WHERE ARE THE LESSON FILES?

Purchase of this Classroom in a Book in any format gives you access to the lesson files you’ll need to complete the exercises in the book.

1  Go to adobepress.com/DreamweaverCIB2022.
2  Sign in or create a new account.
3  Click Submit.

4  Answer the questions as proof of purchase.
5  The lesson files can be accessed through the Registered Products tab on your Account page.
6  Click the Access Bonus Content link below the title of your product to proceed to the download page. Click the lesson file links to download them to your computer.

**Note:** If you encounter problems registering your product or accessing the lesson files or web edition, go to adobepress.com/support for assistance.

**Note:** If you purchased a digital product directly from adobepress.com or peachpit.com, your product will already be registered. However, you still need to follow the registration steps and answer the proof of purchase question before the Access Bonus Content link will appear under the product on your Registered Products tab.
CONTENTS

GETTING STARTED

About Classroom in a Book ................................................. 1
TinyURLs ........................................................................ 1
Prerequisites ...................................................................... 2
Conventions used in this book ........................................... 2
Bolded text ........................................................................ 2
Code font .......................................................................... 2
Strikethrough .................................................................... 3
Missing punctuation .......................................................... 3
Element references ........................................................... 3
Windows vs. macOS instructions ........................................ 3
Installing the program ....................................................... 4
Updating Dreamweaver to the latest version ....................... 5
Online content ............................................................... 6
Lesson files ...................................................................... 6
Web Edition ...................................................................... 6
Accessing the lesson files and Web Edition ....................... 6
Recommended lesson order ............................................... 7
Bonus material ................................................................... 8
On first launch ................................................................. 8
Choosing a color theme after first launch ......................... 9
Setting up the workspace .................................................. 9
Defining a Dreamweaver site ............................................. 11
Checking for updates ....................................................... 14
Additional resources ....................................................... 15
Adobe Authorized Training Centers .................................. 15
1 CUSTOMIZING YOUR WORKSPACE

Touring the workspace ........................................ 18
Using the Start Screen ........................................... 19
Quick Start ..................................................... 19
Starter Templates .............................................. 20
Create New and Open ......................................... 21
Exploring the New Feature guides ............................. 21
Switching and splitting views .................................. 22
Code view ....................................................... 22
Design view ..................................................... 22
Live view .......................................................... 23
Split view .......................................................... 24
Working with panels ............................................. 24
Minimizing panels .............................................. 25
Closing panels and panel groups .............................. 27
Dragging ............................................................ 27
Floating .............................................................. 28
Grouping, stacking, and docking ............................... 28
Personalizing Dreamweaver ..................................... 29
Saving a custom workspace .................................... 30
Working with toolbars .......................................... 30
Document toolbar .............................................. 30
Standard toolbar .............................................. 30
Common toolbar ............................................... 31
Creating custom keyboard shortcuts ......................... 31
Using the Property inspector ................................... 33
Using the HTML tab ........................................... 33
Using the CSS tab ............................................. 34
Accessing image properties ................................... 34
Accessing table properties ................................... 34
Using the Related Files interface ............................ 34
Using the Tag Selector interface ............................ 36
Using the CSS Designer ....................................... 37
Sources ........................................................... 38
@media ............................................................ 39
Selectors ........................................................... 39
Properties ......................................................... 40
All and Current modes .......................................... 41
Using the Visual Media Query (VMQ) interface .................. 42
Using the DOM Viewer ........................................ 42
Using element dialogs, displays, and inspectors ............. 43
*Position Assist dialog* ....................................... 43
*Element Display* ............................................. 43
*Image Display* ............................................... 44
*Text Display* ................................................ 44
*Setting up version control in Dreamweaver* ............... 44
Exploring, experimenting, and learning ..................... 46
Review questions ............................................. 47
Review answers ................................................ 47

2 HTML BASICS

What is HTML? ............................................... 50
Where did HTML begin?..................................... 50
*What comprises HTML?* ................................ 51
Frequently used HTML elements ........................... 52
*HTML tags* .................................................. 53
*HTML character entities* .................................. 55
What’s new in HTML5 ........................................ 55
*HTML5 tags* ............................................... 56
*Semantic web design* ..................................... 57
*New techniques and technology* ......................... 58
Review questions ............................................. 59
Review answers ................................................ 59

3 CSS BASICS

What is CSS? ................................................ 62
HTML vs. CSS formatting ................................... 62
HTML defaults ............................................... 64
*HTML5 defaults?* .......................................... 65
*Browser antics* ............................................. 66
CSS box model ................................................. 67
Applying CSS styling ....................................... 69
*Cascade theory* ............................................. 70
*Inheritance theory* ........................................ 72
*Descendant theory* ....................................... 72
Speciﬁcity theory .................................................. 73
CSS Designer .......................................................... 74
Multiples, classes, and ids, oh my! .............................. 80
Applying formatting to multiple elements ....................... 80
Using CSS shorthand .............................................. 80
Creating class attributes .......................................... 81
Creating id attributes .............................................. 82
CSS3 features and effects ......................................... 82
CSS3 overview and support ...................................... 84
Review questions ..................................................... 85
Review answers ....................................................... 85

4 WORKING WITH CODE 86
Creating HTML code ............................................... 88
Writing code manually ............................................. 88
Writing code automatically ....................................... 91
Working with multicursor support .............................. 95
Commenting your code .......................................... 97
Working with CSS preprocessors ............................... 99
Enabling a preprocessor ......................................... 99
Creating the CSS source ﬁle .................................... 101
Compiling CSS code .............................................. 105
Nesting CSS selectors ........................................... 108
Importing other style sheets .................................... 110
Learning more about preprocessors .......................... 112
Using linting support .......................................... 112
Selecting code ..................................................... 114
Using line numbers .............................................. 114
Using tag selectors .............................................. 115
Using parent tag selectors .................................... 117
Collapsing code ................................................... 118
Expanding code ................................................... 119
Accessing Split Code view ...................................... 120
Previewing assets in Code view ............................... 121
Review questions ............................................... 123
Review answers .................................................... 123
7  WORKING WITH TEMPLATES

Creating Dreamweaver templates ................................................................. 190
Removing unneeded components ................................................................. 191
Modifying a Bootstrap layout ....................................................................... 196
Modifying text formatting in a Bootstrap element ......................................... 201
Adding template boilerplate and placeholders ................................................. 203
Fixing semantic errors .................................................................................... 206
Inserting metadata ........................................................................................ 209
Validating HTML code ..................................................................................... 210
Working with editable regions ........................................................................ 212
  Image carousel ............................................................................................... 212
  Card-based section ....................................................................................... 212
  List-based section ....................................................................................... 212
  Inserting a new Bootstrap element ............................................................. 213
  Inserting an editable region ......................................................................... 216
  Inserting an editable optional region .......................................................... 219
Working with child pages ................................................................................ 222
  Creating a new page ...................................................................................... 222
  Adding content to child pages ..................................................................... 224
  Adding metadata to a child page .................................................................. 226
Updating a template ......................................................................................... 227
  Removing an optional region from a child page ............................................ 231
  Removing unused sections from a child page ................................................. 232
Review questions .............................................................................................. 234
Review answers ................................................................................................ 235

8  WORKING WITH TEXT, LISTS, AND TABLES

Previewing the completed file ........................................................................ 238
Creating and styling text ................................................................................ 240
  Importing text .............................................................................................. 240
  Duplicating Bootstrap rows ......................................................................... 244
  Deleting unused Bootstrap components ...................................................... 247
Creating lists ..................................................................................................... 248
  Basing content structures on lists ............................................................... 254
  Pasting multiple elements in Live view ...................................................... 258
  Creating new list-based items ...................................................................... 259
Creating and styling tables ................................................. 262
Creating tables from scratch ........................................ 262
Copying and pasting tables ............................................... 267
Styling tables with CSS .................................................. 268
Styling table cells ......................................................... 271
Adding header rows to tables ......................................... 272
Controlling table display ............................................... 274
Inserting tables from other sources ................................. 276
Creating semantic text structures .................................. 278
Adding and formatting caption elements ......................... 282
Spell-checking webpages .............................................. 284
Finding and replacing text ............................................. 285
Review questions .......................................................... 292
Review answers ............................................................. 293

9 WORKING WITH IMAGES .................................................. 294
Web image basics ......................................................... 296
Vector graphics ............................................................. 296
Raster graphics ............................................................. 296
Raster image file formats ............................................... 300
Inserting an image ........................................................ 303
Inserting images in Design view ..................................... 306
Resizing images ............................................................. 308
Inserting Photoshop Images .......................................... 311
Inserting images using the Assets panel ......................... 317
Adapting images to mobile design .................................. 319
Using the Insert menu .................................................... 321
Working with the Insert panel ........................................ 322
Inserting images into the site template ......................... 324
Adding CSS classes to template structures .................... 328
Adding images to a Bootstrap carousel ....................... 331
Styling headings and text in a Bootstrap carousel ........ 334
Self-paced exercise: Inserting images in child pages .... 336
Review questions .......................................................... 339
Review answers ............................................................. 339
12 WORKING WITH MOBILE DESIGN

Responsive design ........................................ 416
Mobile-first design ........................................ 416
Testing responsiveness in Dreamweaver ............... 416
Media type properties ..................................... 422
Media queries .............................................. 422
Media query syntax ...................................... 423
Working with the Visual Media Query interface ....... 423
Introducing web frameworks ............................... 427
Setting up web-hosted fonts ............................... 428
Setting the base font ..................................... 429
Building font stacks with web-hosted fonts ............ 436
Specifying font size ....................................... 438
Creating custom media queries ......................... 443
Adding rules to a media query ......................... 445
Controlling text styling with media queries .......... 448
Changing component visibility with media queries .. 453
Controlling Bootstrap element alignment ............. 460
Revising Bootstrap template structure ................. 464
Adapting content to mobile design .................... 469
Checking pages for design conflicts .................. 469
Making HTML tables responsive ....................... 471
Previewing pages using Real-Time Preview ............ 477
Fixing issues for mobile design and desktop .......... 482
Review questions ......................................... 485
Review answers ........................................... 485

APPENDIX: TINY URLS

INDEX
Lesson overview

In this lesson, you’ll learn how to work with code and do the following:

- Write code using code hinting and Emmet shorthand.
- Set up a CSS preprocessor and create SCSS styling.
- Use multiple cursors to select and edit code.
- Collapse and expand code entries.
- Use Live Code view to test and troubleshoot dynamic code.
- Use Inspect mode to identify HTML elements and associated styling.
- Access and edit attached files using the Related Files interface.

