, 469–477
  assigning keyframes, 470
  bitmap fills, 465–466
  choosing fonts, 466–467
  converting objects to symbols, 470, 472
  creating animation, 472
  creating layer effects, 473–475
  creating objects, 454, 456–460
  creating text effects, 468–469
  Default panel, 455
  described, 449
  Design panels, 455
  Development panels, 456
  exporting movies, 479–481
  frames, 469–470
  gradient fills, 463–464
  grouping objects, 461–462
  importing art, 459–460
  importing sound files into, 476
  launching for the first time, 449
  layout, components of, 449
  layers, 451, 470, 473–475
  Library panel, 460
  Line tools, 456, 457
  making shapes with tools, 458–459

www.it-ebooks.info

modifying objects, 460–466
objects to symbols, converting, 470, 472
optimizing movies, tips for, 483–485
panels, 454–456
programming with ActionScript, 477–478
publishing movies, 481–483
rotating and skewing objects, 463
saving work, 459
scaling objects, 462–463
scenes, 450–451
selecting and deleting objects, 457–458
setting font characteristics, 467–468
setting stroke characteristics, 456–457
sound and, 476
straightening and smoothing objects, 463
stretching and distorting objects, 460–461
symbols, 470
synchronizing sound and action, 477
the Stage, 450
the Timeline, 451–452
Tools panel, 452–454
tweening, 472–473
flipping images, 180–181
flipping and rotating images, 183
floating images, 153–155
FONT element
coloring text, 119
comparing font size with heading size, 117
overriding default fond faces, 117–119
specifying a size, 116–117
using relative size, 117
font face, 117–118
fonts
consistency in selection of, 67
embedding in images, 162–163
in Macromedia Flash, 466–468
monospace, 113
proportionally spaced, 13
sans serif, 117
serif, 117
font size
comparing with heading size, 117
overriding default, 117–119
relative, 117
specifying, 116–117
fonts, proportionally spaced, 113
foreground color, 197
formatting tables
adding, 211–212
adding images to, 232
alignment of, 220
background colors, 234–240
background images, 233–234
borderless, 218
borders, 217–218
cell padding, 218–219
cell spacing, 219–220
defined, 211
with forms, 222–224
horizontal alignment, 220–227
preventing word wrap, 228–229
rules, 218
setting size, 211–216
with sidebar images, 225–227
spanning rows and columns, 229–232
vertical alignment, 228
formatting text
aligning and indenting, 119–123
aligning images with, 153–157
BASE element, 116–119
BLOCKQUOTE element, indenting with, 122–123
boldfacing, 112–113
cascading style sheets (CSS), 286–287, 308–309
CENTER element, 120
character styles, 111–115
choosing character sets, 123–127
color, 119, 160, 197
default fonts, overriding, 117–119
deprecation and, 120–122
FONT element, 116–119
formatting in Blogger, 515–516
heading elements, 43
horizontal lines to divide, 44–47
italicizing, 112
line breaks, 43–44
in Macromedia Flash, 466–469
as placeholder for missing image, 150–152
preformatting, 113–114
special characters and entities, 127–135
style, maintaining consistency in, 67
subscripts and superscripts, 114–115
white space, 47–48
word wrap around images, 155–157
form processing reference model, 804–815
forms
access keys, specifying, 360–361
adding, 331–334
border colors, 161
BUTTON element, 356–357
check boxes, 343–344
controls, 331–332
custom buttons, creating, 353–354
default choice, setting, 346, 351
defined, 804
graphical buttons, 354–355
hidden fields, 357–358

www.it-ebooks.info

forms *(continued)*
INPUT buttons, 352–355
INPUT element, 332–333
INPUT types, 332–333
Java Server Pages/JavaServlets, handling of data by, 362
labels, 358–359
length of, 13
list boxes, 346–351
menu types, 349–350
multiple selections, allowing for, 350–351
naming elements, 333–334
option buttons, grouping, 344–345
option buttons, naming, 334
OPTION element, 348
password fields, 335
pop-up menus, 346–351
as pre-access device, 13
purpose of, 331
Reset button, 352–353
SELECT element, 348
Submit button, 352
submitting, 361–363
using tables with, 222–224
tab order, 359–360
text areas, 339–342
text boxes, 334–339
text input, options for, 342–346
validation of in JavaScript, 396–400
values, specifying, 349
in WML, 667
frames
adding foreign elements to, 265
alternatives to, 263–265
with Animation Timelines (Macromedia Flash), 469–470
backup navigation, 266
borders between, 260–262
contents of, 253
customizing, 260–263
custom-named frames, targeting, 258–259
defined, 873
designing frame layouts, 241–244
iframes, 263–264
keyframes, 470
legal challenges to, 267
margins, 262–263
naming conventions, 254
navigation, 241, 243–244
percentage sizing, 146, 250
pixel sizing, 148, 250
preventing resizing, 253
purpose of, 241–242
relative sizing, 250–253
reserved frame names, 259–260
screen size, 266
scroll bar options, 263–265
server side includes, 164–265
of someone else's pages on your site, 267
specifying names and contents, 253–255
too few, 265
too many, 266
framesets
alternatives to, 263–265
creating, 244-258
defined, 241, 244, 873
and HTML pages compared, 245–246
nesting, 255–257
NOFRAMES element, adding, 257–258
setting columns and rows, 246–250
frame spoofing, 59
Free-For-All (FFA) Web directories, 29
freeware, 57
Frequently Asked Questions (FAQ) section, 627
FrontPage
to create users and passwords, 101
server extensions, 85
FTP. *See* File Transfer Protocol (FTP)
FTP, anonymous, 869
functionality, 68–69
function, defined, 379

G

Gateway Interface, Common (CGI)
defined, 871
handling of form data by, 362
GIF files, 34, 137, 873
Google, 17, 18
acquisition of Blogger by, 509
advertising with, 579–585, 586
AdWords accounts, 579–585, 586
form interface, illustration of, 334
providing meaningful information to, 23–24
searching for blogs on, 513
URL, 20
Gopher, 873
.gov extension, 80
Gradient fills, 463–464
Graphical buttons, 354–355
Graphics
ALT tag, importance of, 23
copyrighted material, 142–143
to enhance navigation bars, 412–413
free artwork, downloading, 139
GIF files, 34, 137, 873
JPEG files, 138, 875

www.it-ebooks.info

original art, finding artists, 139–140
PNG files, 138
public domain photos, 138–139
stock photos, 140–141
3D programs for, 187–192
trademarks, 143

H

handlers, event
alphabetical list of, 386–387
assignment of, 371–372
in JavaScript, 386–387
header, 873
heading, 43
heading size, 117
height attribute, 146–147, 213
help balloons, 151
hexadecimal (hex) color notation, 198, 498
hidden fields, adding to forms, 357–358
hit, 873
home pages, 407–410, 873–874
barrier syndrome, 409
defined, 873–874
navigation blunders, avoiding, 407–410
HomeSite+, 60–61, 146
Hometown Web-building site (AOL), 518–519
horizontal lines, adding, 44–47
horizontal rule, 874
horizontal spacing, 147–148
horizontal vs. vertical navigation bars, 417–418
Hotmail, 32
hover advertisements, 592
HTML editors, 51, 874
HTML (Hypertext Markup Language)
adding text, 42–48
authoring tools, 48–63
block-level elements, structuring pages with, 39
BODY element, 38
browser compatibility testers, 58–59
character entities and codes, 128–131
classic structure of, 37–38
code validators, 58
container relationships, 38–39, 221
defined, 874
defining elements with start and end tags, 38
Dreamweaver, 61–63
editor, 51, 874
elements, redefining, 274–275
e-mail, formatting, 32
English color names in, 199–200
HEAD element, 38
and HomeSite+, 60–61
inline elements, fine-tuning with, 40
link checkers, 58
lists, compared to WML lists, 672–673
Microsoft Word, for writing code, 54, 53
and Notepad, 60
redefining with cascading styles sheets, 274
server monitors, 59–60
technical and math character entities and codes, 131–135
text editors, 49–50
TITLE element, 38
word processors and, 54–55
WYSIWYG programs, 51–54
XML compliance, 637–639, 654–655
HTML 4
defined, 705–706
and XHTML 1.0 compared, 712–715
HTML 4.01 specification, 681–699
attributes, index of, 685–699
elements, index of, 682–685
HTML text
adding, 42–48
adding line breaks, 43–44
dividing with horizontal lines, 44–47
heading elements, 43
white space, 47–48
HTTP (Hypertext Transfer Protocol), 72, 124, 874
hyperlink. *See* link(s)
hypertext, 874
Hypertext Markup Language (HTML)
adding text, 42–48
authoring tools, 48–63
block-level elements, structuring pages with, 39
BODY element, 38
browser compatibility testers, 58–59
character entities and codes, 128–131
classic structure of, 37–38
code validators, 58
container relationships, 38–39, 221
defined, 874
defining elements with start and end tags, 38
Dreamweaver, 61–63
editor, 51, 874
elements, redefining, 274–275
e-mail, formatting, 32
English color names in, 199–200
HEAD element, 38
and HomeSite+, 60–61
inline elements, fine-tuning with, 40

www.it-ebooks.info

Hypertext Markup Language (HTML) *(continued)*
link checkers, 58
lists, compared to WML lists, 672–673
Microsoft Word, for writing code, 54, 53
and Notepad, 60
redefining with cascading styles sheets, 274
server monitors, 59–60
technical and math character entities and codes, 131–135
text editors, 49–50
TITLE element, 38
word processors and, 54–55
WYSIWYG programs, 51–54
XML compliance, 637–639, 654–655
Hypertext Markup Language 4 (HTML 4)
defined, 705–706
and XHTML 1.0 compared, 712–715
Hypertext Markup Language 4.01 (HTML 4.01) specification, 681–699
attributes, index of, 685–699
elements, index of, 682–685
Hypertext Markup Language text (HTML text)
adding, 42–48
adding line breaks, 43–44
dividing with horizontal lines, 44–47
heading elements, 43
white space, 47–48
hypertext references, 75
Hypertext Transfer Protocol. *See* HTTP

I

iCANN (Internet Corporation for Assigned Names and Numbers), 80
identify theft, 602
ID selectors, 275–276
i18 attributes, 41
IETF. *See* Internet Engineering Task Force (IETF)
if . . . else statement, 381–382
iframes, 263–264
image editing tools
Fireworks, 174, 175, 178, 179, 183
Painter, 174, 175, 178, 180, 183
Paint Shop Pro, 174, 176, 178, 180, 183–184
Photoshop, 173, 174, 178, 179, 180, 181–182
image links, 157–161
borders, 158–160
coloring, 160
multiple, using image maps, 160–161
image maps, 23
defined, 160, 874
making multiple links with, 160–161
image object, 393
images, 874
adding to tables, 232
aligning text with, 153–157
alignment settings, 155
artistic effects filters, 184–187
background, adding, 163–164
background, in tables, 233–234
borders around, 148–150
calculating sizes for tiling, 164
clip art, 176–177
copyrighted materials, using, 142–143
cropping, 177–178
defined, 874
editing tools. *See* image editing tools
embedding special fonts, 162–163
enlargement (in Macromedia Fireworks), 147
floating, 153–155
free, off the Web, 138–141
free artwork, downloading, 139
graphics files, 137–138
height and width, changing, 146–147
horizontal and vertical spacing, 147–148
importing into Macromedia Flash, 459–460
inserting into Web pages, 145–150
Irfan View program, 176
legal concerns, 141–143
to link files, 157–161
low-resolution to speed download time, 152–153
maps, 23
modifying, 176–177
multiple, aligning with text, 157
original art, finding artists, 139–140
placeholders, 150–153
public domain photographs,138–139
resizing and resampling, 178–180
rotating and flipping, 180–184
sharpening and blurring, 184
stock photographs, 140–141
swapping with rollovers, 415–417, 430–432
3D graphics programs, 187–192
thumbnail, to speed up page downloads, 161–162
trademarks, 143
word wrap around, 155–157
images, background
adding, 163–164
sidebar, with tables, 225–227
specifying for tables, 233–234

