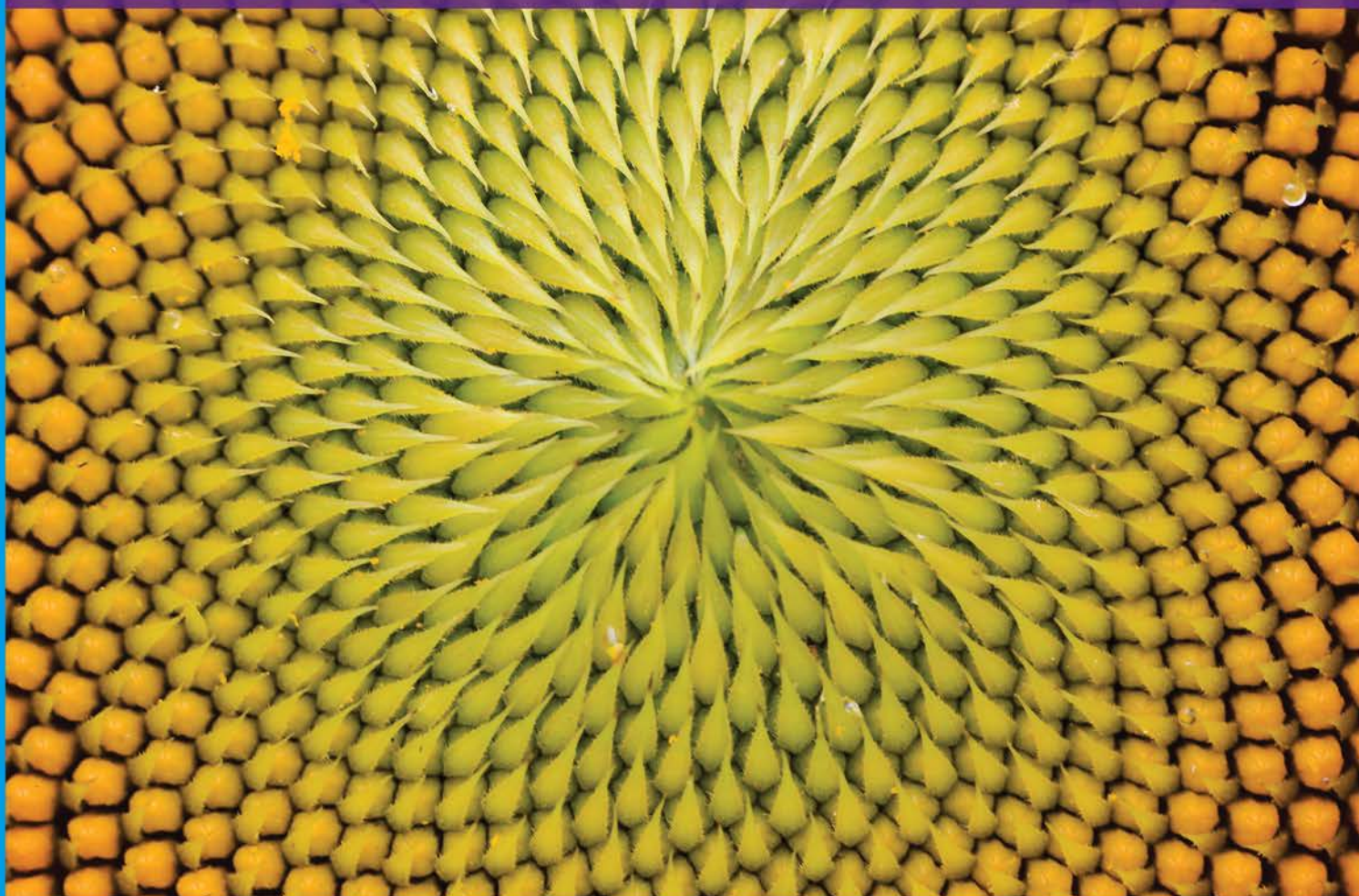


NEW PERSPECTIVES

CAREY

HTML 5 and CSS

Comprehensive

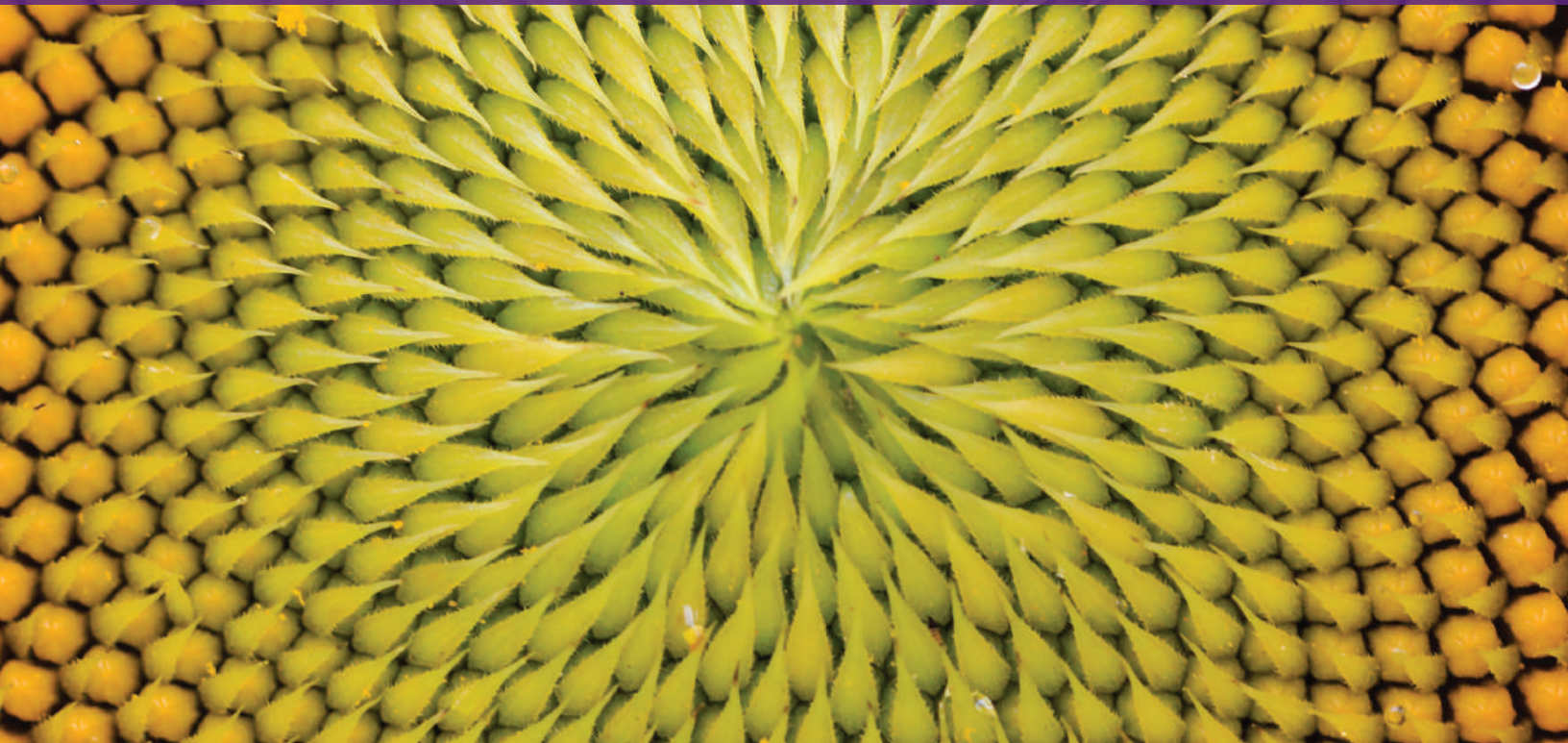


NEW PERSPECTIVES

PATRICK CAREY

HTML 5 and CSS

Comprehensive



Australia • Brazil • Mexico • Singapore • United Kingdom • United States

This is an electronic version of the print textbook. Due to electronic rights restrictions, some third party content may be suppressed. Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. The publisher reserves the right to remove content from this title at any time if subsequent rights restrictions require it. For valuable information on pricing, previous editions, changes to current editions, and alternate formats, please visit www.cengage.com/highered to search by ISBN#, author, title, or keyword for materials in your areas of interest.

Important Notice: Media content referenced within the product description or the product text may not be available in the eBook version.