This lesson will take about 90 minutes to complete. To get the lesson files used in this lesson, download them from the webpage for this book at adobepress.com/DreamweaverCIB2022. Define a new site for the lesson04 folder, as described in the “Getting Started” section at the beginning of this book. Name the new site lesson04.
Dreamweaver’s claim to fame is as a visually based HTML editor, but its code-editing features don’t take a back seat to its graphical interface, and they offer few compromises to professional coders and developers.
Creating HTML code

As one of the leading WYSIWYG HTML editors, Dreamweaver allows users to create elaborate webpages and applications without touching or even seeing the code that does all the work behind the scenes. But for many designers, working with the code is not only a desire but a necessity.

Dreamweaver has always made it as easy to work with a page in Code view as it is in Design view or Live view. Dreamweaver can unify your entire web development team by providing a single platform that can handle almost any task.

You’ll often find that a specific task is actually easier to accomplish in Code view than in Live view or Design view alone. In the following exercises, you’ll learn more about how Dreamweaver makes working with the code an effortless and surprisingly enjoyable task.

Writing code manually

As you complete this and the next eight lessons, you will have numerous opportunities to view and edit code by hand. But for anyone jumping directly to this lesson, this exercise will provide a quick overview of the topic. One way to experience Dreamweaver’s code-writing and editing tools is to create a new file.

1 Define a site based on the lesson04 folder downloaded from your account page, as described in the “Getting Started” section at the beginning of the book.

2 Select Developer from the Workspace menu.

All the code-editing tools work identically in either workspace, but the Developer workspace focuses on the Code view window and provides a better experience for the following exercises.
3 Choose File > New.

The New Document dialog appears.

4 Choose New Document > HTML > None. Click Create.

Dreamweaver creates the basic structure of a webpage automatically. The cursor will normally appear at the beginning of the code when you are using the Developer workspace.

As you can see, Dreamweaver provides color-coded tags and markup to make it easier to read, but that’s not all. It also offers code hinting for ten different web development languages, including but not limited to HTML, CSS, JavaScript, and PHP.

5 Choose File > Save.

6 Name the file myfirstpage.html and save it in the lesson04 folder.

Note: In all screen shots, we use the Classic color theme, which can be selected in Preferences. See the “Getting Started” section at the beginning of the book for more details.
7 Insert the cursor after the opening <body> tag. Press Enter/Return to create a new line. Type <

A code-hinting window appears, showing you a list of HTML-compatible codes you can select from.

8 Type d

The code-hinting window filters to code elements that start with the letter d. You can continue to type the tag name directly or use this list to select the desired element. By using the list, you can eliminate simple typing errors.

9 Press the Down Arrow key.

The dd tag in the code-hinting window is highlighted.

10 Continue pressing the Down Arrow key until the tag div is highlighted. Press Enter/Return.

The tag name div is inserted in the code. The cursor remains at the end of the tag name, waiting for your next input. For example, you could complete the tag name or enter various HTML attributes. Let’s add an id attribute to the div element.

Note: Depending on the settings in your program, tags may close automatically, and you may have to move the cursor to complete the next step. This behavior can be turned off or adjusted in the Code Hints section of Preferences.

11 Press the spacebar to insert a space.

The hinting menu opens again, displaying a different list; this time the list contains various appropriate HTML attributes.
12 Type **id** and press Enter/Return.

Dreamweaver creates the *id* attribute, complete with equals sign and quotation marks. Note that the cursor appears within the quotation marks, ready for your entry.

13 Type **wrapper** and press the Right Arrow key once. The cursor moves outside the closing quotation mark.

14 Type **>**

When you type the >, Dreamweaver closes the *div* element automatically. As you see, the program can provide a lot of help as you write code manually. But it can help you write code automatically too.

15 Choose File > Save.

**Writing code automatically**

*Emmet* is a web-developer toolkit that was added to Dreamweaver a while ago and enables you to supercharge your code-writing tasks. When you enter short-hand characters and operators, Emmet enables you to create whole blocks of code with just a few keystrokes. In the following exercise you will experience the power of Emmet.

1 If necessary, open **myfirstpage.html**.

2 In the Code view window, insert the cursor within the *div* element and press Enter/Return to create a new line.

Note: In HTML5, quotation marks are no longer required for tag attributes. However, older browsers and applications may not display the code properly without them. It does no harm to use them, so continue to add them to your code.
Emmet is enabled by default and works whenever you are typing in Code view. In most websites a navigation menu appears at the top of the page. HTML5 uses the `<nav>` element as the foundation of site navigation. You will insert the menu and learn how to populate it with menu items.

3 Type `nav` and press Tab.

Dreamweaver creates the opening and closing tags all at once. The cursor appears inside the `nav` element, ready for you to add another element, some content, or both.

HTML navigation menus are usually based on an unordered list, which consists of a `<ul>` element with one or more child `<li>` elements. Emmet allows you to create multiple elements at the same time, and by using one or more operators, you can specify whether the subsequent elements follow the first (`+`) or are nested one within the other (`>`).

4 Type `ul>li` and press Tab.

A `<ul>` element containing one list item appears. The greater-than symbol (`>`) is used to create the parent–child structure you see here. By adding another operator, you can create several list items.

5 Choose Edit > Undo.

The code reverts to the `ul>li` shorthand. It’s easy to adapt this shorthand markup to create a menu with five items.

6 Edit the existing shorthand phrase as highlighted `ul>li*5` and press Tab.
A new unordered list appears, this time with five \(<li>\) elements. The asterisk (*), is the mathematical symbol for multiplication, so this latest change says “\(<li>\) times 5.”

To create a proper menu, you also need to add a hyperlink to each menu item.

7 Press Ctrl+Z/Cmd+Z or choose Edit > Undo.

The code reverts to the ul>li*5 shorthand.

8 Edit the existing shorthand phrase as highlighted: ul>li*5>a

If you guessed that adding the markup \(>a\) would create a hyperlink child element for each link item, you are correct. Emmet can also create placeholder content. Let’s use it to insert some text in each link item.

9 Edit the shorthand phrase as highlighted: ul>li*5>a{Link}

Adding text within braces passes it to the final structure of the hyperlink, but we’re not done yet. You can also increment the items, such as Link 1, Link 2, Link 3, and so on, by adding a variable character ($).

10 Edit the shorthand phrase as highlighted ul>li*5>a{Link $} and press Tab.

**Note:** The cursor must be outside the brace before you press Tab.

The new menu appears fully structured, with five link items and hyperlink placeholders incremented 1 through 5. The menu is nearly complete. The only things missing are targets for the href attributes. You could add them now using another Emmet phrase, but let’s save that change for the next exercise.

11 Insert the cursor after the closing \(</nav>\) tag.

Press Enter/Return to create a new line.

Let’s see how easy it is to use Emmet to add a header element to your new page.

12 Type header and press Tab.

As with the \(<nav>\) element you created earlier, the opening and closing header tags appear, with the cursor positioned to insert the content. We will model the header after one you will use in Lesson 6, “Creating a Page Layout.” You need to
add two text components: an `<h2>` for the company name and a `<p>` element for the motto. Emmet provides a method for adding not only the tags but also the content.

13 Type `h2{Favorite City Tour}+p{Travel with a purpose}` and press Tab.

The two elements appear complete and contain the company name and motto. Note how you added the text to each item using braces. The plus (+) sign designates that the `<p>` element should be added as a peer to the heading.

14 Insert the cursor after the closing `</header>` tag.
15 Press Enter/Return to insert a new line.

As you can see, Emmet enables you to quickly build complex multifaceted parent–child structures like the navigation menu and the header, but it doesn’t stop there. As you string together several elements with placeholder text, you can even add `id` and `class` attributes. To insert an id, start the name with the hash symbol (#); to add a class, start the name with a dot (.)

16 Type `main#content>aside.sidebar1>p(lorem)^article>p(lorem100)^aside.sidebar2>p(lorem)` and press Tab.

Note: The entire phrase may wrap to more than one line in Code view, but make sure there are no spaces or line breaks within the markup.
A `<main>` element is created with three child elements (aside, article, aside), along with id and class attributes. The caret (^) symbol in the shorthand is used to ensure that the article and aside.sidebar2 elements are created as siblings of aside.sidebar1. Within each child element, you should see a paragraph of placeholder text.

Emmet includes a *Lorem* generator to create blocks of placeholder text automatically. When you add lorem in parentheses after an element name, such as p(lorem), Emmet will generate 30 words of placeholder content. To specify a larger or smaller amount of text, just add a number at the end, such as p(lorem100) for 100 words.

Let’s finish up the page with a footer element containing a copyright statement.

17 Insert the cursor after the closing `<main>` tag. Create a new line. Type `footer{Copyright 2022 Favorite City Tour. All rights reserved.}` and press Tab.

18 Save the file.

Using a few shorthand phrases, you have built a complete webpage structure and some placeholder content. You can see how Emmet can supercharge your code-writing tasks. Feel free to use this amazing toolkit at any time to add a single element or a complex, multifaceted component. It’s there anytime you need it.

This exercise has barely scratched the surface of what Emmet can do. It is simply too powerful to fully describe in just a few pages. But you got a good peek at its capabilities.

Check out https://emmet.io to learn more about Emmet. Check out https://docs.emmet.io/cheat-sheet/ for a handy Emmet shorthand cheat sheet.

