
Adobe
Dw
Adobe
Dreamweaver CC
2019 release
CLASSROOM IN A BOOK®
The official training workbook from Adobe
Jim Maivald

Contents

3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35

36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
B2-2

B2-3

B2-4

B2-5

B2-6

B2-7

B2-8

B2-9

B2-10

B2-11

B2-12

B2-13

B2-14

B2-15

B2-16

B2-17

B2-18

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

B3-2

B3-3

B3-4

B3-5

B3-6

B3-7

B3-8

B3-9

B3-10

B3-11

B3-12

B3-13

B3-14

B3-15

B3-16

B3-17

B3-18

B3-19

B3-20

B3-21

B3-22

B3-23

B3-24

B3-25

B3-26

B3-27

B3-28

B3-29

B3-30

B3-31

B3-32

B3-33

B3-34

B3-35

B3-36

B3-37

B3-38

B3-39

B3-40

B3-41

B3-42

B3-43

B3-44

B3-45

B3-46

B3-47

B3-48

B3-49

B3-50

B3-51

B3-52

B3-53

98

99

100

101

102

103

104

105

106

107

108

109

B4-2

B4-3

B4-4

B4-5

B4-6

B4-7

B4-8

B4-9

B4-10

B4-11

110

111

112

113

114

115

116

117

118

119

120

121

122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154

155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187

188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220

221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253

254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286

287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319

320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352

353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385

386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418

419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451

452

B14-2

B14-3

B14-4

B14-5

B14-6

B14-7

B14-8

B14-9

B14-10

B14-11

B14-12

B14-13

B14-14

B14-15

B14-16

B14-17

B14-18

B14-19

B14-20

B14-21

B14-22

B14-23

B14-24

B14-25

B14-26

B14-27

B14-28

B14-29

B14-30

B14-31

B14-32

B14-33

B14-34

B14-35

B14-36

B14-37

B14-38

B14-39

B14-40

B14-41

B14-42

B14-43

B14-44

B14-45

B14-46

B14-47

B14-48

B14-49

B14-50

B14-51

B14-52

B14-53

B14-54

B14-55

B14-56

B14-57

B14-58

B14-59

B14-60

B15-2

B15-3

B15-4

B15-5

B15-6

B15-7

B15-8

B15-9

B15-10

B15-11

B15-12

B15-13

B15-14

B15-15

B15-16

B15-17

B15-18

B15-19

B15-20

B15-21

B15-22

B15-23

B15-24

B15-25

B15-26

B15-27

B15-28

B15-29

B15-30

B15-31

B15-32

B15-33

B15-34

B15-35

B15-36

B15-37

B15-38

B15-39

B15-40

B15-41

B15-42

B15-43

B15-44

B15-45

B15-46

B15-47

B15-48

B15-49

B15-50

B15-51

B15-52

B15-53

B15-54

B15-55

B15-56

B15-57

B15-58

B15-59

B15-60

B15-61

B15-62

B15-63

B15-64

B16-2

B16-3

B16-4

B16-5

B16-6

B16-7

B16-8

B16-9

B16-10

B16-11

B16-12

B16-13

B16-14

B16-15

B16-16

B16-17

B16-18

B16-19

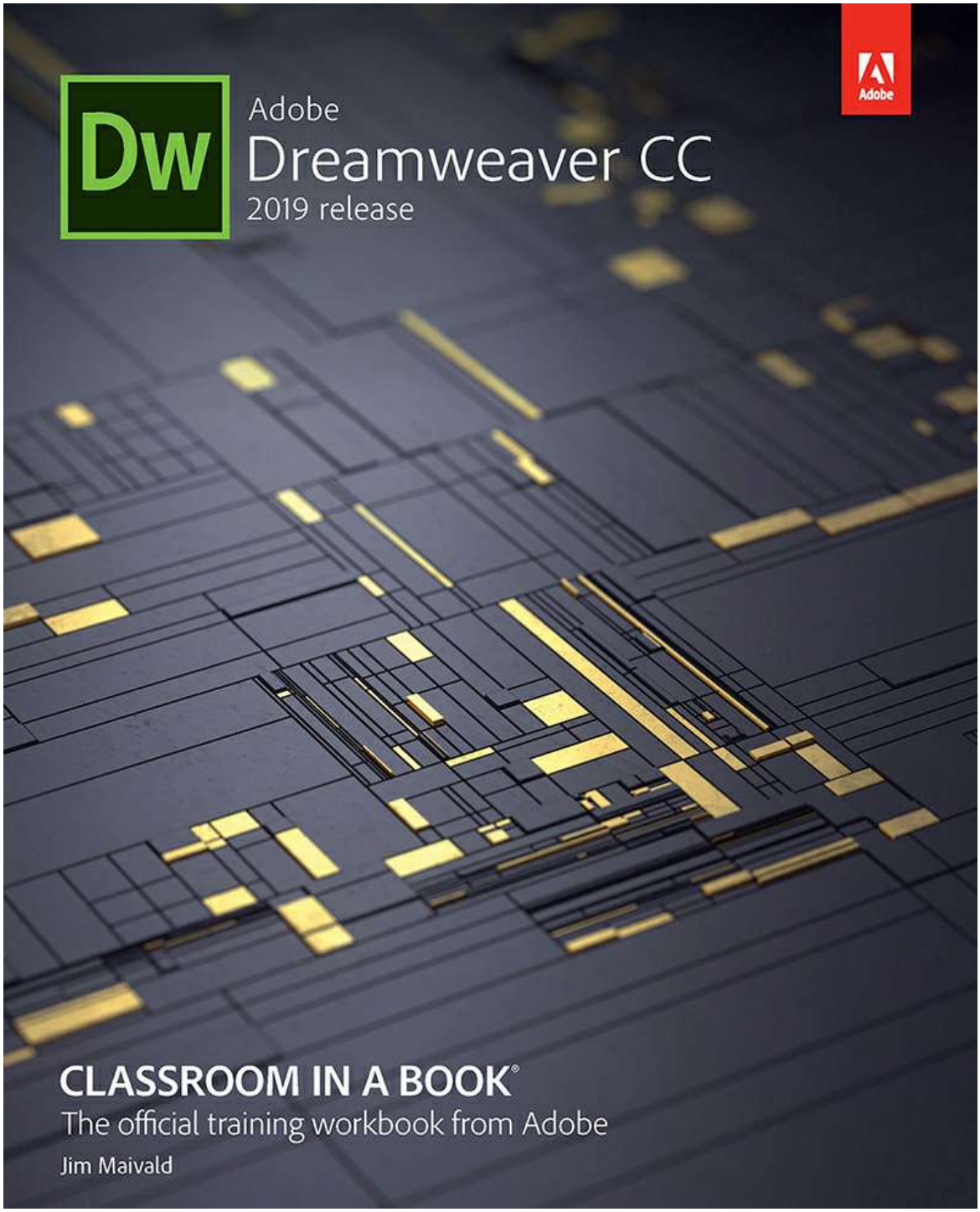
B16-20

B16-21

B16-22

B16-23

B16-24


Adobe
Dw
Adobe
Dreamweaver CC
2019 release
CLASSROOM IN A BOOK®
The official training workbook from Adobe
Jim Maivald

Adobe Dreamweaver CC Classroom in a Book ® 2019 release

The official training workbook from Adobe

Jim Maivald

Adobe Dreamweaver CC Classroom in a Book® (2019 release)

Adobe Press is an imprint of Pearson Education, Inc. For the latest on Adobe Press books, go to www.adobepress.com. To report errors, please send a note to errata@peachpit.com. For information regarding permissions, request forms and the appropriate contacts within the Pearson Education Global Rights & Permissions department, please visit www.pearsoned.com/permissions/.

Adobe Systems Incorporated, 345 Park Avenue, San Jose, California 95110-2704, USA

Writer: James J Maivald
Executive Editor: Laura Norman
Development Editor: Robyn G. Thomas
Technical Reviewer: Candyce Mairs
Senior Production Editor: Tracey Croom
Copyeditor: Scout Festa
Composition: Kim Scott, Bumpy Design
Proofreader: Patricia Pane
Indexer: Valerie Haynes Perry

Cover Illustration: Dimitris Ladopoulos (Athens, Greece), behance.net/gallery/59061121/Algorithm
Cover Designer: Eddie Yuen
Interior Designer: Mimi Heft

ISBN-13: 978-0-13-526214-6

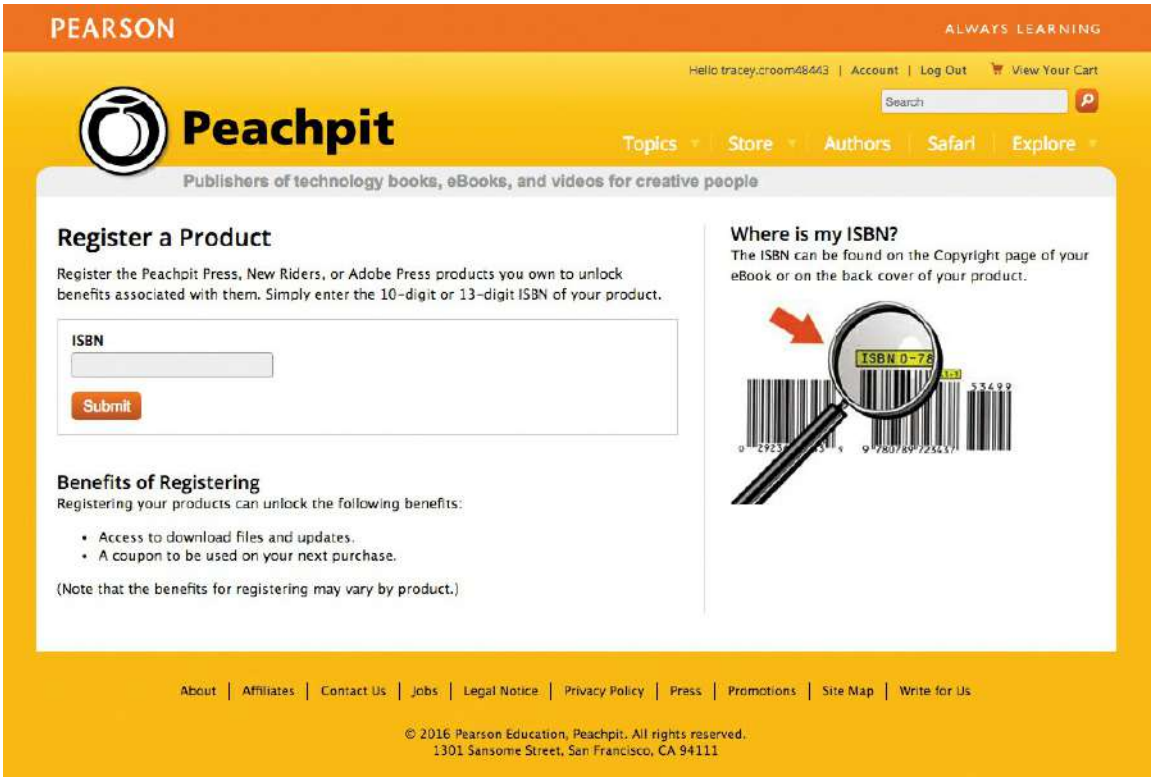
ISBN-10: 0-13-526214-3

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.

You 'll find the files you need on your **Account** page at peachpit.com on the **Registered Products** tab.

- Go to www.peachpit.com/register.
- Sign in or create a new account.
- Enter the ISBN: **9780135262146**



- Answer the questions as proof of purchase.
- The lesson files can be accessed through the Registered Products tab on your Account page.
- 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.

PEARSON ALWAYS LEARNING

| Account | Log Out View Your Cart

Search

Peachpit

Topics | Store | Authors | Safari | Explore

Publishers of technology books, eBooks, and videos for creative people

Home > Account

Account

Digital Purchases | Registered Products | Wish List | Saved Content

Registered Products (What is this?)

Adobe Dreamweaver CC Classroom in a Book (2019 release)
Registered Jan 6, 2019
Benefit: Access Bonus Content

Register Another Product

Digital Product Voucher
(What is this?)

Enter your code to receive a digital product or lesson files.

Submit your code

Personal Information
Edit Email & User Name
Change Password
Manage Billing & Shipping Addresses
Edit Your Profile

Newsletter Subscriptions
Manage Subscriptions

Discount Codes
Manage Codes
About Discount Codes

Contents

Getting Started

Adobe® Dreamweaver CC is one of the leading web-authoring programs available. Whether you create websites for others for a living or plan to create one for your own business, Dreamweaver offers all the tools you need to get professional-quality results.

About Classroom in a Book

Adobe Dreamweaver CC Classroom in a Book® (2019 release) is part of the official training series for graphics and publishing software developed with the support of Adobe product experts.

The lessons are designed so that you can learn at your own pace. If you're new to Dreamweaver, you'll learn the fundamentals of putting the program to work. If you are an experienced user, you'll find that Classroom in a Book teaches many advanced features, including tips and techniques for using the latest version of Dreamweaver.

Although each lesson includes step-by-step instructions for creating a specific project, you'll have room for exploration and experimentation. You can follow the book from start to finish or complete only those lessons that correspond to your interests and needs. Each lesson concludes with a review section containing questions and answers on the subjects you've covered.

TinyURLs

At several points in the book, I reference external information available on the Internet. The uniform resource locators (URLs) for this information are often long and unwieldy, so I have provided custom TinyURLs in many places for your convenience. Unfortunately, the TinyURLs sometimes expire over time and no longer function. If you find that a TinyURL doesn't work, look up the actual URL provided in the appendix.

Prerequisites

Before using *Adobe Dreamweaver CC Classroom in a Book (2019 release)*, you should have a working knowledge of your computer and its operating system. Be sure you know how to use the mouse, standard menus, and commands, as well as how to open, save, and close files. If you need to review these techniques, see the printed or online documentation included with your Windows or macOS operating system.

Conventions used in this book

Working in Dreamweaver means you'll be working with code. We have used several

conventions in the following lessons and exercises to make working with the code in this book easier to follow and understand.

Bolded text

Certain names, words, and phrases will be bolded from time to time, usually when first cited in an instruction. This styling will include text, other than HTML or CSS code, that needs to be entered into program dialogs or into the body of a webpage, like this:

Type **Insert main heading here**

Filenames, like **mygreen-styles.css,** will also be bolded as needed to identify crucial resources or targets of a specific step or exercise. Be aware that these same names may not be bolded in introductory descriptions or general discussion. Be sure to identify all resources required in a specific exercise prior to commencing it.

Code font

In many instructions, you will be required to enter HTML code, CSS rules and properties, and other code-based markup. To distinguish the markup from the instructional text, the entries will be styled with a code font, like this:

Examine the following code: `<h1>Heading goes here</h1>`

In instances where you must enter the markup yourself, the entry will be formatted in color, like this:

Insert the following code: **`<h1>Heading goes here</h1>`**

Enter the code exactly as depicted, being careful to include all punctuation marks and special characters.

Strikethrough

In several exercises, you will be instructed to delete markup that already exists within the webpage or style sheet. In those instances, the targeted references will be identified with strikethrough formatting, like this:

Delete the following values:

Click here to view code image

```
margin: 10px 20px 10px 20px;
background-image: url(images/fern.png), url(images/stripe.png);
```

Be careful to delete only the identified markup so that you achieve the following result:

Click here to view code image

```
margin: 10px 10px;
background-image: url(images/fern.png);
```

In most cases, white space differences will not affect the resulting display or operation of the code, but you should always attempt to match the depicted code exactly.

Missing punctuation

HTML code, CSS markup, and JavaScript often require the use of various punctuation, such as periods (.), commas (,), and semicolons (;), and can be damaged by their incorrect usage or placement. Consequently, I have omitted periods and other punctuation expected in a sentence or paragraph from an instruction or hyperlink whenever it may cause confusion or a possible error, as in the following two instructions:

Enter the following code: **`<h1>Heading goes here</h1>`**

Type the following link: **`https://adobe.com`**

Element references

Within the body of descriptions and exercise instructions, elements may be referenced by name or by class or id attribute. When an element is identified by its tag name, it will appear as `<section>` or `section`. When referenced by its class attribute, the name will appear with a leading period (.) in a code-like font, like this: `.content` or `.sidebar1`. References to elements by their id attribute will appear with a leading hash (#) and in a code font, like this: `#top`. This practice matches the way these elements appear in Dreamweaver's tag selector interface.

Windows vs. macOS instructions

In most cases, Dreamweaver performs identically in both Windows and macOS. Minor differences exist between the two systems, mostly because of platformspecific issues out of the control of the program. Most of these are simply differences in keyboard shortcuts, how dialogs are displayed, and how buttons are named. In most cases, screen shots were made in the macOS version of Dreamweaver and may appear different from your own screen.

Where specific commands differ, they are noted within the text. Windows commands are listed first, followed by the macOS equivalent, such as Ctrl+C/Cmd+C. Common abbreviations are used for all commands whenever possible, as follows:

Windows	**macOS**
Control = Ctrl	Command = Cmd
Alternate = Alt	Option = Opt

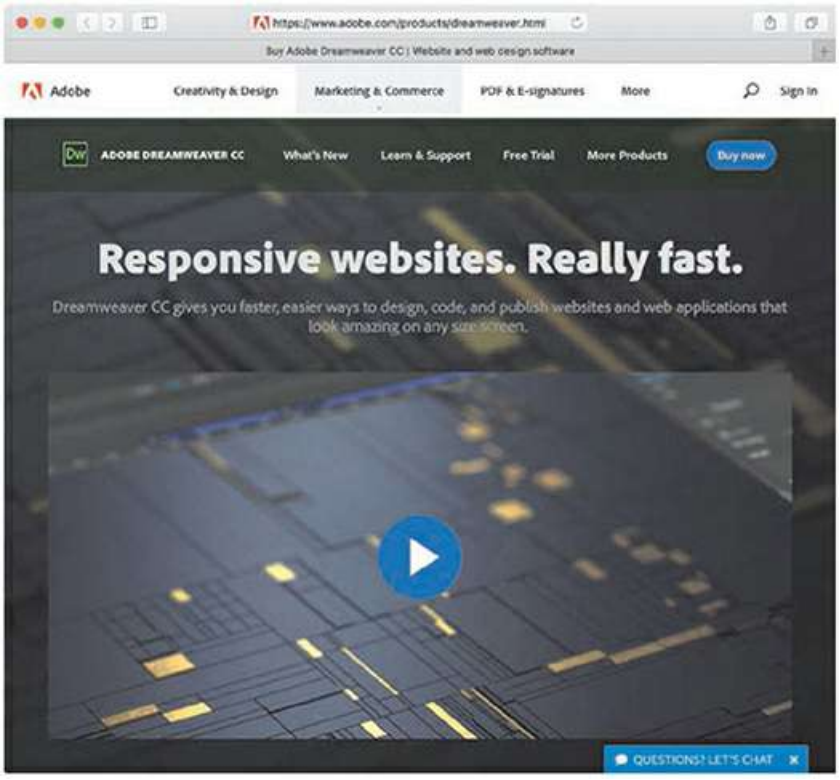
As lessons proceed, instructions may be truncated or shortened to save space, with the

assumption that you picked up the essential concepts earlier in the lesson. For example, at the beginning of a lesson you may be instructed to select Edit > Copy or "press Ctrl+C/Cmd+C." Later, you may be told to "copy" text or a code element. These should be considered identical instructions.

If you find you have difficulties in any particular task, review earlier steps or exercises in that lesson. In some cases, if an exercise is based on concepts covered earlier you will be referred to the specific lesson.

Installing the program

Before you perform any exercises in this book, verify that your computer system meets the hardware requirements for Dreamweaver, that it's correctly configured, and that all required software is installed.



If you do not have Dreamweaver, you will first have to install it from Creative Cloud. Adobe Dreamweaver must be purchased separately; it is not included with the lesson files that accompany this book. Go to **helpx.adobe.com/dreamweaver/system-requirements.html** to obtain the system requirements.