**New Perspectives on HTML 5 and CSS,
8th Edition Comprehensive**
Patrick Carey

SVP, Higher Education Product Management:
Erin Joyner

VP, Product Management: Mike Schenk
Product Director: Lauren Murphy
Product Team Manager: Kristin McNary
Product Assistant: Tom Benedetto
Director, Learning Design: Rebecca von Gillern
Senior Manager, Learning Design: Leigh Hefferon
Learning Designer: Kate Mason
Vice President, Marketing – Science, Technology,
& Math: Jason Sakos
Senior Marketing Director: Michele McTighe
Marketing Manager: Cassie L Cloutier
Marketing Development Manager:
Samantha Best
Director, Content Creation: Juliet Steiner
Senior Manager, Content Creation: Patty Stephan
Content Manager: Christina Nyren
Director, Digital Production Services:
Krista Kellman
Digital Delivery Lead: Justin Maniaci
Technical Editors: John Freitas and Danielle Shaw
Developmental Editors: Deb Kaufmann and
Ann Shaffer
Production Service/Composition: Lumina
Datamatics Ltd.
Design Director: Jack Pendleton
Designer: Erin Griffin
Text Designer: Althea Chen
Cover Template Designer: Wing-Ip Ngan,
Ink Design, Inc.
Cover image(s): Mai Phongsook/Shutterstock.com

© 2021, 2017 Cengage Learning, Inc.

Unless otherwise noted, all content is © Cengage.

WCN: 02-300

ALL RIGHTS RESERVED. No part of this work covered by the copyright herein may be reproduced or distributed in any form or by any means, except as permitted by U.S. copyright law, without the prior written permission of the copyright owner.

For product information and technology assistance, contact us at
**Cengage Customer & Sales Support, 1-800-354-9706 or
support.cengage.com.**

For permission to use material from this text or product,
submit all requests online at **www.cengage.com/permissions.**

Library of Congress Control Number: 2019940290

ISBN: 978-0-357-10714-0

Cengage
200 Pier 4 Boulevard
Boston, MA 02210
USA

Cengage is a leading provider of customized learning solutions with employees residing in nearly 40 different countries and sales in more than 125 countries around the world. Find your local representative at **www.cengage.com.**

Cengage products are represented in Canada by Nelson Education, Ltd.

To learn more about Cengage platforms and services, register or access your online learning solution, or purchase materials for your course, visit **www.cengage.com.**

Notice to the Reader

Publisher does not warrant or guarantee any of the products described herein or perform any independent analysis in connection with any of the product information contained herein. Publisher does not assume, and expressly disclaims, any obligation to obtain and include information other than that provided to it by the manufacturer. The reader is expressly warned to consider and adopt all safety precautions that might be indicated by the activities described herein and to avoid all potential hazards. By following the instructions contained herein, the reader willingly assumes all risks in connection with such instructions. The publisher makes no representations or warranties of any kind, including but not limited to, the warranties of fitness for particular purpose or merchantability, nor are any such representations implied with respect to the material set forth herein, and the publisher takes no responsibility with respect to such material. The publisher shall not be liable for any special, consequential, or exemplary damages resulting, in whole or part, from the readers' use of, or reliance upon, this material.

Preface

The New Perspectives Series' critical-thinking, problem-solving approach is the ideal way to prepare students to transcend point-and-click skills and take advantage of all that HTML 5 and CSS has to offer.

In developing the New Perspectives Series, our goal was to create books that give students the software concepts and practical skills they need to succeed beyond the classroom. We've updated our proven case-based pedagogy with more practical content to make learning skills more meaningful to students. With the New Perspectives Series, students understand *why* they are learning *what* they are learning, and are fully prepared to apply their skills to real-life situations.

"I love this text because it provides detailed instructions and real-world application examples. It is ideal for classroom and online instruction. At the end of the term my students comment on how much they've learned and put to use outside the classroom."

—Customer at
St. Johns River
Community College

About This Book

This book provides thorough coverage of HTML 5 and CSS, and includes the following:

- Up-to-date coverage of using HTML 5 to create structured websites
- Instruction on the most current CSS styles to create visually-interesting pages and captivating graphical designs
- Working with browser developer tools to aid in the creation and maintenance of fully-functioning websites

New for this edition!

- Coverage of CSS grid styles for creating grid-based layouts.
- Exploration of new CSS styles for filters and transformations.
- New hands-on demo pages to interactively explore HTML and CSS concepts.
- New coding challenges for each tutorial to focus on specific tasks and concepts.
- New debugging challenges for each tutorial to explore how to fix malfunctioning websites.

System Requirements

This book assumes that students have an Internet connection, a text editor, and a current browser that supports HTML 5 and CSS. The following is a list of the most recent versions of the major browsers at the time this text was published: Internet Explorer 11, Microsoft Edge 44, Firefox 66, Safari 12.1, Opera 60, and Google Chrome 75. More recent versions may have come out since the publication of this book. Students should go to the web browser home page to download the most current version. All browsers interpret HTML 5 and CSS code in slightly different ways. It is highly recommended that students have several different browsers installed on their systems for comparison and, if possible, access to a mobile browser or a mobile emulator. Students might also want to run older versions of these browsers to highlight compatibility issues. The screenshots in this book were produced using Google Chrome 75 running on Windows 10 (64-bit), unless otherwise noted. If students are using different devices, browsers, or operating systems, their screens might vary from those shown in the book; this should not present any problems in completing the tutorials.

“New Perspectives texts provide up-to-date, real-world application of content, making book selection easy. The step-by-step, hands-on approach teaches students concepts they can apply immediately.”

—Customer at
Southeastern Technical
College

The New Perspectives Approach

Context

Each tutorial begins with a problem presented in a “real-world” case that is meaningful to students. The case sets the scene to help students understand what they will do in the tutorial.

Hands-on Approach

Each tutorial is divided into manageable sessions that combine reading and hands-on, step-by-step work. Colorful screenshots help guide students through the steps. **Trouble?** tips, which anticipate common mistakes or problems, help students stay on track and continue with the tutorial.

VISUAL OVERVIEW

Visual Overviews

Each session begins with a Visual Overview, a two-page spread that includes colorful, enlarged figures with numerous callouts and key term definitions, giving students a comprehensive preview of the topics covered in the session, as well as a handy study guide.

PROSKILLS

ProSkills Boxes

ProSkills boxes provide guidance for applying concepts to real-world, professional situations, involving one or more of the following soft skills: decision making, problem solving, teamwork, verbal communication, and written communication.

KEY STEP

Key Steps

Important steps are highlighted in yellow with attached margin notes to help students pay close attention to completing the steps correctly and avoid time-consuming rework.

INSIGHT

InSight Boxes

InSight boxes offer expert advice and best practices to help students achieve a deeper understanding of the concepts behind the software features and skills.

TIP

Margin Tips

Margin Tips provide helpful hints and shortcuts for more efficient use of the software. The Tips appear in the margin at key points throughout each tutorial, giving students extra information when and where they need it.

TRY IT

Try It tips point to demo pages provided with the data folder for interactive exploration of key concepts.

REVIEW

Assessment

Retention is a key component to learning. At the end of each session, a series of Quick Check multiple choice questions helps students test their understanding of the material before moving on. New with this edition are Coding Challenges and debugging exercises that focus on a few key challenges. Engaging end-of-tutorial Review Assignments and Case Problems have always been a hallmark feature of the New Perspectives Series. Colorful bars and brief descriptions accompany the exercises, making it easy to understand both the goal and level of challenge a particular assignment holds.

CODE

DEBUG

APPLY

CHALLENGE

CREATE

REFERENCE

Reference

Within each tutorial, Reference boxes appear before a set of steps to provide a succinct summary or preview of how to perform a task. In addition, each book includes a combination Glossary/Index to promote easy reference of material.

GLOSSARY/INDEX

INTRODUCTORY

COMPREHENSIVE

Our Complete System of Instruction

Coverage To Meet Your Needs

Whether you're looking for just a small amount of coverage or enough to fill a semester-long class, we can provide you with a textbook that meets your needs.

- Introductory books contain an average of 5 to 8 tutorials and include essential skills on the books concepts.
- Comprehensive books, which cover additional concepts and skills in depth, are great for a full-semester class, and contain 9 to 12+ tutorials.

So, if you are looking for just the essential skills or more complete in-depth coverage of a topic, we have an offering available to meet your needs. Go to our web site or contact your Cengage sales representative to find out what else we offer.

MindTap

MindTap is a personalized learning experience with relevant assignments that guide students to analyze, apply, and improve thinking, allowing you to measure skills and outcomes with ease.