www.it-ebooks.info

IMG element, 40, 145, 232, 332
IMG tag, 22, 23
inactivity timeout, 874
incremental uploads, 626
.info extension, 80
informational Web sites, 4–6
inline elements, 40, 872
INPUT buttons, 352–355
  custom, 353–354
  graphical, 354–355
  Reset, 352–353
  Submit, 352
INPUT element, 332–333
instance, 470
integration, 524–525
Interface, Common Gateway (CGI)
  defined, 871
  handling of form data by, 362
International Organization for Standardization (ISO)
  ISO 8859 family of character sets, 124–125
  ISO 8859 original ECMA versions of, 126
  ISO 10646 (Unicode), 123
Internet
  access to, 874
  advertising agencies, 588–589
  common fears about, 600–603
  defined, 874
  finding reliable information sources on, 10–12
  research firms, selected URLs for, 1
Internet addresses (URLs)
  absolute, 72
  addressing resources with, 72–73
  color information, 207
  defined, 72, 88
  examples of for Web site types, 6–7, 9–10
  Internet research firms, 12
  music sources, 494
  online shopping malls, 538
  partial, 72
  prefixes, 73
  privacy policies, samples of, 604
  relative, 73
  trademark research sites, 601
  uses of, 72
Internet advertising
  accepting on personal Web sites, 607
  affiliate marketing, 589–590
  with banners, 17, 32–35, 587–588, 607, 608
  via e-mail, 590–591
  with Google, 586, 579–585
  hover, 592
  Internet advertising agencies, 588–589
  mailing lists, 591–592
  multimedia ads, 34
  newsletters, 592, 593
  online, typical costs per month, 33
  with search engines other than Google, 585–587
  traditional vehicles for, 35
Internet Advertising Bureau, 33
Internet affiliate, 869
Internet animation
  hazards of, 429–430
  using with Macromedia Flash, 449–485
Internet browser(s)
  compatibility. *See* browser compatibility
  defined, 882
  handling of images in tables, 233
  help balloons, 151
  sniffing, 404
  support for CSS properties, 270
  support for Document Object Model (DOM), 437–441
  testing for use with cascading style sheets, 270
  tool tips, 151–152
  tracking use of, 59
  uploading via HTTP, 107
  using more than one to test Web projects, 54
  versions, programming for, 59
  visitor-customized preference settings, 618–621
Internet business (e-commerce), 872
  advertising, 579–592, 607
  affiliate stores, 544, 545
  B2B (business-to-business), 870
  B2C (business-to-commerce), 870
  chargebacks, 575
  checks, 541–542, 543, 576–577
  credit cards, 541, 542, 571–573, 575
  credit fraud, 575–601–602
  digital certificates, 536, 537, 542
  e-cash, 542–544, 577
  encryption, 601, 603
  general stores, 530–531
  Internet buying patterns, 544, 546–547
  online computer stores, 532, 533
  online shopping malls, 534–535, 536–538, 605
  online superstores, 531–532
  open source software, 538–540
  PayPal accounts, 536, 561–562, 577
  personal information, asking customers to provide, 603
  privacy policies, 603, 604

www.it-ebooks.info

Internet business (e-commerce) *(continued)*
return and refund policies, 573–575, 603, 605–606
security and, 536, 601–603
selling on eBay, 549–569
service sites, 532, 534
setup, options for, 529
shopping carts, 535–540
single product sites, 529
specialty stores, 529–530
taking payment, 540–544
Yahoo! Store, 536
Internet buying (eBay), 539–569
after the auction, 565–566
Buy Now option, 560
checks as payment option, 562–563
debit cards, 541, 571–573
eBay template, 566–567
enhancing success with, 566
establishing yourself as a reputable dealer, 564–565
fees for selling on, 552–554
modifying the eBay template, 567
monitoring your auction, 565
overview of, 549–550
payment and shipping options, 561–564
PayPal accounts, 561–562
setting a start and finish time, 565
setting up an account, 551–552
setting up an auction, 554–560
setting up a Dutch (multiple item) auction, 560–576
setting up an eBay store, 568
shipping merchandise, 563
template, eBay, 566–567
Internet directories, 27–29
categorization style of, 27–29
Free-For-All sites, 29
ranking within, 27–28
registering with, 28–29
and search engines compared, 27
Web site promotion via, 17, 27–29
Yahoo!, 27–29
Internet domain extensions
.biz, 80
.com, 80
.coop, 80
.gov, 80
.info, 80
.mil, 80
.name, 80
.net, 80
Internet domain name
buying a taken name, 80
celebrity names as, 597
choosing, 79–82
country-specific, 83
disputes over, 596–598
extensions, 79–80, 599
legal concerns about, 595–596
maintaining, 598–599
multiple-year discounts, 83
registering derivatives, 599
registrar, 82–83
registration delays, 84
registration pitfalls, 83–84
researching online, 598
securing, 79–84
theft of, 93–94
trademarks as, 597
transferring, 82
Uniform Domain Name Resolution Policy (UDRP), 597–598
Internet Engineering Task Force (IETF), 874
Internet mail (e-mail)
advertising via, 590–591
for communicating with customers, 31
defined, 872
extra accounts, 95
formatting, plain text vs. HTML, 32
locating addresses, 31–32, 591–592
mailing lists, 591–592
sending with mailto links, 77
spam, 17, 31, 590, 629, 880
unsubscribing option, 32
username, 872
Web site promotion via, 17
Internet page design
color, 66
content, 68
functionality and, 68–69
navigation, 67
text style, 67
Internet page downloads, speeding up
using low-resolution images, 152–153
using thumbnail images, 161–162
Internet page elements, dynamic
animation, hazards of, 429
browser support for, 437–441
color fades, 445–447
Document Object Model (DOM) and, 435–447
positioning, 435–448
purpose of, 429
using rollovers to change, 430–435
Internet page generation program, 878

www.it-ebooks.info

Internet page planning
absolute technology, reasons for using, 14
budgeting for, 15
deciding on complexity levels, 14–15
facts, reliable sources of, 10–12
leading-edge technology and, 14
setting goals, 13
surveying site visitors, 12–13
Web site types, analyzing, 3–10
Internet Presence Provider (IPP), 874
Internet Protocol (IP), 879, 881
Internet Relay Chat (IRC), 874–875
internet server
categories of, 84
choosing, 84–87
co-located, 84, 88–89
dedicated, 84, 88
defined, 84
monitor services, 59–60
non-virtual hosting, 87–88
and operating system, reliable combinations, 85
price-service ratios, 89–90
server extensions, 85
space needs, determining, 87
uploading pages with File Transfer Protocol (FTP), 102–106
uploading pages with Web browsers via HTTP, 107
virtual hosting, 84, 88, 89
Internet Service Provider (ISP), 15, 79, 660, 876
Internet shopping carts, 535–540
defined, 535–536, 880
commercial software for, 538, 539
online shopping malls, 536–538
open source software for, 538, 540
security, 536
Internet site, 876. *See also* Web site
Internet site visitors
browser preference, settings by, 618–621
comfort level, 621
getting statistics about, 97–98
Internet fears, soothing, 601–603
surveying, 12–13
in Web site planning, 13
Internet space providers
autoresponders, 97
contracts with, 91
control panels to set site options, 98–101
customer service, 92
deadlines for special deals, 94–95
domain name registration, 94
domain name theft, 93–94
extra e-mail accounts, 95
finding, 84–91
getting visitor statistics, 97–98
investigating, 90–91
redirecting messages, 95–97
reliability of, 91
reselling a Web space, 101
scams, avoiding, 92–95
technical support, 92
testing responsiveness, 91–92
unlimited traffic or space, 94
Internet standards, 876
Internet telephony, 876
Internet theft of public domain music, 493
Internet URLs
absolute, 72
addressing resources with, 72–73
color information, 207
defined, 72, 88
examples of for Web site types, 6–7, 9–10
Internet research firms, 12
music sources, 494
online shopping malls, 538
partial, 72
prefixes, 73
privacy policies, samples of, 604
relative, 73
trademark research sites, 601
uses of, 72
Internet Web page
defined, 882
loading of, 387–388
navigation bars, 410–412
unloading of, 388
Internet Web page design
color, 66
content, 68
functionality and, 68–69
navigation, 67
text style, 67
Internet Web page downloads, speeding up
using low-resolution images, 152–153
using thumbnail images, 161–162
Internet Web page elements, dynamic
animation, hazards of, 429
browser support for, 437–441
color fades, 445–447
Document Object Model (DOM) and, 435–447
positioning, 435–448
purpose of, 429
using rollovers to change, 430–435
Internet Web page generation program, 878

www.it-ebooks.info

Internet Web page planning
absolute technology, reasons for using, 14
budgeting for, 15
deciding on complexity levels, 14–15
facts, reliable sources of, 10–12
leading-edge technology and, 4
setting goals, 13
surveying site visitors, 12–13
Web site types, analyzing, 3–10
Intranet, 876
intrinsic event, 821, 822
IP. *See* Internet Protocol (IP)
IP address, 876
IPP. *See* Internet Presence Provider
IRC server. *See* Internet Relay Chat (IRC)
Irfan View, 176–177
ISO (International Organization for Standardization)
ISO 8859 family of character sets, 124–125
ISO 8859 original ECMA versions of, 126
ISO 10646 (Unicode), 123
ISP. *See* Internet Service Provider (ISP)
iSpace, 55, 191
italics, 112

J

Jasc Paint Sho Pro. *See* Paint Shop Pro
Java applet parameters, 498, 500, 506
Java applets, 498–506
adding alternative content, 499
to create link lists, 423–425
defined, 498
as objects, 502–505
parameters, 498, 500, 506
sources of, 505
using, 499–502
JavaArray object, 866
JavaClass object, 867
JavaObject, 867
JavaPackage object, 867
JavaScript
adding statements, 371–378
assignment operator, 366–367
browser compatibility, 386, 400–404
calling a statement, 379–380
clicking and double-clicking, 391–392
combining statements into functions, 379–380
comments, 375
defined, 875
document object, 393
double slash (//) comment market, 374–375
event handlers, 386–387
events, using to trigger scripts, 386–393
form validation, 396–400
if, if . . . else statements, 381–382
image object, 393
JavaArray object, 866
JavaClass object, 866
and Java compared, 365
keystokes, 392–393
literals, 368, 369–371
loops, 382–385
methods, 395–396
mouse movements, 388–391
in navigation bars, 415–417
object-oriented programming, 393–396
objects, 865–867
operators, 369–371
page loads and unloads, 388–389
practice using, 365–366
processing statements, 375–376
properties, 394–395
reserved words, 367, 865
rollovers, 388, 415–417, 430
<SCRIPT> tag, 373–374
unexpected behavior, 376–378
values, 368–371
variables, 366–368, 380
window object, 393
and WMLScript compared, 660
JavaScript, statements in
adding, 371–378
calling, 379–380
combining into functions, 379–380
defined, 371
double slash (//) comment marker, 374–375
if, if . . . else, 381–382
including comments, 375
processing, 375–376
<SCRIPT> tag, 373–374
unexpected behavior, 376–377
use of quotation marks in, 372–373
JavaScript values
affecting with operators, 369–371
event handlers, 371–372
types, 368
Java Server Pages (JSP), handling of form data by, 362
Java Servlets, handling of form data by, 362
Jeeves (Ask Jeeves search engine), 20, 587
JPEG files, 138, 875
JScript, 386
jukebox, 875

www.it-ebooks.info

K

Kat's Power Tools (KPT), 184
keyframes, 470
keys, access, 360–361
keyword, 22, 875