Go to **www.adobe.com/creativecloud/plans.html** to sign up for Adobe Creative Cloud. Dreamweaver may be purchased with the entire Creative Cloud family or as a standalone app. Adobe also allows you to try Creative Cloud and the individual applications for seven days for free.

Check out **www.adobe.com/products/dreamweaver.html** to learn more about the different options for obtaining Dreamweaver.

Updating Dreamweaver to the latest version

Although Dreamweaver is downloaded and installed on your computer hard drive, periodic updates are provided via Creative Cloud. Some updates provide bug fixes and security patches, while others supply amazing new features and capabilities.

The lessons in this book are based on Dreamweaver CC (2019 release) and may not work properly in earlier versions of the program. To check which version is installed on your computer, choose Help > About Dreamweaver in Windows or Dreamweaver > About Dreamweaver on macOS. A window will display the version number of the application and other pertinent information.

If you have an earlier version of the program installed, you will have to update Dreamweaver to the latest version. You can check the status of your installation by opening the Creative Cloud manager and logging in to your account.

Windows

macOS

Check out **helpx.adobe.com/creative-cloud/help/download-install-trial.html** to learn how to download and install a limited-period trial of Creative Cloud to your computer or laptop.

Online content

Your purchase of this Classroom in a Book includes online materials provided by way of your Account page on peachpit.com.

Lesson files

To work through the projects in this book, you will need to download the lesson files from peachpit.com. You can download the files for individual lessons, or it may be possible to download them all in a single file.

Warning

Do not copy one lesson folder into any other lesson folder. The files and folders for each lesson cannot be used interchangeably.

Web Edition

The Web Edition is an online interactive version of the book providing an enhanced learning experience. Your Web Edition can be accessed from any device with a connection to the Internet; it contains the following:

- The complete text of the book
- Hours of instructional video keyed to the text
- Interactive quizzes

In addition, the Web Edition may be updated when Adobe adds significant feature updates between major Creative Cloud releases. To accommodate the changes, sections of the online book may be updated or new sections may be added.

Accessing the lesson files and Web Edition

If you purchased an ebook from peachpit.com or adobepress.com, your Web Edition will automatically appear under the Digital Purchases tab on your Account page. Click the *Launch* link to access the product. Continue reading to learn how to register your product to get access to the lesson files.

If you purchased an ebook from a different vendor or you bought a print book, you must register your purchase on peachpit.com to access the online content:

. Go to www.peachpit.com/register.

. Sign in or create a new account.

- Enter the ISBN: **9780135262146.**
- Answer the questions as proof of purchase.
- The Web Edition will appear on the Digital Purchases tab on your Account page. Click the *Launch* link to access the product.

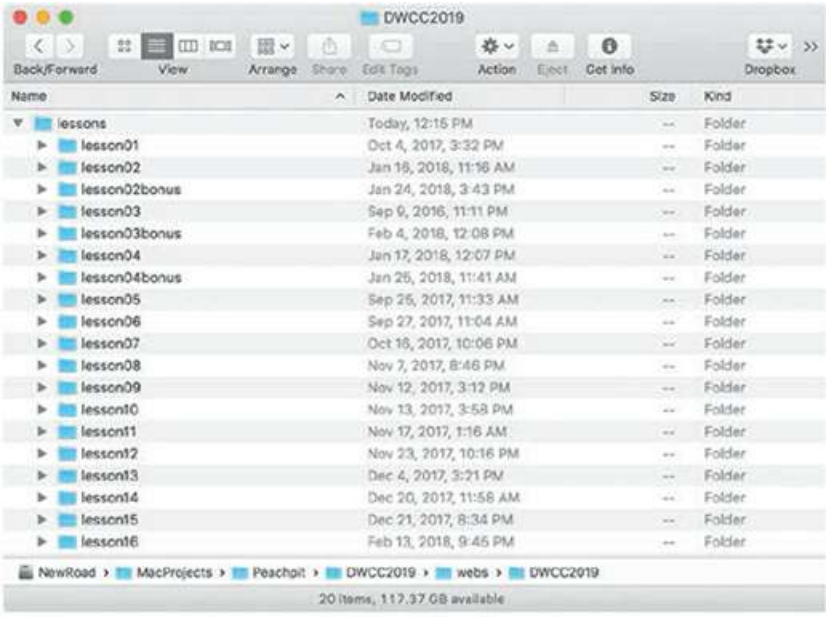
  The lesson files can be accessed through the Registered Products tab on your Account page. Click the Access Bonus Content link below the title of your product to proceed to the download page. Click the lesson file links to download them to your computer.

  The files are compressed into ZIP archives to speed up download time and to protect the contents from damage during transfer. You must uncompress (or "unzip") the files to restore them to their original size and format before you use them with the book. Modern Mac and Windows systems are set up to open ZIP archives by simply double-clicking.

> **Note**
>
> Windows may treat Zip archives like a file folder, allowing you to access the contents without decompressing it first. To use the files in Dreamweaver, you must decompress each archive first.

- Do one of the following:

  - If you downloaded **DWCC2019_lesson_files.zip**, unzipping the archive will produce a folder named **DWCC2019_Lesson_Files** containing all the lesson files used by the book.
  - If you downloaded the lessons individually, create a new folder on your hard drive and name it **DWCC2019**. Unzip the individual lesson files to this folder. That way, all the lesson files will be stored in one location. Do not share or copy files between lessons.



Note

The files are updated from time to time, so the dates depicted in screen shots may be different from the ones shown.

Recommended lesson order

The training in this book is designed to take you from A to Z in basic to intermediate website design, development, and production. Each new lesson builds on previous exercises, using supplied files and assets to create an entire website. We recommend you download all lesson files at once.

Start with Lesson 1 and proceed through the entire book to Lesson 12. Continue with the online Lessons 13 through 16 (refer to the "Bonus material" section for more information about the online material).

I recommend that you do not skip any lessons, or even individual exercises. Although ideal, this method may not be a practicable scenario for every user. So each lesson folder contains all the files needed to complete every exercise within it using partially completed or staged assets, allowing you to complete individual lessons out of order, if desired.

However, don't assume that the staged files and customized templates in each lesson represent a complete set of assets. It may seem that these folders contain duplicative materials, but these "duplicate" files and assets, in most cases, cannot be used interchangeably in other lessons and exercises. Doing so will probably cause you to fail to achieve the goal or desired results of the exercise.

For that reason, you should treat each folder as a standalone website. Copy the lesson folder to your hard drive, and create a new site for that lesson using the Site Setup dialog. Do not define sites using subfolders of existing sites. Keep your sites and assets in their original folders to avoid conflicts.

One suggestion is to organize the lesson folders in a single web or sites master folder near the root of your hard drive. But avoid using the Dreamweaver application folder. In most cases, you'll want to use a local web server as your testing server, which is described in Lesson 11, "Publishing to the Web."

Bonus material

We've provided additional material for Lessons 2, 3, and 4 on the Peachpit website. This book has so much great material that we couldn't fit it all in the printed pages, so we placed Lessons 14 through 16 on the Peachpit website as well:

Lesson 2, "HTML Basics Bonus"

Lesson 3, "CSS Basics Bonus"

Lesson 4, "Creating Web Assets Using Photoshop Generator Bonus"

Lesson 14, "Working with a Web Framework"

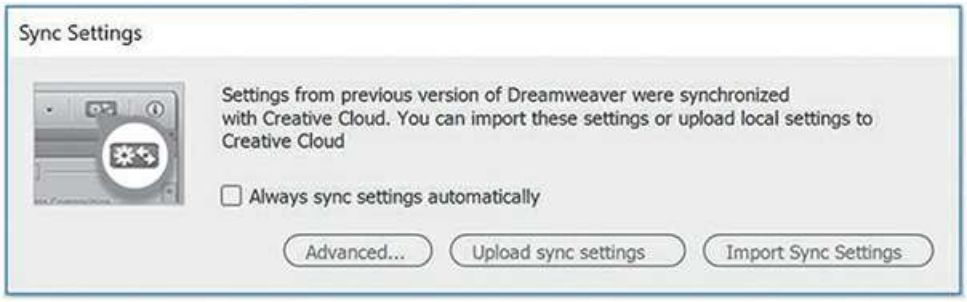
Lesson 15, "Adapting Content to Responsive Design"

Lesson 16, "Working with Web Animation and Video"

You will find these on your account page (Lessons & Update Files tab) once you register your book, as described earlier in "Accessing the lesson files and Web Edition."

On first launch

Right after installation or upon first launch, Dreamweaver CC will display several introduction screens. First, the Sync Settings dialog will appear. If you are a user of previous versions of Dreamweaver, select Import Sync Settings to download your existing program preferences. If this is the first time you've used Dreamweaver, select Upload Sync Settings to sync your preferences to your Creative Cloud account.



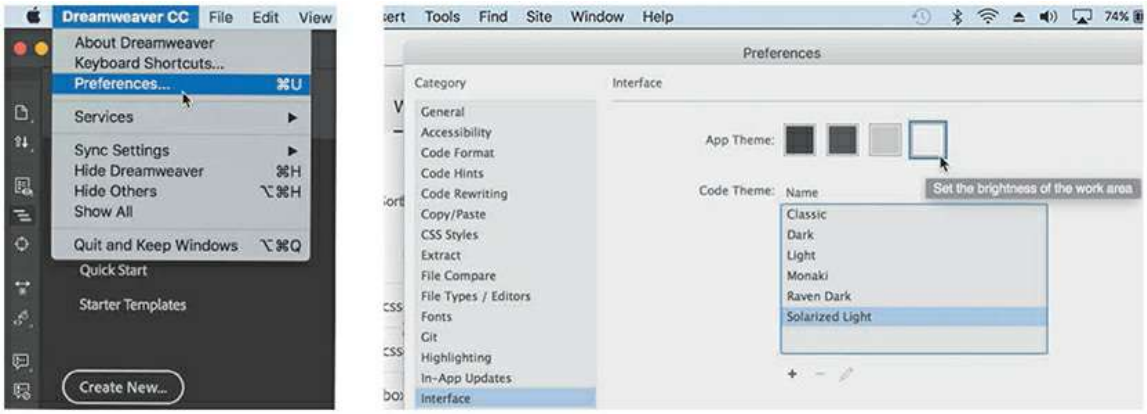
In the book, I use the lightest interface themes for the screen shots. This was done both to save ink in printing and to place less stress on the environment. Feel free to pick the color themes you prefer.

Choosing the program color theme

If you purchased the book after you installed and launched Dreamweaver, you may be using a different color theme than the one pictured in most screen shots in the book. All exercises will function properly using any color theme, but if you want to configure your interface to match the one shown, complete the following steps.

- Select Edit > Preferences in Windows or Dreamweaver CC > Preferences in macOS. The Preferences dialog appears.
- Select the Interface category.
- Select the lightest App Theme color.

  Select **Solarized Light** from the Code Theme menu.



The interface changes to the new theme. Depending on which app theme you select, the code theme may change automatically. The changes are not permanent yet. If you close the dialog, the theme will revert to the original colors.

. Click the Apply button.

The theme changes are now permanent.

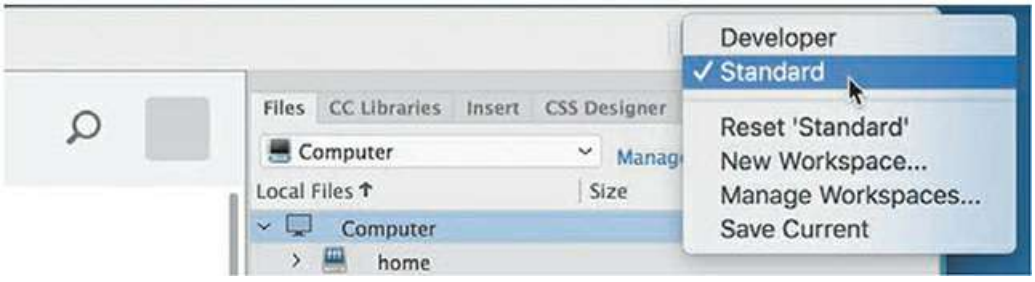
. Click the Close button.

Feel free to change the color theme at any time. Often users select the theme that works best in their normal working environment. The lighter themes work best in well-lighted rooms, while the darker themes work best in indirect or controlled lighting environments used in some design offices. All exercises will work properly in any theme color.

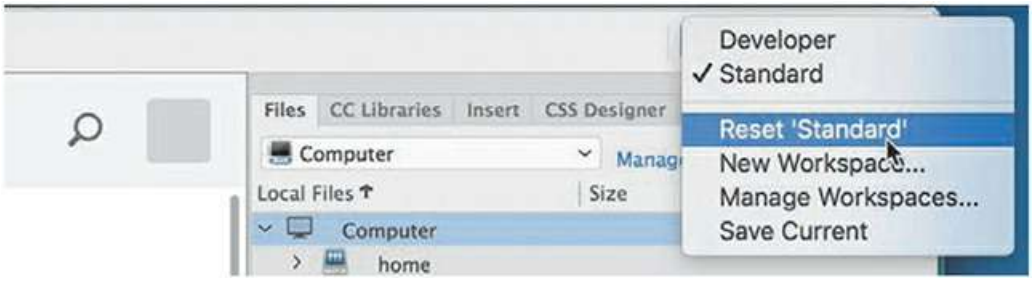
Setting up the workspace

Dreamweaver CC (2019 release) includes two main workspaces to accommodate various computer configurations and individual workflows. For this book, the Standard workspace is recommended.

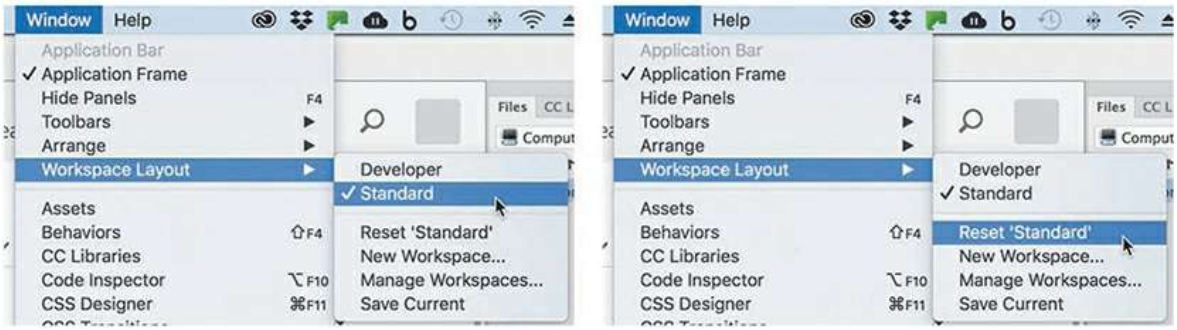
. If the Standard workspace is not displayed by default, you can select it from the Window > Workspace menu.



. If the default Standard workspace has been modified—where certain toolbars and panels are not visible (as they appear in the figures in the book)—you can restore the factory settings by choosing Reset ‘Standard’ from the Workspace drop-down menu.



These same options can be accessed from the Window > Workspace Layout menu.



Most of the figures in this book show the Standard workspace. When you finish the lessons in this book, experiment with each workspace to find the one that you prefer, or build your own configuration and save the layout under a custom name.

For a more complete description of the Dreamweaver workspaces, see Lesson 1, "Customizing Your Workspace."

Defining a Dreamweaver site

In the course of completing the following lessons, you will create webpages from scratch and use existing files and resources that are stored on your hard drive. The resulting webpages and assets make up what's called your *local* site. When you are ready to upload your site to the Internet (see Lesson 11, "Publishing to the Web"), you publish your completed files to a web-host server, which then becomes your *remote* site. The folder structures and files of the local and remote sites are usually mirror images of one another.

The first step is to define your local site.

Warning

You must unzip the lesson files before you create your site definition.

. Launch Adobe Dreamweaver CC (2019 release) or later.

. Open the Site menu.

The Site menu provides options for creating and managing standard Dreamweaver sites.

. Choose New Site.

The Site Setup dialog appears.

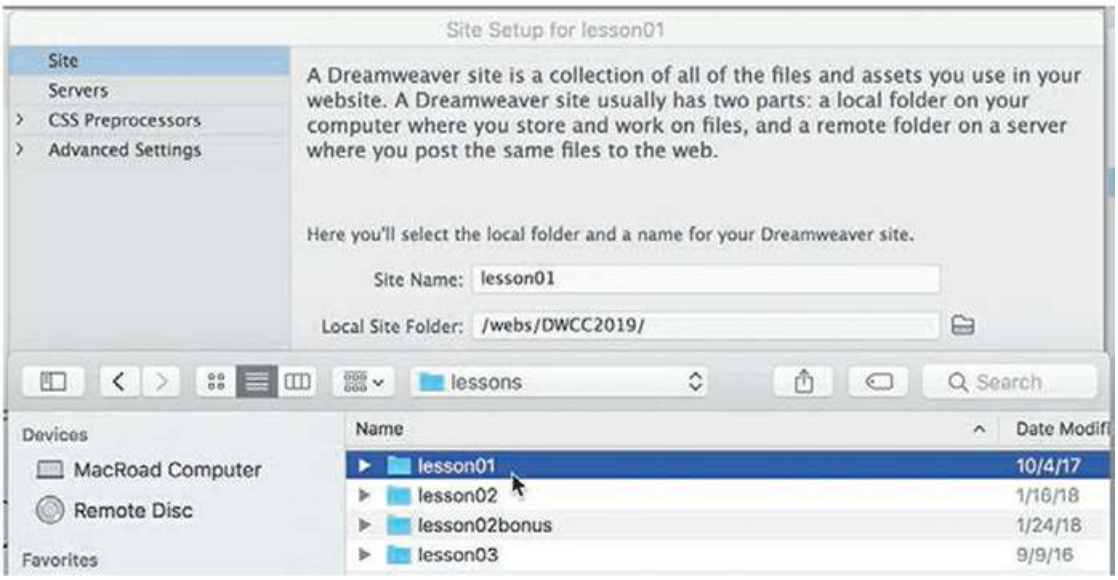
To create a standard website in Dreamweaver, you need only name it and select the local site folder. The site name should relate to a specific project or client and will appear in the Files panel Site drop-down menu. This name is intended for your own purposes only; it will not be seen by the public, so there are no limitations to the name you can create. Use a name that clearly describes the purpose of the website. For the purposes of this book, use the name of the lesson you intend to complete, such as lesson01, lesson02, lesson03, and so on.

Note

The main folder that contains the site will be referred to throughout the book as the site root folder.

. Type **lesson01** or another name, as appropriate, in the Site Name field.

- Next to the Local Site Folder field, click the Browse For Folder icon.
- Navigate to the appropriate folder containing the lesson files you downloaded from peachpit.com (as described earlier), and click Select/Choose.



You could click Save at this time and begin working on your new website, but you'll add one more piece of handy information.

Note

Lesson files must be decompressed prior to defining the site.

- Click the arrow next to the Advanced Settings category to reveal the categories listed there. Select Local Info.

Although it's not required, a good policy for site management is to store different file types in separate folders. For example, many websites provide individual folders for images, PDFs, videos, and so on. Dreamweaver assists in this endeavor by including an option for a default images folder.

Later, as you insert images from other locations on your computer, Dreamweaver will use this

setting to automatically move the images into the site structure.