**Working with multicursor support**

Have you ever wanted to edit more than one line of code at a time? Dreamweaver offers multicursor support. This feature allows you to select and edit multiple lines of code at once to speed up a variety of mundane tasks. Let’s take a look at how it works.
1 If necessary, open myfirstpage.html as it appears at the end of the previous exercise.

The file contains a complete webpage with nav, header, main, and footer elements. The content features classes and several paragraphs of placeholder text. The <nav> element includes five placeholders for a navigation menu, but the href attributes are empty. For the menu and links to appear and behave properly, you need to add a filename, URL, or placeholder element to each link. In HTML, the hash mark (#) is used as placeholder content until the final link destinations can be added.

2 Insert the cursor between the quotation marks in the href="" attribute in Link 1.

Normally, you would have to add a hash mark (#) to each attribute individually. Multicursor support makes this task much easier, but don’t be surprised if it takes you a little practice. Note that all the link attributes are aligned vertically on consecutive lines.

3 Hold the Alt key (Windows) or Option key (macOS) and drag the mouse down through all five links.

Using the Alt/Option key enables you to select code or insert cursors in consecutive lines. Be careful to drag down in a straight line. If you slip a little to the left or right, you may select some of the surrounding markup. If that happens, you can just start over. When you are finished, you should see a cursor flashing in the href attribute for each link.

Customizing the Common toolbar

Some of the code-editing exercises in this lesson require tools that may not appear in the interface by default. The Common toolbar was previously called the Coding toolbar and appeared only in Code view. The toolbar appears in all views, but some tools may be visible only when the cursor is inserted directly in the Code view window.

If the exercise calls for a tool that is not visible, even with the cursor in the proper position, you may need to customize the toolbar yourself. This can be done by first clicking the Customize Toolbar icon and then enabling the tools within the Customize Toolbar dialog. At the same time, feel free to disable tools you don’t use.
4 Type #
The hash mark (#) appears in all five attributes at the same time.

The Ctrl/Cmd key enables you to select code or insert cursors in nonconsecutive lines of code.

5 Hold the Ctrl/Cmd key and click to insert the cursor between the p and the > bracket in each of the three opening <p> tags in the <main> element.

6 Press the spacebar to insert a space, and type class="first"

The attribute appears simultaneously in all three <p> tags.

7 Save the file.
Multicursor support can save tons of time in repetitive code-editing tasks.

Commenting your code

Comments allow you to leave notes within the code—invisible in the browser—to describe the purpose of certain markup or provide important information to other coders. Although you can add markup or provide important information to other coders. Although you can add comments manually at any time, Dreamweaver has a built-in feature that can speed up the process.

1 Open myfirstpage.html using the Developer workspace.

2 Insert the cursor after the opening tag <aside class="sidebar1">

3 Click the Apply Comment icon 📝.
A pop-up menu appears with several comment options. Dreamweaver supports comment markup for various web-compatible languages, including HTML, CSS, JavaScript, and PHP.

4 Choose Apply HTML Comment.

An HTML comment block appears, with the text cursor positioned in the center.

5 Type **Insert customer testimonials into Sidebar 1**

The comment appears in gray between the <!-- and --> markup. The tool can also apply comment markup to existing text.

6 Insert the cursor after the opening tag `<aside class="sidebar2">`.

7 Type **Sidebar 2 should be used for content related to the tour or product**

8 Select the text created in step 7.
Click the Apply Comment icon ![Apply Comment](image)
A pop-up menu opens.

9 Select Apply HTML Comment.

Dreamweaver applies the <!-- and --> markup to the selection. If you need to remove existing comment markup from a selection, click the Remove Comment icon ![Remove Comment](image) in the toolbar.

10 Save all files.
You’ve created a basic webpage, complete with placeholder text. The next step is to style the page. Dreamweaver supports CSS preprocessors. In the next exercise, you’ll learn how to set up and create CSS styling using a preprocessor.

**Working with CSS preprocessors**

One of the biggest additions to Dreamweaver over the last few years was support for industry-standard CSS preprocessors. Known by the acronyms LESS (Leaner CSS), Sass (Syntactically Awesome Style Sheets), and SCSS (Sassy CSS), these are scripting languages that enable you to extend the capabilities of cascading style sheets with a variety of productivity enhancements that can then be compiled in a standard CSS file. These languages provide a variety of benefits for designers and developers who prefer to write their code by hand, including speed, ease of use, reusable snippets, variables, logic, calculations, and much more. No other software is needed to work in these preprocessors, but Dreamweaver also supports other frameworks, such as Compass and Bourbon.

In this exercise, you’ll get a taste of how easy it is to use preprocessors with Dreamweaver as well as what advantages they offer compared to a regular CSS workflow.

**Enabling a preprocessor**

Support for CSS preprocessors is site-specific and must be enabled for each site defined in Dreamweaver, as desired. To enable LESS, Sass, or SCSS, you first define a site and then enable the CSS Preprocessors option within the Site Definition dialog.

1. Select Site > Manage Sites.
   
   The Manage Sites dialog appears.

2. Select lesson04 in the Manage Sites window.
   
   Click the Edit icon at the bottom of the Manage Sites window.

![Manage Sites dialog](image)
The Site Definition dialog for lesson04 appears.

3 Select the **CSS Preprocessors** option in the Site Definition dialog.

The CSS Preprocessors option contains six subcategories, including General, Source & Output, and options for various Compass and Bourbon frameworks. You can check out the Dreamweaver Help topics for more information on these frameworks. For this exercise, you need only the features that are built into the program itself.

4 Select the General category.

When selected, this category features the on/off switch for the LESS, Sass, or SCSS compiler, as well as various options for how the languages operate. For our purposes, the default settings will work fine.

5 Select the Enable Auto Compilation On File Save checkbox to enable the preprocessor compiler, if necessary.

When this is enabled, Dreamweaver will automatically compile your CSS from your LESS, Sass, or SCSS source files whenever they are saved. Some designers and developers use the root folder of the site for compilation. In this case, we'll separate the source and output files in distinct folders.

6 Select the Source & Output category.

This category enables you to designate the source and output folders for your CSS preprocessor. The default option targets the folder where the source file is saved.

7 Select the Define Output Folder option.
When you enable this option, Dreamweaver displays a file path pointing to a folder (css). This folder doesn't exist yet, but Dreamweaver will create it automatically. If you want to use a different folder, you will have to use the Browse For Folder icon to select or create the folder.

8 Click the Browse For Folder icon beside the Source Folder field.

9 Navigate to the Site Root folder.

10 Select the existing Sass folder, and click Select Folder/Choose.

11 Save the changes and click Done to return to your site.

The CSS preprocessor is enabled, and the source and output folders are now designated. Next, you'll create the CSS source file.

**LESS or Sass—the choice is yours**

LESS and Sass offer similar features and functions, so which one should you choose? That's hard to say. Some think that LESS is easier to learn but that Sass offers more powerful functionality. Both make the chore of writing CSS by hand faster and easier and, more importantly, provide significant advantages for maintaining and extending your CSS over time. There are lots of opinions on which preprocessor is better, but you'll find that it comes down to personal preference.

Before you decide, check out the following links to get some informed perspectives:

- blog.udemy.com/less-vs-sass/
- css-tricks.com/sass-vs-less/
- keycdn.com/blog/sass-vs-less

Dreamweaver provides two syntaxes for Sass. In this lesson, we use SCSS (Sassy CSS), which is a form of Sass that is written like and looks more like regular CSS.

**Creating the CSS source file**

When using a preprocessor workflow, you do not write the CSS code directly. Instead, you write rules and other code in a source file that is then compiled to the output file. For the following exercise, you'll create a Sass source file and learn some of the functions of that language.
1. Select Standard from the Workspace menu.

2. Choose Window > Files to display the Files panel, if necessary. Select lesson04 from the Site List dropdown menu, if necessary.

3. If necessary, open myfirstpage.html and switch to Split view. The webpage is unstyled at the moment.


The New Document dialog appears. This dialog allows you to create all types of web-compatible documents. In the Document Type section of the dialog, you will see the LESS, Sass, and SCSS file types. We’ll use SCSS in the following exercises. SCSS is a flavor of Sass that uses a syntax that is similar to regular CSS and that many users find easier to learn and work with.

5. Choose New Document > SCSS.

Click the Create button.

A new blank SCSS document appears in the document window.

6. Save the file as favorite-styles.scss in the Sass folder you targeted as the Source folder in the previous exercise.
You don’t need to create the CSS file; the compiler in Dreamweaver will do that for you. You’re all set to start working with Sass. The first step is to define variables. Variables are programmatic constructs that enable you to store CSS specifications you want to use multiple times, such as colors in your site theme. By using a variable, you have to define it only once. If you need to change it in the future, you can edit one entry in the style sheet and all the instances of the variable will update automatically.

7 Insert the cursor into line 2 of `favorite-styles.scss`.
Type `$logoyellow: #ED6;` and press Enter/Return.
You’ve created your first variable. This is the main yellow color of the site theme. Let’s create the rest of the variables.

8 Type
```
$darkyellow: #ED0;
$lightyellow: #FF3;
$logoblue: #069;
$darkblue: #089;
/lightblue: #08A;
$font-stack: "Trebuchet MS", Verdana, Arial, Helvetica, sans-serif;
```
and press Enter/Return to create a new line.

Entering the variables on separate lines makes them easier to read and edit but does not affect how they perform. Just make sure you add a semicolon (`;`) at the end of each variable.

Let’s start the style sheet with the base or default styling of the body element. SCSS markup in most cases looks just like regular CSS, except in this case you’ll use one of your variables to set the font family.

9 Type `body` and press the spacebar.
Type `{` and press Enter/Return.
When you typed the opening brace ({), Dreamweaver created the closing brace automatically. When you created the new line, the cursor was indented by default, and pressing Enter/Return moved the closing brace to the following line. You can also use Emmet to enter the settings more quickly.

10 Type `ff$font-stack` and press Tab.

![CSS code with Emmet shorthand](image)

The shorthand expands to `font-family: $font-stack;`.