For instructors: personalized teaching becomes yours with a Learning Path that is built with key student objectives. Control what students see and when they see it. Use as-is, or match to your syllabus exactly: hide, rearrange, add, or create your own content.

For students: a unique Learning Path of relevant readings, multimedia, and activities that guide you through basic knowledge and comprehension to analysis and application.

Better outcomes: empower instructors and motivate students with analytics and reports that provide a snapshot of class progress, time in course, engagement, and completion rates.

The MindTap for HTML 5 and CSS includes coding labs, study tools, and interactive quizzing, all integrated into an eReader that includes the full content of the printed text.

Instructor Resources

We offer more than just a book. We have all the tools you need to enhance your lectures, check students' work, and generate exams in a new, easier-to-use and completely revised package. This book's Instructor's Manual, Cengage testbank, PowerPoint presentations, data files, solution files, figure files, and a sample syllabus are all available at sso.cengage.com.

Acknowledgments

I would like to thank the people who worked so hard to make this book possible. Special thanks to my developmental editors, Deb Kaufmann and Ann Shaffer, for their hard work, attention to detail, and valuable insights, and to Content Manager, Christina Nyren, who has worked tirelessly in overseeing this project and made my task so much easier with enthusiasm and good humor. Other people at Cengage who deserve credit are Kristin McNary, Program Team Lead;

Kate Mason, Learning Designer; Tom Benedetto, Product Assistant; Erin Griffin, Art Director; Fola Orekoya, Manufacturing Planner; Lumina Datamatics Ltd., Compositor, as well as John Freitas and Danielle Shaw, Technical Editors.

This book is dedicated to my wife Joan who is my inspiration and role model for her good humor, dedication, and tireless support.

– **Patrick Carey**

BRIEF CONTENTS

HTML

Level I Tutorials

- Tutorial 1** Getting Started with HTML 5 HTML 1
Creating a Website for a Food Vendor
- Tutorial 2** Getting Started with CSS HTML 85
Designing a Website for a Fitness Club

Level II Tutorials

- Tutorial 3** Designing a Page Layout HTML 175
Creating a Website for a Chocolatier
- Tutorial 4** Graphic Design with CSS HTML 273
Creating a Graphic Design for a Genealogy Website
- Tutorial 5** Designing for the Mobile Web HTML 361
Creating a Mobile Website for a Daycare Center

Level III Tutorials

- Tutorial 6** Working with Tables and Columns HTML 451
Creating a Program Schedule for a Radio Station
- Tutorial 7** Designing a Web Form HTML 517
Creating a Survey Form
- Tutorial 8** Enhancing a Website with Multimedia HTML 601
Working with Sound, Video, and Animation
- Tutorial 9** Getting Started with JavaScript HTML 681
Creating a Countdown Clock
- Tutorial 10** Exploring Arrays, Loops, and Conditional Statements . . . HTML 751
Creating a Monthly Calendar
- Appendix A** Color Names with Color Values, and HTML Character Entities HTML A1
- Appendix B** HTML Elements and Attributes HTML B1
- Appendix C** Cascading Styles and Selectors HTML C1
- Appendix D** Making the Web More Accessible HTML D1
- Appendix E** Designing for the Web HTML E1
- Appendix F** Page Validation with XHTML HTML F1

Glossary

REF 1

Index

REF 11

TABLE OF CONTENTS

Preface	iii
---------------	-----

HTML LEVEL I TUTORIALS

Tutorial 1 Getting Started with HTML 5

<i>Creating a Website for a Food Vendor</i>	HTML 1
---	--------

SESSION 1.1.....HTML 2

Exploring the World Wide Web.....	HTML 4
Networks	HTML 4
Locating Information on a Network	HTML 4
Web Pages and Web Servers	HTML 4
Introducing HTML	HTML 5
The History of HTML.....	HTML 5
Tools for Working with HTML.....	HTML 6
Content Management Systems and Frameworks.....	HTML 7
Testing your Code.....	HTML 7
Exploring an HTML Document	HTML 8
The Document Type Declaration.....	HTML 8
Introducing Element Tags.....	HTML 9
The Element Hierarchy.....	HTML 10
Introducing Element Attributes.....	HTML 11
Handling White Space	HTML 12
Viewing an HTML File in a Browser.....	HTML 12
Creating an HTML File.....	HTML 13
Creating the Document Head.....	HTML 15
Setting the Page Title	HTML 16
Adding Metadata to the Document	HTML 16
Adding Comments to Your Document.....	HTML 18
Session 1.1 Quick Check	HTML 21

SESSION 1.2

HTML 22	
Writing the Page Body	HTML 24
Using Sectioning Elements.....	HTML 24
Comparing Sections in HTML 4 and HTML 5 ...	HTML 26
Using Grouping Elements	HTML 26
Using Text-Level Elements.....	HTML 29
Linking an HTML Document to a Style Sheet	HTML 32
Working with Character Sets and Special Characters	HTML 33
Character Encoding	HTML 33
Character Entity References.....	HTML 34
Working with Inline Images.....	HTML 36
Line Breaks and Other Empty Elements	HTML 38
Working with Block Quotes and Other Elements ...	HTML 39
Session 1.2 Quick Check	HTML 45

SESSION 1.3

HTML 46	
Working with Lists	HTML 48
Ordered Lists	HTML 48
Unordered Lists	HTML 49
Description Lists.....	HTML 51
Navigation Lists	HTML 55
Working with Hypertext Links.....	HTML 57
Turning an Inline Image into a Link	HTML 59
Specifying the Folder Path	HTML 60
Absolute Paths.....	HTML 61
Relative Paths	HTML 61
Setting the Base Path	HTML 62
Linking to a Location within a Document.....	HTML 63

Marking Locations with the id Attribute	HTML 63
Linking to an id	HTML 63
Anchors and the name Attribute	HTML 63
Linking to the Internet and Other Resources	HTML 64
Linking to a Web Resource	HTML 64
Linking to an Email Address	HTML 65
Linking to a Phone Number	HTML 67
Working with Hypertext Attributes	HTML 68
Validating Your Website	HTML 69
Session 1.3 Quick Check	HTML 71
Review Assignments	HTML 76
Case Problems	HTML 79
Tutorial 2 Getting Started with CSS	
<i>Designing a Website for a Fitness Club</i>	HTML 85
SESSION 2.1	HTML 86
Introducing CSS	HTML 88
Types of Style Sheets	HTML 88
Viewing a Page Using Different Style Sheets . . .	HTML 89
Exploring Style Rules	HTML 92
Browser Extensions	HTML 92
Embedded Style Sheets	HTML 93
Inline Styles	HTML 93
Style Specificity and Precedence	HTML 94
Style Inheritance	HTML 94
Browser Developer Tools	HTML 95
Creating a Style Sheet	HTML 96
Writing Style Comments	HTML 96
Defining the Character Encoding	HTML 97
Importing Style Sheets	HTML 98
Working with Color in CSS	HTML 98
Color Names	HTML 98
RGB Color Values	HTML 99
HSL Color Values	HTML 101
Defining Semi-Opaque Colors	HTML 102
Setting Text and Background Colors	HTML 102
Employing Progressive Enhancement	HTML 106
Session 2.1 Quick Check	HTML 107
SESSION 2.2	HTML 108
Exploring Selector Patterns	HTML 110
Contextual Selectors	HTML 110
Attribute Selectors	HTML 113
Working with Fonts	HTML 117
Choosing a Font	HTML 117
Exploring Web Fonts	HTML 119
The @font-face Rule	HTML 120
Setting the Font Size	HTML 123
Absolute Units	HTML 123
Relative Units	HTML 123
Scaling Fonts with ems and rems	HTML 124
Using Viewport Units	HTML 125
Sizing Keywords	HTML 125
Controlling Spacing and Indentation	HTML 127
Working with Font Styles	HTML 129
Aligning Text Horizontally and Vertically	HTML 130
Combining All Text Formatting in a Single Style	HTML 131
Session 2.2 Quick Check	HTML 133
SESSION 2.3	HTML 136
Formatting Lists	HTML 138
Choosing a List Style Type	HTML 138
Creating an Outline Style	HTML 138