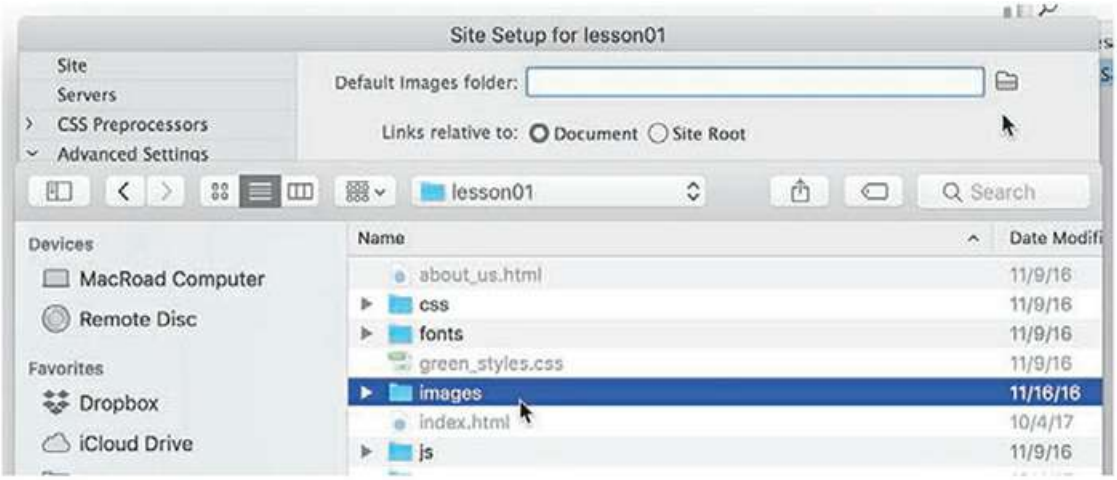
Note

The folder that contains the image assets will be referred to throughout the book as the site default images folder or the default images folder.

. Next to the Default Images Folder field, click the Browse For Folder icon. When the dialog opens, navigate to the appropriate images folder for that lesson or site and click Select/Choose.

Note

Resource folders for images and other assets should always be contained within the main site root folder.



The path to the images folder appears in the Default Images Folder field. The next step would be to enter your site domain name in the Web URL field.

. Enter http://green-start.org for the lessons in this book, or enter your own website URL, in the Web URL field.

Note

The Web URL is not needed for most static HTML sites, but it's required for working with sites using dynamic applications or to connect to databases and a testing server.

You've entered all the information required to begin your new site. In subsequent lessons, you'll add information that will enable you to upload files to your remote and testing servers.

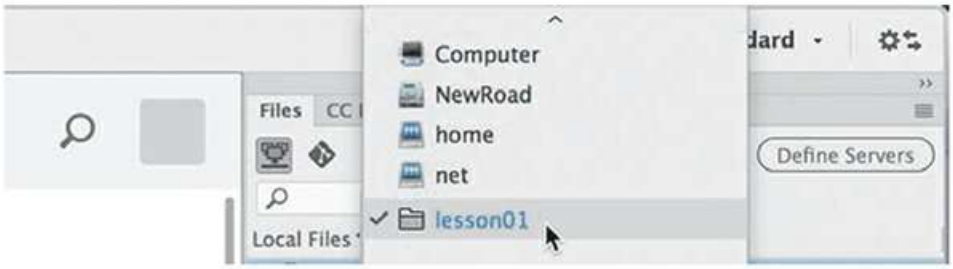
. In the Site Setup dialog, click Save.

The Site Setup dialog closes.

Whenever a site is selected or modified, Dreamweaver will build, or rebuild, a cache of every file in the folder. The cache identifies relationships between the webpages and the assets within sites and will assist you whenever a file is moved, renamed, or deleted to update links or other referenced information.

. Click OK to build the cache, if necessary.

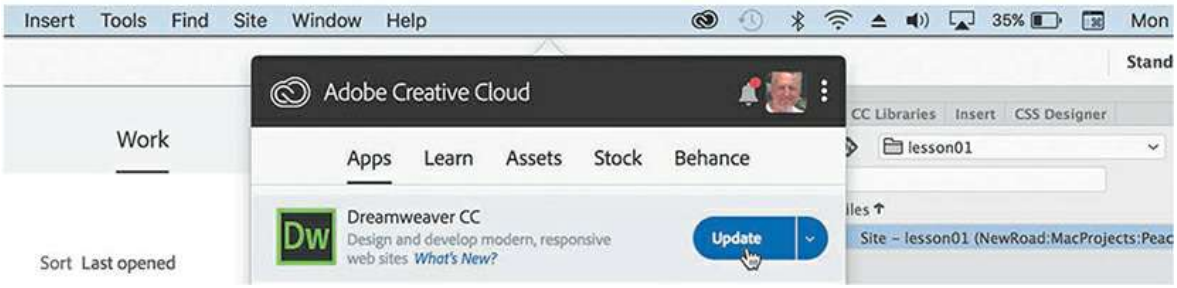
In the Files panel, the new site name appears in the site list drop-down menu. As you add more site definitions, you can switch between the sites by selecting the appropriate name from this menu.



Setting up a site is a crucial first step in beginning any project in Dreamweaver. Knowing where the site root folder is located helps Dreamweaver determine link pathways and enables many sitewide options, such as orphaned-file checking and Find and Replace.

Checking for updates

Adobe periodically provides software updates. To check for updates in the program, choose Help > Updates in Dreamweaver. An update notice may also appear in the Creative Cloud update desktop manager.



For book updates and bonus material, visit your Account page on peachpit.com and select the Lesson & Update Files tab.

Additional resources

Adobe Dreamweaver CC Classroom in a Book (2019 release) is not meant to replace documentation that comes with the program or to be a comprehensive reference for every feature. Only the commands and options used in the lessons are explained in this book. For comprehensive information about program features and tutorials, refer to these resources:

Adobe Dreamweaver Learn & Support: https://helpx.adobe.com/dreamweaver/tutorials.html (accessible in Dreamweaver by choosing Help > Dreamweaver Tutorial) is where you can find and browse tutorials, help, and support on Adobe.com.

Dreamweaver Help: helpx.adobe.com/support/dreamweaver.html is a reference for application features, commands, and tools (press F1 or choose Help > Dreamweaver Help).

Adobe Forums: forums.adobe.com lets you tap into peer-to-peer discussions and questions and answers on Adobe products.

Resources for educators: adobe.com/education and edex.adobe.com offer a treasure trove of information for instructors who teach classes on Adobe software. You'll find solutions for education at all levels, including free curricula that use an integrated approach to teaching Adobe software and that can be used to prepare for the Adobe Certified Associate exams.

Also check out these useful links:

Adobe Add-ons: exchange.adobe.com/addons is a central resource for finding tools, services, extensions, code samples, and more to supplement and extend your Adobe products.

Adobe Dreamweaver CC product home page: adobe.com/products/dreamweaver.html has more information about the product.

Adobe Authorized Training Centers

Adobe Authorized Training Centers offer instructor-led courses and training on Adobe products. Go to training.adobe.com/training/partner-finder.html to find a directory of AATCs.

1 Customizing your Workspace

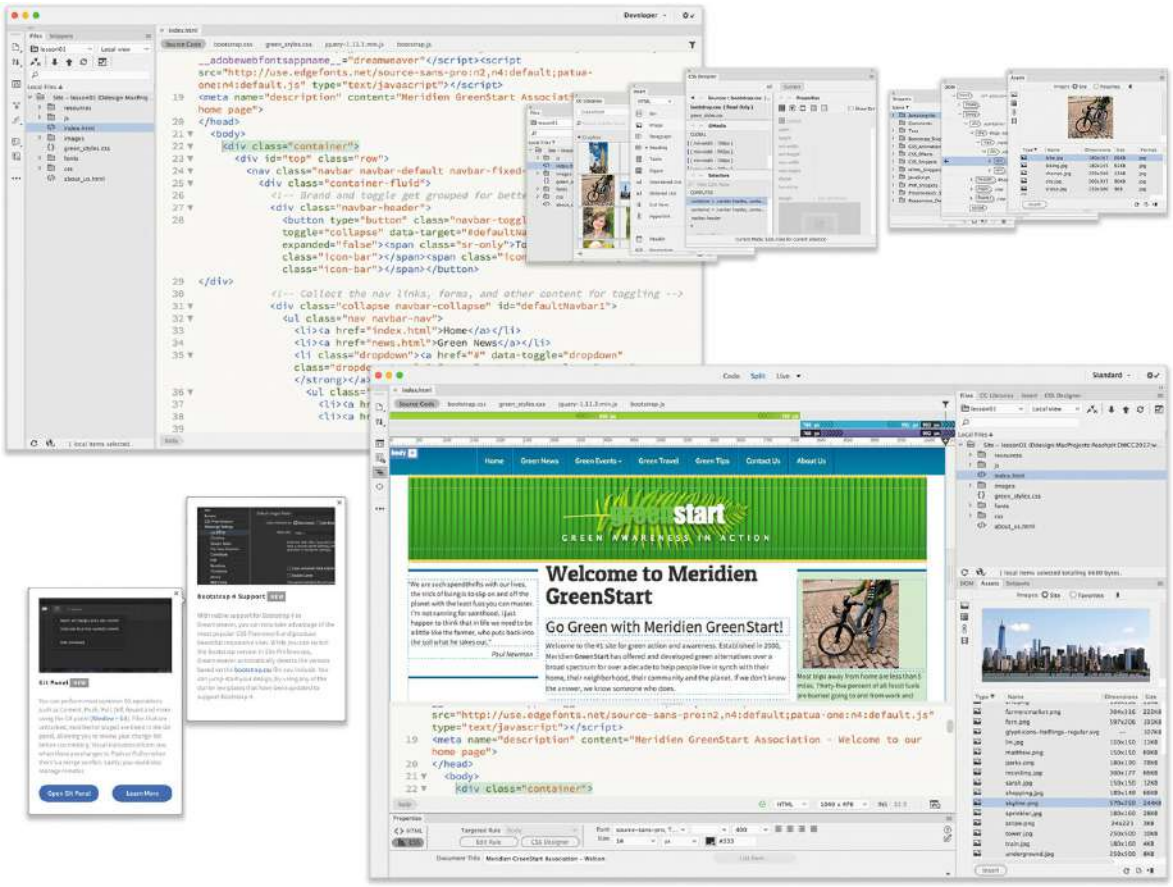
Lesson overview

In this lesson, you'll familiarize yourself with the Dreamweaver CC (2019 release) program interface and learn how to do the following:

- Use the program Welcome screen
- Switch document views
- Work with panels
- Select a workspace layout
- Adjust toolbars
- Personalize preferences
- Create custom keyboard shortcuts
- Use the Property inspector
- Use the Extract workflow

This lesson will take about 1 hour to complete. Please log in to your account on peachpit.com to download the files for this lesson, or go to the "Getting Started" section at the beginning of this book and follow the instructions under "Accessing the Lesson Files and Web Edition." Store the files on your computer in a convenient location. Define a site based on the lesson01 folder.

Your Account page is also where you'll find any updates to the lesson files. Look on the Lesson & Update Files tab to access the most current content.



You'd probably need a dozen programs to perform all the tasks that Dreamweaver can do—and none of them would be as fun to use.

Touring the workspace

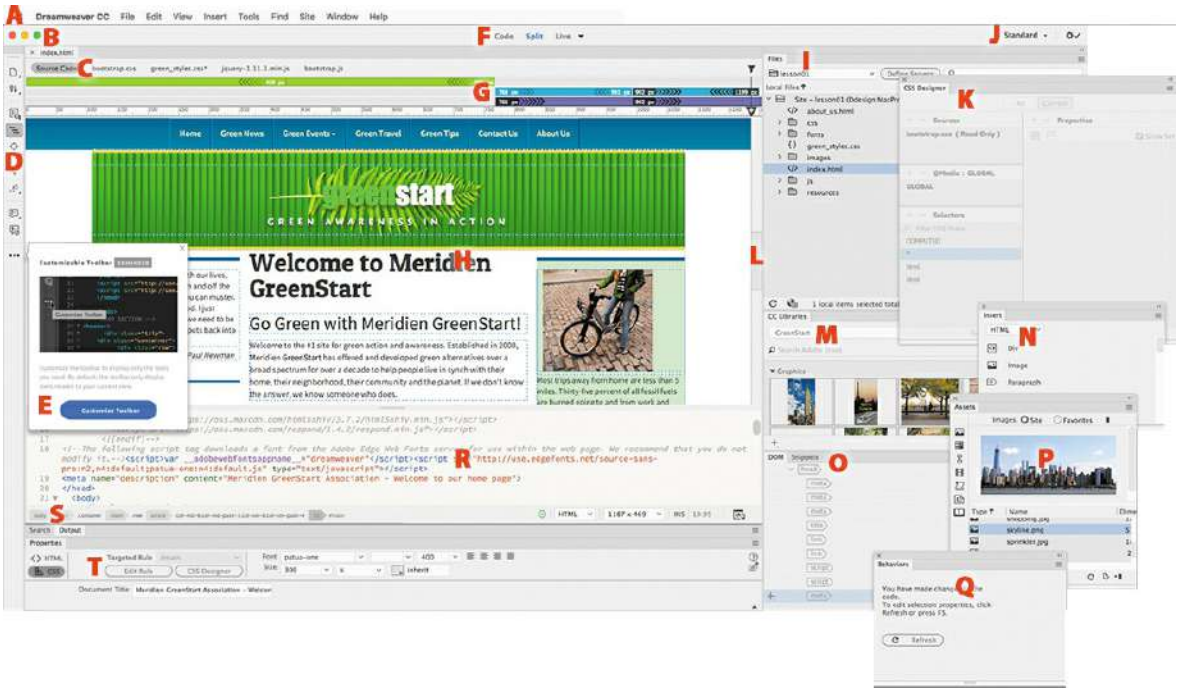
Dreamweaver is the industry-leading Hypertext Markup Language (HTML) editor, with good reasons for its popularity. The program offers an incredible array of design and code-editing tools. Dreamweaver offers something for everyone.

Note

Before you begin this lesson, download the lesson files and create a new website for lesson01 as described in the "Getting Started" section at the beginning of the book.

Coders love the range of enhancements built into the Code view environment, and developers enjoy the program's support for a variety of programming languages and code hinting. Designers marvel at seeing their text and graphics appear in an accurate What You See Is What You Get (WYSIWYG) depiction as they work, saving hours of time previewing their designs in browsers. Novices certainly appreciate the program's simple-to-use and power-packed interface. No matter

what type of user you are, if you use Dreamweaver, you don't have to compromise.



The Dreamweaver interface features a vast array of user-configurable panels and toolbars. Take a moment to familiarize yourself with the names of these components.

A. Menu bar

B. Document tab

C. Related files interface

D. Common toolbar

E. New Features

F. Document toolbar

G. Visual Media Query (VMQ) interface

H. Live/Design views

I. Files panel

J. Workspace menu

K. CSS Designer

L. Scrubber

M. CC Libraries panel

N. Insert panel

O. DOM panel

P. Assets panel

Q. Behaviors panel

R. Code view

S. Tag selectors

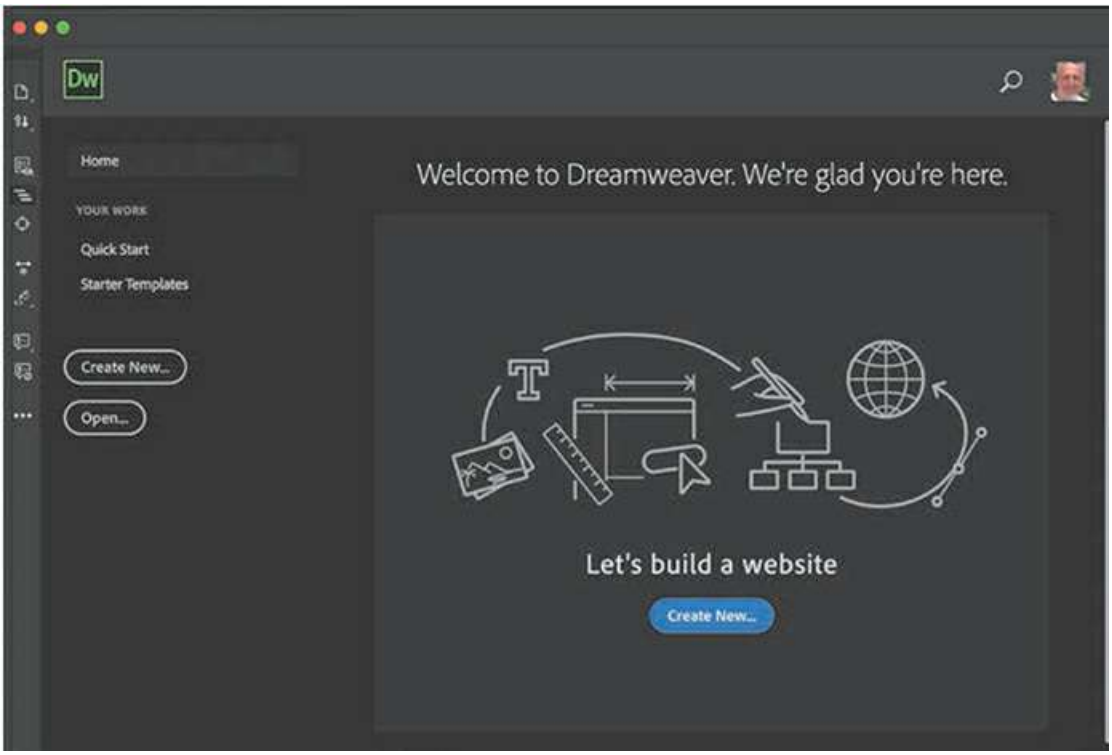
T. Property inspector

You'd think a program with this much to offer would be dense, slow, and unwieldy, but you'd be wrong. Dreamweaver provides much of its power via dockable panels and toolbars that you can display or hide and arrange in innumerable combinations to create your ideal workspace. In most cases, if you don't see a desired tool or panel, you'll find it in the Window menu.

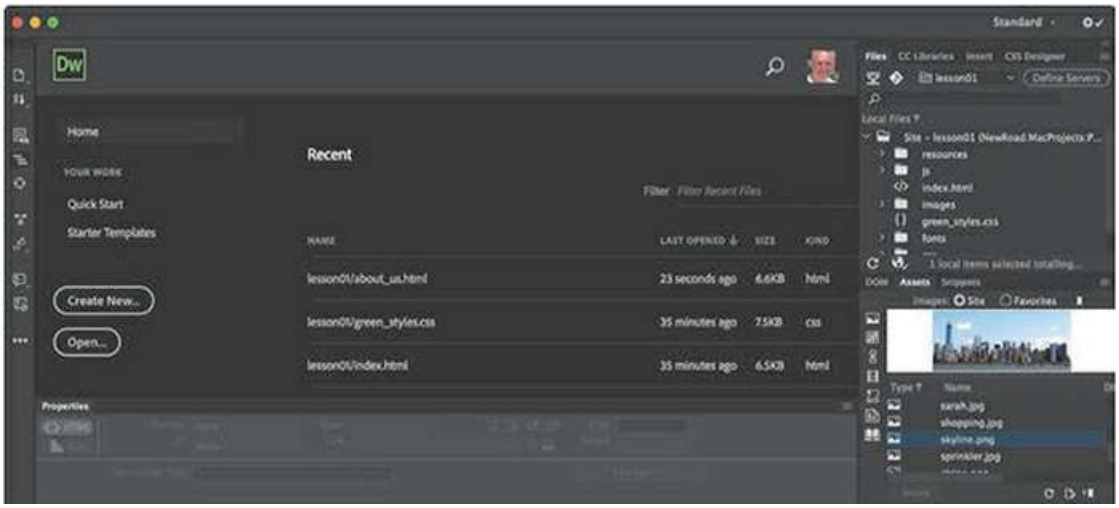
This lesson introduces you to the Dreamweaver interface and gets you in touch with some of the power hiding under the hood. We don't spend a lot of time in the upcoming lessons teaching you how to perform basic activities within the interface; that's the intention of this lesson. So take some time to go through the following descriptions and exercises to familiarize yourself with the basic operations of the program interface. Feel free to refer to this lesson anytime you need a refresher on the program's many dialogs and panels and how they function.

Using the Start Screen

Once the program is installed and the initial setup is completed, you'll see the new Dreamweaver Start Screen. This screen provides quick access to recent pages, easy creation of a variety of page types, and a direct connection to several key Help resources. The Start Screen appears when you first start the program or when no other documents are open. The Start Screen has gotten a facelift in this version of Dreamweaver and deserves a quick review to check out what it offers. For example, it now has two main options: Quick Start and Starter Templates and two buttons for creating new files and opening existing ones. If you have never used the program before the center of the Start Screen may prompt you "Build a Website."