L

labels, adding to forms, 358–359
Language, Hypertext Markup (HTML)
  adding text, 42–48
  authoring tools, 48–63
  block-level elements, structuring pages with, 39
  BODY element, 38
  browser compatibility testers, 58–59
  character entities and codes, 128–131
  classic structure of, 37–38
  code validators, 58
  container relationships, 38–39, 221
  defined, 874
  defining elements with start and end tags, 38
  Dreamweaver, 61–63
  editor, 51, 874
  elements, redefining, 274–275
  e-mail, formatting, 32
  English color names in, 199–200
  HEAD element, 38
  and HomeSite+, 60–61
  inline elements, fine-tuning with, 40
  link checkers, 58
  lists, compared to WML lists, 672–673
  Microsoft Word, for writing code, 54, 53
  and Notepad, 60
  redefining with cascading styles sheets, 274
  server monitors, 59–60
  technical and math character entities and codes, 131–135
  text editors, 49–50
  TITLE element, 38
  word processors and, 54–55
  WYSIWYG programs, 51–54
  XML compliance, 637–639, 654–655
the law and public domain music, 493
laws, Anti-spam, 591
layers
  adding, 315–322
  in Animation Timeline, 470, 473–475
  background color, 321–322
  clipping content, 322–324
  CSS positioning properties, 315–316
  hidden overflow, 325–326
  Macromedia Flash, 451
  moving around the screen, 435–437
  nested elements, 320–321
  normal flow, 317
  overlapping, 321
  screen size and, 441–445
  stacking with z-order, 321
  transparency, 321–322
  visibility of, 324
  visible overflow, 324–325, 327
layout
  natural breakpoints, 71–72
  planning, 69–70
  topic-specific Web pages, 70–71
Library panel, 460
line breaks, adding, 43–44
link checkers, 58
link colors, changing, 197–198
link errors, 74–75
links
  adding, 73–78
  avoiding errors, 74–75
  BASE element, locking relative URLs with, 77–78
  colors of, 197–198
  creating in Blogger, 516
  defined, 876
  display of, 418–426
  to downloadable file, 77
  frames as target for, 258–259
  hypertext references, 75
  importance of, 68
  internal with local anchors, 76
  to news articles, 77
  newsgroup, 77
  ordered and unordered lists, 420–425
  plain text listings, 418–420
  SELECT element, creating menus with, 425–426
  sending e-mail with mailto links, 77
  specific newsgroup article, 77
links, image, 157–161
  borders, 158–160
  coloring, 160
  multiple, using image maps, 160–161
Linux
  HTML editor sources, 51
  text editor sources, 50
  WYSIWYG editor sources, 53
list boxes, 346–351
  allowing multiple selections, 350–351
  creating, 348
  purpose of, 346–347
  setting menu types, 349–350
  specifying values, 349

www.it-ebooks.info

list owner, 876
list server, 876
literals
  affecting with operators, 369–371
  defined, 368
  value types, 368
LiveJournal, 520–521
Load event, 387
local anchors, 76
log file, 876
logical operator, 371
logs, 20
loops, in JavaScript, 382–385
Lycos (search engine), 18, 20, 585, 586

M

Macintosh
  HTML editor sources, 52
  text editor sources, 50
  WYSISYG editor sources, 54
Macromedia Fireworks
  applying artistic effects filters, 186
  cropping, 178
  described, 174–175
  image enlargement in, 147
  resizing and resampling, 179
  rotating and flipping, 183
  sharpening and blurring, 184
Macromedia Flash
  adding actions, 478–479
  adding layers, 470, 471
  adding MP3 sound, 477
  adding objects to layers, 470, 471
  adding text, 466
  as an advertising format, 34
  Animation Timelines, 469–477
  assigning keyframes, 470
  bitmap fills, 465–466
  choosing fonts, 466–467
  converting objects to symbols, 470, 472
  creating animation, 472
  creating layer effects, 473–475
  creating objects, 454, 456–460
  creating text effects, 468–469
  Default panel, 455
  described, 449
  Design panels, 455
  Development panels, 456
  exporting movies, 479–481
  frames, 469–470
  gradient fills, 463–464
  grouping objects, 461–462
  importing art, 459–460
  importing sound files into, 476
  launching for the first time, 449
  layers, 451, 470, 473–475
  layout, components of, 449
  Library panel, 460
  Line tools, 456, 457
  making shapes with tools, 458–459
  modifying objects, 460–466
  objects to symbols, converting, 470, 472
  optimizing movies, tips for, 483–485
  panels, 454–456
  programming with ActionScript, 477–478
  publishing movies, 481–483
  rotating and skewing objects, 463
  saving work, 459
  scaling objects, 462–463
  scenes, 450–451
  selecting and deleting objects, 457–458
  setting font characteristics, 467–468
  setting stroke characteristics, 456–457
  sound and, 476
  the Stage, 450
  straightening and smoothing objects, 463
  stretching and distorting objects, 460–461
  symbols, 470
  synchronizing sound and action, 477
  the Timeline, 451–452
  Tools panel, 452–454
  tweening, 472–473
Macromedia Flash Animation Timeline
  adding actions, 478–479
  adding layers, 470, 471
  adding MP3 sound, 477
  adding objects to layers, 470, 471
  adding text, 466
  as an advertising format, 34
  Animation Timelines, 469–477
  assigning keyframes, 470
  bitmap fills, 465–466
  choosing fonts, 466–467
  converting objects to symbols, 470, 472
  creating animation, 472
  creating layer effects, 473–475
  creating objects, 454, 456–460
  creating text effects, 468–469
  Default panel, 455
  described, 449
  Design panels, 455
  Development panels, 456
  exporting movies, 479–481
  frames, 469–470
  gradient fills, 463–464
  grouping objects, 461–462
  importing art, 459–460
  importing sound files into, 476

www.it-ebooks.info

launching for the first time, 449
layers, 451, 470, 473–475
layout, components of, 449
Library panel, 460
Line tools, 456, 457
making shapes with tools, 458–459
modifying objects, 460–466
objects to symbols, converting, 470, 472
optimizing movies, tips for, 483–485
panels, 454–456
programming with ActionScript, 477–478
publishing movies, 481–483
rotating and skewing objects, 463
saving work, 459
scaling objects, 462–463
scenes, 450–451
selecting and deleting objects, 457–458
setting font characteristics, 467–468
setting stroke characteristics, 456–457
sound and, 476
the Stage, 450
straightening and smoothing objects, 463
stretching and distorting objects, 460–461
symbols, 470
synchronizing sound and action, 477
the Timeline, 451–452
Tools panel, 452–454
tweening, 472–473
Macromedia Shockwave, 34
mailbox, 876
mail filtering, 876
mailing lists, 591–592, 876
mail server, 876
mailto link, 77
maintenance
browser preference settings, 618–621
common tasks, 621–622
consistency, importance of, 621–622
evaluating input from testers and visitors, 617–618
hiring outside testers, 615–617
server logs, reviewing, 618
server monitors, 622
site improvement tools, 622
testing spreadsheet, 614
testing your site, 613–614
user scenario testing, 614–615
visitors' changes, coping with, 618–621
visitor comfort, 621
making your own music, 491–492
managing text
aligning and indenting, 119–123
aligning images with, 153–157
BASE element, 116–119
BLOCKQUOTE element, indenting with, 122–123
boldfacing, 112–113
cascading style sheets (CSS), 286–287, 308–309
CENTER element, 120
character styles, 111–115
choosing character sets, 123–127
color, 119, 160, 197
default fonts, overriding, 117–119
deprecation and, 120–122
FONT element, 116–119
formatting in Blogger, 515–516
heading elements, 43
horizontal lines to divide, 44–47
italicizing, 112
line breaks, 43–44
in Macromedia Flash, 466–469
as placeholder for missing image, 150–152
preformatting, 113–114
special characters and entities, 127–135
style, maintaining consistency in, 67
subscripts and superscripts, 114–115
white space, 47–48
word wrap around images, 155–157
manipulating tables
adding, 211–212
adding images to, 232
alignment of, 220
background colors, 234–240
background images, 233–234
borderless, 218
borders, 217–218
cell padding, 218–219
cell spacing, 219–220
defined, 211
with forms, 222–224
horizontal alignment, 220–227
preventing word wrap, 228–229
rules, 218
setting size, 211–216
with sidebar images, 225–227
spanning rows and columns, 229–232
vertical alignment, 228
mapping, 123
maps, image, 23
defined, 160, 874
making multiple links with, 160–161
margins, frames, 262–263
Markup Language, Wireless (WML)
cards, 664–666
defined, 659–660
format, specifying, 667–669
forms in, 667

www.it-ebooks.info

Markup Language, Wireless (WML) *(continued)*
lists, compared to HTML, 672–673
menus, 669–671
multiple-choice menus, 671–672
required values, specifying, 669
tags, 662–663
transforming XHTML into, 673–676
user interaction, 666–673
using, 660–662
and XHTML compared, 659
Markup Language, Wireless script (WMLscript), 660
Markup Language, Wireless 2.0 (WML2.0)
adding style to, 849–850
attribute expression syntax, 815–821
BACK key, 827
Base Module, 840
change history, 864
Context and Navigation Module, 841–848
default style sheet, 850, 851
document conformance, 848
documents, identification of, 826
Document Type Definition (DTD) for, 797–798, 850–851
elements, 851–859
event model, 821–825
Events Module, 841
form processing reference model, 804–815
Forms Module, 834–837
Hypertext Module, 834
Image Module, 838
IMG element, 839
informative references, 793–794
Link Module, 840
Lists Module, 838
markup elements and attributes, 832–848
Metainformation Module, 839–840
navigation reference processing model, 800–804
normative references, 792–793
Object Module, 840
Presentation Module, 841
specification, 787–864
static conformance requirements, 859–864
Structure Module, 832–833
Style Sheet Module, 840
Tables Module, 837–838
terminology and conventions, 794–796
Text Module, 833
timer processing, 829–830
user agent behavior, 799–832
user agent conformance rules, 830–832
wm1:do element, 827–829
XHTML, acceptance of, 830
mapping, 123
marketing, affiliate, 589–590
marketing and production
accepting on personal Web sites, 607
affiliate marketing, 589–590
banner exchanges, 17, 35
via banners, 17, 32–35, 587–588, 607, 608
via e-mail, 17, 590–591
with Google, 586, 579–585
hover, 592
Internet advertising agencies, 588–589
Internet buying patterns, capitalizing on, 544, 546–547
issuing press releases, 29–32
mailing lists, 591–592
media, types of, 34–35
multimedia ads, 34
newsletters, 592, 593
online, typical costs per month, 33
via search engines, 17, 18–27
with search engines other than Google, 585–587
setting up reciprocal links, 32
traditional advertising vehicles for, 35
Web directories, 17, 27–29
markup language, 876
Markup Language, Hypertext (HTML)
adding text, 42–48
authoring tools, 48–63
block-level elements, structuring pages with, 39
BODY element, 38
browser compatibility testers, 58–59
character entities and codes, 128–131
classic structure of, 37–38
code validators, 58
container relationships, 38–39, 221
defined, 874
defining elements with start and end tags, 38
Dreamweaver, 61–63
editor, 51, 874
elements, redefining, 274–275
e-mail, formatting, 32
English color names in, 199–200
HEAD element, 38
and HomeSite+, 60–61
inline elements, fine-tuning with, 40
link checkers, 58
lists, compared to WML lists, 672–673
Microsoft Word, for writing code, 54, 53
and Notepad, 60

www.it-ebooks.info

redefining with cascading styles sheets, 274
server monitors, 59–60
technical and math character entities and codes, 131–135
text editors, 49–50
TITLE element, 38
word processors and, 54–55
WYSIWYG programs, 51–54
XML compliance, 637–639, 654–655
Markup Language, Vector (VML), 652
marquee, 876
mathematical variables
arithmetic operator, 370–371
assigning different values, 368
assignment operator, 366–367
comparison operator, 369–370
declaring, 366
defined, 366
logical operator, 371
name application, 380
naming, 366–368
scope of, 380
valid and invalid, 368
value types, 368
menus, popup, 346–351
creating, 348
and list menus compared, 350
purpose of, 346–347
setting menu types, 349
specifying values, 349
message board, 877
metadata, 877
META element, 124
<META> tag, 22
methods
defined, 395
JavaScript, 395–396
Microsoft Developer Network Web site, 200–201
Microsoft Windows
character sets for, 127
HTML editor sources, 51
text editor sources, 50
WYSIWYG editor sources, 53
Microsoft Word, for writing HTML code, 54, 53
MIDI files, 488
.mil extension, 80
MNG (Multiple Network Graphics), 138
mobile computing devices, 659. *See also* Wireless Markup Language (WML)
modem, 877
moderator, 877
Model, Document Object (DOM)
browser support for, 437–441
calculating absolute position under, 433–437
defined, 872
purpose of, 393–394
monospace font, 113
mouse movements, JavaScript, 388–391
movies. *See also* Macromedia Flash
adding sound to, 476–477
exporting, 479–481
optimizing, tips for, 483–485
publishing, 481–483
synchronizing sound and action, 477
MP3 sound
adding to Micromedia Flash movies, 477
compared to other audio files, 488
and custom music, 493
multimedia
adding audio, 487–494
adding Java applets, 498–506
defined, 877
embedding video, 494–497
Multiple Network Graphics (MNG), 138
music
buying, 493–494
making your own, 491–492
online sources of, 494
public domain, 493