Using Images for List Markers	HTML 141
Setting the List Marker Position.	HTML 142
Working with Margins and Padding.	HTML 143
Setting the Padding Space	HTML 144
Setting the Margin and the Border Spaces	HTML 146
Using Pseudo-Classes and Pseudo-Elements	HTML 149
Pseudo-Classes	HTML 149
Pseudo-classes for Hypertext	HTML 152
Pseudo-Elements	HTML 154
Generating Content with CSS	HTML 155
Displaying Attribute Values.	HTML 156
Inserting Quotation Marks.	HTML 157
Validating Your Style Sheet.	HTML 158
Session 2.3 Quick Check	HTML 160
Review Assignments	HTML 166
Case Problems	HTML 169

HTML LEVEL II TUTORIALS

Tutorial 3 Designing a Page Layout

Creating a Website for a Chocolatier HTML 175

SESSION 3.1 **HTML 176**

Introducing the display Style	HTML 178
Creating a Reset Style Sheet.	HTML 178
Exploring Page Layout Designs.	HTML 182
Fixed, Fluid, and Elastic Layouts	HTML 182
Working with Width and Height.	HTML 184
Setting Maximum and Minimum Dimensions. . .	HTML 184
Centering a Block Element	HTML 187
Vertical Centering	HTML 188
Floating Page Content.	HTML 189
Clearing a Float	HTML 193
Refining a Floated Layout	HTML 197

Working with Container Collapse	HTML 201
Session 3.1 Quick Check	HTML 204

SESSION 3.2 **HTML 206**

Introducing Grid Layouts.	HTML 208
Overview of Grid-Based Layouts.	HTML 208
Fixed and Fluid Grids	HTML 209
CSS Frameworks	HTML 210
Introducing CSS Grids.	HTML 210
Creating a CSS Grid	HTML 213
Working with Grid Rows and Columns	HTML 215
Track Sizes with Fractional Units	HTML 217
Repeating Columns and Rows	HTML 218
Applying a Grid Layout.	HTML 219
Outlining a Grid	HTML 221
Placing Items within a Grid	HTML 223
Placing Items by Row and Column	HTML 224
Using the span Keyword	HTML 226
Placing Grid Items by Area.	HTML 228
Defining the Grid Gap	HTML 232
Managing Space within a Grid.	HTML 234
Alignment for a Single Grid Cell	HTML 235
Aligning the Grid	HTML 235
Session 3.2 Quick Check	HTML 237
SESSION 3.3 HTML 238	
Positioning Objects	HTML 240
The CSS Positioning Styles.	HTML 240
Relative Positioning	HTML 240
Absolute Positioning.	HTML 241
Fixed and Inherited Positioning	HTML 244
Using the Positioning Styles	HTML 244

Handling Overflow	HTML 254
Clipping an Element	HTML 257
Stacking Elements	HTML 258
Session 3.3 Quick Check	HTML 260
Review Assignments	HTML 267
Case Problems	HTML 269

Tutorial 4 Graphic Design with CSS

<i>Creating a Graphic Design for a Genealogy Website</i>	<i>HTML 273</i>
--	-----------------

SESSION 4.1 HTML 274

Creating Figure Boxes	HTML 276
Exploring Background Styles	HTML 280
Tiling a Background Image	HTML 281
Attaching the Background Image	HTML 283
Setting the Background Image Position	HTML 283
Defining the Extent of the Background	HTML 284
Sizing and Clipping an Image	HTML 285
The background Property	HTML 286
Adding Multiple Backgrounds	HTML 288
Working with Borders	HTML 290
Setting Border Width and Color	HTML 290
Setting the Border Design	HTML 291
Creating Rounded Corners	HTML 293
Applying a Border Image	HTML 297
Session 4.1 Quick Check	HTML 301

SESSION 4.2 HTML 302

Creating Drop Shadows	HTML 304
Creating a Text Shadow	HTML 304
Creating a Box Shadow	HTML 306
Applying a Color Gradient	HTML 312
Creating a Linear Gradient	HTML 312

Gradients and Color Stops	HTML 315
Creating a Radial Gradient	HTML 316
Repeating a Gradient	HTML 320
Creating Semi-Transparent Objects	HTML 322
Session 4.2 Quick Check	HTML 324

SESSION 4.3 HTML 326

Transforming Page Objects	HTML 328
Transformations in Three Dimensions	HTML 332
Understanding Perspective	HTML 333
Exploring CSS Filters	HTML 337
Working with Image Maps	HTML 341
Defining a Client-Side Image Map	HTML 341
Applying an Image Map	HTML 345
Session 4.3 Quick Check	HTML 347
Review Assignments	HTML 354
Case Problems	HTML 357

Tutorial 5 Designing for the Mobile Web

<i>Creating a Mobile Website for a Daycare Center</i>	<i>HTML 361</i>
---	-----------------

SESSION 5.1 HTML 362

Introducing Responsive Design	HTML 364
Introducing Media Queries	HTML 365
The @media Rule	HTML 366
Media Queries and Device Features	HTML 367
Applying Media Queries to a Style Sheet	HTML 369
Exploring Viewports and Device Width	HTML 372
Creating a Mobile Design	HTML 375
Creating a Pulldown Menu with CSS	HTML 376
Testing Your Mobile Website	HTML 379
Creating a Tablet Design	HTML 383
Creating a Desktop Design	HTML 387
Session 5.1 Quick Check	HTML 391

SESSION 5.2	HTML 392
Introducing Flexible Boxes	HTML 394
Defining a Flexible Box	HTML 394
Cross-Browser Flexboxes	HTML 395
Setting the Flexbox Flow	HTML 395
Working with Flex Items	HTML 397
Setting the Flex Basis	HTML 397
Defining the Flex Growth	HTML 398
Defining the Shrink Rate	HTML 399
The flex Property	HTML 401
Applying a Flexbox Layout	HTML 402
Reordering Page Content with Flexboxes	HTML 407
Exploring Flexbox Layouts	HTML 409
Aligning Items along the Main Axis	HTML 409
Aligning Flex Lines	HTML 410
Aligning Items along the Cross Axis	HTML 410
Creating a Navicon Menu	HTML 412
Session 5.2 Quick Check	HTML 417
SESSION 5.3	HTML 418
Designing for Printed Media	HTML 420
Previewing the Print Version	HTML 420
Applying a Media Query for Printed Output	HTML 421
Working with the @page Rule	HTML 422
Setting the Page Size	HTML 423
Using the Page Pseudo-Classes	HTML 423
Page Names and the Page Property	HTML 423
Formatting Hypertext Links for Printing	HTML 428
Working with Page Breaks	HTML 431
Preventing Page Breaks	HTML 432
Working with Widows and Orphans	HTML 434

Session 5.3 Quick Check	HTML 437
Review Assignments	HTML 443
Case Problems	HTML 446

HTML LEVEL III TUTORIALS

Tutorial 6 Working with Tables and Columns

<i>Creating a Program Schedule for a Radio Station</i> ..	HTML 451
---	----------

SESSION 6.1

Introducing Web Tables	HTML 454
Marking Tables and Table Rows	HTML 454
Marking Table Headings and Table Data	HTML 456
Adding Table Borders with CSS	HTML 459
Spanning Rows and Columns	HTML 464
Creating a Table Caption	HTML 471
Session 6.1 Quick Check	HTML 475

SESSION 6.2

Creating Row Groups	HTML 478
Creating Column Groups	HTML 482
Exploring CSS Styles and Web Tables	HTML 485
Working with Width and Height	HTML 486
Applying Table Styles to Other Page Elements	HTML 490
Tables and Responsive Design	HTML 492
Designing a Column Layout	HTML 496
Setting the Number of Columns	HTML 496
Defining Columns Widths and Gaps	HTML 498
Managing Column Breaks	HTML 501
Spanning Cell Columns	HTML 503
Session 6.2 Quick Check	HTML 505
Review Assignments	HTML 510
Case Problems	HTML 512

Tutorial 7 Designing a Web Form*Creating a Survey Form* HTML 517**SESSION 7.1** **HTML 518**