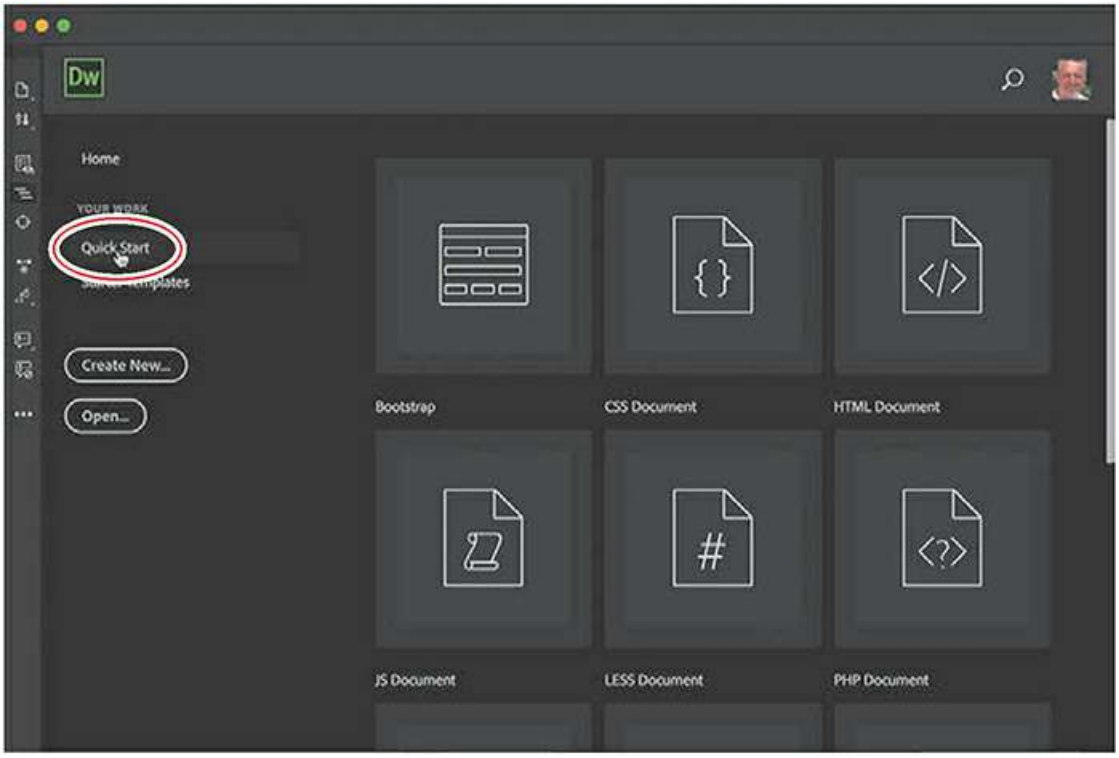


Once you create or open your first file, Dreamweaver will provide a list of the files you last worked on. The list is dynamic, and the last file you worked on will be at the top of the list. To reopen a file, simply click its name.



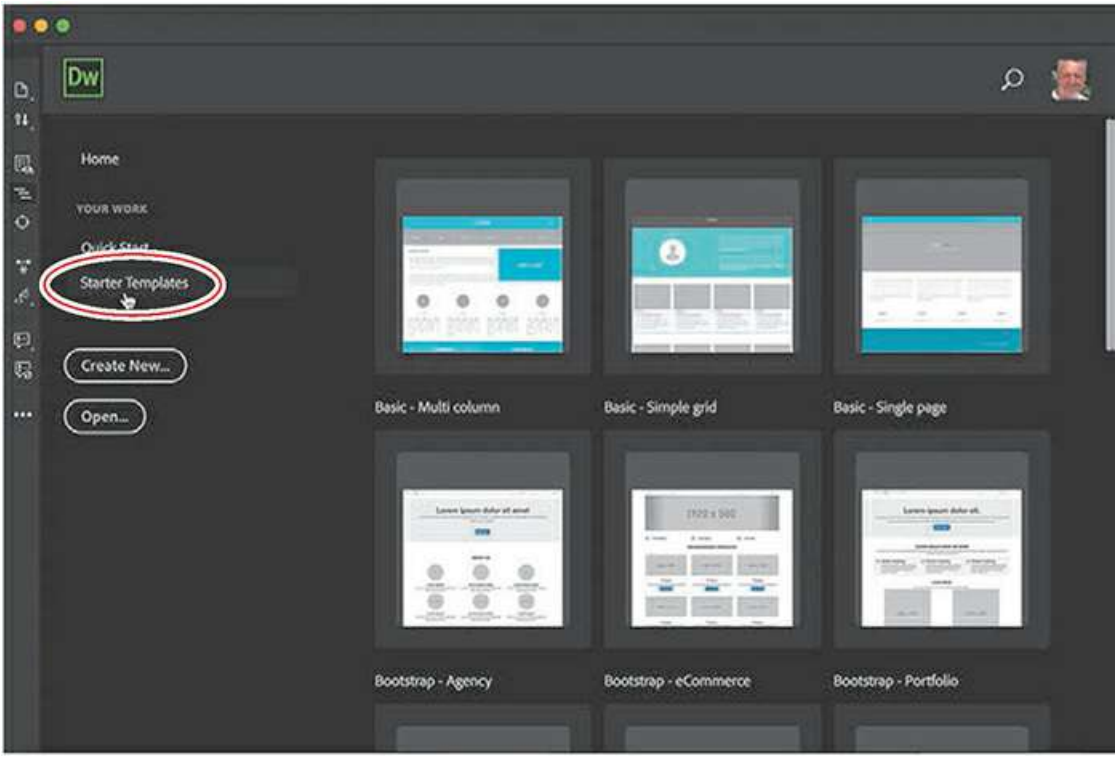
Quick Start

If the Quick Start tab looks familiar, it’s because it has been around in one form or another for many versions of Dreamweaver. As it has always done, it provides instant access to a list of basic web-compatible file types, such as HTML, CSS, JS, PHP, and so on. Just click the file type to start a new document.



Starter Templates

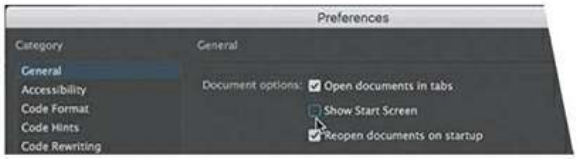
The Starter Templates option enables you to access predefined starter templates that provide responsive styling to support smartphones and mobile devices, as well as the popular Bootstrap framework. The templates can be used to quickly create from scratch a variety of webpages that are already compatible with smartphones and tablets.



Create New and Open

The Create New and Open buttons access the New Document and Open dialogs, respectively. Previous users of Dreamweaver may be more comfortable using these options, which open familiar interfaces for creating new documents or opening existing ones.

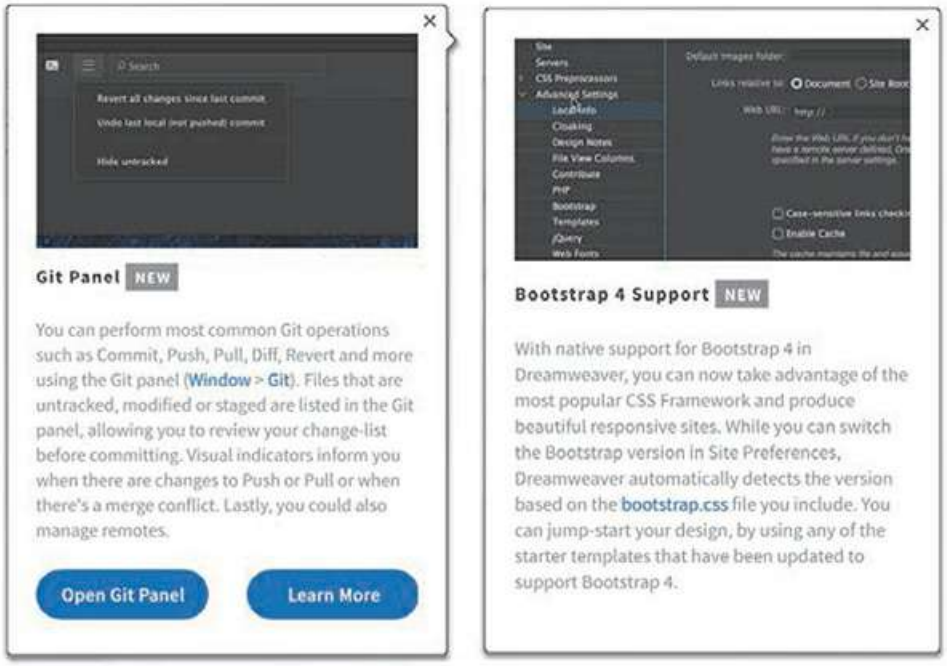
If you do not want to see the Start Screen anymore, you can disable it by accessing the option in General settings in Dreamweaver Preferences and deselecting the checkbox.



Exploring New Feature guides

In Dreamweaver CC, the New Feature guides pop up from time to time as you access various tools, features, or interface options. The pop-ups call your attention to new features or workflows

that have been added to the program and provide handy tips to help you get the most out of them.



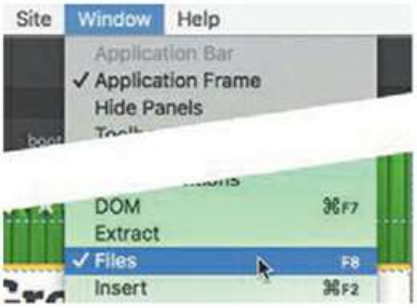
When a tip appears, it may provide more extensive information or a tutorial you can access by following the prompts in the pop-up window. When you are finished, you can close the pop-up by clicking the Close icon in the upper-right corner of each tip. When you close the tip, it will not appear again. You can display the tips again by selecting Help > Reset Contextual Feature Tips.

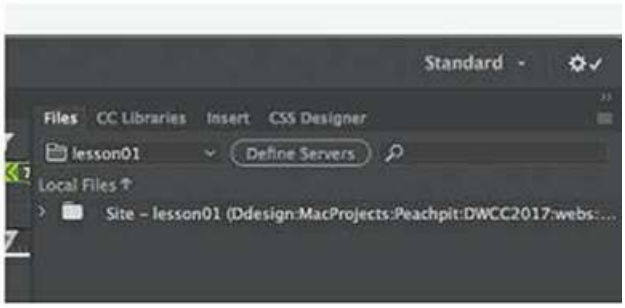
Setting interface preferences

Dreamweaver provides users with extensive controls over the basic program interface. You can set up, arrange, and customize the various panels and menus to your own liking. One of the first places you should visit before you begin the lessons in this book is the Dreamweaver Preferences dialog.

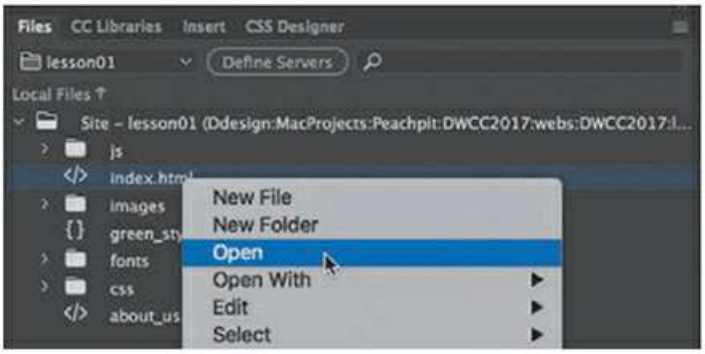
As with other Adobe applications, the Preferences dialog provides specific settings and specifications that dictate how the program looks and functions. Preference settings are normally persistent, meaning that they remain in effect even after the program is shut down and relaunched. There are far too many options in this dialog to cover in one lesson, but let's make a couple of changes to give you a taste of what you can do. Some features of the program are not visible until you create or open a file for editing.

- Define a new site based on the lesson01 folder as described in the "Getting Started" section at the beginning of the book.
- Select Window > Files or press F8 to display the Files panel.



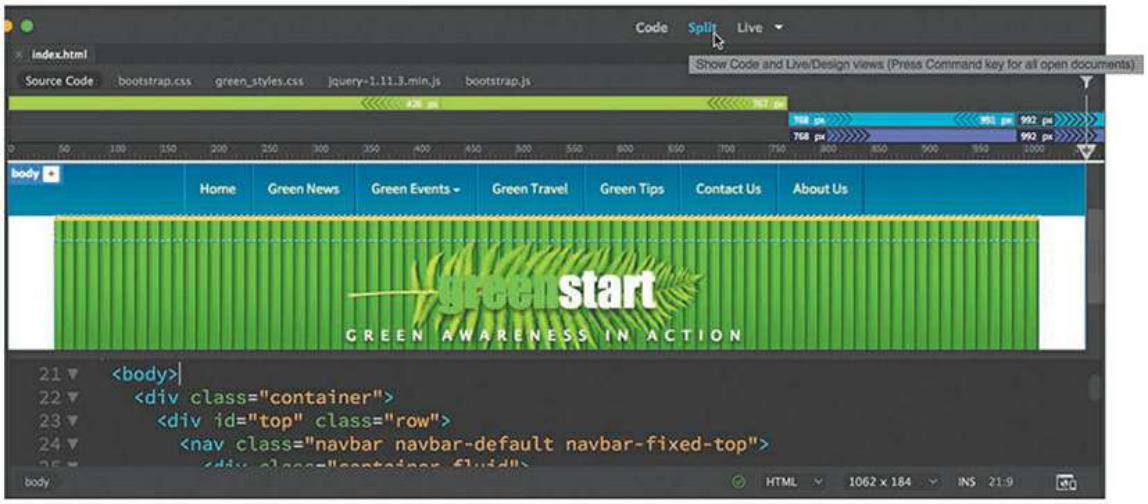


- In the File panel, select **lesson01** from the drop-down menu and reveal the site file list in the panel, if necessary.
- Right-click the file **index.html** from the lesson01 folder and choose Open from the context menu. You can also double-click a file in the list to open it.



The file opens in the document window. If the program has not been used previously, the file may open in Live view. To get the full appreciation of the upcoming changes, let's also display the code-editing interface at the same time.

- Select Split view at the top of the document window, if necessary.

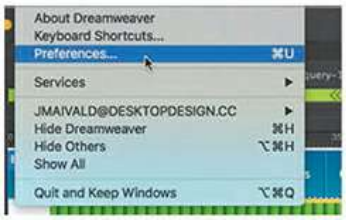


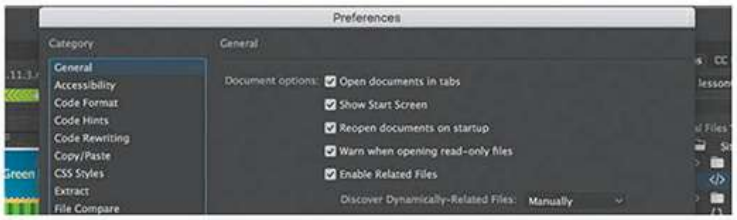
If you did not complete the "Getting Started" section at the beginning of the book, you will see

that Dreamweaver is sporting a new dark color scheme in Code view. Some users love it; some hate it. You can change it completely or merely tweak it in Preferences. If you have already changed your interface theme, skip to the next exercise.

. In Windows, select Edit > Preferences.

In macOS, select Dreamweaver CC > Preferences.





The Preferences dialog appears.

. Select the Interface category.

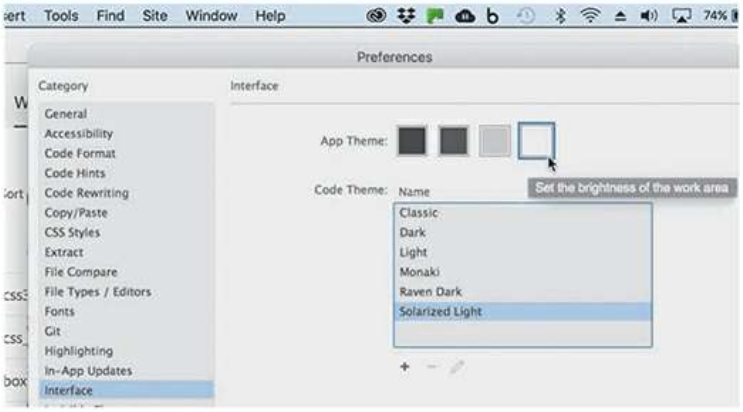
As you see in the dialog, Dreamweaver gives you control over the overall color theme as well as the code-editing window. You may change one or both.

Many designers work in controlled lighting environments and prefer the dark interface themes that are now the default in most Adobe applications. For the purposes of this book, all screen shots from this point forward are taken in the lightest theme. This saves ink during printing, for less impact on the environment. You may continue to use the dark theme if you prefer, or switch now so that your screen will match the illustrations in the book.

. In the App Theme window, select the lightest theme.

The theme of the entire interface changes to light gray. You will notice that the Code Theme setting changes to *Light* at the same time. If you prefer, you can switch the code theme back to *Dark* or choose another. The screen shots for code editing in the book use the *Solarized Light* theme.

. Select *Solarized Light* in the Code Theme window.



At the moment, the changes are not permanent. If you click the Close button in the dialog, the theme would revert to dark.

- Click Apply in the lower-right corner of the dialog.

  The changes have now been applied permanently.
- Click Close.

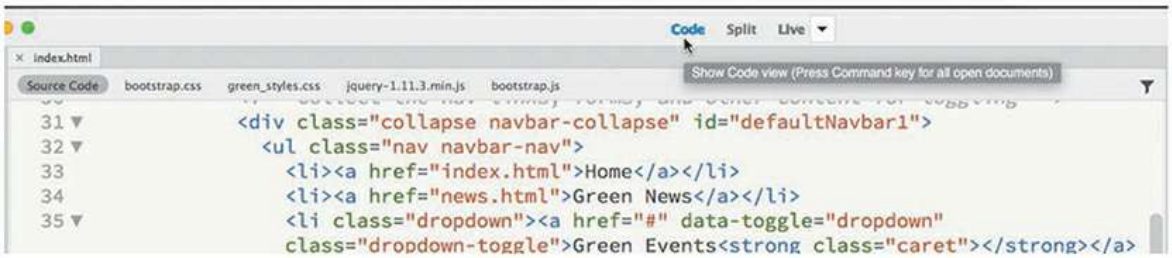
Saved preferences persist from session to session and through each workspace.

Switching and splitting views

Dreamweaver offers dedicated environments for coders and designers.

Code view

Code view focuses the Dreamweaver workspace exclusively on the HTML code and a variety of code-editing productivity tools. To access Code view, click the Code view button in the Document toolbar.

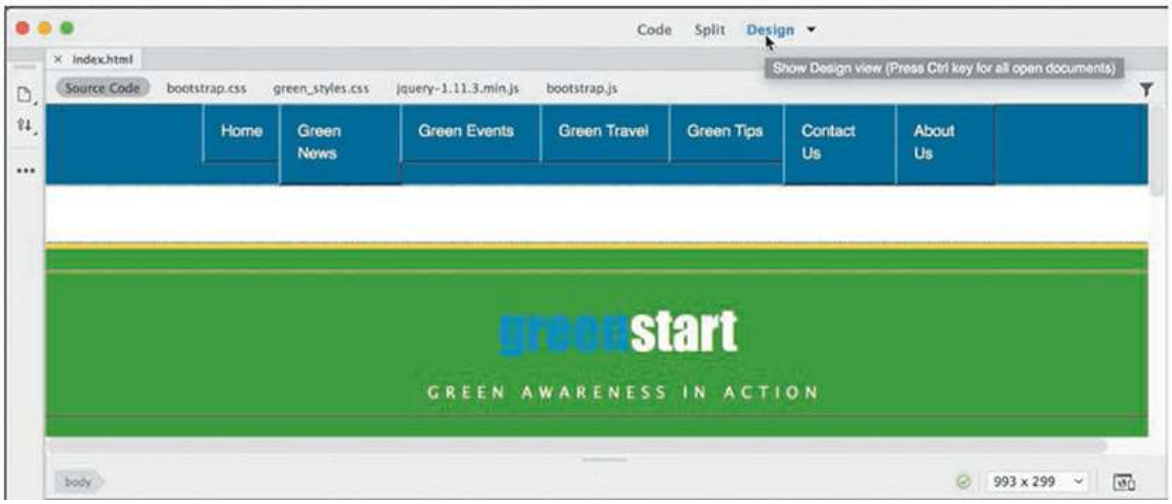


Code view

Design view

Design view shares the document window with Live view and focuses the Dreamweaver workspace on its classic WYSIWYG editor. In the past, Design view provided a reasonable facsimile of the webpage as it would appear in a browser, but with the advancements in CSS and HTML, it is no longer as WYSIWYG as it once was. Although it can be difficult to use in some situations, you'll find that it does offer an interface that speeds up the creation and editing of your content. And at the moment, it's also the only way to access certain Dreamweaver tools or workflows, as you will see in the upcoming lessons.