11 Press Enter/Return to create a new line within the body rule. Type `c` and press Tab.

![CSS code with color shorthand](image)

The shorthand expands to `color: #000;`. The default color is acceptable.

12 Hold the Alt/Cmd key and press the Right Arrow key to move the cursor to the end of the current line of code.

13 Press Enter/Return to create a new line. Type `m0` and press Tab.

![CSS code with margin shorthand](image)

The shorthand expands to `margin: 0;`. completing the basic styling for the body element. Before you save the file, this is a good time to see how preprocessors do their work.
Compiling CSS code

You have completed the specifications for the body element. But you have not created the styling directly in a CSS file. Your entries were made entirely in the SCSS source file. In this exercise, you will see how the compiler that is built into Dreamweaver generates the CSS output.

1. Display the Files panel, if necessary, and expand the list of site files.

The site consists of one HTML file and three folders: Sass, images, and css.

2. Expand the view of the css and Sass folders.

The Sass folder contains favorite-styles.scss and _base.scss. The css folder contains favorite-styles.css. This file did not exist when you started the lesson. It was generated automatically when you created the SCSS file and saved it into the site folder defined as the Source folder. At the moment, the CSS file should contain no CSS rules or markup. It’s also not referenced in the sample webpage.

Note: The favorite-styles.css file should have been created automatically in the previous exercise when the SCSS file was saved. If you do not see the .css file, you may need to shut down and relaunch Dreamweaver.
3 Select the document tab for *myfirstpage.html*. Choose View > Split > Code-Live. Choose View > Split > Horizontally, if necessary.

The document window is split into two windows top to bottom, showing the rendered webpage in one and the code in the other. The page shows only default HTML styling at this point.

4 In the Code view window, insert the cursor after the opening `<head>` tag and press Enter/Return to insert a new line.

5 Type `link` and press Tab.

The shorthand expands to a `<link>` reference for a style sheet. It comes in with two attributes, `rel` and `href`. You’ll use the `href` attribute to link the webpage to the generated CSS file.

6 Insert the cursor between the quotation marks in the `href` attributes.

7 Type `/css/`

As you type, Dreamweaver displays a hinting menu for the file structure of the site. Once you type the second backslash, you should see the CSS file created automatically by the preprocessor.

8 Press the Down Arrow key to highlight the filename *favorite-styles.css*. 
9 Press Enter/Return.

The URL to the CSS output file appears in the attribute. The link to the style sheet is now complete.

The CSS output file is now referenced by the webpage. In the Live view window, there should be no difference in the styling, but you should now see `favorite-styles.css` displayed in the Related Files interface.

**Note:** If you accidentally saved the SCSS file before this step, you may see styling in the HTML file and another filename in the Related Files interface.

10 Select `favorite-styles.css` in the Related Files interface.

Code view displays the contents of `favorite-styles.css`, which is empty at the moment. An asterisk appears next to the filename in the document tab for `favorite-styles.scss`, indicating that the file has been changed but not saved.

11 Choose Window > Arrange > Tile.

The webpage and the SCSS source file appear side by side in the program window.
12 Insert the cursor anywhere in the **favorite-styles.scss** document window and choose File > Save All.

**Note:** It must be said that the changes in the SCSS file have not been saved until this point. If the program crashes before you save, you will have to start from scratch.

After a moment, the display of **myfirstpage.html** changes, showing the new font and margin settings. The Code view window also updates to display the new contents of **favorite-styles.css**. Each time you save the SCSS source file, Dreamweaver will update the output file.

### Nesting CSS selectors

Targeting CSS styling to one element without accidentally affecting another is a constant challenge for web designers everywhere. Descendant selectors are one method for ensuring that the styling is applied correctly. But creating and maintaining the correct descendant structure becomes more difficult as the site and style sheets grow in size. All preprocessor languages offer some form of nesting for selector names.

In this exercise, you will learn how to nest selectors while styling the navigation menu. First, you’ll set the basic styling for the `<nav>` element itself.

1. In the **favorite-styles.scss** window, insert the cursor after the closing brace (}) on line 13 for the body rule.

2. Create a new line; type **nav {** and press Enter/Return.

   The nav selector and declaration structure are created and ready for your entry. Emmet provides shorthand entries for all CSS properties.

3. Type **bg$logoyellow** and press Tab. Press Enter/Return.

   The shorthand expands to background: $logoyellow, which is the first variable you created in the SCSS source file. This will apply the color #ED6 to the nav element.

4. Type **ta:c** and press Tab. Press Enter/Return.

   The shorthand expands to text-align: center.

5. Type **ov:a** and press Tab. Press Enter/Return.

   The shorthand expands to overflow: auto.
6  Save the source file.

The `<nav>` element in `myfirstpage.html` displays the color #ED6. The menu doesn't look like much yet, but you've only just begun. Next, you'll format the `<ul>` element. Note that the cursor is still within the declaration structure for the `nav` selector.

7  Type `ul {` and press Enter/Return.

The new selector and declaration are created within the `nav` rule.

8  Type `list-style: none; margin: 5px;` and press Tab.

The shorthand expands to show the properties `list-style: none` and `margin: 5px`. These properties reset the default styling of the unordered list, removing the bullets and indent. Next, you'll override the styling of the list items.

9  Press Enter/Return and type `li {`

As before, the new selector and declaration are fully within the `ul` rule.

10 Create the property `display: inline-block` and press Tab.

Press Enter/Return.

The property `display: inline-block` will display all the links in a single row, side by side. The last element to style is the `<a>` for the link itself.

11 Type `a {` and press Enter/Return.

Type `margin: 0;` and press Tab.

The shorthand expands to show the properties `margin`, `padding`, `color`, `text-decoration`, and `background` styling the rule entirely within the `li` rule. Each of the rules styling the navigation menu has been nested one inside the other in a logical, intuitive manner and will result in an equally logical and intuitive CSS output.
Save the file.

The navigation menu in myfirstpage.html is reformatted to display a single line of links, side by side. The CSS output file displays several new CSS rules. The new rules are not nested as in the source file. They are separate and distinct. More surprisingly, the selectors have been rewritten to target the descendant structures of the menu, such as nav ul li a. As you can see, nesting rules in the SCSS source file eliminates the chore of writing complex selectors.

Importing other style sheets

To make CSS styling more manageable, many designers split their style sheets into multiple separate files, such as one for navigation components, another for feature articles, and still another for dynamic elements. Large companies may create an overall corporate standard style sheet and then allow various departments or subsidiaries to write custom style sheets for their own subdomains, products, and purposes. Eventually, all these CSS files need to be brought together and called by the webpages on the site, but this can create a big problem.

Every resource linked to a page creates an HTTP request that can bog down the loading of your pages and assets. This is not a big deal for small sites or lightly traveled ones. But popular, heavily traveled sites with tons of HTTP requests can overload a web server and even cause pages to freeze in a visitor’s browser. Too many experiences like this can cause visitors to flee and never return.

Reducing or eliminating superfluous HTTP calls should be the goal of any designer or developer, but especially those working on large enterprise or highly popular sites. One important technique is to cut down on the number of individual style sheets called by each page. If a page needs to link to more than one CSS file, it’s usually recommended that you designate one file as the main style sheet and then simply import the other files into it, creating one large universal style sheet.
In a normal CSS file, importing multiple style sheets would not produce any benefit, because the import command creates the same type of HTTP request that you’re trying to avoid in the first place. But since you are using a CSS preprocessor, the import command happens before any HTTP request occurs. The various style sheets are imported and combined. Although this makes the resulting style sheet larger, this file is downloaded only once by the visitor’s computer and then cached for their entire visit, speeding up the process overall.

Let’s see how easy it is to combine multiple style sheets into one file.

1. Open myfirstpage.html and switch to Split view, if necessary. Open favorite-styles.scss and choose Window > Arrange > Tile.

   The two files are displayed side by side to make it easier to edit the CSS and see the changes as they occur.

2. In myfirstpage.html, click favorite-styles.css in the Related Files interface.

   Code view displays the content of favorite-styles.css. It contains the output of rules written in the SCSS source file.

3. In favorite-styles.scss, insert the cursor before the body rule (around line 9). Type @import "_base.scss"; and press Enter/Return to insert a new line.

   This command imports the contents of the file _base.scss stored in the Sass folder. The file was created ahead of time to style other portions of your page. At the moment, nothing has changed, because favorite-styles.scss has not been saved yet.

4. Save favorite-styles.scss and observe the changes in myfirstpage.html.

If you correctly followed the instructions on how to create the HTML structure earlier in this lesson, the page should be entirely formatted now. When you examine favorite-styles.css, you will see that dozens of rules were inserted before the body rule. Imported content will be added starting at line 2. This
may be confusing at first since the SCSS file had seven lines of code before the body rule. Although the @import command was at line 9 in the SCSS file, the variables are not passed to the final CSS directly. They are parsed and rendered into each of the rules they affect. Once the content has been imported, normal CSS precedence and specificity take effect. Just make sure that all rules and file references appear after the variables; otherwise, the variables won’t work.

5  Save and close all files.

In this section, you created an SCSS file and learned how to work with a CSS preprocessor. You experienced various productivity enhancements and advanced functionality and have glimpsed just a bit of the breadth and scope of what is possible.

**Learning more about preprocessors**

Check out the following books to learn more about CSS preprocessors and supercharging your CSS workflow:


*Jump Start Sass: Get Up to Speed with Sass in a Weekend*, by Hugo Giraudel and Miriam Suzanne, SitePoint (2016); ISBN: 978-0994182678

**Using linting support**

Dreamweaver provides live code error checking. Linting support is enabled by default in Preferences, which means the program monitors your code writing and flags errors in real time.

1  Open *myfirstpage.html* and switch to Code view. If necessary, select Source Code in the Related Files interface.

2  Insert the cursor after the opening `<article>` tag and press Enter/Return to create a new line.

3  Type `<h1>Insert headline here</h1>`
   
   Note: Dreamweaver will create the opening and closing tags at once. If not, skip to step 5.