N

nagware, 57
named colors, limitations of, 199–200
name, domain
buying a taken name, 80
celebrity names as, 597
choosing, 79–82
country-specific, 83
disputes over, 596–598
extensions, 79–80, 599
legal concerns about, 595–596
maintaining, 598–599
multiple-year discounts, 83
registrar, 82–83
registering derivatives, 599
registration delays, 84
registration pitfalls, 83–84
researching online, 598
securing, 79–84
theft of, 93–94
trademarks as, 597
transferring, 82
Uniform Domain Name Resolution Policy (UDRP), 597–598

www.it-ebooks.info

.name extension, 80
name server, 877
namespaces, 642–643. *See also* Document Type Definition (DTD)
navigation, 67
  adding navigation bars, 410–418
  backup, 266
  displaying links with lists and menus, 418–426
  home page blunders, avoiding, 407–410
  importance of, 407
  orphan pages, avoiding, 410
  providing multiple options for, 68
  tiered, 67
navigation bars
  adding pointers, 413–415
  adding to Web page, 410–412
  graphics to enhance, 412–413
  horizontal vs. vertical, 417–418
  in JavaScript, 415–417
  pointers, 413–415
navigation frame, 241, 243–244
navigation reference processing tool, 800–804
nested elements, 276–279, 320–321
.net extension, 80
netiquette, 877
NetObjects Fusion, 53
network, 877
network adapter, 877
network hub, 877
Network News Transfer Protocol (NNTP), 877
network node, 877
network repeater, 877
network server
  categories of, 84
  choosing, 84–87
  co-located, 84, 88–89
  dedicated, 84, 88
  defined, 84
  monitor services, 59–60
  non-virtual hosting, 87–88
  and operating system, reliable combinations, 85
  price-service ratios, 89–90
  server extensions, 85
  space needs, determining, 87
  uploading pages with File Transfer Protocol (FTP), 102–106
  uploading pages with Web browsers via HTTP, 107
  virtual hosting, 84, 88, 89
newsgroup, 77, 877
newsletters, 592, 593, 629–630
news reader, 878
news server, 878
NNTP. *See* Network News Transfer Protocol (NNTP)
NOEMBED element, 490
NOFRAMES element, 257–258
non-virtual web hosting, 87–88
Notepad, 60
numbered lists, 420, 422
numeric values, 368

O

Object Model, Document (DOM)
  browser support for, 437–441
  calculating absolute position under, 433–437
  defined, 872
  purpose of, 393–394
object-oriented programming, 393–394
objects
  adding to layers in Animation Timeline, 470
  converting to symbols with Animation Timeline, 470
  creating with Macromedia Flash, 454, 456–460
  Java applets as, 502–505
  modifying with Macromedia Flash, 456–466
official standard, 880
omnibus study, 12
online addresses (URLs)
  absolute, 72
  addressing resources with, 72–73
  color information, 207
  defined, 72, 88
  examples of for Web site types, 6–7, 9–10
  Internet research firms, 12
  music sources, 494
  online shopping malls, 538
  partial, 72
  prefixes, 73
  privacy policies, samples of, 604
  relative, 73
  trademark research sites, 601
  uses of, 72
online advertising
  accepting on personal Web sites, 607
  affiliate marketing, 589–590
  with banners, 17, 32–35, 587–588, 607, 608
  via e-mail, 590–591
  with Google, 586, 579–585
  hover, 592

www.it-ebooks.info

Internet advertising agencies, 588–589
mailing lists, 591–592
multimedia ads, 34
newsletters, 592, 593
online, typical costs per month, 33
with search engines other than Google, 585–587
traditional vehicles for, 35
online affiliate, 869
online animation
hazards of, 429–430
using with Macromedia Flash, 449–485
online browser(s)
compatibility. *See* browser compatibility
defined, 882
handling of images in tables, 233
help balloons, 151
sniffing, 404
support for CSS properties, 270
support for Document Object Model (DOM), 437–441
testing for use with cascading style sheets, 270
tool tips, 151–152
tracking use of, 59
uploading via HTTP, 107
using more than one to test Web projects, 54
versions, programming for, 59
visitor-customized preference settings, 618–621
online business (e-commerce), 872
advertising, 579–592, 607
affiliate stores, 544, 545
B2B (business-to-business), 870
B2C (business-to-commerce), 870
chargebacks, 575
checks, 541–542, 543, 576–577
credit cards, 541, 542, 571–573, 575
credit fraud, 575–601–602
digital certificates, 536, 537, 542
e-cash, 542–544, 577
encryption, 601, 603
general stores, 530–531
Internet buying patterns, 544, 546–547
online computer stores, 532, 533
online shopping malls, 534–535, 536–538, 605
online superstores, 531–532
open source software, 538–540
PayPal accounts, 536, 561–562, 577
personal information, asking customers to provide, 603
privacy policies, 603, 604
return and refund policies, 573–575, 603, 605–606
security and, 536, 601–603
selling on eBay, 549–569
service sites, 532, 534
setup, options for, 529
shopping carts, 535–540
single product sites, 529
specialty stores, 529–530
taking payment, 540–544
Yahoo! Store, 536
online buying (eBay), 539–569
after the auction, 565–566
Buy Now option, 560
checks as payment option, 562–563
debit cards, 541, 571–573
eBay template, 566–567
enhancing success with, 566
establishing yourself as a reputable dealer, 564–565
fees for selling on, 552–554
modifying the eBay template, 567
monitoring your auction, 565
overview of, 549–550
payment and shipping options, 561–564
PayPal accounts, 561–562
setting a start and finish time, 565
setting up a Dutch (multiple item) auction, 560–576
setting up an account, 551–552
setting up an auction, 554–560
setting up an eBay store, 568
shipping merchandise, 563
online directories, 27–29
categorization style of, 27–29
Free-For-All sites, 29
ranking within, 27–28
registering with, 28–29
and search engines compared, 27
Web site promotion via, 17, 27–29
Yahoo!, 27–29
online domain name
buying a taken name, 80
celebrity names as, 597
choosing, 79–82
country-specific, 83
disputes over, 596–598
extensions, 79–80, 599
legal concerns about, 595–596
maintaining, 598–599
multiple-year discounts, 83
registering derivatives, 599
registrar, 82–83
registration delays, 84

www.it-ebooks.info

online domain name *(continued)*
registration pitfalls, 83–84
researching online, 598
securing, 79–84
theft of, 93–94
trademarks as, 597
transferring, 82
Uniform Domain Name Resolution Policy (UDRP), 597–598
online mail (e-mail)
advertising via, 590–591
for communicating with customers, 31
defined, 872
extra accounts, 95
formatting, plain text vs. HTML, 32
locating addresses, 31–32, 591–592
mailing lists, 591–592
sending with mailto links, 77
spam, 17, 31, 590, 629, 880
unsubscribing option, 32
username, 872
Web site promotion via, 17
online page planning
absolute technology, reasons for using, 14
budgeting for, 15
deciding on complexity levels, 14–15
facts, reliable sources of, 10–12
leading-edge technology and, 14
setting goals, 13
surveying site visitors, 12–13
Web site types, analyzing, 3–10
online server
categories of, 84
choosing, 84–87
co-located, 84, 88–89
dedicated, 84, 88
defined, 84
monitor services, 59–60
non-virtual hosting, 87–88
and operating system, reliable combinations, 85
price-service ratios, 89–90
server extensions, 85
space needs, determining, 87
uploading pages with File Transfer Protocol (FTP), 102–106
uploading pages with Web browsers via HTTP, 107
virtual hosting, 84, 88, 89
online service, 878
online shopping carts, 535–540
commercial software for, 538, 539
defined, 535–536, 880
online shopping malls, 536–538
open source software for, 3=538, 540
security, 536
online shopping malls, 534–535, 536–538
online space providers
autoresponders, 97
contracts with, 91
control panels to set site options, 98–101
customer service, 92
deadlines for special deals, 94–95
domain name registration, 94
domain name theft, 93–94
extra e-mail accounts, 95
finding, 84–91
getting visitor statistics, 97–98
investigating, 90–91
redirecting messages, 95–97
reliability of, 91
reselling a Web space, 101
scams, avoiding, 92–95
technical support, 92
testing responsiveness, 91–92
unlimited traffic or space, 94
online visitors
browser preference, settings by, 618–621
comfort level, 621
getting statistics about, 97–98
Internet fears, soothing, 601–603
surveying, 12–13
in Web site planning, 13
online tables
adding, 211–212
adding images to, 232
alignment of, 220
background colors, 234–240
background images, 233–234
borderless, 218
borders, 217–218
cell padding, 218–219
cell spacing, 219–220
defined, 211
with forms, 222–224
horizontal alignment, 220–227
preventing word wrap, 228–229
rules, 218
setting size, 211–216
with sidebar images, 225–227
spanning rows and columns, 229–232
vertical alignment, 228
online text
aligning and indenting, 119–123
aligning images with, 153–157
BASE element, 116–119
BLOCKQUOTE element, indenting with, 122–123

www.it-ebooks.info

boldfacing, 112–113
cascading style sheets (CSS), 286–287, 308–309
CENTER element, 120
character styles, 111–115
choosing character sets, 123–127
color, 119, 160, 197
default fonts, overriding, 117–119
deprecation and, 120–122
FONT element, 116–119
formatting in Blogger, 515–516
heading elements, 43
horizontal lines to divide, 44–47
italicizing, 112
line breaks, 43–44
in Macromedia Flash, 466–469
as placeholder for missing image, 150–152
preformatting, 113–114
special characters and entities, 127–135
style, maintaining consistency in, 67
subscripts and superscripts, 114–115
white space, 47–48
word wrap around images, 155–157
online theft of public domain music, 493
online URLs
absolute, 72
addressing resources with, 72–73
color information, 207
defined, 72, 88
examples of, for Web site types, 6–7, 9–10
Internet research firms, 12
music sources, 494
online shopping malls, 538
partial, 72
prefixes, 73
privacy policies, samples of, 604
relative, 73
trademark research sites, 601
uses of, 72
online video
digital sources of, 496–498
embedding, 494–497
movie clips, 496
online Web page
defined, 882
loading of, 387–388
navigation bars, 410–412
unloading of, 388
online Web page design
color, 66
content, 68
functionality and, 68–69
navigation, 67
text style, 67
online Web page downloads, speeding up
using low-resolution images, 152–153
using thumbnail images, 161–162
online Web page elements, dynamic
animation, hazards of, 429
browser support for, 437–441
color fades, 445–447
Document Object Model (DOM) and, 435–447
positioning, 435–448
purpose of, 429
using rollovers to change, 430–435
online Web page generation program, 878
online Web page planning
absolute technology, reasons for using, 14
budgeting for, 15
deciding on complexity levels, 14–15
facts, reliable sources of, 10–12
leading-edge technology and, `4
setting goals, 13
surveying site visitors, 12–13
Web site types, analyzing, 3–10
open source software, 538–540
operator, assignment, 366–367
operators
arithmetic, 370–371
assignment, 366–367
comparison, 367–370
logical, 371
option buttons, 344–345
OPTION element, 346–351, 360
ordered lists, defined, 420
.org, 80
organization
adding links, 73–78
addressing resources with URLs, 72–73
color, 66
content, 68, 69
hierarchical vs. linear, 71
functionality, 68–69
layout, 69–72
navigation, 67
page design vs. site design, 65–69
scanability, 68–69
text style, 67
visitor input concerning, 70
organizational web sites, 6–7
orphan pages, avoiding, 410
overflow
hidden, 325–326
scroll bars as compromise for, 326–327
visible, 324–325