To activate Design view, choose it from the Design/Live view drop-down menu in the Document toolbar. Most HTML elements and basic cascading style sheets (CSS) formatting will be rendered properly within Design view, with the major exceptions being CSS3 properties; dynamic content; interactivity, such as link behaviors, video, audio, and jQuery widgets; and some form elements. In previous versions of Dreamweaver, you spent most of your time in Design view. That will no longer be the case.



Design view

Live view

Live view is the default workspace of Dreamweaver CC. It speeds up the process of developing modern websites by allowing you to *visually* create and edit webpages and web content in a browser-like environment, and it supports and previews most dynamic effects and interactivity.

To use Live view, choose it from the Design/Live view drop-down menu in the Document toolbar. When Live view is activated, most HTML content will function as it would in an actual browser, allowing you to preview and test most dynamic applications and behaviors.

In previous versions of Dreamweaver, the content in Live view was not editable. That has changed. You can edit text, add and delete elements, create classes and IDs, and even style elements, all in the same window. It’s like working on a live webpage right inside Dreamweaver.

Live view is integrally connected to the CSS Designer, allowing you to create and edit advanced CSS styling and build fully responsive webpages without having to switch views or waste time previewing the page in a browser.

Live view

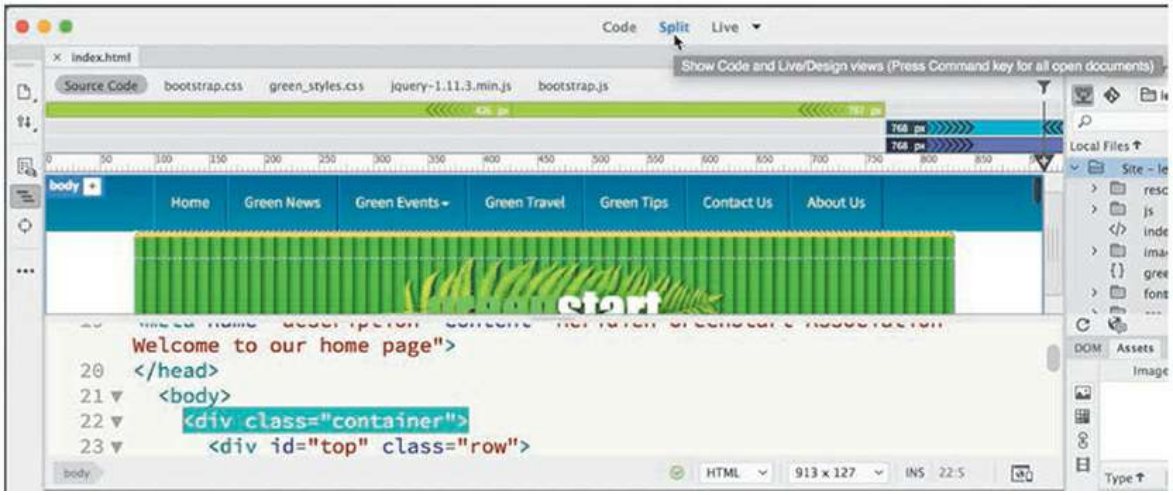
Split view

Split view provides a composite workspace that gives you access to both the design and the code simultaneously. Changes made in either window update in the other in real time.

Note

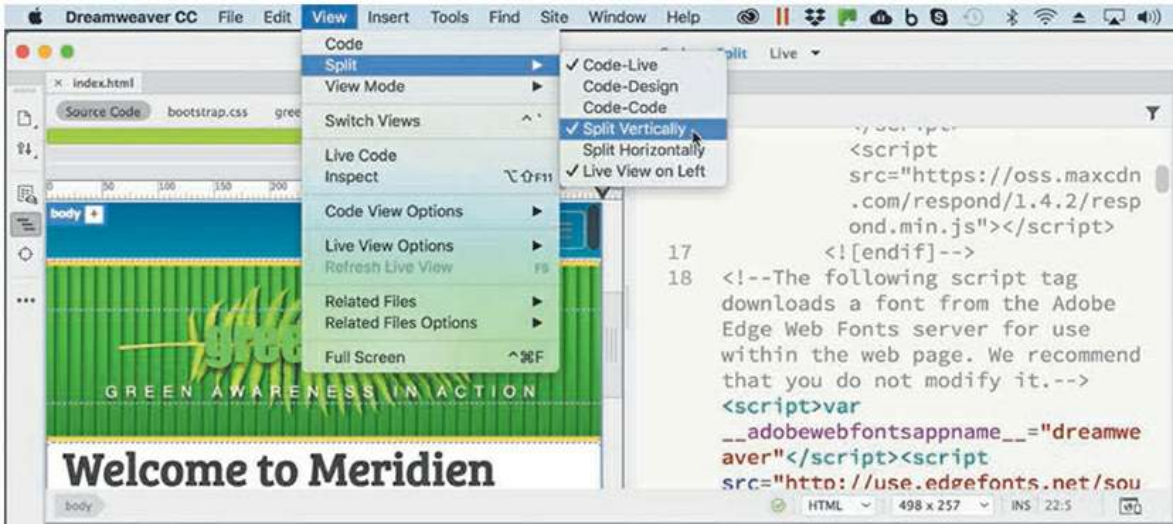
Split view can pair Code view with either Design view or Live view.

To access Split view, click the Split view button in the Document toolbar. Dreamweaver splits the workspace horizontally by default. When using Split view, you can display Code view with either Live view or Design view.



Split view (horizontal)

You can split the screen vertically by selecting the Split Vertically option on the View menu. When the window is split, Dreamweaver also gives you options for how the two windows display. You can put the code window on the top, bottom, left, or right. You can find all these options in the View menu. Most screen shots in the book that show Split view show Design or Live view at top or right.



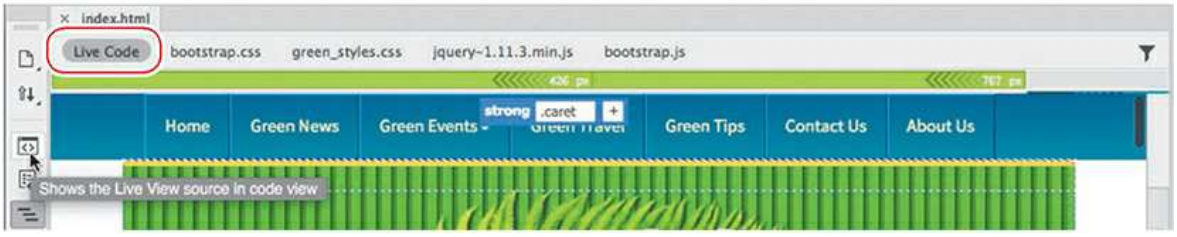
Split view (vertical)

Live Source Code

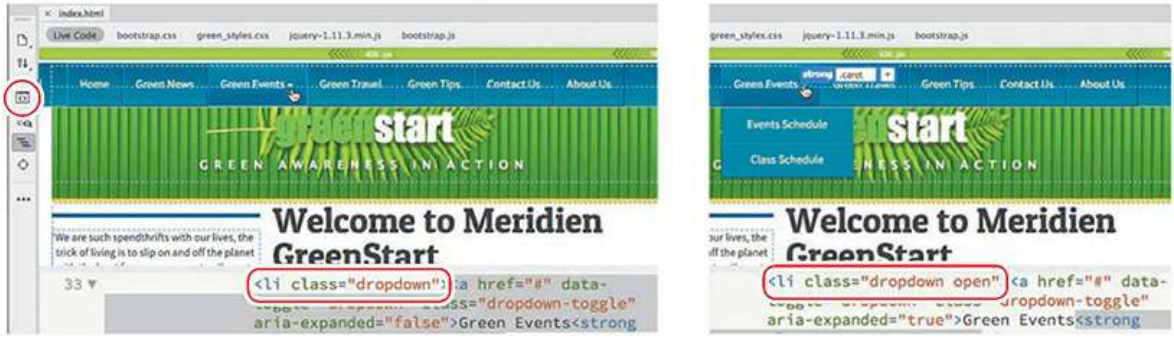
Live Source Code is an HTML code-troubleshooting display mode available whenever Live view is activated. To access Live Source Code, activate Live view and then click the Live Source Code icon in the toolbox at the left side of the document window. While active, Live Source Code displays the HTML code as it would appear in a live browser on the Internet and gives you a peek at how the code changes when the visitor interacts with various parts of the page.

Note

The Live Source Code icon may not appear on the Common toolbar by default. You may need to activate it using the Customize Toolbar icon.



You can see this interaction firsthand by clicking the *Green Events* menu item to open the drop-down menu. In Code view, you will see that the menu item starts with the class `dropdown` and that the class `open` is then added to the code interactively. The `open` class is then removed when you close the menu. Without Live Source Code, you would not be able to see this interaction and behavior.



Live Source Code mode

Be aware that while Live Source Code is active, you will not be able to edit the HTML code, although you can still modify external files, such as linked style sheets. To disable Live Source Code, click the Live Source Code icon again to toggle the mode off.

Inspect mode

Inspect mode is a CSS troubleshooting display mode that is available whenever Live view is activated. It is integrated with the CSS Designer and allows you to rapidly identify CSS styles applied to content within the page by moving the mouse cursor over elements in the webpage. Clicking an element freezes the focus on that item.

The Live view window highlights the targeted element and displays the pertinent CSS rules applied or inherited by that element. You can access Inspect mode at any time by clicking the Live view icon whenever an HTML file is open and then clicking the Inspect icon in the Common toolbar.

Inspect mode

Selecting a workspace layout

A quick way to customize the program environment is to use one of the prebuilt workspaces in Dreamweaver. Experts have optimized these workspaces to put the tools you need at your

fingertips.

Dreamweaver CC (2019 release) includes two prebuilt workspaces: *Standard* and *Developer*. To access these workspaces, choose them from the Workspace menu, located at the upper-right side of the program window.

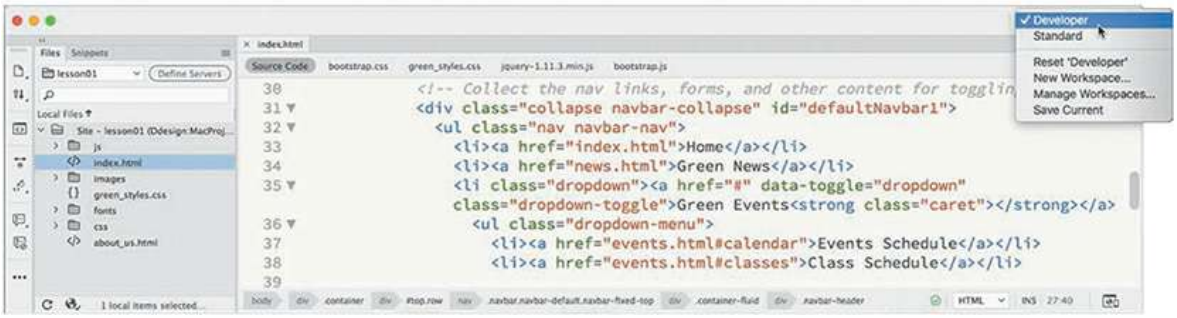
Standard workspace

The Standard workspace focuses the available screen real estate on the Design and Live view window. Standard is the default workspace for screen shots in this book.

Standard workspace

Developer workspace

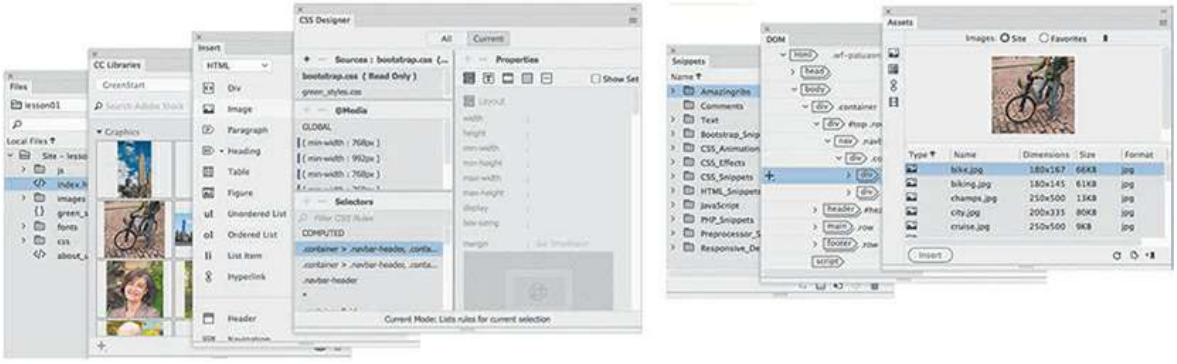
The Developer workspace provides a code-centric layout of tools and panels ideal for coders and programmers. The workspace is focused on Code view.



Developer workspace

Working with panels

Although you can access most commands from the menus, Dreamweaver scatters much of its power in user-selectable panels and toolbars. You can display, hide, arrange, and dock panels at will around the screen. You can even move them to a second or third computer display if you desire.



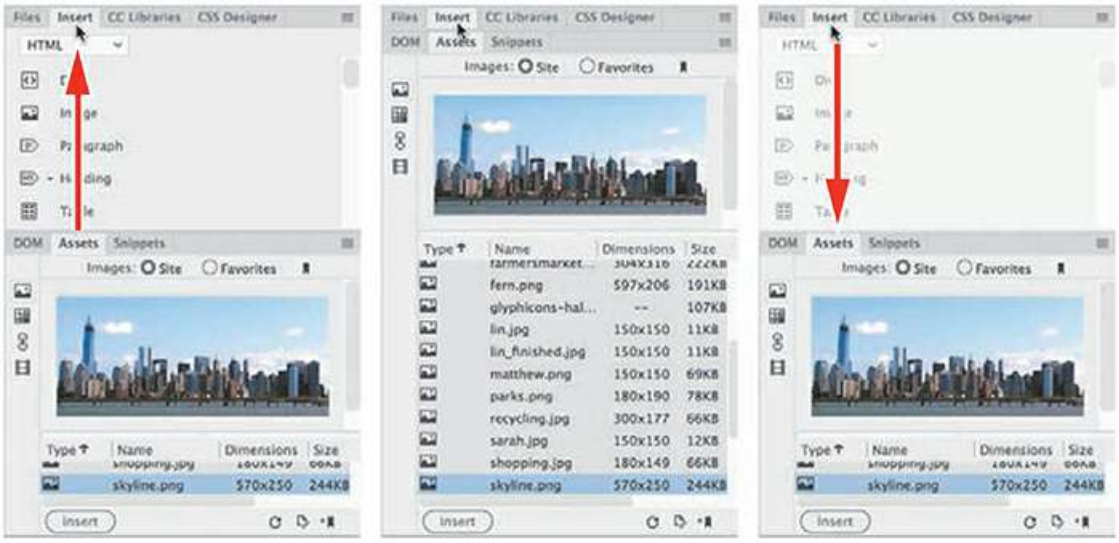
Standard panel grouping

The Window menu lists all the panels available in the program. If you do not see a desired panel on the screen, choose it from the Window menu. A check mark appears next to its name in the menu to indicate that the panel is open and visible. Occasionally, one panel may lie behind another on the screen and be difficult to locate. In such situations, simply choose the desired panel from the Window menu and the panel will rise to the top of the stack.

Minimizing panels

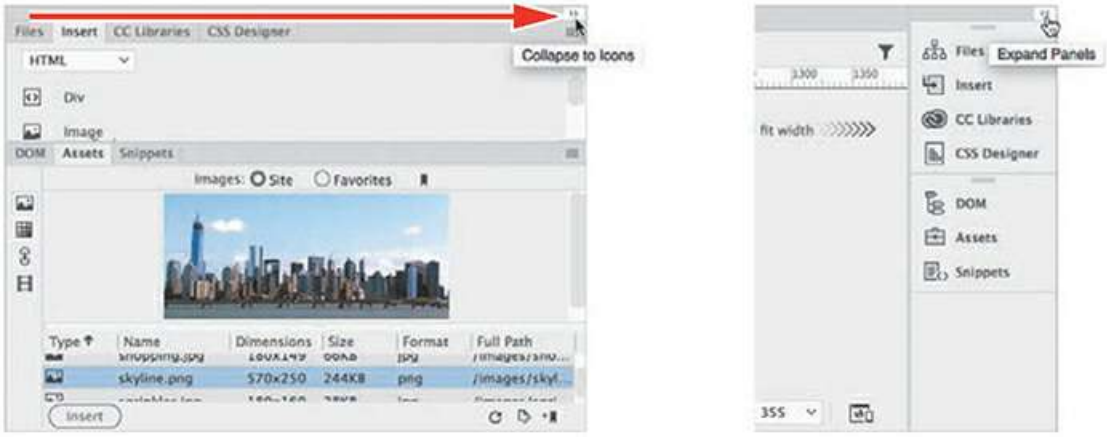
To create room for other panels or to access obscured areas of the workspace, you can minimize or expand individual panels in place. To minimize a standalone panel, double-click the tab containing the panel name. To expand the panel, click the tab once.

Minimizing a panel by double-clicking its tab



Minimizing one panel in a stack using its tab

To recover more screen real estate, you can minimize panel groups or stacks down to icons by double-clicking the title bar. You can also minimize the panels to icons by clicking the double-arrow icon in the panel title bar. When panels are minimized to icons, you access an individual panel by clicking its icon. The selected panel will appear on the left or right of the icon, wherever room permits.



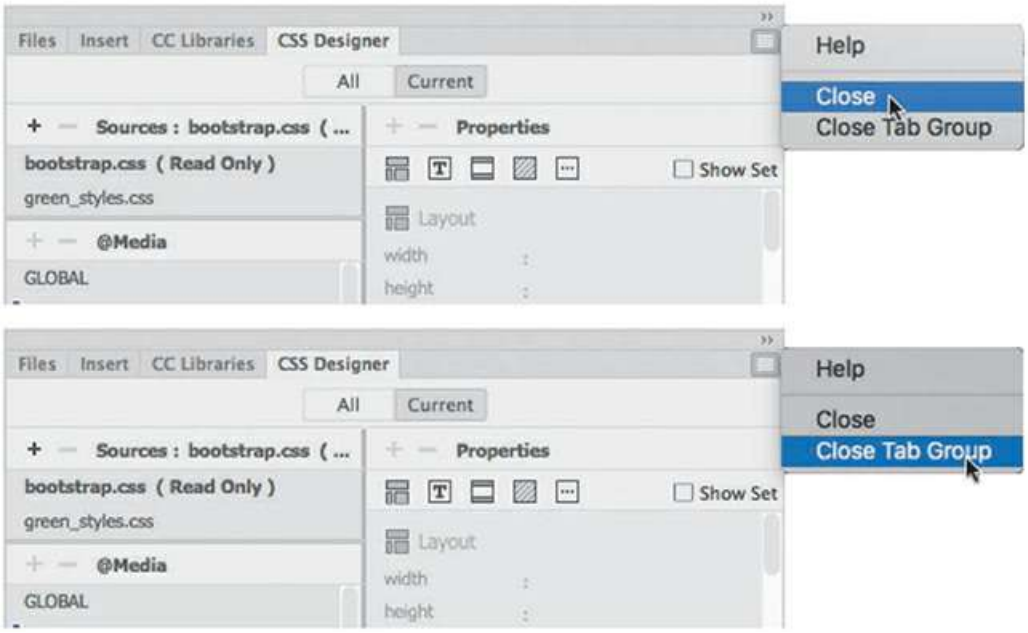
Collapsing a panel to icons

Closing panels and panel groups

Each panel or panel group may be closed at any time. You can close a panel or panel group in several ways; the method often depends on whether the panel is floating, docked, or grouped with another panel.