4  Delete the closing `</h1>` tag.

5  Save the file.
   
   You failed to close the `<h1>` element in step 3. When an error occurs, a red X will appear at the bottom of the document window whenever you save the page.
6 Click the X icon.

The Output panel opens automatically and displays the coding errors. In this case, the message says that the tag must be paired and identifies what line it thinks the error occurs on. The message erroneously targets line 27, but this can happen because of the nature of HTML tags and structures.

7 Double-click the error message.

Dreamweaver focuses on the article in the Code view window that it identifies as containing the error. Since Dreamweaver is looking for the closing tag for the `<h1>` element and flags it. Unfortunately, the error message is not always this accurate. Dreamweaver’s linting function will get you close to the error, but often you will have to look earlier in the code to track down the actual issue yourself.

8 Insert the cursor at the end of the code

```html
<h1>Insert headline here</h1>
```

Note: If your heading closed automatically in step 3, typing `</h1>` will probably not close the tag. Check your preference settings for code rewriting and adjust them as desired.

Dreamweaver should now be closed. If not, go ahead and finish it properly.
9 Save the file.

Once the error is corrected, the red X is replaced by a green checkmark.

10 Right-click the Output panel tab and select Close Tab Group from the context menu.

It’s important to be alert for this icon as you save your work. No other error message will pop up indicating any problems, and you’ll want to catch and correct any errors before uploading your pages to the web server.

Selecting code

Dreamweaver provides several methods for interacting with and selecting code in Code view.

Using line numbers

You can use your cursor to interact with the code in several ways.

1 Open myfirstpage.html, if necessary, and switch to Code view.

2 Scroll down and locate the <nav> element (around line 11).

3 Drag the cursor across the entire element, including the menu items.

Using the cursor in this way, you can select any portion of the code or its entirety. However, using the cursor in this way can be prone to error, causing you to miss vital portions of the code. At times, using line numbers to select whole lines of code is easier.

4 Click the line number beside the <nav> tag.

The entire line is selected within the window.
5  Drag down the line numbers to select the entire <nav> element.

Dreamweaver completely highlights all seven lines. Using line numbers can save a lot of time and avoid errors during selection, but it doesn't take into account the actual structure of the code elements, which may begin and end in the middle of a line. Tag selectors provide a better way to select logical code structures.

**Using tag selectors**

One of the easiest and most efficient ways to select code is to use the tag selectors, as you will frequently do in upcoming lessons.

1  Scroll down and locate the following code:
   `&lt;a href="#"&gt;Link 1&lt;/a&gt;`

2  Insert the cursor anywhere in the text **Link 1**.
   Examine the tag selectors at the bottom of the document window.

   The tag selectors in Code view display the `<a>` tag and all its parent elements, the same way they do in Live or Design view.
3 Select the `<a>` tag selector.

The entire `<a>` element, including its content, is highlighted in Code view. It can now be copied, cut, moved, or collapsed. The tag selectors clearly reveal the structure of the code, even without referring to the Code view display. The `<a>` is a child of the `<li>` element, which is a child of `<ul>`, which is in turn a child of `<nav>`, which is a child of `<div#wrapper>`, and so on.

The tag selectors make it a simple chore to select any part of the code structure.

4 Select the `<ul>` tag selector.

The code for the unordered list is entirely selected.
5  Select the `<nav>` tag selector.

   The code for the entire menu is selected.

6  Select the `<div#wrapper>` tag selector.

   The code for the entire page is now selected. Using the tag selectors allows you to identify and select the structure of any element on your page, but it requires you to identify and select the parent tag yourself. Dreamweaver offers another tool that can do it for you automatically.

**Using parent tag selectors**

Using the parent tag selector in the Code view window makes the job of selecting the hierarchical structure of your page even simpler.

1  Choose Window > Toolbars > Common to display the Common toolbar, if necessary.

2  Insert the cursor anywhere in the text **Link 1**.

   The Select Parent Tag option may not be displayed by default in the Common toolbar. If you see the tool icon, skip to step 4, otherwise you will need to enable this tool before proceeding.

3  Click the Customize Toolbar icon and enable the Select Parent Tag tool before proceeding to step 4.
4 In the Common toolbar, click the Select Parent Tag icon. 

The entire `<a>` element is highlighted.

5 Click the Select Parent Tag icon again or press Ctrl+] / Cmd+] (left bracket).

The entire `<li>` element is selected.

6 Click the Select Parent Tag icon.

The entire `<ul>` element is selected.

7 Press Ctrl+] / Cmd+] until `<div id="wrapper">` is selected.

Each time you click the icon or press the shortcut key, Dreamweaver selects the parent element of the current selection. Once you’ve selected it, you may find working with long sections of code unwieldy. Code view offers other handy options to collapse long sections to make them easier to work with.

**Collapsing code**

Collapsing code is a productivity practice that makes it simple to copy or move large sections of code. Coders and developers also collapse code sections when they are looking for a particular element or section of a page and want to temporarily hide unneeded sections from view. Code can be collapsed either by selection or by logical element.
1 Select the first three Link items in the `<nav>` element.

Note the Collapse icon ▼ along the left edge of Code view; it indicates that the selection is currently expanded.

2 Click the Collapse icon to collapse the selection.

The selection collapses, showing only the first `<li>` element and a snippet of text from it.

You can also collapse code based on logical elements, like `<ul>` or `<nav>`. Notice that each line that contains an opening element tag also displays a Collapse icon.

3 Click the Collapse icon beside the line for the `<nav>` element.

The entire `<nav>` element collapses in the Code window, showing only an abbreviated snippet of the entire element. In either instance, the code hasn’t been deleted or damaged in any way. It still functions and operates as expected. Also, the collapse functionality appears only in Code view in Dreamweaver; on the web or in another application, the code will appear normally. To expand the code, just reverse the process, as described in the following section.

### Expanding code

When the code is collapsed, you can still copy, cut, or move it like you would any other selected element. You can then expand elements one at a time or all at once.

1 Click the Expand icon ➤ beside the line for the `<nav>` element.

The `<nav>` element expands, but the three `<li>` elements collapsed in the previous exercise are still collapsed.

2 Click the Expand icon beside the line for the `<li>` elements.

All collapsed elements are now expanded. Note that the Expand icon for the three `<li>` elements disappears altogether.
Accessing Split Code view

Why should coders be denied the ability to work in two windows at the same time? Split Code view enables you to work in two different documents or two different sections of the same document at once. Take your pick.

1. If necessary, switch to Code view.

The document displays two Code view windows, both focusing on myfirstpage.html.

3. Insert the cursor in the top window and scroll down to the `<footer>` element.
   
   Split Code view enables you to view and edit two different sections of the same file.

4. Insert the cursor in the bottom window and scroll to the `<header>` element.
You can also view and edit the contents of any related file.

5 In the Related Files interface, select **favorite-styles.css**.

The window loads the style sheet into one of the windows. You can work in either window and save your changes in real time. Dreamweaver displays an asterisk (*) on any filename in the interface that has been changed but not saved. If you select File > Save or press Ctrl+S/Cmd+S, Dreamweaver saves the changes in the document where your cursor is inserted. Since Dreamweaver can make changes to documents even when they are not open, this feature allows you to edit and update even the files that are closed but linked to your webpage.

### Previewing assets in Code view

Although you may be a diehard coder or developer, there’s no reason you can’t feel the love from Dreamweaver’s graphical display too. The program provides visual previews of graphic assets and certain CSS properties in Code view.

1 Open **myfirstpage.html**. Select Code view.

   In Code view, you see only the HTML. The graphical assets are simply references that appear in the CSS file **favorite-styles.css**.

2 Click **favorite-styles.css** in the Related Files interface.

   The style sheet appears in the window. Although it’s fully editable, don’t waste your time making any changes to it. Since the file is the output of the SCSS source file, any changes you make will be overwritten the next time the file compiles.

3 Locate the **header** rule (around line 5).

   The **header** consists of two text elements and two images. You should be able to see the image references in the **background** property.

4 Position the cursor over the markup `url(../images/favcity-logo.jpg)` in the **background** property (line 8).

   ![A miniature preview of the company logo appears below the cursor.](image)
Position the cursor over the markup background-color: #ED6; in the background property.

A small color chip appears, displaying the color specified. The preview functions the same way for all color models. You no longer have to guess what image or color you specified before you can see it in Live view or the browser.

In this lesson, you learned a number of techniques to make working with code easier and more efficient. You learned how to write code manually using hinting and auto-code completion and how to write code automatically using Emmet shorthand. You learned how to check code construction using built-in linting support. You learned how to select, collapse, and expand code, as well as how to create HTML comments and view code in different ways.

Overall, you learned that whether you are a visual designer or a hands-on coder, you can rely on Dreamweaver to offer vital features and power that will allow you to create and edit HTML and CSS code without compromises. Remember these techniques as you work through the book and use any of them whenever appropriate.
Review questions

1. In what ways does Dreamweaver assist you in creating new code?
2. What is Emmet, and what functionality does it provide to users?
3. Does Dreamweaver require you to install any additional software to create a LESS, Sass, or SCSS workflow?
4. What feature in Dreamweaver reports code errors when you save a file?
5. True or false? Collapsed code will not appear in Live view or the browser until it is expanded.
6. What Dreamweaver feature provides instant access to files linked or referenced within the document?