www.it-ebooks.info

P

packet, 878
padding, tables, 218, 219
page design
  color, 66
  content, 68
  functionality and, 68–69
  navigation, 67
  text style, 67
page directories (Web), 27–29
  categorization style of, 27–29
  Free-For-All sites, 29
  ranking within, 27–28
  registering with, 28–29
  and search engines compared, 27
  Web site promotion via, 17, 27–29
  Yahoo!, 27–29
page downloads, speeding up
  using low-resolution images, 152–153
  using thumbnail images, 161–162
page elements, dynamic
  animation, hazards of, 429
  browser support for, 437–441
  color fades, 445–447
  Document Object Model (DOM) and, 435–447
  positioning, 435–448
  purpose of, 429
  using rollovers to change, 430–435
page generation program, 878
page planning (Web)
  absolute technology, reasons for using, 14
  budgeting for, 15
  deciding on complexity levels, 14–15
  facts, reliable sources of, 10–12
  leading-edge technology and, 14
  setting goals, 13
  surveying site visitors, 12–13
  Web site types, analyzing, 3–10
Paint Shop Pro
  applying artistic effects filters, 187
  cropping, 178
  described, 174, 176
  resizing and resampling, 180
  rotating and flipping, 183–184
  sharpening and blurring, 184
Painter
  applying artistic effects filters, 186
  cropping, 178
  described, 174, 175
  resizing and resampling, 180
  rotating and flipping, 183
  sharpening and blurring, 184
parameters for Java applets, 498, 500, 506
partial URL, 73
password
  Blogger, 511
  defined, 878
  forms and, 335
password fields, 335
PayPal accounts, 536, 561–562, 577
PDF (Portable Document Format) files, 879
personal home pages, 4
PHP programs, handling of form data by, 362
photographs
  public domain, 138–139
  stock, 140–141
Photoshop
  applying artistic filters, 185
  cropping, 178
  described, 173
  resizing and resampling, 179
  rotating and flipping, 181–182
  sharpening and blurring, 147, 184
pixel, 198
placeholders
  alternative text, 150–152
  low-resolution images, 152–153
planning Web pages
  absolute technology, reasons for using, 14
  budgeting for, 15
  deciding on complexity levels, 14–15
  facts, reliable sources of, 10–12
  leading-edge technology and, 14
  setting goals, 13
  surveying site visitors, 12–13
  Web site types, analyzing, 3–10
plug-in, defined, 487
PNG files, 138
point, defined, 116
pointers, adding to navigation bars, 413–415
policy, acceptable use, 869
political web sites, 4, 8, 9
POP. *See* Post Office Protocol (POP)
Pop Menu applet, 499–502
popup menus, 346–351
  creating, 348
  and list menus compared, 350
  purpose of, 346–347
  setting menu types, 349
  specifying values, 349
popup windows, 879
Portable Document Format. *See* PDF (Portable Document Format) files
portals, 25
port numbers, 878

www.it-ebooks.info

Poser 3D imaging program, 188–189
positioning
  absolute vs. relative, 317–319
  adding layers, 315–322
  background color and transparency in, 321–322
  calculating absolute, 435–437
  clipping layers, 322–324
  CSS properties for, 315–316
  dynamic page elements, 435–447
  in HTML, 316–317
  layer visibility, 324
  nested elements, 320–321
  normal flow, 317
  overflow attributes, 324–327
  overlapping layers, 321
  stacking layers with z-order, 321
Post Office Protocol (POP), 879
preformatting text, 113–114
press releases, 29–32
  content of, 31
  plain text vs. HTML, 32
  reasons to use, 30
  using e-mail to distribute, 31–32
previewing authoring tools, 56–57
price-service ratios, 89–90
privacy policy, 603, 604, 879
privileges, 879
product reviews, 630
programming with ActionScript, 477–478
programs, code validation, 58
promotion and marketing
  accepting on personal Web sites, 607
  affiliate marketing, 589–590
  banner exchanges, 17, 35
  via banners, 17, 32–35, 587–588, 607, 608
  via e-mail, 17, 590–591
  with Google, 586, 579–585
  hover, 592
  Internet advertising agencies, 588–589
  Internet buying patterns, capitalizing on, 544, 546–547
  issuing press releases, 29–32
  mailing lists, 591–592
  media, types of, 34–35
  multimedia ads, 34
  newsletters, 592, 593
  online, typical costs per month, 33
  via search engines, 17, 18–27
  with search engines other than Google, 585–587
  setting up reciprocal links, 32
  traditional advertising vehicles for, 35
  Web directories, 17, 27–29
properties, CSS1, 280–288
  backgrounds, 280–281
  borders, 281–272
  clear, 282–283
  color, 283
  display, 283
  float, 283
  fonts, 283–284
  height, 284
  letter-spacing, 284–285
  line-height, 284
  list-styles, 285–286
  margins, 286
  padding, 286
  text, 286–287
  vertical-align, 287
  white-space, 287
  width, 288
  word-spacing, 288
properties, CSS2, 288–311
  Azimuth, 289
  backgrounds, 289–290
  borders, 290–291
  bottom, 292
  caption-side, 293
  clear, 293
  clip, 293
  color, 293
  content, 293–294
  counter-increment, 294
  counter-reset, 294
  cues, 294–295
  cursor, 295
  direction, 295
  display, 295
  elevation, 295–296
  empty-cells, 296
  float, 296
  fonts, 296–297
  height, 298
  left, 298
  letter-spacing, 298–299
  line-height, 299
  list-styles, 299–300
  margins, 300
  marker-offset, 300
  marks, 300–301
  max-height, 301
  max-width, 301
  min-height, 301
  min-width, 301
  orphans, 302
  outlines, 302–303
  overflow, 303

www.it-ebooks.info

properties, CSS2 *(continued)*
padding, 303
page, 303
page-breaks, 303–304
pause, 304–305
pitch, 305
play-during, 305
position, 305–306
quotes, 306
richness, 306
right, 306
size, 306
speak, 306–307
speech-rate, 307
stress, 308
table-layout, 308
text, 308–309
top, 309
unicode-bidi, 309
vertical-align, 310
visibility, 310
voice-family, 310
volume, 310
white-space, 310–311
widows, 311
width, 311
word-spacing, 311
z-index, 311
properties, JavaScript, 394–395
proportionally spaced fonts, 113
protocol, 879
providers, Web space
autoresponders, 97
contracts with, 91
control panels to set site options, 98–101
customer service, 92
deadlines for special deals, 94–95
domain name registration, 94
domain name theft, 93–94
extra e-mail accounts, 95
finding, 84–91
getting visitor statistics, 97–98
investigating, 90–91
redirecting messages, 95–97
reliability of, 91
reselling a Web space, 101
scams, avoiding, 92–95
technical support, 92
testing responsiveness, 91–92
unlimited traffic or space, 94
public domain
music, 493
photographs, 138–139
push buttons, 353–354

R

radio buttons. *See* option buttons
Radio UserLand, 521–523
ray tracing, 879
reciprocal links, 32, 880
redirect address, 95
refund and return policies, 573–575, 603, 605–606
registrar, 880
regular expressions, 398–399
relative positioning, 317–319
relative sizing, 250
relative URL, 73
resampling images, 178–180
reselling Web space, 101
Reset button, adding to forms, 352–353
resizing, 178–180
resizing in Macromedia Fireworks, 179
resource, 881
RGB (red, green, blue) color combinations, 198
decoding triplets, 198–199
ring, 880
Robots
blocking, 26
purpose of, 19, 21
robots.txt file, 26–27
rollover
changing colors with, 432–433
defined, 388, 430, 880
setting status bar text, 433–435
swapping images, 430–432
root privileges, 879
rotating and flipping in Macromedia Fireworks, 183
rotating images, 181–184

S

sans serif fonts, 117
scanability, 69
scheme, 72
screen size
frames, 266
layers, 441–445
Scripts, JavaScript
adding statements, 371–378
assignment operator, 366–367
browser compatibility, 386, 400–404
calling a statement, 379–380
clicking and double-clicking, 391–392
combining statements into functions, 379–380
comments, 375
defined, 875

www.it-ebooks.info

document object, 393
double slash (//) comment market, 374–375
event handlers, 386–387
events, using to trigger scripts, 386–393
form validation, 396–400
if, if . . . else statements, 381–382
image object, 393
JavaArray object, 866
JavaClass object, 866
and Java compared, 365
keystokes, 392–393
literals, 368, 369–371
loops, 382–385
methods, 395–396
mouse movements, 388–391
in navigation bars, 415–417
object-oriented programming, 393–396
objects, 865–867
operators, 369–371
page loads and unloads, 388–389
practice using, 365–366
processing statements, 375–376
properties, 394–395
reserved words, 367, 865
rollovers, 388, 415–417, 430
<SCRIPT> tag, 373–374
unexpected behavior, 376–378
values, 368–371
variables, 366–368, 380
window object, 393
and WMLScript compared, 660
script statements (in JavaScript)
adding, 371–378
calling, 379–380
combining into functions, 379–380
defined, 371
double slash (//) comment marker, 374–375
if, if . . . else, 381–382
including comments, 375
processing, 375–376
<SCRIPT> tag, 373–374
unexpected behavior, 376–377
use of quotation marks in, 372–373
<SCRIPT> tag, 373–374
script values (in JavaScript)
affecting with operators, 369–371
event handlers, 371–372
types, 368
scroll bars
in frames, 263–265
as a way to control overflow, 326–327
search engines
about, 487
All the Web, 20
AltaVista, 18, 20
ALT tag, 22–23
Ask Jeeves, 20, 587
blocking, 26–27
commonalities of, 19–20
defined, 18, 880
keeping a site searchable, 21–23
logs, 20
META tag, 22
optimizing personal Web sites for, 17, 18–19
portals, 25
purpose of, 4, 18–19
ranking on, 24
Robots and Spiders, 19, 20, 21
submitting a site, 20–21
TITLE tag, 21–22
and Web directories compared, 17
search optimization tools, 607–610
secure server, 536
Secure Sockets Layer (SSL), 880
security, e-commerce, 536, 601–603
serif fonts, 117
SELECT element, 346–351
to create link menus, 425–426
saving space in forms with, 346–351
tab order in forms, 360
selecting authoring tools, 55–57
selecting freeware, 57
selecting HTML editors, 51
server, 871
server monitor services, 59–60, 622
server side includes, 264–265
server, virtual, 84, 882
server, Web
categories of, 84
choosing, 84–87
co-located, 84, 88–89
dedicated, 84, 88
defined, 84
monitor services, 59–60
non-virtual hosting, 87–88
and operating system, reliable combinations, 85
price-service ratios, 89–90
server extensions, 85
space needs, determining, 87
uploading pages with File Transfer Protocol (FTP), 102–106
uploading pages with Web browsers via HTTP, 107
virtual hosting, 84, 88, 89