To close an individual panel that is docked, right-click in the panel tab and choose Close from the context menu. To close an entire group of panels, right-click any tab in the group and choose

Close Tab Group.



To close a floating panel or panel group, click the Close ✖ icon that appears in the upper-right corner of the panel in Windows or in the left corner of the title bar of the panel or panel group in macOS. To reopen a panel, choose the panel name from the Window menu. Reopened panels will sometimes appear floating in the interface. You may use them this way or attach, or dock, them to the sides, top, or bottom of the interface. You will learn how to dock panels later.

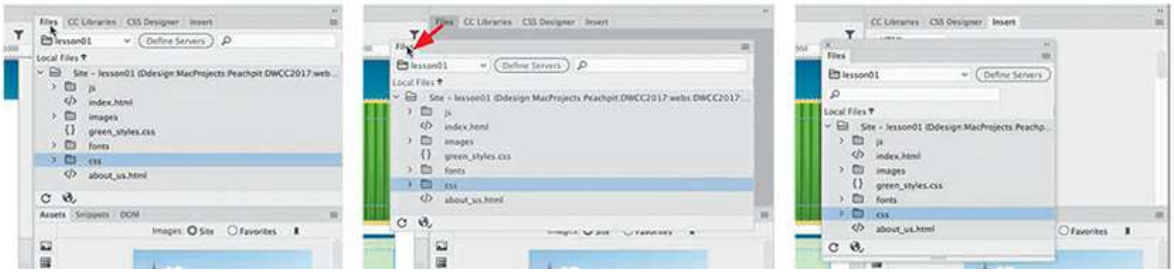
Dragging

You can reorder a panel tab by dragging it to the desired position within the group.

Dragging a tab to change its position

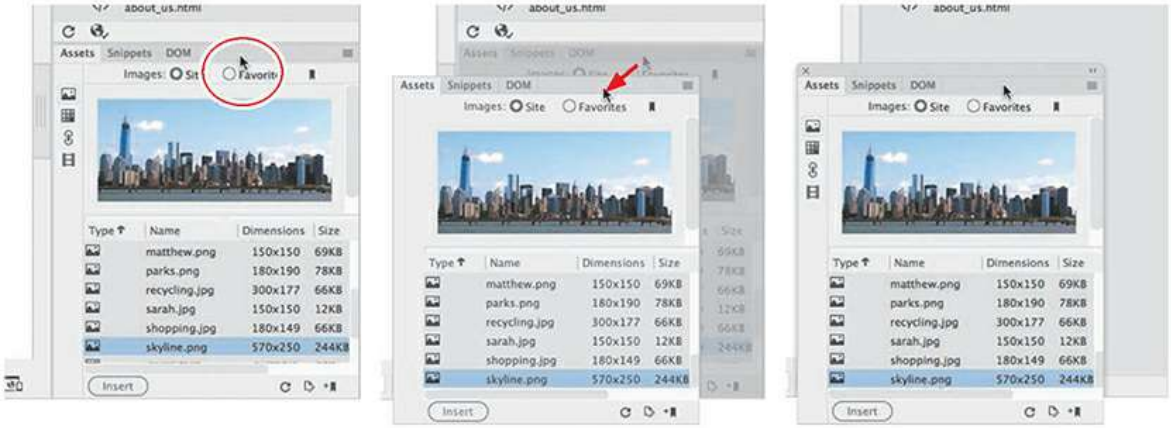
Floating

A panel that is grouped with other panels can be floated separately. To float a panel, drag it from the group by its tab.



Pulling a panel out by its tab

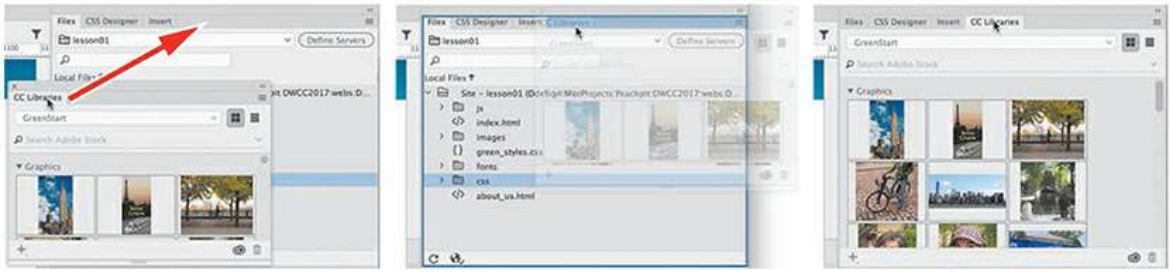
To reposition panels, groups, and stacks in the workspace, simply drag them by the title bar. To pull out a single panel group when it’s docked, grab it by the tab bar.



Dragging a whole docked panel group to a new position

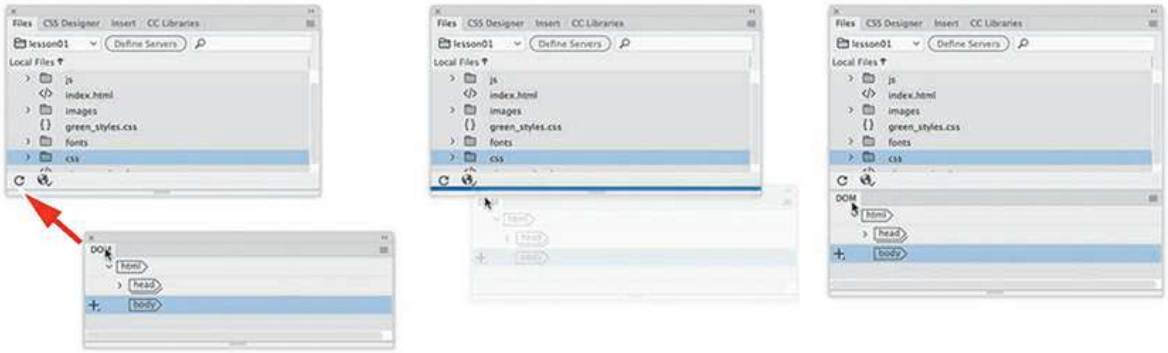
Grouping, stacking, and docking

You can create custom groups by dragging one panel into another. When you’ve moved the panel to the correct position, Dreamweaver highlights the area, called the *drop zone*, in blue. Release the mouse button to create the new group.



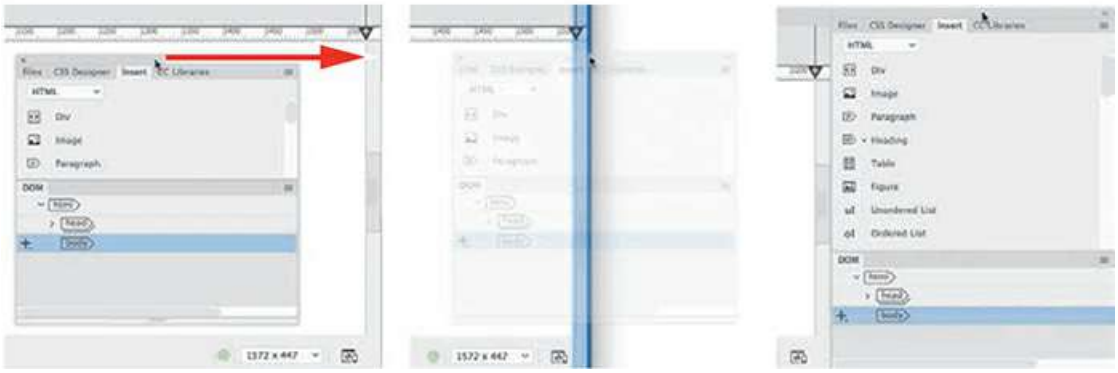
Creating new groups

In some cases, you may want to keep both panels visible simultaneously. To stack panels, drag the desired tab to the top or bottom of another panel. When you see the blue drop zone appear, release the mouse button.



Creating panel stacks

Floating panels can be docked to the right, left, or bottom of the Dreamweaver workspace. To dock a panel, group, or stack, drag its title bar to the edge of the window on which you wish to dock it. When you see the blue drop zone appear, release the mouse button.



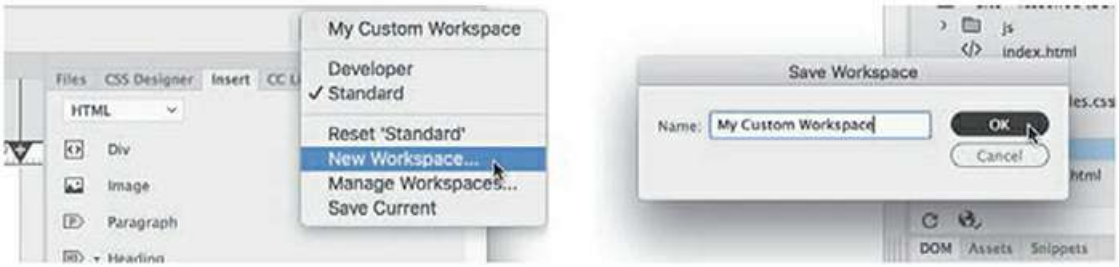
Docking panels

Personalizing Dreamweaver

As you continue to work with Dreamweaver, you'll devise your own optimal workspace of panels and toolbars for each activity. You can store these configurations in a custom workspace of your own naming.

Saving a custom workspace

To save a custom workspace, create your desired configuration of panels, choose New Workspace from the Workspace menu, and then give it a custom name.

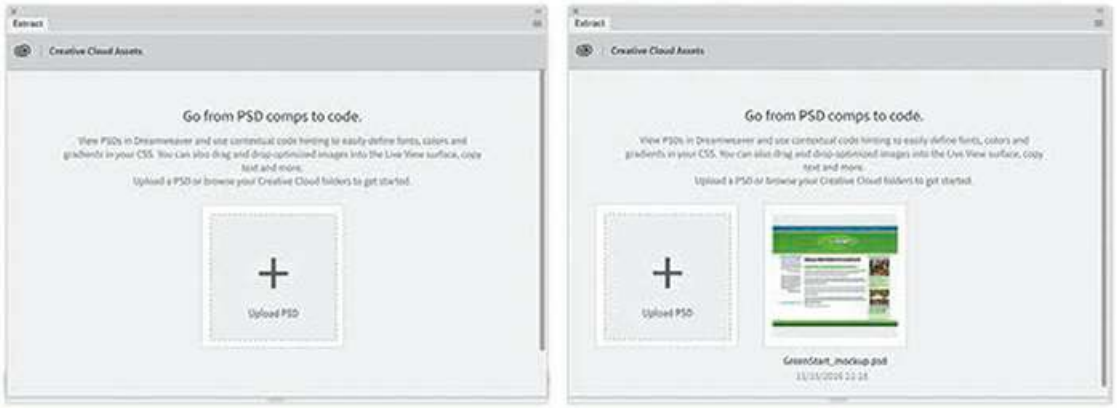


Saving a custom workspace

Working with Extract

Extract is a newer workflow that allows you to create CSS styles and image assets from a Photoshop-based mockup. You can create your webpage design using text and linked or embedded image layers and post the file to Creative Cloud, where Dreamweaver can access the styles, colors, and images to help you build your basic site design.

Build your design in Photoshop using text, images, and effects stored in layers.



Post your file to your Creative Cloud online folder right inside Dreamweaver.

Access the various layers from the Extract panel inside Dreamweaver, copy styles and text, and even download image assets.

Try these features yourself by uploading **GreenStart_mockup.psd**, in the lesson01 resources folder, to your Creative Cloud account online folder. Go to helpx.adobe.com/creative-cloud/help/sync-files.html to learn how to upload files to your Creative Cloud account. In Lesson 5, "Creating a Page Layout," you will learn how to extract CSS styling and image assets from a Photoshop mockup to build a layout for your site template.

Working with toolbars

Some program features are so handy you may want them to be available all the time in toolbar form. Two of the toolbars—Document and Standard—appear horizontally at the top of the document window. The Common toolbar, however, appears vertically on the left side of the screen. You can display the desired toolbar by choosing it from the Window menu.

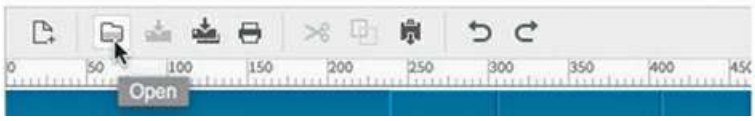
Document toolbar

The document toolbar appears at the very top of the program interface and provides onscreen commands for switching views from Live, Design, Code, and Split views. You can enable this toolbar by selecting Window > Toolbars > Document when a document is open.

Document toolbar

Standard toolbar

The Standard toolbar is an optional toolbar that appears between the Related Files interface and the document window and provides handy commands for various document and editing tasks, such as creating, saving, or opening documents; copying, cutting, and pasting content; and so on. You can enable this toolbar by selecting Window > Toolbars > Standard when a document is open.

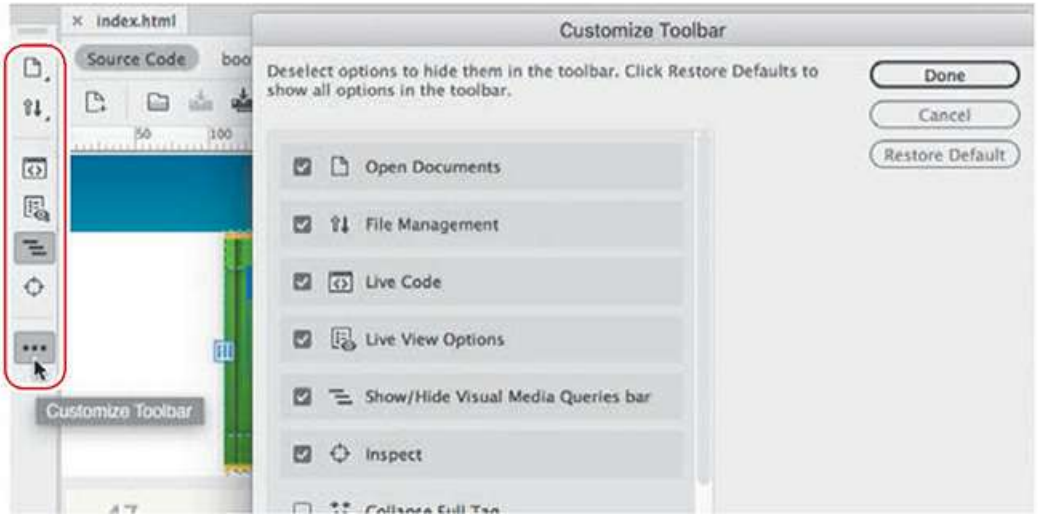


Standard toolbar

Common toolbar

The Common toolbar appears on the left side of the program window and provides a variety of commands for working with code and HTML elements. The toolbar displays six tools by default in Live and Design view. But insert the cursor in the code window and you may see several more.

The Common toolbar was named the Coding toolbar in a previous version of Dreamweaver, and it is now user customizable. You can add and remove tools by selecting the Customize Toolbar icon. Be aware that some tools will be displayed and active only when using the Code view window.



Creating custom keyboard shortcuts

Another powerful feature of Dreamweaver is the ability to create your own keyboard shortcuts as well as edit existing ones. Keyboard shortcuts are loaded and preserved independently of workspaces.

Is there a command you can’t live without that doesn’t have a keyboard shortcut or uses one that’s inconvenient? Create one of your own.

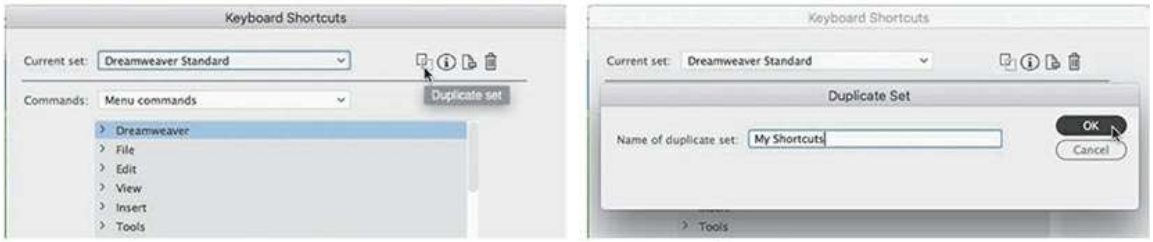
- Choose Edit > Keyboard Shortcuts (Windows) or Dreamweaver CC > Keyboard Shortcuts (macOS).

  You cannot modify the default shortcuts, so you have to create a list of your own.

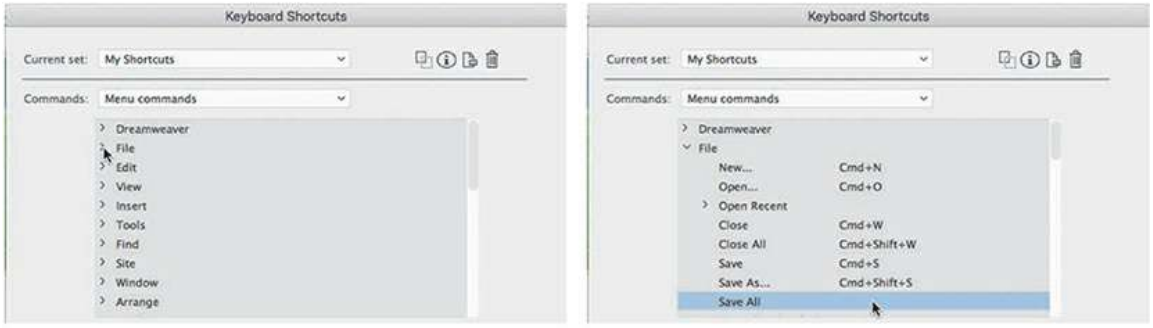
Note

The default keyboard shortcuts are locked and cannot be edited. But you can duplicate the set, save it under a new name, and modify any shortcut within that custom set.

- Click the Duplicate Set icon to create a new set of shortcuts.
- Enter a name in the Name Of Duplicate Set field. Click OK.



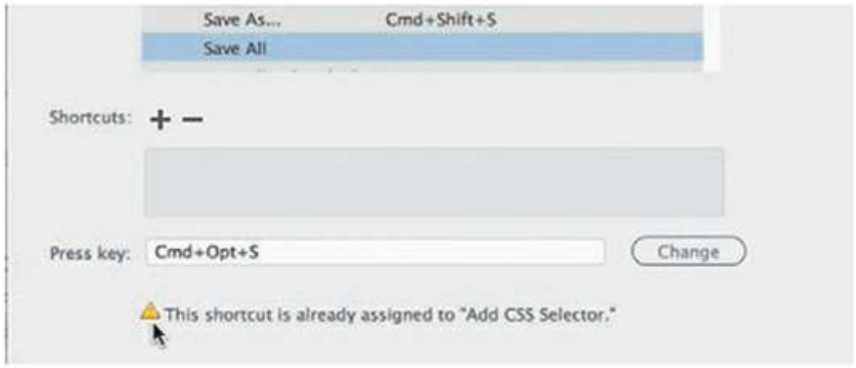
- Choose Menu Commands from the Commands pop-up menu.
- In the Commands window, choose File > Save All.



Note that the Save All command does not have an existing shortcut, although you'll use this command frequently in Dreamweaver.

- Insert the cursor in the Press Key field.