Review answers

1. Dreamweaver provides code hinting and auto-completion for HTML tags, attributes, and CSS styling as you type, along with support for JavaScript and PHP, among other languages.
2. Emmet is a scripting toolkit that creates HTML code by converting shorthand entries into complete elements, placeholders, and even content.
3. No additional software or services are needed to use LESS, Sass, or SCSS. Dreamweaver supports these CSS preprocessors out of the box. You merely have to enable the compiler in the Site Definition dialog.
4. Linting checks the HTML code and structure every time you save a file and then displays a red X icon at the bottom of the document window when an error is detected.
5. False. Collapsing code has no effect on the display or operation of the code outside of Dreamweaver.
6. The Related Files interface appears at the top of the document window and enables users to instantly access and review CSS, JavaScript, and other compatible file types linked to the webpage. In some cases, a file displayed in the interface will be stored on a remote resource on the internet. While the Related Files interface enables you to view the contents of all the files displayed, you will be able to edit only ones that are stored on your local hard drive.
INDEX

A
a element, 53, 165, 169, 354
Absolute hyperlinks, 342–343, 360–364
Access model, 127–128
Adobe Analytics, 127
Adobe Dreamweaver Classroom in a Book
   bonus material, 8
   conventions used, 2–3
   lessons, 1, 6–7
   online resources, 1, 6–7, 15
   prerequisites to, 2
   training centers, 15
Adobe Fonts, 431
Adobe Generator, 136
Adobe Illustrator, 132
Alignment, element, 460–463
All and Current modes, 41–42, 77
Alpha transparency, 299
Alt text, 305
Analytics, 127, 301
Apply HTML Comment, 98
Assets panel, 317–318
Attributes
   border, 272
   class and id, 43
   deleting template, 156–159
   HTML table, 263
   HTML5 class, 81, 82
   HTML5 tag, 91
Audience identification, 126

B
Background
   formatting for screen size, 450
   gradient, 169–173, 447
   images, 179–183
   layers in images, 313, 314
   properties, 169–170
   repeating, 180
Backups
   auto, 231
   uploading template for, 395
Balanced code, 51
Base fonts, 429–436
Berners-Lee, Tim, 50
Bit depth, 298
Block elements, 52
Body, 52
body element, 103–104, 184, 439
Boilerplate (uneditable content)
   adding content, 203–206
   on child pages, 212, 222
   defined, 190–191
   inserting editable regions into, 212
   saving changes to, 331
   text editor editing of, 222, 224
Bolded text, 2
Bootstrap, 20, 427
Bootstrap templates
   adding boilerplate content, 203–206
   adding elements to, 213–216, 259–261
   controls location, 259
   creating new structures, 175–179
   deleting unused components, 247–248
   duplicating rows, 244–246
   for mobile devices, 427, 460–468
   modifying layout in, 196–201
   modifying text formatting, 201–203
   six options for, 141
   style sheet, 148, 166, 196, 443
   styling of, 143–145, 306
   version changes, 165
   working with, 142–143
Borders, 263
Bots, 370
Bourbon, 99
Box model, 67–69
Break points, 449, 451, 462
Broadband access, 299–300
Browse for File function, 363
Browsers
   audience use of, 128
   graphics supported by, 296
   and HTML defaults, 66–67
   link-sharing interface, 363
   screen size for, 416–417
   status bar, 344
   support for HTML varieties, 51
   tables set for, 272
   testing pages in, 477
   validating code for, 210
   web-hosted fonts for, 428

C
Captions, 282–283
Card-based section
   as content model, 191
   deleting components in, 247
   as editable region, 212
Cascade theory, 70
Cascading style sheets (CSS).
   See CSS (cascading style sheets)
Character entities, 55
Child elements, 72, 157, 216
Child pages
   adding content to, 224–227
   code links to, 221
   creating new, 222–223
   deleting components on, 216, 232–233
   inserting images in, 336–338
   internal links on, 357–359
   metadata for, 209, 226–227
   optional region removal, 231
   streamlining to prep for, 191
   uneditable regions of, 212
Chrome, 481
Classes
  added to template structure, 328–331
  attributes of, 43, 81–82
  and element alignment, 460, 461
  pseudo-classes, 354, 379, 473
  syntax for, 260
  weight of, 74
Clipboard, 242
Cloaking, 393, 395–396
Closing panels, 27
Code. See also HTML
  ability to read, 52
  automatic writing of, 91–95
  browser displays of, 66–67
  collapsing, 118–119
  commenting your, 97–99
  compiling CSS, 105–108
  CSS preprocessors, 99–112
  elements of HTML, 51
  errors, 474
  expanding, 119
  font for, 2
  font for coding, 3
  manual writing of, 88–95
  multicursor support, 95–97
  online resources, 86
  readability of, 90, 153
  review questions on, 123
  selecting, 114–118
  textual distinctions for, 2
  use of Emmet for, 91
  validating, 210–211
  visual editing of, 87, 121
Code view
  accessing, 22
  designing in, 88
  image insertion in, 325
  line wraps, 94
  options available in, 88
  previewing assets in, 121–122
  Split Code view, 119–121
  Split view of, 24
  template workflow in, 217
  tool display in, 31
  user interface with, 18
Color
  changes in CSS, 62–64
  Dreamweaver settings, 62, 89
  hints for coding, 89
  of images, 298–300
  of selected elements, 193
  selection in CSS Designer, 79
  text changes in Bootstrap, 201–203
  theme, 9
Columns
  and element alignment, 461
  row/column screen division, 175, 196, 244
  size of table, 274–276
Commands
  accessing, 24
  custom shortcuts for, 31
  Windows vs. macOS, 3–4
  Commenting of code, 97–99
  Common toolbar, 30, 31, 96
  Company name
    in metadata, 209
    placement of, 131
  Compass, 99
  Compatibility
    with browsers, 51, 66–67
    class/id attributes and, 81
  Competition, learning from, 126
  Compiler, CSS, 105–108
  COMPUTED option, 41, 78
  Conflict troubleshooting, 151–153
  Conflicts, design, 469
  Connectivity
    broadband access, 299–300
    to remote sites, 388
  Contact list, 254
  Content and presentation, 68
  Copy
    alternate methods for, 225
    of new menu items, 164–165
    to replace code, 290–291
    of tables, 262, 267–268
  Corporate style sheets, 110
  Create New and Open, 21
  Create tools, 23
  Creative Cloud
    Design view, 22
    Dreamweaver updates via, 5
    Extract in, 143
  fonts in, 431
  page previews on, 479
  Photoshop in, 309
  signing up for, 5
  syncing preferences to, 8
  Typekit fonts in, 133
CSS (cascading style sheets)
  box model, 67–69
  class attributes, 81–82
  compiling code, 105–108
  conflict troubleshooting, 151–153
  CSS3, 82–84, 473, 477, 484
  defined, 62
  in Design view, 23
  and HTML defaults, 64–67
  media types and queries, 422–423
  multiple element formatting, 80
  nesting selectors, 108–110
  online resources, 60
  output files, 107
  preprocessors, 99–112
  Property inspector button for, 34
  review questions, 85
  sample layouts, 140–143
  shorthand, 80–81
  source file creation, 101–104
  style application, 69–74
  support for, 84
  table styling, 268–270
  and table use, 262
  vs. HTML, 62–64
CSS Designer
  All and Current modes, 41–42
  four windows of, 37–42
  Live view integration with, 23
  overview of, 74–79
  for table properties, 269
  Tag Selector interface and, 37
  Tag Selector panel, 37
  CSS Property inspector, 34
  Current mode
    editing rules in, 77
    function of, 41–42
    styling in, 147–148
  Cursor movement, 264
Customization
  of class and id, 81–82
  of the Common toolbar, 31, 96
  of keyboard shortcuts, 31–33
  media queries, 443–460
  for mobile vs. desktop, 134
  of the workspace, 29–30