www.it-ebooks.info

services, value-added
  newsletters and e-zines, 592, 593, 629–630
  online chats, 628
  online training, 631
  product reviews, 633
  tutorials, 630–631
setting audio volume, 490
sets, character
  choosing, 123–127
  defined, 123
  Eastern hemisphere, 127
  ECMA versions, ISO 8859 standards, 126
  ISO 8859 family of, 124–125
  ISO 10646 (Unicode), 123
  Windows-specific, 127
SGML. *See* Standard Generalized Markup Language (SGML)
shareware, authoring tools, 57
sharpening images, 184
sheets, cascading style (CSS)
  absolute vs. relative positioning, 317–319
  browser support for, 270
  building menus with, 277–279
  cascading priorities, 270–274
  and character styles compared, 111
  classes, 275
  CSS1 properties, 280–288
  CSS2 properties, 288–311
  defined, 269, 870
  embedded style sheet, 272
  ID selectors, 275–276
  linking external style sheets, 273–274
  nested elements, 276–279
  positioning properties, 315–316
  redefining HTML elements, 274–275
  scroll bars, 326–327
  style attribute, 270, 271
  style element, 270
  style sheet-creation programs, 312
  <STYLE> tag, 272
  support information sites, 271
  uses of, 269, 274
  Web sites concerning, 313
  XHTML and, 720–721
shopping carts, 535–540
  commercial software for, 538, 539
  defined, 535–536, 880
  online shopping malls, 536–538
  open source software for, 538, 540
  security, 536
sidebar images, using tables with, 115–227
Simple Mail Transfer Protocol (SMTP), 880
Site, AOL Hometown Web-building, 518–519
site design
  color, 66
  content, 68
  functionality and, 68, 69
  navigation, 67
  text style, 67
site improvement tools, 622
site visitors, Internet
  browser preference, settings by, 618–621
  comfort level, 621
  getting statistics about, 97–98
  Internet fears, soothing, 601–603
  surveying, 12–13
  in Web site planning, 13
site (Web) directories, 27–29
  categorization style of, 27–29
  Free-For-All sites, 29
  ranking within, 27–28
  registering with, 28–29
  and search engines compared, 27
  Web site promotion via, 17, 27–29
  Yahoo!, 27–29
size, absolute, 250
size, screen
  frames, 266
  layers, 441–445
SMTP. *See* Simple Mail Transfer Protocol Spam (SMTP)
software navigation bars
  adding pointers, 413–415
  adding to Web page, 410–412
  graphics to enhance, 412–413
  horizontal vs. vertical, 417–418
  in JavaScript, 415–417
sound
  importing sound files into Macromedia Flash, 476–477
  MP3, adding to Flash movies, 477
  synchronizing with action, 477
sound (MP3 sound)
  adding to Micromedia Flash movies, 477
  compared to other audio files, 488
  and custom music, 493
sources for digital audio, 490–494
space needs, 87
space providers, Web
  autoresponders, 97
  contracts with, 91
  control panels to set site options, 98–101
  customer service, 92
  deadlines for special deals, 94–95
  domain name registration, 94
  domain name theft, 93–94
  extra e-mail accounts, 95
  finding, 84–91

www.it-ebooks.info

getting visitor statistics, 97–98
investigating, 90–91
redirecting messages, 95–97
reliability of, 91
reselling a Web space, 101
scams, avoiding, 92–95
technical support, 92
testing responsiveness, 91–92
unlimited traffic or space, 94
spacing, proportional for fonts, 113
spacing, tables, 218–220
spam, 17, 31, 590, 629, 880
spam, laws against, 591
special characters
common HTML character entities and codes, 128–131
HTML technical and math character entities and codes, 131–135
numbering of, 127–128
specialized attribute(s), 42
Spiders
blocking, 26
purpose of, 19, 20
SSL. *See* Secure Socket Layers (SSL)
standard, 880
Standard Generalized Markup Language (SGML), 880
Standardization, International Organization for (ISO)
ISO 8859 family of character sets, 124–125
ISO 8859 original ECMA versions of, 126
ISO 10646 (Unicode), 123
standard-size banners, 33
statements, in JavaScript
adding, 371–378
calling, 379–380
combining into functions, 379–380
defined, 371
double slash (//) comment marker, 374–375
if, if . . . else, 381–382
including comments, 375
processing, 375–376
<SCRIPT> tag, 373–374
unexpected behavior, 376–377
use of quotation marks in, 372–373
status bar text, setting with rollovers, 433–435
stock photographs, 140–141
storage space providers (Web)
autoresponders, 97
contracts with, 91
control panels to set site options, 98–101
customer service, 92
deadlines for special deals, 94–95
domain name registration, 94
domain name theft, 93–94
extra e-mail accounts, 95
finding, 84–91
getting visitor statistics, 97–98
investigating, 90–91
redirecting messages, 95–97
reliability of, 91
reselling a Web space, 101
scams, avoiding, 92–95
technical support, 92
testing responsiveness, 91–92
unlimited traffic or space, 94
storage space, unlimited, 94
stores, affiliate, 544, 545
string values, 368
style attribute, 270–271
going inline with, 271
style element, 270
styles, character, 111–115
boldface, 112–113
defined, 111
italics, 112
preformatting, 113–114
proportionally spaced fonts, 113
superscripts and subscripts, 114–115
style-sheet creation programs, 312
style sheets, cascading (CSS)
absolute vs. relative positioning, 317–319
browser support for, 270
building menus with, 277–279
cascading priorities, 270–274
and character styles compared, 111
classes, 275
CSS1 properties, 280–288
CSS2 properties, 288–311
defined, 269, 870
embedded style sheet, 272
ID selectors, 275–276
linking external style sheets, 273–274
nested elements, 276–279
positioning properties, 315–316
redefining HTML elements, 274–275
scroll bars, 326–327
style attribute, 270, 271
style element, 270
style sheet-creation programs, 312
<STYLE> tag, 272
support information sites, 271
uses of, 269, 274
Web sites concerning, 313
XHTML and, 720–721

www.it-ebooks.info

<STYLE> tag, embedding CSS with, 272
subdomains, 87–88
Submit button, adding to forms, 352
subscriber, 881
subscripts, 114–115
superscripts, 114–115
swapping banners, 34–35
syndication source, 522
syndrome, barrier, 409
system, address verification (AVS), 575

T

TABLE element, 212, 332
tables
  adding, 211–212
  adding images to, 232
  alignment of, 220
  background colors, 234–240
  background images, 233–234
  borderless, 218
  borders, 217–218
  cell padding, 218–219
  cell spacing, 219–220
  defined, 211
  with forms, 222–224
  horizontal alignment, 220–227
  preventing word wrap, 228–229
  rules, 218
  setting size, 211–216
  with sidebar images, 225–227
  spanning rows and columns, 229–232
  specifying background color for, 234–240
  vertical alignment, 228
tab order, 359–360t
tag, 881
tag, ALT, 22–23
task, 800
TCP/IP. *See* Transmission Control Protocol/Internet Protocol (TCP/IP)
telecommuting, 881
Telnet, 881
testing authoring tools, 56–57
text
  aligning and indenting, 119–123
  aligning images with, 153–157
  BASE element, 116–119
  BLOCKQUOTE element, indenting with, 122–123
  boldfacing, 112–113
  cascading style sheets (CSS), 286–287, 308–309
  CENTER element, 120
  character styles, 111–115
  choosing character sets, 123–127
  color, 119, 160, 197
  default fonts, overriding, 117–119
  deprecation and, 120–122
  FONT element, 116–119
  formatting in Blogger, 515–516
  heading elements, 43
  horizontal lines to divide, 44–47
  italicizing, 112
  line breaks, 43–44
  in Macromedia Flash, 466–469
  as placeholder for missing image, 150–152
  preformatting, 113–114
  special characters and entities, 127–135
  style, maintaining consistency in, 67
  subscripts and superscripts, 114–115
  white space, 47–48
  word wrap around images, 155–157
text areas, 339–342
  default contents, setting, 342
  sizing, 339, 340
  word wrap, setting, 339–341
text boxes, 334–339
  disabled attribute, 339
  locking content of, 338–339
  maximum length of input, defining, 336–337
  and password fields compared, 335
  prefilling, 337–338
  purpose of, 334
  readonly attribute, 338–339
  size, setting, 335–336
text editors, 49–50, 881
text file, 881
text-level elements, 40
text style, 67
theft of public domain music, 493
3D graphics programs, 187–192
  Bryce, 189, 190
  DAZ3D free model, 192
  iSpace, 191
  Poser, 188–189
  trueSpace, 189–191
thumbnails
  Blogger templates, 513
  images, 161
tiered navigation, 67
tiling, background images, 164
Timeline, Animation (Macromedia Flash)
  adding actions, 478–479
  adding layers, 470, 471
  adding MP3 sound, 477
  adding objects to layers, 470, 471
  adding text, 466
  as an advertising format, 34

www.it-ebooks.info

Animation Timelines, 469–477
assigning keyframes, 470
bitmap fills, 465–466
choosing fonts, 466–467
converting objects to symbols, 470, 472
creating animation, 472
creating layer effects, 473–475
creating objects, 454, 456–460
creating text effects, 468–469
Default panel, 455
described, 449
Design panels, 455
Development panels, 456
exporting movies, 479–481
frames, 469–470
gradient fills, 463–464
grouping objects, 461–462
importing art, 459–460
importing sound files into, 476
launching for the first time, 449
layers, 451, 470, 473–475
layout, components of, 449
Library panel, 460
Line tools, 456, 457
making shapes with tools, 458–459
modifying objects, 460–466
objects to symbols, converting, 470, 472
optimizing movies, tips for, 483–485
panels, 454–456
programming with ActionScript, 477–478
publishing movies, 481–483
rotating and skewing objects, 463
saving work, 459
scaling objects, 462–463
scenes, 450–451
selecting and deleting objects, 457–458
setting font characteristics, 467–468
setting stroke characteristics, 456–457
sound and, 476
the Stage, 450
straightening and smoothing objects, 463
stretching and distorting objects, 460–461
symbols, 470
synchronizing sound and action, 477
the Timeline, 451–452
Tools panel, 452–454
tweening, 472–473
TITLE tag, 21–22
tools, authoring
browser capability testers, 58–59
choosing, 55–57
code validation programs, 58
demos, trying out, 56–57
Dreamweaver, 55, 61–63, 146, 174
freeware, 57
HomeSite+, 60–61
HTML editors, 51
iSpace, 55
link checkers, 58
Notepad, 60
server monitor services, 59–60
shareware, 57
text editors, 49–50
types of, 49
word processors, 54–55
WYSIWYG programs, 51–54
tools, image editing
Fireworks, 174, 175, 178, 179, 183
Painter, 174, 175, 178, 180, 183
Paint Shop Pro, 174, 176, 178, 180, 183–184
Photoshop, 173, 174, 178, 179, 180, 181–182
tool tips, 151–152
topic-specific Web pages, 70–71
trademarks, 143
Anti-Cybersquatting Consumer Protection Act, 597, 598
cease and desist orders, 597
defined, 595
protecting, 597
researching online, 599–600, 601
trading banners, 34–35
traffic, 881
transactions, Web-based business (e-commerce), 872
advertising, 579–592, 607
affiliate stores, 544, 545
B2B (business-to-business), 870
B2C (business-to-commerce), 870
chargebacks, 575
checks, 541–542, 543, 576–577
credit cards, 541, 542, 571–573, 575
credit fraud, 575–601–602
digital certificates, 536, 537, 542
e-cash, 542–544, 577
encryption, 601, 603
general stores, 530–531
Internet buying patterns, 544, 546–547
online computer stores, 532, 533
online shopping malls, 534–535, 536–538, 605
online superstores, 531–532
open source software, 538–540
PayPal accounts, 536, 561–562, 577
personal information, asking customers to provide, 603
privacy policies, 603, 604
return and refund policies, 573–575, 603, 605–606

www.it-ebooks.info

transactions, Web-based business (e-commerce) *(continued)*
security and, 536, 601–603
selling on eBay, 549–569
service sites, 532, 534
setup, options for, 529
shopping carts, 535–540
single product sites,, 529
specialty stores, 529–530
taking payment, 540–544
Yahoo! Store, 536
Transmission Control Protocol/Internet Protocol (TCP/IP), 879, 881
transparent processes, 73
trueSpace 3D imaging program, 189–191
trying out tools, 56–57
tutorials, 630–631
tweening, 472–473
Type Definition, Document (DTD)
overview, 652–653
in XHTML 1.0, 716–717
in XML, 642–643, 652–653
type faces (fonts)
consistency in selection of, 67
embedding in images, 162–163
in Macromedia Flash, 466–468
monospace, 113
proportionally spaced, 13
sans serif, 117
serif, 117
TypePad, 519–520
types of Web sites, analyzing
commercial sites, 3, 8–10
informational sites, 4–6
organizational sites, 6–7
personal sites, 3, 4
political sites, 4, 8, 9
typical charges for banners, 33