Press Ctrl+Alt+S (Windows) or Cmd+Opt+S (macOS).



Note the error message indicating that the keyboard combination you chose is already assigned to a command. Although we could reassign the combination, let's choose a different one.

- Press Ctrl+Alt+Shift+S (Windows) or Ctrl+Cmd+S (macOS).

This combination is not currently being used, so let's assign it to the Save All command.

- Click the Change button.

The new shortcut is now assigned to the Save All command.

- Click OK to save the change.

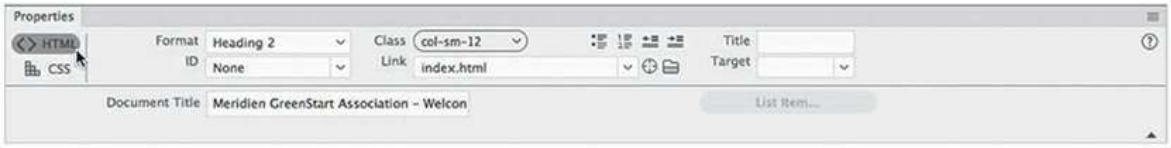
You have created your own keyboard shortcut—one you can use in upcoming lessons. Whenever an instruction in one of the lessons says “save all files,” use this keyboard shortcut.

Using the Property inspector

One tool vital to your workflow is the Property inspector. In the predefined Dreamweaver workspaces, the Property inspector is no longer a default component. If it is not visible in your program interface, you can display it by selecting Window > Properties and then dock it to the bottom of the document window, as described earlier. The Property inspector is context-driven and adapts to the type of element you select.

Using the HTML tab

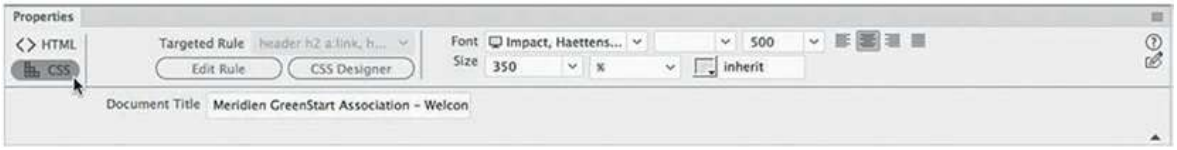
Insert the cursor into any text content on your page and the Property inspector provides a means to quickly assign some basic HTML codes and formatting. When the HTML button is selected, you can apply heading or paragraph tags as well as bold, italics, bullets, numbers, and indenting, among other formatting and attributes. The Document Title metadata field is also available in the bottom half of the Property inspector. Enter your desired document title in this field, and Dreamweaver adds it automatically to the document `<head>` section. If you don’t see the full Property inspector, click the triangle icon in the lower-right corner of the panel to expand its display.



HTML Property inspector

Using the CSS tab

Click the CSS button to quickly access commands to assign or edit CSS formatting.



CSS Property inspector

Accessing image properties

Select an image in a webpage to access the image-based attributes and formatting controls of the Property inspector.

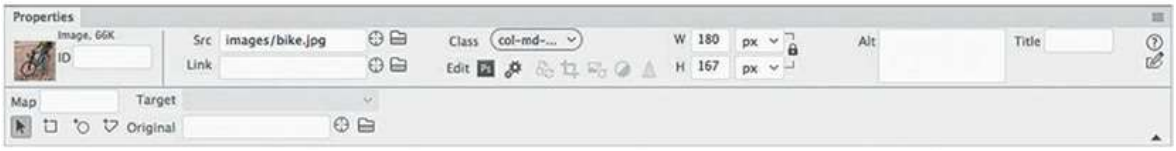


Image Property inspector

Accessing table properties

To access table properties, insert your cursor in a table and then click the table tag selector at the bottom of the document window.

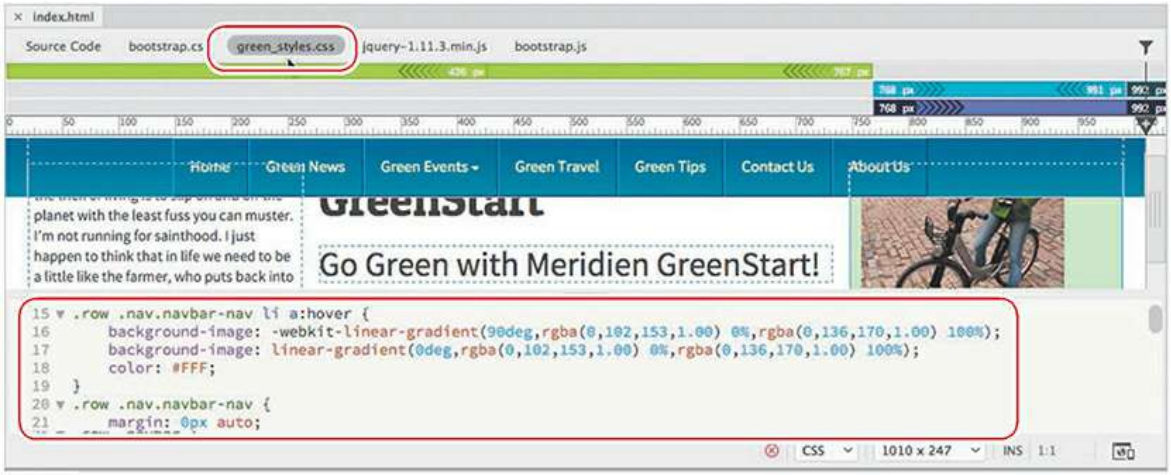
Table Property inspector

Using the Related Files interface

Webpages are often built with multiple external files that provide styling and programming assistance. Dreamweaver enables you to see all the files linked to, or referenced by, the current document by displaying the filenames in the Related Files interface at the top of the document window. This interface displays the name of any external file and will actually display the contents of each file—if the contents are available—when you simply select the filename in the display.

The Related Files interface lists all external files linked to a document.

To view the contents of the referenced file, click the name. Dreamweaver splits the document window and shows the contents of the selected file in the Code view window. If the file is stored locally, you'll even be able to edit the contents of the file when it's selected.



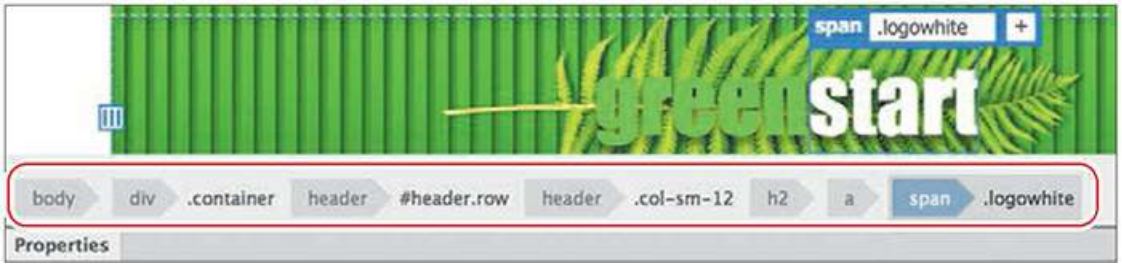
Use the Related Files interface to edit locally stored files linked to the webpage.

To view the HTML code contained within the main document, click the Source Code option in the interface.

Choose the Source Code option to see the contents of the main document.

Using tag selectors

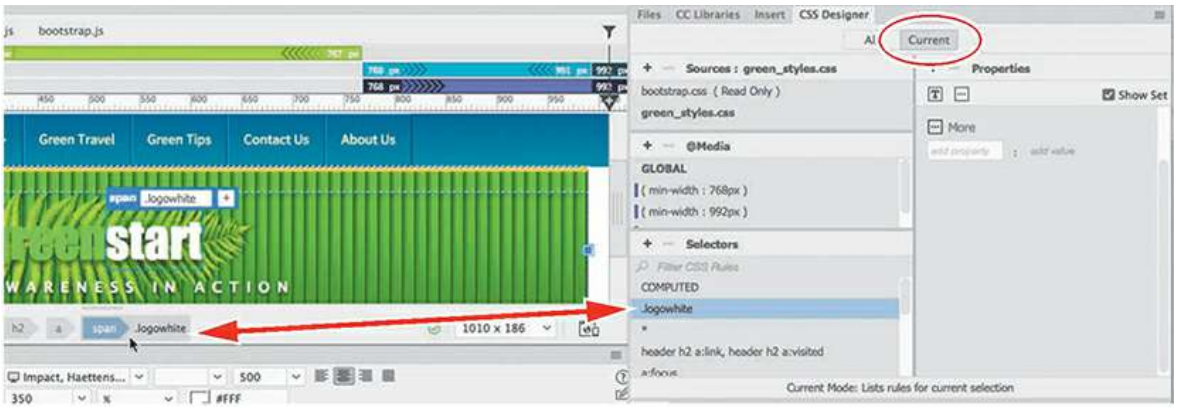
One of the most important features of Dreamweaver is the tag selector interface that appears at the bottom of the document window. This interface displays the tags and element structure in any HTML file pertinent to the insertion point of, or that is selected by, the cursor. The display of tags is hierarchical, starting at the document root at the left of the display and listing each tag or element in order based on the structure of the page and the selected element.



The display in the tag selector interface mimics the structure of the HTML code based on your selection.

The tag selectors also enable you to select any of the elements displayed by simply clicking a tag. When a tag is selected, all the content and child elements contained within that tag are also selected.

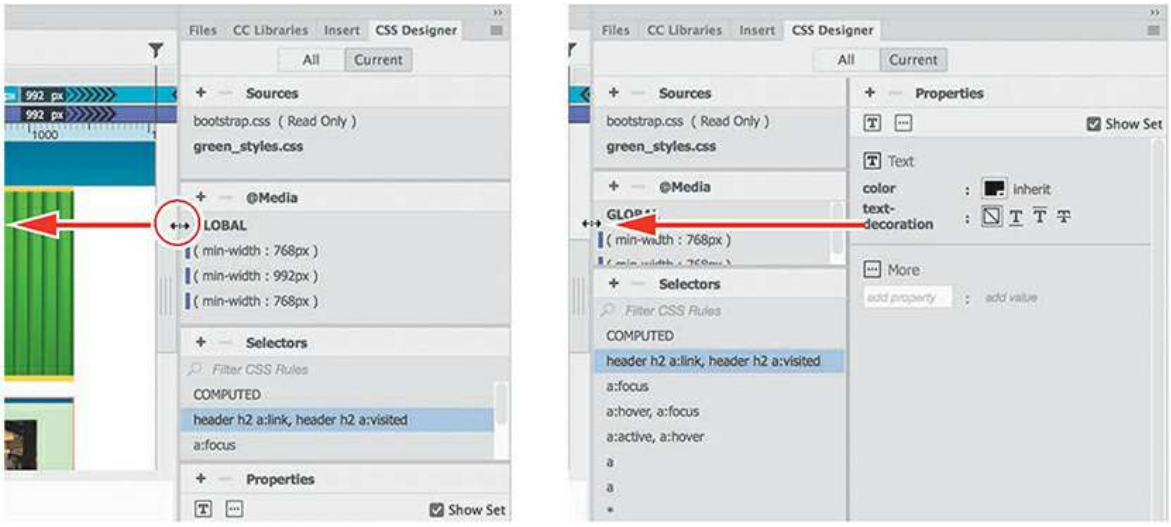
The tag selector interface is closely integrated with the CSS Designer panel. You may use the tag selectors to help you style content or to cut, copy, paste, and delete elements.



Using the CSS Designer

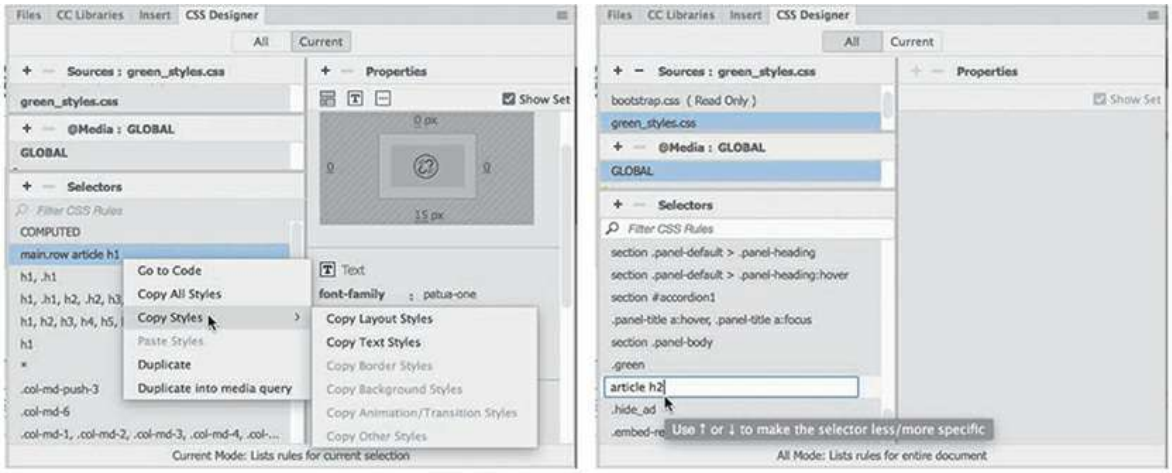
The CSS Designer is a powerful tool for visually inspecting, creating, editing, and troubleshooting CSS styling. The panel adapts to the size of the available workspace and

displays in either a one- or two-column layout.



The CSS Designer can be displayed in one or two columns. Simply drag the edge of the document window to the left or right until the panel displays the desired number of columns.

CSS Designer allows you to copy and paste CSS styles from one rule to another. You can also decrease or increase the specificity of new selector names by pressing the up or down arrow keys, respectively.

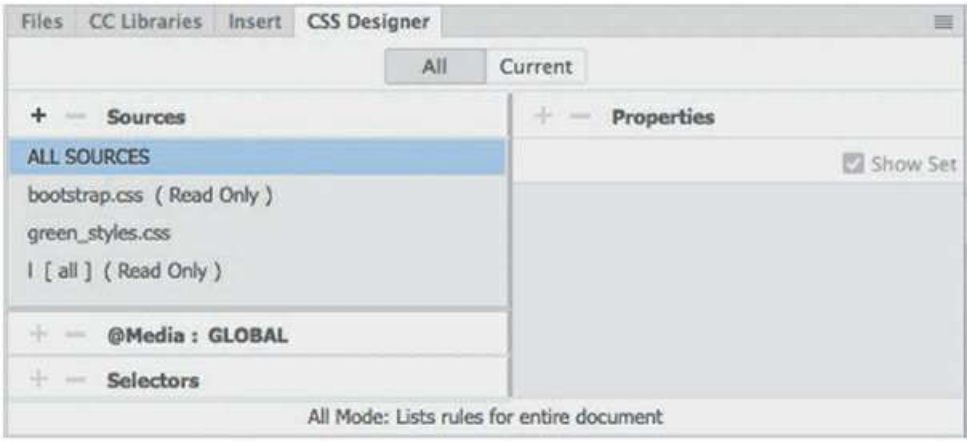


Copy and paste styles from one rule to another (left). Make selectors more or less specific by using the arrow keys (right).

The CSS Designer panel consists of four windows: Sources, @Media, Selectors, and Properties.

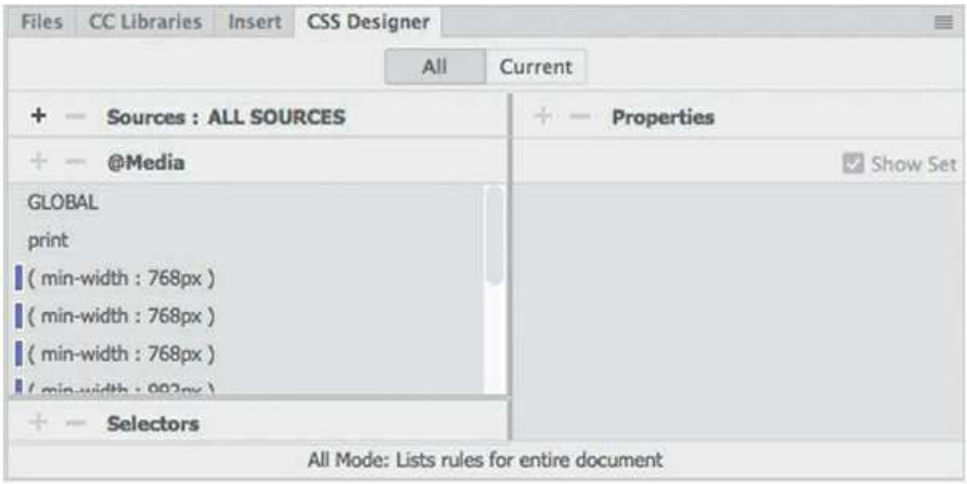
Sources

The Sources window allows you to create, attach, define, and remove internal embedded and external, linked style sheets.



@Media

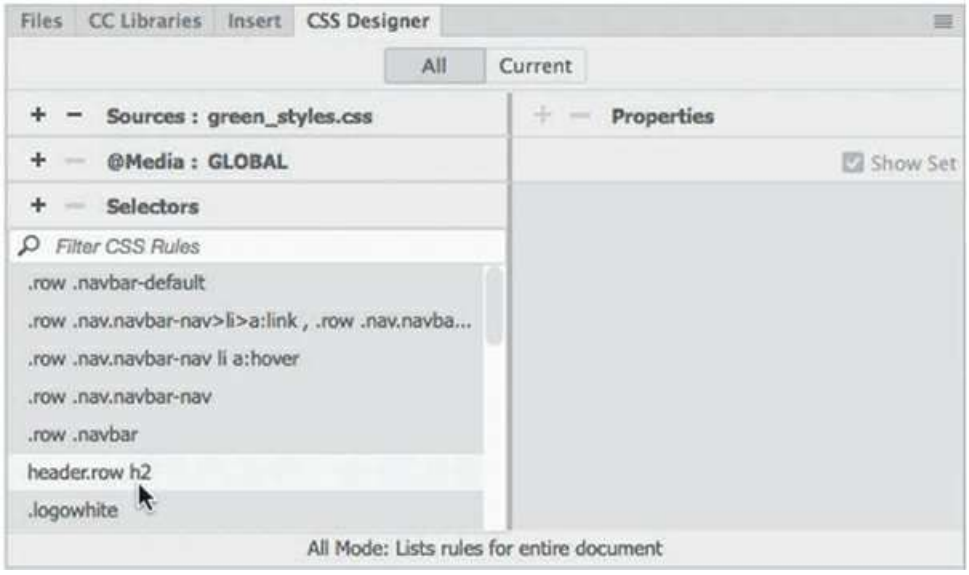
The @Media window is used to define media queries to support various types of media and devices.



Selectors

The Selectors window is used to create and edit the CSS rules that format the components and content of your page. Once a selector, or rule, is created, you define the formatting you wish to apply in the Properties window.

In addition to allowing you to create and edit CSS styling, the CSS Designer can also be used to identify styles already defined and applied, and to troubleshoot issues or conflicts with these styles.