D
Data, 263
Declarations and Selectors, 69
Defaults
  color settings, 62
  email programs, 366
  HTML, 64–67
  hyperlink, 364
  Site Definition dialog, 174
Degradation, graceful, 431
Deletion, component
  Bootstrap templates, 156–159, 247–248
  in DOM panel, 255
  Dreamweaver templates, 191–196
Demographic analysis, 127, 301
Dependent files, 404
Deprecated tags, 62
Descendent theory, 72–73, 108
Description, webpage, 209–210
Design. See also Mobile design;
  Styling; Web design
  audience as guiding, 126
  conflict troubleshooting, 469–477
  CSS as basis for, 61
  effect of browsers on, 66–67
  mockups for, 132–133
  options for page, 140
  predefined layouts for, 140–143
  template as basis for, 222
Design view
  function of, 22–23
  image insertion in, 325
  inserting images from, 306–308
  Style Rendering command, 68
  table preview in, 264
  template workflow in, 217
  text formatting in, 242
Designers
  HTML capabilities of, 52
  transition to HTML5, 58
  use of CSS, 61
  use of Dreamweaver, 18
Desktop computers
  designing for, 127–128, 134
  font troubleshooting, 482–484
  hiding links on, 356
  mobile-first design and, 416
  screen size for, 416
  testing pages on, 479
Destination links, 374–376
Developer workspace, 88
Disabled menu, 157–158
Displays, 43–46, 443
Dithering, 298
div element, 455
Docking panels, 29
Document Title field, 209
Document toolbar, 30
DOM (Document Object Model)
  panel
    adding elements via, 213
    creating new elements in,
    162–163
    for element deletion, 255
    for table selection, 273
    in the workspace, 42
Download times, 298
DPI (dots per inch), 297
Dragging tabs, 27
Dreamweaver, 8, 18, 416–417
Dreamweaver (2022 release)
  color theme, 9
  Design/Live view changes in, 23
  Git as supported by, 44–45
  as HTML editor, 50
  installing, 4–5
  interface, 18
  local/remote site setup, 11–14
  Photoshop compatibility, 311
  resources and training centers, 15
  updates to, 5–6, 14–15
  users of, 18
  Windows vs. macOS, 3–4
  workspace setup, 9–10
Dreamweaver templates
  adding editable regions to, 217
  child pages, 222–227
  creating, 190–191
  as design beginning, 222, 233
  as dynamic, 221
  editable regions, 212–221
  lesson on, 188
  locked areas of, 190–191
  removing components, 191–196
  review questions, 234–235
  2022 options, 141
  updating, 227–233
  when to use, 233
Drop zone, 28, 29
Dynamic templates, 221
E
Edge Web Fonts, 431, 433, 435, 436
Editable regions, 191, 212–221, 332
Editorial content
  completed pages preview, 238–240
  creating and styling text, 240–244
  in editable regions, 216
  font size for, 438
  models containing, 191
  spell-checking, 284–285
Element Display, 43, 367, 374–376
Elements
  boilerplate, 190–191
  class/id identification of, 304
  creating multiple, 92
  cursor focus on, 165
  defined, 52
  in Design view, 23
  dialogs, displays, and inspectors, 43–46
  for display vs. meaning, 57
  Extract panel styling of, 145–151
  formatting multiple, 80
  highlighting, 76
  HTML, 52–55, 169
  locked on a screen, 376–378
mobile device alignment, 460–463
new DOM panel, 162–163
parent and child, 72
pasting multiple, 258
properties display for, 77
references to, 3
removing unneeded, 191–196
resizing, 196–201
selecting, 192–194
structural display of, 36
styles applied to, 38, 40
understanding formats of, 65
Em measurements, 377
Email
  address wrapping, 369
  creating links, 365–366
  robots and, 370
  templates, 141
  web-hosted links, 346
Embedded styling, 69
Emmet, 91–92
Empty/void elements, 52
Encryption, 388
Entities, 55
Errors
  broken link checks, 383, 402
  in CSS syntax, 71
  fixing semantic, 206–208
  formatting failure from, 474
  found via HTML, 52
  linting support for, 112–114
  from manual link entry, 351
  spell checking, 284–285
Expanding code, 119
Experimentation, 46
Explicit encryption, 388
External file use, 34
External hyperlinks, 342, 360–364
Extract panel
  gradient backgrounds in, 169–173
  mockup styling via, 143–145
  styling elements via, 145–151
  for text, 153–156
F
Favorite City Tour scenario, 128
Files
  cloaking, 395–396
  dependent, 404
  named “temp,” 220
  opening consecutive, 238
  previewing completed, 238–240
  raster image formats, 300–303
  synchronization of, 408
Find and Replace, 285–291
Firefox, 128
Floating panels, 28
Folders
  cloaking, 395–396
  nonpublic, 391
  organization of, 8
Font Squirrel, 431
Fonts
  base, 429–436
  em measurements and, 377
  for mobile design, 482–484
  sizing, 438–442
  support for, 437
  Typekit, 133
  web-hosted, 428–442
  web-safe, 431
Footer, 184
Formatting. See also Styling
captions, 282–283
CSS boxes, 68
CSS vs. HTML, 62–64
header components, 448–453
with HTML lists, 248
images, 34
media queries for, 422
mobile navigation menus, 445–447
modifying text, 201–203
multiple elements, 80
and quality of content, 69
responsive tables, 471–477
tables, 263, 268
Frameworks, web, 427
Freezes, 229
FTP (File Transfer Protocol), 388
FTP over SSL/TLS, 388
FTP site setup, 389–394
Functionality test, 394
G
GIF (Graphics Interchange Format), 300
Git repository, 44–46
GLOBAL attribute, 426
Google, 81, 126, 431
Google Analytics, 127
Google Chrome, 128
Gradient backgrounds, 169–173, 447
Grammatical errors, 285
Graphics. See also Images, two categories of, 296–300
Group environments
  check in/check out for, 393
  cloaked files and, 396
  synchronization in, 408, 412
  templates for, 190
Grouping panels, 28–29
H
h2 element, 207–208
Hardware requirements, 4–5
Head, 52, 209
Headers
  adding backgrounds to, 179–183
  carousel styling for, 334–336
  creating/styling text for, 240–241
  font size for, 438
  formatting for screen size, 448–453
  hiding, 453–460
  layer, 174
  locked on the screen, 376–378
  in responsive tables, 472–473
  for tables, 272–274
Help resources, 19, 21
Hierarchy, CSS, 74
High color, 299
Highlighting, 76
Hinting of code, 89–90
Home links, 352–357
Home page, 396–402
Horizontal rule, 52, 206
Horizontal navigation, 131
hr elements, 206–207
HTML. See also Code
additions in HTML5, 55–58
as the backbone of the web, 49
comments, 98
composition of, 51–52
creating code, 88–95
creating lists, 248–254
defaults, 64–67
defined, 50
frequently used elements, 52–55
HTML 4, 58, 65
id attributes in, 82, 370
origins of, 50–51
overview of, 48
review questions, 59
table attributes, 263, 272
textual distinctions for, 2, 3
validating template code, 210–211
vs. CSS, 62–64
HTML elements
in Design view, 23
DOM Viewer for structure of, 42
Live view editing, 23, 43
Property inspector tab for, 33
structural display of, 36
HTML5
defaults, 65–66
id attributes in, 81, 82
new additions in, 55–58
semantic rules, 206
starter layouts in, 141
support, 84
tag attributes, 91
HTTP requests, 110
Hyperlinks
browser display of, 344
checking validity of, 383, 402
destination links, 374–376
email, 346
image-based, 367–370
internal, 349–359
lesson on, 340, 384
pseudo-classes, 354
review questions, 385
syntax for, 375
telephone, 381–382
types of, 342–343
Hypertext, 342
I
Ids
attributes, 43, 370, 374
creating, 81–82, 374
id-based links, 375–376
weight of, 74
Image carousel
adding images to, 331–334
and component visibility, 455
as content model, 191
deleting components in, 247
as editable region, 212
for home page, 400
list-structured, 254
as optional, 216
styling headings and text in, 331–334
Image Display, 44
Images
adding background, 179–183
adding to carousel, 331–334
Assets panel insertion,
317–318
onto child pages, 336–338
Design view insertion, 306–308
extracting mockup assets, 173–175
formatting, 34
home page, 400
image-based links, 367
Image menu insertion, 320–322
Image panel insertion, 322–324
inserting, 303–306
lesson on, 294
list as structure for, 254
for mobile devices, 319–320
from Photoshop, 311–316
placeholder, 212
resizing, 308–311, 316
review questions, 339
screen size and, 448–449
template insertion, 324–328
web image basics, 296–303
Image element, 304
Implicit encryption, 388
Importing text, 240–244
Inheritance theory, 72
Inline elements, 52, 69
Insert menu, 321–322
Insert panel, 322–324
Inspectors, 43–46
Installation, program, 4–5
Internal hyperlinks
on child pages, 357–359
defined, 342
home links, 352–357
relative, 349–352
within a webpage, 371–373
Internet access, 299–300
iOS, 481, 482
IP addresses, 390
ISP (Internet Service Provider), 390
JPEG (Joint Photographic Experts Group), 300
jQuery Mobile, 427
jQuery UI, 427
Keyboard shortcuts, 31–33
Languages. See HTML; LESS; Sass; SCSS
Layout
creating menu items, 164–165
creating new elements, 162–163
CSS conflicts, 151–153
deleting menu components, 156–159
footer, 184–185
gradient backgrounds, 169–173
header, 179–183
inserting menu items, 159–161
mockup image extraction, 173–175
mockup text extraction, 153–156
modifying Bootstrap, 196–201
new Bootstrap structures, 175–179
online resources, 138
power positions in, 131
review questions, 186
Living standard, HTML as, 51, 84
Local site
defining your, 11–14
synchronized with remote, 408–412
term defined, 388
Locked template areas. See Boilerplate (uneditable content)
Logos
in layered header, 179
placement of, 131, 132
on site template, 324
on varying screen sizes, 448–453
as vector graphics, 296
Lorem generator, 95
MAC
macOS, 3–4, 51
Maintenance, 387
Manual coding, 88–91
Lessons
accessing files for, 6–7
on coding, 86
on CSS, 60
design of, 1
on Dreamweaver
templates, 188
on HTML, 48
on images, 294
on mobile design, 414
on navigation, 340
on page layout, 138
for publishing websites, 386
recommended order of, 7–8
on web design, 124
on workspaces, 16
Li element, 54, 169
Line number selection, 114–115
Line wraps, 94
Linked styling, 69
Links. See Hyperlinks
Linting support, 112–114
List-based section
as content model, 191
creating new items in, 259–261
deleting components in, 247
as editable region, 212
Lists
content structure via, 254–261
creating, 248–254
lesson on, 236
review questions, 292–293
Live view
as default, 43, 63
defined, 23
external link creation, 360–364
Extract panel styling in, 146
pasting multiple elements in, 258
selecting elements in, 460
text formatting in, 242
Microsoft Access, 262
Microsoft Edge, 128
Microsoft Excel, 262, 276
Microsoft Word, 267
Minimizing panels, 25–26
Mobile design
content adaptation for, 469–477
controlling Bootstrap alignment, 460–463
custom media queries, 443–460
troubleshooting, 482–484
of images, 319–320
lesson on, 414
media queries, 422–423
media query syntax, 423
media type properties, 422
page component list for, 130
responsive, 129, 416–421, 427
review questions, 485
template revisions for, 464–468
testing, 463
VMQ interface, 423–427
web frameworks, 427
web-hosted fonts, 428–442
wireframes for, 134
Mobile devices
font issues with, 482
hyperlinks on, 355
image optimization for, 299, 316
telephone links for, 381
templates for, 20, 198
testing pages on, 479
web access via, 127, 128
Mobile-first design, 134, 416
Mockups
adapting templates to, 143
as basis for page design, 140
extracting image assets, 173–175
extracting styling from, 165
extracting text from, 153–156
how to make, 132–136
Monochrome, 298
MS DOS, 51
Multicursor support, 95–97
Multiple element placement, 258
N
Named anchor, 370, 374
Navbar element, 154
Navigation
as boilerplate, 203
Bootstrap bar, 175
destination links, 374–376
e-mail links, 364–366
external links, 360–364
formatting for screen size, 455–457
horizontal, 131
HTML, 92
hyperlinks, 342–343
id attributes and, 82
image-based links, 367–370
internal links, 349–359
lesson on, 340, 343, 384
locking elements onscreen, 376–378
menu in the power position, 132
menu styling, 378–381
menu visibility, 371, 376
for mobile devices, 421
review questions, 385
targeting page elements, 370–373
telephone links, 381–382
text styling, 204
thumbnails for designing, 130
Nesting selectors, 108–110
New Document dialog, 21
New Feature guides, 21
Notepad, 222, 224