U

Uniform Domain Name Resolution Policy (UDRP), 597–598
uniform resource locators (URLs)
absolute, 72
addressing resources with, 72–73
color information, 207
defined, 72, 88
examples of for Web site types, 6–7, 9–10
Internet research firms, 12
music sources, 494
online shopping malls, 538
partial, 72
prefixes, 73
privacy policies, samples of, 604
relative, 73
trademark research sites, 601
uses of, 72
Universal Character Set (UCS), 123
Unix, 881
unlimited
bandwidth, 94
traffic, 94
unload event, 387
updating
adding extra value, 629–631
browser compatibility problems, 639–640
content, 625
establishing a schedule for, 626–627
finding new material, 627
Frequently Asked Questions (FAQs), 627
incremental uploads, 626
making HTML XML-compliant, 637–639
need for, 625
newsletters and e-zines, 629–630
online chats, 628
online training, 631
product reviews, 630
transitioning to XHTML 1.0, 637–640
tutorials, 630–631
visitor-generated material, 627–629
and XML, 631–640
upload
defined, 881
incremental, 626
uploading pages, 101–107
transferring with FTP, 102–107
with web browsers via HTTP, 107
unordered lists, defined, 420
upsampling images, 178
upstreaming, 522
URL, absolute, 72
ULR extensions
.biz, 80
.com, 80
.coop, 80
.gov, 80
.info, 80
.mil, 80
.name, 80
.net, 80
URL, partial, 73
URLs
absolute, 72
addressing resources with, 72–73
color information, 207
defined, 72, 88
examples of for Web site types, 6–7, 9–10
Internet research firms, 12
music sources, 494

www.it-ebooks.info

online shopping malls, 538
partial, 72
prefixes, 73
privacy policies, samples of, 604
relative, 73
trademark research sites, 601
uses of, 72
usability testing, 70
use of banners, 32–35
use policy, acceptable, 869
user name, 882
using animation
hazards of, 429–430
using with Macromedia Flash, 449–485
using public domain music, 493
using text
aligning and indenting, 119–123
aligning images with, 153–157
BASE element, 116–119
BLOCKQUOTE element, indenting with, 122–123
boldfacing, 112–113
cascading style sheets (CSS), 286–287, 308–309
CENTER element, 120
character styles, 111–115
choosing character sets, 123–127
color, 119, 160, 197
default fonts, overriding, 117–119
deprecation and, 120–122
FONT element, 116–119
formatting in Blogger, 515–516
heading elements, 43
horizontal lines to divide, 44–47
italicizing, 112
line breaks, 43–44
in Macromedia Flash, 466–469
as placeholder for missing image, 150–152
preformatting, 113–114
special characters and entities, 127–135
style, maintaining consistency in, 67
subscripts and superscripts, 114–115
white space, 47–48
word wrap around images, 155–157

V

validation programs for code, 58
value-added services
newsletters and e-zines, 592, 593, 629–630
online chats, 628
online training, 631
product reviews, 633
tutorials, 630–631
values, in JavaScript
affecting with operators, 369–371
event handlers, 371–372
types, 368
values-setting attribute(s), 42
variables
arithmetic operator, 370–371
assigning different values, 368
assignment operator, 366–367
comparison operator, 369–370
declaring, 366
defined, 366
logical operator, 371
name application, 380
naming, 366–368
scope of, 380
valid and invalid, 368
value types, 368
VBScript, 882
Vector Markup Language (VML), 652
verification system, address (AVS), 575
VeriSign, 536, 537, 602
vertical vs. horizontal navigation bars, 417–418
video
digital sources of, 496–498
embedding, 494–497
movie clips, 496
virtual domain, 88
virtual server, 84, 882
virtual Web hosting, 88, 89–90
visibility and contrast in color(s), 206
VisiBone Webmaster's Color Lab, 201
VisionQuest Images VQ Color Site, 201, 202
visitors
browser preference, settings by, 618–621
comfort level, 621
getting statistics about, 97–98
Internet fears, soothing, 601–603
surveying, 12–13
in Web site planning, 13
VML (Vector Markup Language), 652
volume, setting, 490
VQ Color Site (VisionQuest Images), 201, 202

W

warm colors, 206–207
Web
access to, 874
advertising agencies, 588–589
common fears about, 600–603
defined, 874
finding reliable information sources on, 10–12
research firms, selected URLs for, 1

www.it-ebooks.info

Web addresses (URLs)
absolute, 72
addressing resources with, 72–73
color information, 207
defined, 72, 88
examples of for Web site types, 6–7, 9–10
Internet research firms, 12
music sources, 494
online shopping malls, 538
partial, 72
prefixes, 73
privacy policies, samples of, 604
relative, 73
trademark research sites, 601
uses of, 72
Web address extensions
.biz, 80
.com, 80
.coop, 80
.gov, 80
.info, 80
.mil, 80
.name, 80
.net, 80
Web advertising
accepting on personal Web sites, 607
affiliate marketing, 589–590
with banners, 17, 32–35, 587–588, 607, 608
via e-mail, 590–591
with Google, 586, 579–585
hover, 592
Internet advertising agencies, 588–589
mailing lists, 591–592
multimedia ads, 34
newsletters, 592, 593
online, typical costs per month, 33
with search engines other than Google, 585–587
traditional vehicles for, 35
Web affiliate, 869
Web authoring tools
browser capability testers, 58–59
choosing, 55–57
code validation programs, 58
demos, trying out, 56–57
Dreamweaver, 55, 61–63, 146, 174
freeware, 57
HomeSite+, 60–61
HTML editors, 51
iSpace, 55
link checkers, 58
Notepad, 60
server monitor services, 59–60
shareware, 57
text editors, 49–50
types of, 49
word processors, 54–55
WYSIWYG programs, 51–54
Web-based business transactions (e-commerce), 872
advertising, 579–592, 607
affiliate stores, 544, 545
B2B (business-to-business), 870
B2C (business-to-commerce), 870
chargebacks, 575
checks, 541–542, 543, 576–577
credit cards, 541, 542, 571–573, 575
credit fraud, 575–601–602
digital certificates, 536, 537, 542
e-cash, 542–544, 577
encryption, 601, 603
general stores, 530–531
Internet buying patterns, 544, 546–547
online computer stores, 532, 533
online shopping malls, 534–535, 536–538, 605
online superstores, 531–532
open source software, 538–540
PayPal accounts, 536, 561–562, 577
personal information, asking customers to provide, 603
privacy policies, 603, 604
return and refund policies, 573–575, 603, 605–606
security and, 536, 601–603
selling on eBay, 549–569
service sites, 532, 534
setup, options for, 529
shopping carts, 535–540
single product sites, 529
specialty stores, 529–530
taking payment, 540–544
Yahoo! Store, 536
Web-based buying (eBay), 539–569
after the auction, 565–566
Buy Now option, 560
checks as payment option, 562–563
debit cards, 541, 571–573
eBay template, 566–567
enhancing success with, 566
establishing yourself as a reputable dealer, 564–565
fees for selling on, 552–554
modifying the eBay template, 567
monitoring your auction, 565
overview of, 549–550
payment and shipping options, 561–564
PayPal accounts, 561–562

www.it-ebooks.info

setting a start and finish time, 565
setting up an account, 551–552
setting up an auction, 554–560
setting up a Dutch (multiple item) auction, 560–576
setting up an eBay store, 568
shipping merchandise, 563
template, eBay, 566–577
Web browser(s)
compatibility. *See* browser compatibility
defined, 882
handling of images in tables, 233
help balloons, 151
sniffing, 404
support for CSS properties, 270
support for Document Object Model (DOM), 437–441
testing for use with cascading style sheets, 270
tool tips, 151–152
tracking use of, 59
uploading via HTTP, 107
using more than one to test Web projects, 54
versions, programming for, 59
visitor-customized preference settings, 618–621
Web-building site (AOL Hometown), 518–519
WebCrawler (search engine), 18, 20
Web directories, 27–29
categorization style of, 27–29
Free-For-All sites, 29
ranking within, 27–28
registering with, 28–29
and search engines compared, 27
Web site promotion via, 17, 27–29
Yahoo!, 27–29
Web domain name
buying a taken name, 80
celebrity names as, 597
choosing, 79–82
country-specific, 83
disputes over, 596–598
extensions, 79–80, 599
legal concerns about, 595–596
maintaining, 598–599
multiple-year discounts, 83
registering derivatives, 599
registrar, 82–83
registration delays, 84
registration pitfalls, 83–84
researching online, 598
securing, 79–84
theft of, 93–94
trademarks as, 597
transferring, 82
Uniform Domain Name Resolution Policy (UDRP), 597–598
Web hosting, virtual, 88, 89–90
Web image(s), 874
adding to tables, 232
aligning text with, 153–157
alignment settings, 155
artistic effects filters, 184–187
background, adding, 163–164
background, in tables, 233–234
borders around, 148–150
calculating sizes for tiling, 164
clip art, 176–177
copyrighted materials, using, 142–143
cropping, 177–178
defined, 874
editing tools. *See* image editing tools
embedding special fonts, 162–163
floating, 153–155
free artwork, downloading, 139
free, off the Web, 138–141
graphics files, 137–138
height and width, changing, 146–147
horizontal and vertical spacing, 147–148
importing into Macromedia Flash, 459–460
inserting into Web pages, 145–150
Irfan View program, 176
legal concerns, 141–143
to link files, 157–161
low-resolution to speed download time, 152–153
modifying, 176–177
multiple, aligning with text, 157
original art, finding artists, 139–140
placeholders, 150–153
public domain photographs,138-139
resizing and resampling, 178–180
rotating and flipping, 180–184
sharpening and blurring, 184
stock photographs, 140–141
swapping with rollovers, 415–417, 430–432
3D graphics programs, 187–192
thumbnail, to speed up page downloads, 161–162
trademarks, 143
word wrap around, 155–157
Web mail (e-mail)
advertising via, 590–591
for communicating with customers, 31
defined, 872
extra accounts, 95
formatting, plain text vs. HTML, 32