Introducing Web Forms HTML 520

Parts of a Web Form HTML 520

Forms and Server-Based Programs HTML 521

Starting a Web Form HTML 522

Interacting with the Web Server HTML 523

Creating a Field Set HTML 525

Marking a Field Set HTML 525

Adding a Field Set Legend HTML 526

Creating Input Boxes HTML 528

Input Types HTML 528

Input Types and Virtual Keyboards HTML 531

Adding Field Labels HTML 532

Designing a Form Layout HTML 534

Defining Default Values and Placeholders HTML 539

Session 7.1 Quick Check HTML 543

SESSION 7.2 **HTML 544**

Entering Date and Time Values HTML 546

Creating a Selection List HTML 547

Working with select Attributes HTML 549

Grouping Selection Options HTML 551

Creating Option Buttons HTML 553

Creating Check Boxes HTML 556

Creating a Text Area Box HTML 558

Session 7.2 Quick Check HTML 561

SESSION 7.3 **HTML 562**

Entering Numeric Data HTML 564

Creating a Spinner Control HTML 564

Creating a Range Slider HTML 566

Suggesting Options with Data Lists HTML 569

Working with Form Buttons HTML 572

Creating a Command Button HTML 572

Creating Submit and Reset Buttons HTML 572

Designing a Custom Button HTML 575

Validating a Web Form HTML 575

Identifying Required Values HTML 575

Validating Based on Data Type HTML 577

Testing for a Valid Pattern HTML 578

Defining the Length of the Field Value HTML 580

Applying Inline Validation HTML 581

Using the focus Pseudo-Class HTML 581

Pseudo-Classes for Valid and Invalid Data HTML 583

Session 7.3 Quick Check HTML 586

Review Assignments HTML 592

Case Problems HTML 595

Tutorial 8 Enhancing a Website with Multimedia*Working with Sound, Video, and Animation* . . . HTML 601**SESSION 8.1** **HTML 602**

Introducing Multimedia on the Web HTML 604

Understanding Codecs and Containers HTML 604

Understanding Plug-Ins HTML 605

Working with the audio Element HTML 607

Browsers and Audio Formats HTML 607

Applying Styles to the Media Player HTML 610

Providing a Fallback to an Audio Clip HTML 613

Exploring Embedded Objects HTML 615

Plug-In Attributes HTML 615

Plug-Ins as Fallback Options HTML 616

Session 8.1 Quick Check HTML 616

SESSION 8.2 HTML 618

Exploring Digital Video	HTML 620
Video Formats and Codecs	HTML 620
Using the HTML 5 video Element	HTML 621
Adding a Text Track to Video	HTML 624
Making Tracks with WebVTT	HTML 625
Placing the Cue Text	HTML 628
Applying Styles to Track Cues	HTML 630
Using Third-Party Video Players	HTML 634
Exploring the Flash Player	HTML 635
Embedding Videos from YouTube	HTML 636
HTML 5 Video Players	HTML 637
Session 8.2 Quick Check	HTML 639

SESSION 8.3 HTML 640

Creating Transitions with CSS	HTML 642
Introducing Transitions	HTML 642
Setting the Transition Timing	HTML 644
Delaying a Transition	HTML 647
Creating a Hover Transition	HTML 647
Animating Objects with CSS	HTML 652
The @keyframes Rule	HTML 652
Applying an Animation	HTML 655
Controlling an Animation	HTML 658
Session 8.3 Quick Check	HTML 666
Review Assignments	HTML 673
Case Problems	HTML 676

Tutorial 9 Getting Started with JavaScript

<i>Creating a Countdown Clock</i>	<i>HTML 681</i>
---	-----------------

SESSION 9.1 HTML 682

Introducing JavaScript	HTML 684
Server-Side and Client-Side Programming	HTML 684

The Development of JavaScript	HTML 685
Working with the script Element	HTML 686
Loading the script Element	HTML 686
Inserting the script Element	HTML 687
Creating a JavaScript Program	HTML 689
Adding Comments to your JavaScript Code	HTML 689
Writing a JavaScript Command	HTML 690
Understanding JavaScript Syntax	HTML 691
Debugging Your Code	HTML 692
Opening a Debugger	HTML 692
Inserting a Breakpoint	HTML 694
Applying Strict Usage of JavaScript	HTML 695
Session 9.1 Quick Check	HTML 697

SESSION 9.2 HTML 698

Introducing Objects	HTML 700
Object References	HTML 701
Referencing Object Collections	HTML 701
Referencing an Object by ID and Name	HTML 703
Changing Properties and Applying Methods	HTML 704
Object Properties	HTML 704
Applying a Method	HTML 704
Writing HTML Code	HTML 705
Working with Variables	HTML 709
Declaring a Variable	HTML 709
Variables and Data Types	HTML 710
Using a Variable	HTML 711
Working with Date Objects	HTML 711
Creating a Date Object	HTML 712
Applying Date Methods	HTML 713
Setting Date and Time Values	HTML 716
Session 9.2 Quick Check	HTML 717

SESSION 9.3	HTML 718
Working with Operators and Operands	HTML 720
Using Assignment Operators	HTML 720
Calculating the Days Left in the Year	HTML 721
Working with the Math Object	HTML 723
Using Math Methods	HTML 723
Using Math Constants	HTML 728
Working with JavaScript Functions	HTML 730
Calling a Function	HTML 732
Creating a Function to Return a Value	HTML 733
Running Timed Commands	HTML 734
Working with Time-Delayed Commands	HTML 734
Running Commands at Specified Intervals	HTML 734
Controlling How JavaScript Works with Numeric Values	HTML 736
Handling Illegal Operations	HTML 736
Defining a Number Format	HTML 737
Converting Between Numbers and Text	HTML 737
Session 9.3 Quick Check	HTML 739
Review Assignments	HTML 744
Case Problems	HTML 746
Tutorial 10 Exploring Arrays, Loops, and Conditional Statements	
<i>Creating a Monthly Calendar</i>	HTML 751
SESSION 10.1	HTML 752
Introducing the Monthly Calendar	HTML 754
Reviewing the Calendar Structure	HTML 755
Adding the calendar() Function	HTML 756
Introducing Arrays	HTML 757
Creating and Populating an Array	HTML 758
Working with Array Length	HTML 761

Reversing an Array	HTML 763
Sorting an Array	HTML 764
Extracting and Inserting Array Items	HTML 765
Using Arrays as Data Stacks	HTML 766
Session 10.1 Quick Check	HTML 769
SESSION 10.2	HTML 770
Working with Program Loops	HTML 772
Exploring the for Loop	HTML 772
Exploring the while Loop	HTML 774
Exploring the do/while Loop	HTML 775
Comparison and Logical Operators	HTML 776
Program Loops and Arrays	HTML 777
Array Methods to Loop Through Arrays	HTML 780
Running a Function for Each Array Item	HTML 781
Mapping an Array	HTML 781
Filtering an Array	HTML 782
Session 10.2 Quick Check	HTML 785
SESSION 10.3	HTML 786
Introducing Conditional Statements	HTML 788
Exploring the if Statement	HTML 789
Nesting if Statements	HTML 791
Exploring the if else Statement	HTML 793
Using Multiple else if Statements	HTML 794
Completing the Calendar App	HTML 796
Setting the First Day of the Month	HTML 797
Placing the First Day of the Month	HTML 798
Writing the Calendar Days	HTML 799
Highlighting the Current Date	HTML 801
Displaying Daily Events	HTML 803

Managing Program Loops and Conditional Statements	HTML 806
Exploring the break Command	HTML 806
Exploring the continue Command	HTML 806
Exploring Statement Labels	HTML 807
Session 10.3 Quick Check	HTML 809
Review Assignments	HTML 815
Case Problems	HTML 817

**Appendix A Color Names with Color Values,
and HTML Character Entities HTML A1**

Appendix B HTML Elements and Attributes . . . HTML B1

Appendix C Cascading Styles and Selectors . . HTML C1

Appendix D Making the Web

More Accessible HTML D1

Appendix E Designing for the Web HTML E1

Appendix F Page Validation with XHTML . . . HTML F1

GLOSSARY REF 1

INDEX REF 11

Getting Started with HTML 5

Creating a Website for a Food Vendor

Case | *Curbside Thai*

Sajja Adulet is the owner and master chef of Curbside Thai, a restaurant owner and now food truck vendor in Charlotte, North Carolina that specializes in Thai dishes. Sajja has hired you to develop the company's website. The website will display information about Curbside Thai, including the truck's daily locations, menu, catering opportunities, and contact information. Sajja wants the pages to convey the message that customers will get the same great food and service whether they order in the restaurant or from the food truck. Some of the materials for these pages have already been completed by a former employee and Sajja needs you to finish the job by converting that work into a collection of web page documents. To complete this task, you'll learn how to write and edit HTML 5 code and how to get your HTML files ready for display on the World Wide Web.