O
Online resources
for Adobe Dreamweaver Classroom in a Book, 1, 6–7
on coding, 86
on CSS, 60
design of, 1
on Dreamweaver templates, 188
on HTML, 48
on images, 294
on mobile design, 414
on navigation, 340
for page layout, 138
for publishing websites, 386
TinyURLs, 1, 414
on web design, 124
for working with code, 86
on workspaces, 16
Opening files, 238
Optimization
color and, 299
for device size, 316
FTP performance, 392
for mobile devices, 416
via web frameworks, 427
Optimize setting, 314
Optional regions
image carousel as, 212, 216
inserting, 219–221
removing, 231–232
Ordered lists, 249–251
Overwrites, file, 405
Padding, 81
Palette, 298
Panels, 24–29
closing, 27
floating, 28
grouping, stacking, and docking, 28–29
list of available, 25
minimizing, 25–26
reordering, 27
user interface, 18
Paragraphs
content of, 240
list as structure for, 254
returns for, 242, 244
Parent elements
in Bootstrap templates, 196
in DOM panel, 216
inheritance theory and, 72
selecting, 117–118
Password, FTP, 390–391, 392
Paste
of elements in Live view, 258
of new menu items, 164–165
to replace code, 290–291
of tables, 262, 267–268
Personalization, 29–30
Phone links, 381–382
Photoshop
in Creative Cloud, 309
Extract from, 143–145, 145
extracting text from, 153–156
image correction in, 308
image insertion from, 311–316
mockups, 132–136
transfers to Dreamweaver, 139
Pixels
Bootstrap templates and, 198
characteristics of, 296
of desktop browsers, 128
of mobile devices, 134
of predefined layouts, 143
in raster graphics, 296–297
responsive design and, 129
and screen-size formatting, 453
Placeholders
adding content, 203–206
in card-based section, 212
defined, 191
home page, 397
replacing graphic, 303, 306, 311, 325
on third-party web hosts, 405, 408, 410
PNG (Portable Network Graphics), 301
Point to File function, 352, 359
Pop-ups, 21
Position Assist dialog, 43
Power positions, 131
Preprocessors, 99–112
Prerequisites, 2
Presentation and content, 68
Print design, 129
Printers, 297
Productivity tools, 189
Proofs of concept, 132–134
Properties
adding table, 269
background, 169–170, 179
in CSS Designer, 76, 77
dialogs, displays, and inspectors, 43–46
manual creation of, 260
of media types, 422
order for layered, 179
uninheritable, 72
window in CSS Designer, 39, 75, 79
property inspector
  document title field, 209
  in element and image displays, 43–44
  floating, 177
  how to use, 33–34
  resizing images via, 308
  for table attributes, 265
pseudo-classes, 354, 379, 473
publishing to the web
  cloaking files/folders, 395–396
  ftp troubleshooting, 392
  home page creation, 396–402
  lesson on, 386
  local/remote synchronization, 408–412
  prelaunch checklist, 402
  putting your site online, 403–408
  remote site setup, 388–394
  review questions, 413
  testing server for, 394
punctuation, missing, 3
purpose, knowing your, 126
q
  quick response (qr) code, 480, 481
  quick start, 19–20
  quotation marks, 91
r
  raster graphics, 296–300
  rds (remote development services), 389
  read only files, 148, 166
  readability
    code, 90
    headers for, 272
    lists as aid to, 248
    styling and, 477
  real-time preview, 477–482
  regular expressions, 285
  related files interface, 30, 34–35
  relative hyperlinks, 342–343, 349–352
  remote site
    cloaking and, 395
    defining your, 11–14
  publishing to, 403–408
  setup for, 388–394
  synchronized with local, 408–412
  term defined, 388–389
repeating background, 180
replace function, 285–291
research, audience, 127, 301
resizing
  elements, 196–201
  email to prevent wrapping, 369
  fonts, 438–442
  images, 308–311, 316
  for mobile responsiveness, 416–421
resolution, image, 297–298, 416
responsive web design
  defined, 129
  how to create, 416–421
  starter layouts as, 141
  for tables, 471–477
  troubleshooting, 469
review questions
  for coding, 123
  css, 85
  html, 59
  on images, 339
  on mobile design, 485
  on navigation, 385
  for page layout, 186
  on publishing to the web, 413
  on text, lists, tables, 292–293
  on web design, 137
  workspace, 47
robots, 370
root, 52
root directory, 391, 395
rows
  added to tables, 265–267
  duplicating template, 244–246
  row/column screen division, 175, 196
rules, css
  conflict between, 73
  conflict with style sheets, 151
  for custom media queries, 445–448
  defined, 69
  hierarchy of, 74
for media queries, 426–427
overlap in, 74
shorthand for writing, 80–81
syntax for, 71
textual distinctions for, 2, 3
rules, semantic, 206
s
safari, 481
sass (syntactically awesome style sheets)
  creating source file for, 101–104
  defined, 99
  resources on, 112
save all command, 231
saving work, 230, 231, 358, 359
screen size
  bootstrap defaults for, 198
  and element alignment, 460–463
  mobile design and, 416–417, 425
  before mobile devices, 416
  revising template for, 464–468
  targeting styling to, 443
scrubber tool, 417–421
scss (sassy css)
  defined, 99
  source files, 101–105
  vs. less, 99
search engines
  access to uploaded material, 396
  link-sharing interface, 363
  metadata for, 209–210
  structure and ranking with, 81, 278
security, 388, 481
selectors
  advanced, 476
  for component deletion, 192–194
  and declarations, 69
  nesting css, 108–110
  special characters, 73
selectors window, 39, 40, 75, 445
semantic text structures, 278–281
semantic web design, 57
creating new Bootstrap, 175–179
in the DOM panel, 162
DOM Viewer of, 42
duplicating rows in, 244–246
of HTML, 52
HTML as, 49
list-based content, 254–261
revising Bootstrap template, 464–468
and search engine rank, 81
semantic text structures, 278–281
structural elements, 52–53
Tag Selector display of, 36
thumbnails for designing, 130
Style Rendering command, 68
Style sheets
body element, 103–104
Bootstrap, 148, 166
combining, 110
importing, 110–112
for mobile devices, 423, 425
placement of new items in, 261
pseudo-classes on, 354
variables, 103
Styling. See also CSS (cascading style sheets)
applying CSS, 69–74
Bootstrap style sheet, 148
copy and paste of CSS, 38
custom media queries for, 443
default HTML, 64–67
effect of browsers on, 66–67
extracting text, 165–168
hierarchy of, 74
hyperlinks, 354–355
navigation menu, 378–381
predefined layouts, 143–145
Properties window display of, 40
tables, 268–272
targeting, 108
text in a carousel, 334–336
via Extract panel, 145–151
Support
with CSS preprocessors, 99, 112
font, 437
HTML, 84
linting, 112–114
SVG (Scalable Vector Graphic), 296
Switching views, 22–24
Sync Settings, 8
Synchronization, site, 408–412
Syntax, media query, 423

Table properties, 34
Tables, 262–283
adding data to, 264
adding rows to, 265–267
borders, 263
caption elements, 282–283
cell size, 274–276
controlling display, 274–276
copying and pasting, 267–268
CSS styling for, 268–270
design conflict checks, 471
header row addition, 272–274
inserted from other sources, 276–278
lesson on, 236
from scratch, 262–267
semantic text structures, 278–281
styling cells, 271–272

Tag Selector display of, 36
Tags, HTML
attributes in HTML5, 91
automatic closing of, 90
list of, 52–54
new in HTML5, 56–57
parent tag selection, 117–118
purpose of, 51
selection, 115–117, 191
weight of, 74

Target menu, 364
Technology
browser support and, 51
continuously evolving, 84
new HTML5, 58
vs. programs, 50
Telephone links, 381–382
Web design
desktop/mobile compromise, 134–136
with HTML5, 58
for mobile devices, 134
online resources, 124
Photoshop tools for, 134
questions guiding, 126–128
responsive. See Responsive
web design
review questions, 137
semantic, 57
thumbnails and wireframes, 128–134

Web Edition, 6
Web frameworks, 427
Web-hosted fonts, 428–442
Web hosts, 390, 391, 405
Web-safe color palette, 299
WebDav (Web Distributed
Authoring and Versioning), 389
Webpages
adaptability of mobile, 423
browser displays of, 66–67
design conflict checks, 469
font issues, 482–484
FTP connection, 392
template, 227–233
Uploading site, 403–408
URLs, 414
Username, FTP, 390–391, 392

Web Design, 6
Web development, 126–128
local and remote site setup, 11–14
publishing. See Publishing to
the web
standards for, 65
Weight of rules, 73, 74
What You See Is What You Get
(WYSIWYG), 18, 22
Whitespace, ignoring, 287
Wildcards, 73
Window menu
  list of panels in, 25
  opening panels from, 27
Windows computers
  Dreamweaver use on, 3–4
  percent of users on, 128
Wireframes, 132–134
Workspaces
  CSS Designer, 37–42
  customizing, 29–30, 62
  default images folder, 303
  dialogs, displays, and inspectors, 43–46
DOM Viewer, 42
  experimenting with, 46
  group environment, 190, 393, 396, 408
  keyboard shortcuts, 31–33
  New Feature guides, 21
  panels, 24–29
  Property inspector, 33–34
  Related Files interface, 34–35
  review questions, 47
  screen size for, 416–417
  setting up, 9–10
  Start Screen, 19–21
  Switching/splitting views, 22–24
Tag Selector interface, 36–37
  toolbars, 30–31
  touring the, 18–19
  user interface, 18
  VMQ interface, 42
Wrapping of content,
  196–201, 369
Write/Read access, 148, 166
Y
  Yahoo, 126
Z
  ZIP archives, 7