www.it-ebooks.info

Web mail (e-mail) *(continued)*
locating addresses, 31–32, 591–592
mailing lists, 591–592
sending with mailto links, 77
spam, 17, 31, 590, 629, 880
unsubscribing option, 32
username, 872
Web site promotion via, 17
Webmaster, 882
Webmaster's Color Lab (VisiBone), 201
Web page
defined, 882
loading of, 387–388
navigation bars, 410–412
unloading of, 388
Web page design
color, 66
content, 68
functionality and, 68–69
navigation, 67
text style, 67
Web page downloads, speeding up
using low-resolution images, 152–153
using thumbnail images, 161–162
Web page elements, dynamic
animation, hazards of, 429
browser support for, 437–441
color fades, 445–447
Document Object Model (DOM) and, 435–447
positioning, 435–448
purpose of, 429
using rollovers to change, 430–435
Web page generation program, 878
Web page planning
absolute technology, reasons for using, 14
budgeting for, 15
deciding on complexity levels, 14–15
facts, reliable sources of, 10–12
leading-edge technology and, `4
setting goals, 13
surveying site visitors, 12–13
Web site types, analyzing, 3–10
Web Pages That Suck site, 104
Web-safe colors, 200–203
Web server
categories of, 84
choosing, 84–87
co-located, 84, 88–89
dedicated, 84, 88
defined, 84
monitor services, 59–60
non-virtual hosting, 87–88
and operating system, reliable combinations, 85
price-service ratios, 89–90
server extensions, 85
space needs, determining, 87
uploading pages with File Transfer Protocol (FTP), 102–106
uploading pages with Web browsers via HTTP, 107
virtual hosting, 84, 88, 89
Web shopping carts, 535–540
defined, 535–536, 880
commercial software for, 538, 539
online shopping malls, 536–538
open source software for, 538, 540
security, 536
Web site animation
hazards of, 429–430
using with Macromedia Flash, 449–485
Web site, defined, 882
Web site design
color, 66
content, 68
functionality and, 68, 69
navigation, 67
text style, 67
Web site types, analyzing
commercial sites, 3, 8–10
informational sites, 4–6
organizational sites, 6–7
personal sites, 3, 4
political sites, 4, 8, 9
Web site visitors
browser preference, settings by, 618–621
comfort level, 621
getting statistics about, 97–98
Internet fears, soothing, 601–603
surveying, 12–13
in Web site planning, 13
Web space, defined, 882
Web space providers
autoresponders, 97
contracts with, 91
control panels to set site options, 98–101
customer service, 92
deadlines for special deals, 94–95
domain name registration, 94
domain name theft, 93–94
extra e-mail accounts, 95
finding, 84–91
getting visitor statistics, 97–98
investigating, 90–91
redirecting messages, 95–97
reliability of, 91

www.it-ebooks.info

reselling a Web space, 101
scams, avoiding, 92–95
technical support, 92
testing responsiveness, 91–92
unlimited traffic or space, 94
WebSphere Home Page Builder, 53
WebTrends software, 20
Web URLs
absolute, 72
addressing resources with, 72–73
color information, 207
defined, 72, 88
examples of for Web site types, 6–7, 9–10
Internet research firms, 12
music sources, 494
online shopping malls, 538
partial, 72
prefixes, 73
privacy policies, samples of, 604
relative, 73
trademark research sites, 601
uses of, 72
white space, HTML text, 47–48
width attribute, 146–147, 213
window object, 393
Windows
character sets for, 127
HTML editor sources, 51
text editor sources, 50
WYSIWYG editor sources, 53
Wireless Markup Language (WML)
cards, 664–666
defined, 659–660
format, specifying, 667–669
forms in, 667
lists, compared to HTML, 672–673
menus, 669–671
multiple-choice menus, 671–672
required values, specifying, 669
tags, 662–663
transforming XHTML into, 673–676
user interaction, 666–673
using, 660–662
and XHTML compared, 659
Wireless Markup Language 2.0 (WML2.0)
adding style to, 849–850
attribute expression syntax, 815–821
BACK key, 827
Base Module, 840
change history, 864
Context and Navigation Module, 841–848
default style sheet, 850, 851
document conformance, 848
documents, identification of, 826
Document Type Definition (DTD) for, 797–798, 850–851
elements, 851–859
event model, 821–825
Events Module, 841
form processing reference model, 804–815
Forms Module, 834–837
Hypertext Module, 834
Image Module, 838
IMG element, 839
informative references, 793–794
Link Module, 840
Lists Module, 838
markup elements and attributes, 832–848
Metainformation Module, 839–840
navigation reference processing model, 800–804
normative references, 792–793
Object Module, 840
Presentation Module, 841
specification, 787–864
static conformance requirements, 859–864
Structure Module, 832–833
Style Sheet Module, 840
Tables Module, 837–838
terminology and conventions, 794–796
Text Module, 833
timer processing, 829–830
user agent behavior, 799–832
user agent conformance rules, 830–832
wm1:do element, 827–829
XHTML, acceptance of, 830
Wireless Markup Language script (WMLscript), 660
WML
cards, 664–666
defined, 659–660
format, specifying, 667–669
forms in, 667
lists, compared to HTML, 672–673
menus, 669–671
multiple-choice menus, 671–672
required values, specifying, 669
tags, 662–663
transforming XHTML into, 673–676
user interaction, 666–673
using, 660–662
and XHTML compared, 659
WMLScript, 660
WML 2.0
adding style to, 849–850
attribute expression syntax, 815–821
BACK key, 827
Base Module, 840

www.it-ebooks.info

WML 2.0 *(continued)*
change history, 864
Context and Navigation Module, 841–848
default style sheet, 850, 851
document conformance, 848
documents, identification of, 826
Document Type Definition (DTD) for, 797–798, 850–851
elements, 851–859
event model, 821–825
Events Module, 841
form processing reference model, 804–815
Forms Module, 834–837
Hypertext Module, 834
Image Module, 838
IMG element, 839
informative references, 793–794
Link Module, 840
Lists Module, 838
markup elements and attributes, 832–848
Metainformation Module, 839–840
navigation reference processing model, 800–804
normative references, 792–793
Object Module, 840
Presentation Module, 841
specification, 787–864
static conformance requirements, 859–864
Structure Module, 832–833
Style Sheet Module, 840
Tables Module, 837–838
terminology and conventions, 794–796
Text Module, 833
timer processing, 829–830
user agent behavior, 799–832
user agent conformance rules, 830–832
wm1:do element, 827–829
XHTML, acceptance of, 830
word processors, coding HTML with, 54–55
word processor text
aligning and indenting, 119–123
aligning images with, 153–157
BASE element, 116–119
BLOCKQUOTE element, indenting with, 122–123
boldfacing, 112–113
cascading style sheets (CSS), 286–287, 308–309
CENTER element, 120
character styles, 111–115
choosing character sets, 123–127
color, 119, 160, 197
default fonts, overriding, 117–119
deprecation and, 120–122
FONT element, 116–119
formatting in Blogger, 515–516
heading elements, 43
horizontal lines to divide, 44–47
italicizing, 112
line breaks, 43–44
in Macromedia Flash, 466–469
as placeholder for missing image, 150–152
preformatting, 113–114
special characters and entities, 127–135
style, maintaining consistency in, 67
subscripts and superscripts, 114–115
white space, 47–48
word wrap around images, 155–157
word wrap
around images, 155–157
preventing in tables, 228–229
setting in text boxes (forms), 339–341
values, 341
work-for-hire (WFH) contract, 142
working with tables
adding, 211–212
adding images to, 232
alignment of, 220
background colors, 234–240
background images, 233–234
borderless, 218
borders, 217–218
cell padding, 218–219
cell spacing, 219–220
defined, 211
with forms, 222–224
horizontal alignment, 220–227
preventing word wrap, 228–229
rules, 218
setting size, 211–216
with sidebar images, 225–227
spanning rows and columns, 229–232
vertical alignment, 228
working with text
aligning and indenting, 119–123
aligning images with, 153–157
BASE element, 116–119
BLOCKQUOTE element, indenting with, 122–123
boldfacing, 112–113
cascading style sheets (CSS), 286–287, 308–309
CENTER element, 120
character styles, 111–115
choosing character sets, 123–127
color, 119, 160, 197
default fonts, overriding, 117–119
deprecation and, 120–122
FONT element, 116–119
formatting in Blogger, 515–516

www.it-ebooks.info

heading elements, 43
horizontal lines to divide, 44–47
italicizing, 112
line breaks, 43–44
in Macromedia Flash, 466–469
as placeholder for missing image, 150–152
preformatting, 113–114
special characters and entities, 127–135
style, maintaining consistency in, 67
subscripts and superscripts, 114–115
white space, 47–48
word wrap around images, 155–157
World Wide Web (WWW), 882
access to, 874
advertising agencies, 588–589
common fears about, 600–603
defined, 874
finding reliable information sources on, 10–12
research firms, selected URLs for, 1
World Wide Web advertising
accepting on personal Web sites, 607
affiliate marketing, 589–590
with banners, 17, 32–35, 587–588, 607, 608
via e-mail, 590–591
with Google, 586, 579–585
hover, 592
Internet advertising agencies, 588–589
mailing lists, 591–592
multimedia ads, 34
newsletters, 592, 593
online, typical costs per month, 33
with search engines other than Google, 585–587
traditional vehicles for, 35
World Wide Web directories, 27–29
categorization style of, 27–29
Free-For-All sites, 29
ranking within, 27–28
registering with, 28–29
and search engines compared, 27
Web site promotion via, 17, 27–29
Yahoo!, 27–29
World Wide Web page
defined, 882
loading of, 387–388
navigation bars, 410–412
unloading of, 388
World Wide Web page design
color, 66
content, 68
functionality and, 68–69
navigation, 67
text style, 67
World Wide Web page downloads, speeding up
using low-resolution images, 152–153
using thumbnail images, 161–162
World Wide Web page elements, dynamic
animation, hazards of, 429
browser support for, 437–441
color fades, 445–447
Document Object Model (DOM) and, 435–447
positioning, 435–448
purpose of, 429
using rollovers to change, 430–435
World Wide Web page generation program, 878
World Wide Web page planning
absolute technology, reasons for using, 14
budgeting for, 15
deciding on complexity levels, 14–15
facts, reliable sources of, 10–12
leading-edge technology and, 14
setting goals, 13
surveying site visitors, 12–13
Web site types, analyzing, 3–10
WYSIWYG (What You See Is What You Get) programs, 51–54, 61

X

x-axis. *See* axis
XHTML (eXtensible Hypertext Markup Language)
cascading style sheets (CSS) and, 720–721
defined, 705, 872
Mobile Profile document type, 797
need for, 706–707
transforming into WML, 673–676
using with other namespaces, 710–711
XHTML 1.0
browser compatibility problems, 656–657
compatibility issues, 715–716
DTDs, 716–717
element prohibitions, 717
HTML compatibility, 637–639, 654–655, 717–722
and HTML 4.0 compared, 639, 655–656, 712–715
normative definition of, 709–712
specification, 701–725
transitioning to, 637–640, 653–657
XML (eXtensible Markup Language)
as alternative to frames, 265
classifying content vs. defining presentation, 633, 643–644
CSS technology, 644–645
customizing tags, 633–634, 651

www.it-ebooks.info

XML (eXtensible Markup Language) *(continued)*
defined, 631, 706
designing with, 641–642
documents, defining the structure of, 649–651
Document Type Definition (DTD) in, 634–636, 642–643, 652–653
features of, 631
and HTML compared, 631
namespaces, 642–643
overview, 631–632
rules, 642
vocabularies, 636–637, 642–643
XSL style sheet, specifying, 645–649
XML 1.0
autodetection of character encodings, 781–783
character classes, 775–779
conformance, 769–770
deterministic content models, 780–781
documents, handling, 734–745
expansion of entity and character references, 779–780
logical structures, 746–757
notation, 770–772
origin and goals, 732
physical structures, 757–769
production notes, 785
references, 772–775
and SGML, 779
specification, 727–785
terminology, 732–734
W3C XML core working group, 784–785
W3C XML working group, 783–784
XSL (eXtensible Style Language), 265, 631, 873

Y

Yahoo! (search engine), advertising with, 587
Yahoo!Shops, 606
Yahoo! Store, 536, 537
Yahoo! Web directory
categorizationa by, 27
ranking within, 27–28
registration proces, 28–29
y-axis. *See* axis

Z

z-axis. *See* aAxis
z-order, 321, 882

www.it-ebooks.info