OBJECTIVES

Session 1.1

- Explore the history of the web
- Create the structure of an HTML document
- Insert HTML elements and attributes
- Insert metadata into a document
- Define a page title

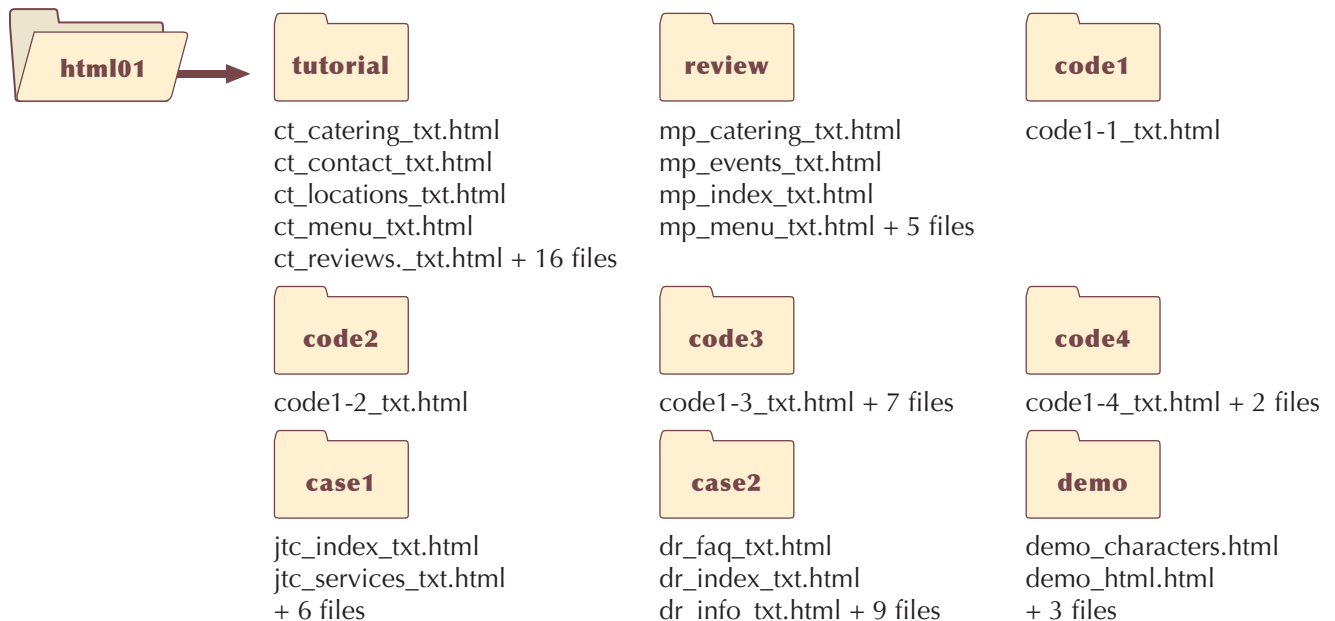
Session 1.2

- Mark page structures with sectioning elements
- Organize page content with grouping elements
- Mark content with text-level elements
- Insert inline images
- Insert symbols based on character codes

Session 1.3

- Mark content using lists
- Create a navigation list
- Link to files within a website with hypertext links
- Link to email addresses and telephone numbers

STARTING DATA FILES



Session 1.1 Visual Overview:

The **document type declaration** is a processing instruction indicating the markup language used in the document.

The **<html>** tag marks the beginning of the HTML document.

The **<head>** tag marks the **document head** containing information about the document.

The **<meta>** tag marks metadata containing information about the document.

The **<body>** tag marks the **document body** containing all of the content that will appear in the page.

An **opening tag** marks the start of the element content; this tag marks the start of page footer.

An **HTML comment** is a descriptive note added to the HTML file.

The **<title>** tag marks the page title that appears on the browser title bar or browser tab.

A **closing tag** marks the end of the element content; this tag marks the end of the page footer.

```
<!DOCTYPE html>
<html>
  <head>
    <!-- Curbside Thai Starting Page -->
    <meta charset="utf-8" />
    <link href="ct_base.css" rel="stylesheet" />
    <link href="ct_layout1.css" rel="stylesheet" />
    <title>Curbside Thai</title>
  </head>
  <body>
    <header>
      
    </header>
    <nav>
      <a href="ct_about.html"></a>
      <a href="ct_locations.html"></a>
      <a href="ct_menu.html"></a>
      <a href="ct_reviews.html"></a>
      <a href="ct_catering.html"></a>
      <a href="ct_contact.html"></a>
    </nav>
    <footer>
      Curbside Thai &#8226; 411 Belde Drive, Charlotte NC 28201 &#8226; 704-555-1151
    </footer>
  </body>
</html>
```

© Kzenon/Shutterstock.com;
© martiapunts/Shutterstock.com;
© Brian A Jackson/Shutterstock.com;
© sayhmog/Shutterstock.com;
© rangizzz/Shutterstock.com

The Structure of an HTML Document

Document as it appears
in the browser.



The exact layout of the
document elements is
determined by a style
sheet and not by the
document markup.

Kzenon/Shutterstock.com; © Courtesy Patrick Carey; martiapunts/Shutterstock.com; Brian A Jackson/Shutterstock.com; A Studios/Shutterstock.com; rangizzz/Shutterstock.com

Exploring the World Wide Web

It is no exaggeration to say that the World Wide Web has had as profound an effect on human communication as the printing press. One key difference is that operation of the printing press was limited to a few select tradesmen but on the web everyone can be a publisher of a website. Before creating your first website, you'll examine a short history of the web because that history impacts the way you write code for your web pages. You'll start by exploring the basic terminology of computer networks.

Networks

A **network** is a structure in which information and services are shared among devices known as **nodes** or **hosts**. A host can be any device that is capable of sending or receiving data electronically. The most common hosts that you will work with are desktop computers, laptops, tablets, mobile phones, and printers.

A host that provides information or a service to other devices on the network is called a **server**. For example, a print server provides printing services; a file server provides storage space for saving and retrieving files. The device receiving these services is called a **client**. A common network design is the **client-server network**, in which the clients access information provided by one or more servers.

Networks are classified based on the range of devices they cover. A network confined to a small geographic area, such as within a building or department, is referred to as a **local area network** or **LAN**. A network that covers a wider area, such as several buildings or cities, is called a **wide area network** or **WAN**. Wide area networks typically consist of two or more interconnected local area networks. The largest WAN in existence is the **Internet**, which incorporates an almost uncountable number of networks and hosts involving computers, mobile devices (such as phones, tablets, and so forth), MP3 players, and gaming systems.

Locating Information on a Network

The biggest obstacle to effectively using the Internet is the network's sheer scope and size. Most of the early Internet tools required users to master a bewildering array of terms, acronyms, and commands. Because network users had to be well versed in computers and network technology, Internet use was largely limited to programmers and computer specialists working for universities, large businesses, and the government.

The solution to this problem was developed in 1989 by Timothy Berners-Lee and other researchers at the CERN nuclear research facility near Geneva, Switzerland. They needed an information system that would make it easy for their researchers to locate and share data on the CERN network, and so developed a system of hypertext documents. **Hypertext** is a method of organization in which data sources are interconnected through a series of links or **hyperlinks** activated to jump from one data source to another. Hypertext is ideally suited for the Internet because end users don't need to know where a service is located—they only need to know how to activate the link. The effectiveness of this technique quickly spread beyond Geneva and was adopted across the Internet. The totality of these interconnected hypertext documents became known as the **World Wide Web**. The fact that the Internet and the World Wide Web are synonymous in many users' minds is a testament to the success of the hypertext approach.

Web Pages and Web Servers

Documents on the web are stored on **web servers** in the form of **web pages** and accessed through a software program called a **web browser**. The browser retrieves the

document from the web server and renders it in a form readable on a client device. However, because there is a wide selection of client devices ranging from desktop computers to mobile phones to screen readers that relay data aurally, each web page must be written in code that is compatible with every device. How does the same document work with so many different devices? To understand, you need to look at how web pages are created.

Introducing HTML

A web page is a simple text file written in **HTML (Hypertext Markup Language)**. You've already read about hypertext, but what is a markup language? A **markup language** is a language that describes the content and structure of a document by "marking up" or tagging, different document elements. For example, this tutorial contains several document elements such as the tutorial title, main headings, subheadings, paragraphs, figures, figure captions, and so forth. Using a markup language, each of these elements could be tagged as a distinct item within the "tutorial document." Thus, a Hypertext Markup Language is a language that supports tagging distinct document elements and connecting documents through hypertext links.

The History of HTML

In the early years, no single organization defined the rules or **syntax** of HTML. Browser developers were free to define and modify the language in different ways that, of course, led to problems as different browsers supported different "flavors" of HTML and a web page that was written based on one browser's standard might appear totally different when rendered by another browser. Ultimately, a group of web designers and programmers called the **World Wide Web Consortium**, or the **W3C**, settled on a set of standards or specifications for all browser manufacturers to follow. The W3C has no enforcement power, but, because using a uniform language is in everyone's best interest, the W3C's recommendations are usually followed, though not always immediately. Each new version of HTML goes through years of discussion and testing before it is formally adopted as the accepted standard. For more information on the W3C and its services, see its website at www.w3.org.

By 1999, HTML had progressed to the fourth version of the language, **HTML 4.01**, which provided support for multimedia, online commerce, and interactive scripts running within the web page. However, there were still many incompatibilities in how HTML was implemented across different browsers and how HTML code was written by web developers. The W3C sought to take control of what had been a haphazard process and enforce a stricter set of standards in a different version of the language called **XHTML (Extensible Hypertext Markup Language)**. By 2002, the W3C had released the specifications for XHTML 1.1. But XHTML 1.1 was intended to be only a minor upgrade on the way to XHTML 2.0, which would correct many of the deficiencies found in HTML 4.01 and become the future language of the web. One problem was that XHTML 2.0 would not be backward compatible with HTML and, as a result, older websites could not be easily brought into the new standard.

Web designers rebelled at this development and, in response, the **Web Hypertext Application Technology Working Group (WHATWG)** was formed in 2004 with the mission to develop a rival version to XHTML 2.0, called **HTML 5**. Unlike XHTML 2.0, HTML 5 would be compatible with earlier versions of HTML and would not apply the same strict standards that XHTML demanded. For several years, it was unclear which specification would win out; but by 2006, work on XHTML 2.0 had completely stalled and the W3C issued a new charter for WHATWG to develop HTML 5 as the de facto standard for the next generation of HTML. You can learn more about WHATWG and its current projects at www.whatwg.org. The current version of HTML is HTML 5.2, which achieved Recommendation status in 2017.

TIP

You can find out which browsers support the features of HTML 5 by going to the website caniuse.com.

As HTML has evolved, features and code found in earlier versions of the language are often **deprecated**, or phased out, and while deprecated features might not be part of HTML 5, that doesn't mean that you won't encounter them in your work—indeed, if you are maintaining older websites, you will often need to interpret code from earlier versions of HTML. Moreover, there are still many older browsers and devices in active use that do not support HTML 5. Thus, a major challenge for website designers is writing code that takes advantage of HTML 5 but is still accessible to older technology.

Figure 1–1 summarizes some of the different versions of HTML that have been implemented over the years. You can read detailed specifications for these versions at the W3C website.

Figure 1–1**HTML version history**

Version	Date	Description
HTML 1.0	1989	The first public version of HTML
HTML 2.0	1995	HTML version that added interactive elements including web forms
HTML 3.2	1997	HTML version that provided additional support for web tables and expanded the options for interactive form elements and a scripting language
HTML 4.01	1999	HTML version that added support for style sheets to give web designers greater control over page layout and appearance, and provided support for multimedia elements such as audio and video
XHTML 1.0	2001	A reformulation of HTML 4.01 using the XML markup language in order to provide enforceable standards for HTML content and to allow HTML to interact with other XML languages
XHTML 2.0	discontinued in 2009	The follow-up version to XHTML 1.1 designed to fix some of the problems inherent in HTML 4.01 syntax
HTML 5.0	2012	HTML version providing support for mobile design, semantic page elements, column layout, form validation, offline storage, and enhanced multimedia
HTML 5.2	2017	The current version of HTML 5

This book focuses on HTML 5, but you will also review some of the specifications for HTML 4.01 and XHTML 1.1. Deprecated features from older versions of HTML will be noted as such in the text.

Tools for Working with HTML

Because HTML documents are simple text files, the first tool you will need is a text editor. You can use a basic text editor such as Windows Notepad or TextEdit for the Macintosh, but it is highly recommended that you use one of the many inexpensive editors that provide built-in support for HTML. These editors include syntax checking to weed out errors and automatic insertion of HTML code. Some of the more popular HTML editors are Notepad++ (notepad-plus-plus.org), Eclipse (www.eclipse.org), and CoffeeCup (www.coffeecup.com).

These enhanced editors are a good way to start learning HTML and they will be all you need for most basic projects, but professional web developers working on large websites will quickly gravitate toward using a web **IDE (Integrated Development Environment)**, which is a software package providing comprehensive coverage of all

phases of the development process from writing HTML code to creating scripts for programs running on web servers. Some of the popular IDEs for web development include Adobe Dreamweaver (www.adobe.com), Aptana Studio (www.aptana.com), NetBeans IDE (netbeans.org), and Komodo IDE (komodoide.com). Web IDEs can be very expensive, but most software companies will provide a free evaluation period for you to test their product to see if it meets your needs.

Content Management Systems and Frameworks

You can also invest in a **web content management system (wcms)** which provides authoring tools for website content and administration. Management systems provide prepackaged templates so that users can get websites up and running with only a minimal knowledge of HTML. Popular content management systems include WordPress (www.wordpress.org), Joomla (www.joomla.org), and Drupal (www.drupal.org). Content management systems are not without drawbacks. A wcms can be expensive to maintain and put extra load on server resources. In addition, the templates and authoring tools can be difficult to modify if they don't exactly meet your needs.

A website usually involves the integration of many technologies and languages beyond HTML, including databases for storing and retrieving data and programs running on the web server for managing electronic commerce and communication. Managing all those technologies is the job of a **web framework** that provides the foundation of the design and deployment of web applications. Popular frameworks include Ruby on Rails (rubyonrails.org), ASP.NET (www.asp.net), AngularJS (angularjs.org), and Django (www.djangoproject.com).

Choosing among all these tools might seem intimidating to you. The bottom line is that no matter what tools you use, the final code for the website is written in HTML. So, even if that code is generated by a framework or content management system, you still need to understand HTML to effectively manage your website. In this book, we'll try to keep things as simple as possible: just you, a text editor, and a web browser creating a foundation for future study.

Testing your Code

TIP

You can analyze each browser for its compatibility with HTML 5 at the website www.html5test.com.

Once you've written your code, you can test whether your HTML code employs proper syntax and structure by validating it at the W3C validation website (validator.w3.org). **Validators**, like the one available through the W3C website, are programs that test code to ensure that it contains no syntax errors. The W3C validator will highlight all of the syntax errors in your document with suggestions about how to fix those errors.

Finally, you'll need to test it to ensure that your content is rendered correctly. You should test your code under a variety of screen resolutions, on several different browsers and, if possible, on different versions of the same browser because users are not always quick to upgrade their browsers. What may look good on a widescreen monitor might look horrible on a mobile phone. At a minimum you should test your website using the following popular browsers: Google Chrome, Internet Explorer, Apple Safari, Mozilla Firefox, and Opera.

It is not always possible to load multiple versions of the same browser on one computer, so, in order to test a website against multiple browser versions, professional designers will upload their code to online testing services that report on the website's compatibility across a wide range of browsers, screen resolutions, and devices, including both desktop and mobile devices. Among the popular testing services are BrowserStack (www.browserstack.com), CrossBrowserTesting (www.crossbrowstesting.com), and Browsera (www.browsera.com). Most of these sites charge a monthly connection fee with a limited number of testing minutes, so you should not upload your code until you are past the initial stages of development.

Exploring an HTML Document

Now that you have reviewed the history of the web and some of the challenges in developing your own website, you will look at the code of an actual HTML file. To get you started, Sajja Adulet has provided you with the `ct_start.html` file containing the code for the initial page users see when they access the Curbside Thai website. Open Sajja's file now.

TIP

All HTML files have the file extension `.html` or `.htm`.

To open the `ct_start.html` file:

1. Use the editor of your choice to open the `ct_start.html` file from the `html01` ► tutorial folder.

Figure 1–2 shows the complete contents of the file as viewed in the Notepad++ editor.

Figure 1–2

Elements and attributes from an HTML document

```

<!DOCTYPE html>
<html>
<head>
  <title>Curbside Thai</title>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1" />
  <link href="ct_base.css" rel="stylesheet" />
  <link href="ct_layout1.css" rel="stylesheet" />
</head>
<body>
  <header>
    
  </header>
  <nav>
    <a href="ct_about.html"></a>
    <a href="ct_locations.html"></a>
    <a href="ct_menu.html"></a>
    <a href="ct_reviews.html"></a>
    <a href="ct_catering.html"></a>
    <a href="ct_contact.html"></a>
  </nav>
  <footer>
    Curbside Thai &#8226; 411 Belde Drive, Charlotte NC 28201 &#8226; 704-555-1151
  </footer>
</body>
</html>

```

Trouble? Depending on your editor and its configuration, the text style applied to your code might not match that shown in Figure 1–2. This is not a problem. Because HTML documents are simple text files, any text styles are a feature of the editor and have no impact on how the document is rendered by the browser.

2. Scroll through the document to become familiar with its content but do not make any changes to the text.

The Document Type Declaration

The first line in an HTML file is the document type declaration or doctype, which is a processing instruction indicating the markup language used in the document. The browser uses the document type declaration to know which standard to use for displaying the content. For HTML 5, the doctype is entered as

```
<!DOCTYPE html>
```

You might also see the doctype entered in lowercase letters as

```
<!doctype html>
```

Both are accepted by all browsers. Older versions of HTML had more complicated doctypes. For example, the doctype for HTML 4.01 is the rather foreboding

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
```

You might even come across older HTML files that do not have a doctype. Because early versions of HTML did not require a doctype, many browsers interpret the absence of the doctype as a signal that the page should be rendered in **quirks mode**, based on styles and practices from the 1990s and early 2000s. When the doctype is present, browsers will render the page in **standards mode**, employing the most current specifications of HTML. The difference between quirks mode and standards mode can mean the difference between a nicely laid-out page and a confusing mess, so always put your HTML 5 file in standards mode by including the doctype.

Introducing Element Tags

The fundamental building block in every HTML document is the **element tag**, which marks an element in the document. A **starting tag** indicates the beginning of that element, while an **ending tag** indicates the ending. The general syntax of a two-sided element tag is

```
<element>content</element>
```

where *element* is the name of the element, *content* is the element's content, *<element>* is the starting tag, and *</element>* is the ending tag. The following code marks a paragraph element enclosed within the `<p>` and `</p>` tags:

```
<p>Welcome to Curbside Thai.</p>
```

The `<p></p>` tags indicate the presence of a paragraph and the text *Welcome to Curbside Thai.* comprises the paragraph text.

Not every element tag encloses document content. **Empty elements** are elements that are either nontextual (such as images) or contain directives to the browser about how the page should be treated. An empty element is entered using one of the following forms of the **one-sided element tag**:

```
<element />
```

or

```
<element>
```

The following `br` element indicates the presence of a line break in the text, and thus does not contain any content:

```
<br />
```

Note that, while this code could also be entered as `
`, the ending slash `/>` form is required in XHTML documents as well as other markup languages. While HTML 5 allows for either form, it's a good idea to get accustomed to using the ending slash `/>` form if you intend to work with other markup languages. We'll follow the `/>` convention in the code in this book.

Elements can contain other elements, which are called **nested elements**. In the following code, the `em` element (used to mark emphasized text) is nested within the paragraph element by placing the `` tag completely within the `<p>` tag.

Proper syntax:

```
<p>Welcome to <em>Curbside Thai</em>.</p>
```

When nesting one element inside of another, the entire code of the inner element must be contained within the outer element, including opening and closing tags. Thus, it would not be correct to place the closing tag for the `em` element outside of the `p` element as in the following code:

Improper syntax:

```
<p>Welcome to <em>Curbside Thai</p>.</em>
```

Now that you've examined the basics of tags, you'll examine at how they're organized within an HTML file.

The Element Hierarchy

The entire structure of an HTML document can be thought of as a set of nested elements in a hierarchical tree. At the top of the tree is the `html` element marking the entire document. Within the `html` element is the `head` element enclosing information about the document itself and the `body` element enclosing the content of the web page. Thus, the general structure of an HTML file, like the one shown in Figure 1–2, is

```
<!DOCTYPE html>
<html>
<head>
  head content
</head>

<body>
  body content
</body>
</html>
```

where *head content* and *body content* are nested elements placed within the document head and body. Note that the `body` element is always placed after the `head` element.

REFERENCE

Creating the Basic Structure of an HTML File

- To create the basic structure of an HTML file, enter the tags

```
<!DOCTYPE html>
<html>
<head>
  head content
</head>

<body>
  body content
</body>
</html>
```

where *head*, *content*, and *body content* contain nested elements that mark the content of the head and body sections.

TIP

Attributes can be listed in any order but they must come after the element name and be separated from each other by a blank space; each attribute value must be enclosed within single or double quotation marks.

Introducing Element Attributes

Elements often contain one or more **element attributes** providing additional information about the purpose of the element or how the element should be handled by the browser. The general syntax of an element attribute within a two-sided tag is

```
<element attr1="value1" attr2="value2" ...>
    content
</element>
```

Or, for a one-sided tag

```
<element attr1="value1" attr2="value2" ... />
```

where *attr1*, *attr2*, and so forth are attributes associated with *element* and *value1*, *value2*, and so forth are the corresponding attribute values. For example, the following code adds the `id` attribute with the value "intro" to the `<p>` tag in order to identify the paragraph as an introductory paragraph.

```
<p id="intro">Welcome to Curbside Thai.</p>
```

Each element has its own set of attributes but, in addition to these element-specific attributes, there is a core set of attributes that can be applied to almost every HTML element. Figure 1–3 lists some of the most commonly used core attributes; others are listed in Appendix B.

Figure 1–3

Commonly used core HTML attributes

Attribute	Description
<code>class="text"</code>	Defines the general classification of the element
<code>dir="ltr rtl auto"</code>	Defines the text direction of the element content as left-to-right, right-to-left, or determined by the browser
<code>hidden</code>	Indicates that the element should be hidden or is no longer relevant
<code>id="text"</code>	Provides a unique identifier for the element
<code>lang="text"</code>	Specifies the language of the element content
<code>style="definition"</code>	Defines the style or appearance of the element content
<code>tabindex="integer"</code>	Specifies the tab order of the element (when the tab button is used to navigate the page)
<code>title="text"</code>	Assigns a title to the element content

For attributes that do not require a value, HTML supports **attribute minimization** by removing the attribute value completely. For example, the `hidden` attribute used in the following code does not require a value; its mere presence indicates that the marked paragraph should be hidden in the rendered page.

```
<p hidden>Placeholder Text</p>
```

Attribute minimization is another example of how HTML 5 differs from other markup languages such as XHTML in which minimization is not allowed and all attributes must have attribute values.

Adding an Attribute to an Element

- To add an attribute to an element, enter

```
<element attr1="value1" attr2="value2" ...>
  content
</element>
```

where *attr1*, *attr2*, and so forth are HTML attributes associated with *element* and *value1*, *value2*, and so forth are the corresponding attribute values.

Handling White Space

An HTML file is composed only of text characters and white-space characters. A **white-space character** is any empty or blank character such as a space, tab, or line break. The browser reading the HTML code ignores the presence of white-space characters between element tags and makes no distinction between spaces, tabs, or line breaks. Thus, a browser will treat the following two pieces of code the same:

```
<p>Welcome to <em>Curbside Thai</em>.</p>
```

and

```
<p>
  Welcome to <em>Curbside Thai</em>.
</p>
```

The browser also collapses consecutive occurrences of white-space characters into a single occurrence, so that the text of the paragraph in the following code is still treated as “Welcome to Curbside Thai”, ignoring the extra white spaces between “Curbside” and “Thai”.

```
<p>
  Welcome to <em>Curbside      Thai</em>.
</p>
```

The bottom line is that it doesn’t matter how you lay out your HTML code because the browser is only interested in the text content and not how that text is entered. This means you can make your file easier to read by indenting lines and by adding extra white-space characters to separate one code block from another. However, this also means that any formatting you do for the page text to make the code more readable, such as tabs or extra white spaces, is *not* transferred to the web page.

Viewing an HTML File in a Browser

The structure of the HTML file shown in Figure 1–2 should now be a little clearer, even if you don’t yet know how to interpret the meaning and purpose of each of element and attribute. To see what this page looks like, open it within a web browser.

To open the `ct_start.html` file in a web browser:

1. Open your web browser. You do not need to be connected to the Internet to view local files stored on your computer.
2. After your browser loads its home page, open the `ct_start.html` file from the `html01 ▶ tutorial` folder. Figure 1–4 shows the page as it appears on a mobile phone and on a tablet device. The two devices have different screen widths, which affects how the page is rendered.