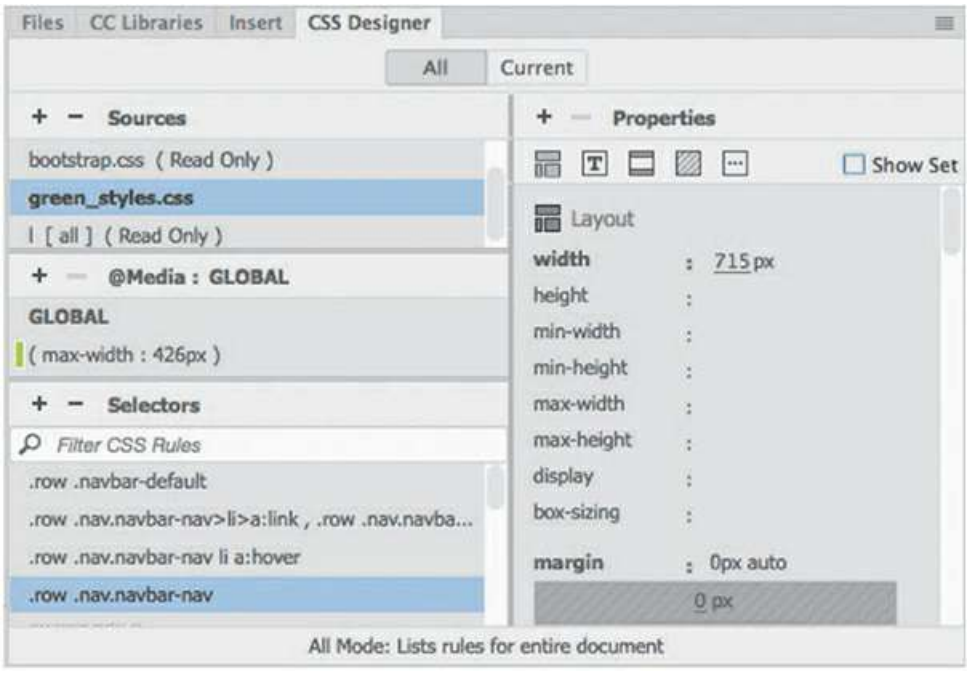


Properties

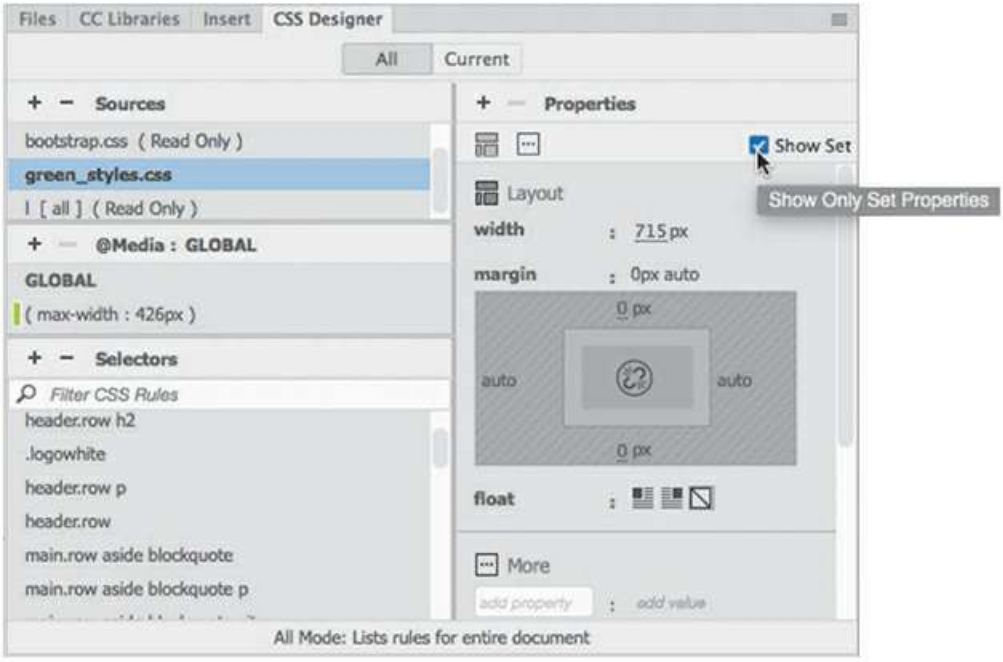
The Properties window features two basic modes. By default, the Properties window displays all available CSS properties in a list, organized in five categories: Layout, Text, Borders, Background, and More. You can scroll down the list and apply styling as desired or click the icon to jump to that category of the Properties panel.

Note

Deselect the Show Set option to see all the CSS Designer categories.



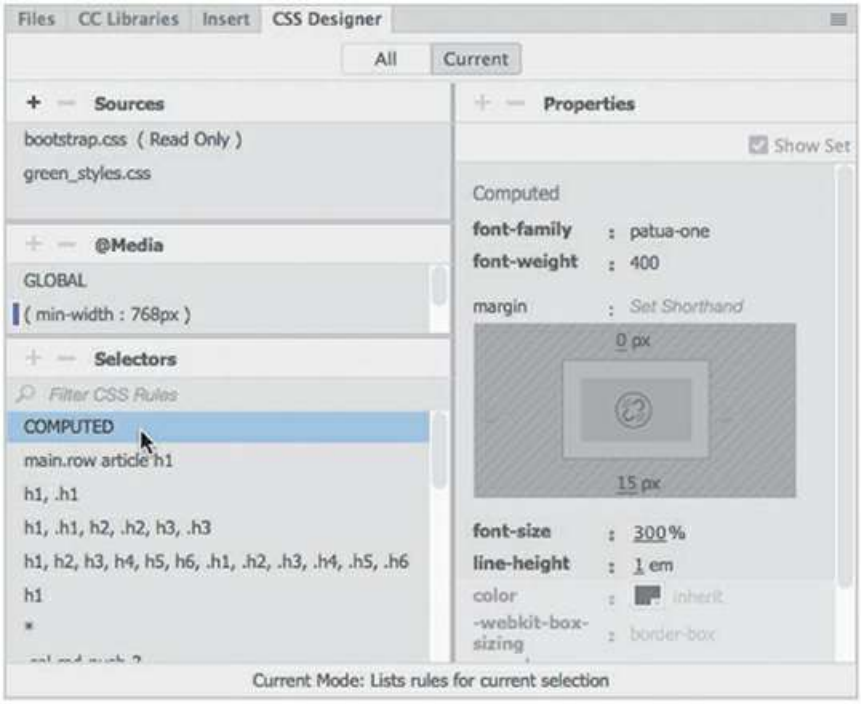
The second mode can be accessed by selecting Show Set at the upper-right edge of the window. In this mode, the Properties panel will filter the list down to only the properties actually applied to the rule chosen in the Selectors window. In either mode, you can add, edit, or remove style sheets, media queries, rules, and properties.



Selecting the Show Set option limits the property display to only the properties that are styled.

The Properties panel also features a COMPUTED option that displays the aggregated list of

styles applied to the selected element when the Current button in CSS Designer is selected. The COMPUTED option will then appear anytime you select an element or component on the page. When you're creating any type of styling, the code created by Dreamweaver complies with industry standards and best practices.



The COMPUTED option collects in one place all styles applied to the selection.

All and Current modes

The CSS Designer has two buttons at the top of the panel, All and Current, that enable specific functions and workflows within the panel.

When the All button is selected, the panel allows you to create and edit CSS style sheets, media queries, rules, and properties. When the Current button is selected, the CSS troubleshooting functions are enabled, allowing you to inspect individual elements in a webpage and assess existing styling properties applied to a selected element. In this mode, however, you will notice that some of the normal features in the CSS Designer are disabled. For example, when in Current mode you are able to edit existing properties and add new style sheets, media queries, and rules that apply to the selected element, but you cannot delete existing style sheets, media queries, or rules. This interaction works the same way in all document views modes.

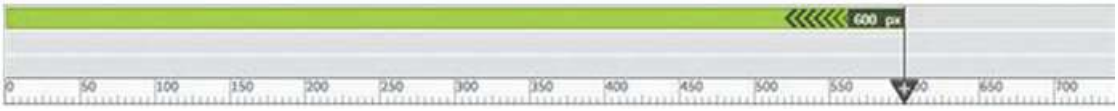
When the Current button is selected, the CSS Designer displays all styling associated with a selected element

In addition to using the CSS Designer, you may also create and edit CSS styling manually within Code view while taking advantage of many productivity enhancements, such as code hinting and auto-completion.

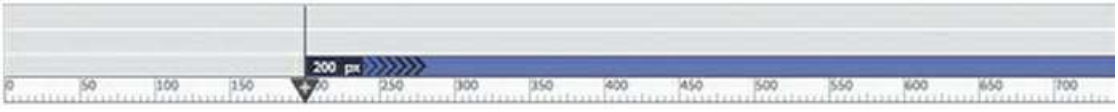
Using the Visual Media Query (VMQ) interface

The Visual Media Query (VMQ) interface is a newer feature of Dreamweaver. Appearing above the document window, the VMQ interface allows you to visually inspect and interact with existing media queries, as well as create new ones on the fly using a simple point-and-click interface.

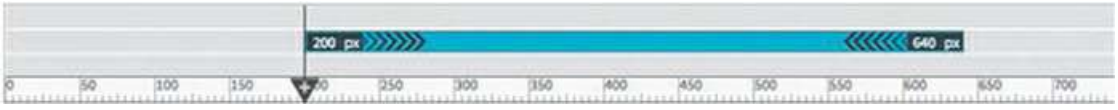
Open any webpage that is formatted by a style sheet with one or more media queries. If necessary, enable the VMQ interface by toggling the VMQ icon. The VMQ interface will appear above the document window and display color-coded bars that specify the type of media query that has been defined. Media queries using only a max-width specification will be displayed in green. Media queries using only a min-width specification will be displayed in purple. Ones that use both will be displayed in blue.



Max-width media query in the VMQ interface



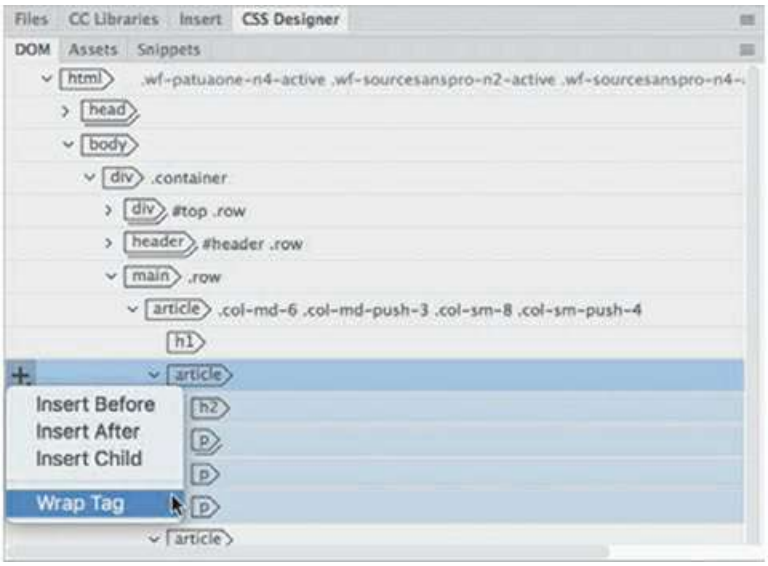
Min-width media query in the VMQ interface



Media using both max-width and min-width specifications

Using the DOM Viewer

The DOM Viewer allows you to view the Document Object Model (DOM) to quickly examine the structure of your webpage as well as interact with it to select, edit, and move existing elements and insert new ones. You'll find that it makes working in complex HTML structures simple.



Using element dialogs, displays, and inspectors

As Dreamweaver moves to make Live view the default workspace, it has driven the development of new methods for editing and managing HTML elements. You will find a handful of new dialogs, displays, and inspectors that provide instant access to important element properties and specifications. All of them, except the Text Display, allow you to add class or id attributes to the selected element and even insert references to those attributes into your CSS style sheets and media queries.

Position Assist dialog

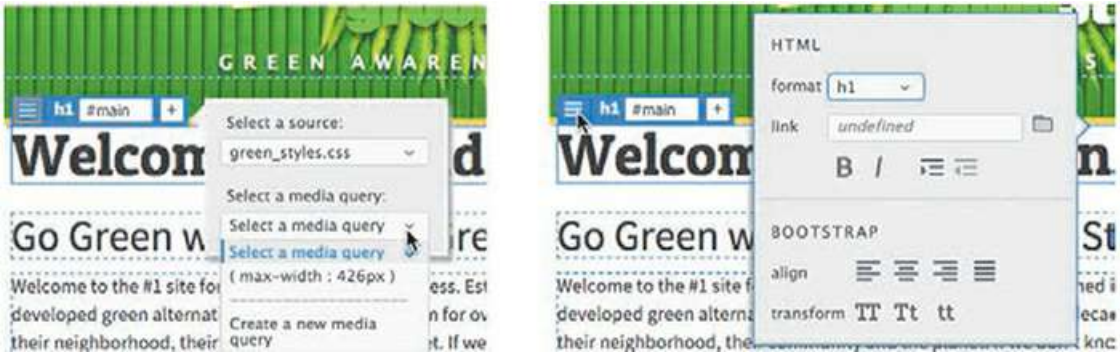
The Position Assist dialog appears whenever new elements are being inserted in Live view, using either the Insert menu or Insert panel. Typically, the Position Assist dialog will offer the options Before, After, Wrap, and Nest. Depending on what type of element is selected and what item is targeted by the cursor, one or more of the options may be grayed out.

The Position Assist dialog allows youto control how elements and components are inserted in Live view.

Element Display

The Element Display appears whenever you select an element in Live view. When an element is selected in Live view, you can change the selection focus by pressing the up and down arrow keys; the Element Display will then highlight each element in the page, in turn, based on its position in the HTML structure.

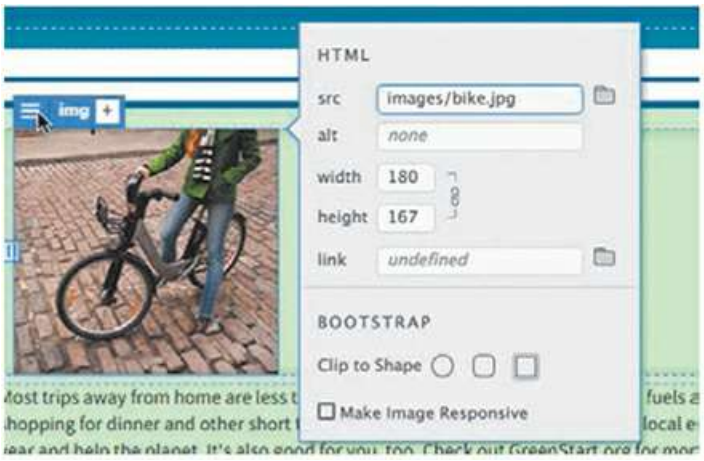
The Element Display features a Quick Property Inspector icon where you can instantly access properties such as formats, links, and alignment. The Element Display also allows you to add a class or id to the selected element or to edit a class or id.



The Element Display enables you to quickly apply classes, IDs, and links, as well as perform basic formatting.

Image Display

The Image Display provides a Quick Property inspector from which you can access the image source, alt text, and width and height attributes; it also contains a field from which you can add a hyperlink.



The Image Display gives you quick access to the image source and allows you to add hyperlinks.

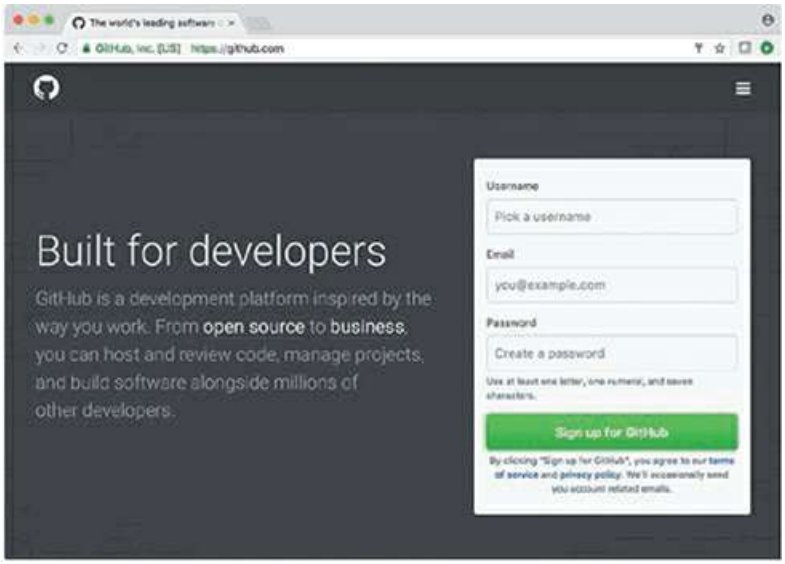
Text Display

The Text Display appears whenever you select a portion of text in Live view. The Text Display allows you to apply bold , italic , and hyperlink <a> markup to the selected text. Double-click the text to open the orange editing box. When you select some text, the Text Display will appear. When you are finished editing the text, click just outside the orange box to complete and accept the changes. Press Esc to cancel the changes and return the text to its previous state.

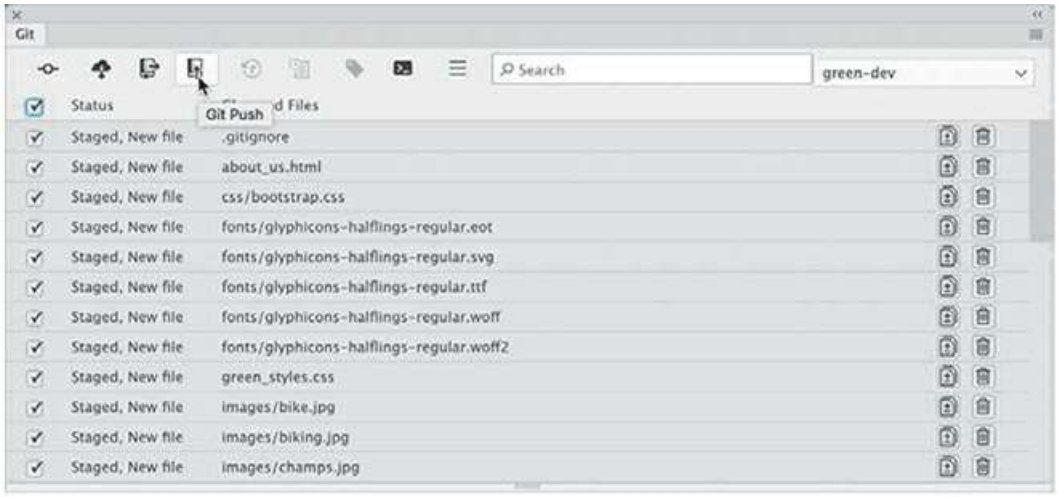
The Text Display lets you apply bold, italics, and hyperlink markup to selected text.

Setting up version control in Dreamweaver

Dreamweaver CC (2019 release) supports Git, a popular open source version control system for managing the source code of your websites. Such systems are very valuable for preventing conflicts and loss of work when you have a number of people working together on a project.



You set up a Git repository within your site definition dialog. After that, you can connect your site to your remote repository and push and pull changes as needed.



Go to https://helpx.adobe.com/dreamweaver/using/git-support.html for full instructions on how to set up Git version control for your own project.

Exploring, experimenting, and learning

The Dreamweaver interface has been carefully crafted over many years to make the job of webpage design and development fast and easy. It's a design in progress. It's always changing and evolving. If you think you already know the program, you're wrong. Install the latest version and check it out. Feel free to explore and experiment with various menus, panels, and options to create the ideal workspace and keyboard shortcuts to produce the most productive environment for your own purposes. You'll find the program endlessly adaptable, with power to spare for any task. Enjoy.

Review questions

1. Where can you access the command to display or hide any panel?
2. Where can you find the Code, Split, Design, and Live view buttons?
3. What can be saved in a workspace?
4. Do workspaces also load keyboard shortcuts?
5. What happens in the Property inspector when you insert the cursor into various elements on the webpage?
6. What features in the CSS Designer make it easy to build new rules from existing ones?
7. What can you do with the DOM Viewer?
8. Does the Element Display appear in Design or Code view?
9. What is Git?

Review answers

1. All panels are listed in the Window menu.
2. The Code, Split, Design, and Live view buttons are components of the Document toolbar.
3. Workspaces can save the configuration of the document window, the open panels, and the panels' size and position on the screen.
4. No. Keyboard shortcuts are loaded and preserved independently of a workspace.
5. The Property inspector adapts to the selected element, displaying pertinent information and formatting commands.
6. The CSS Designer allows you to copy and paste styles from one rule to another.
7. The DOM Viewer allows you to visually examine the DOM and select and insert new elements and edit existing ones.
8. No. The Element Display is visible only in Live view.
9. Git is an open source version control system that enables you to manage your website source code.

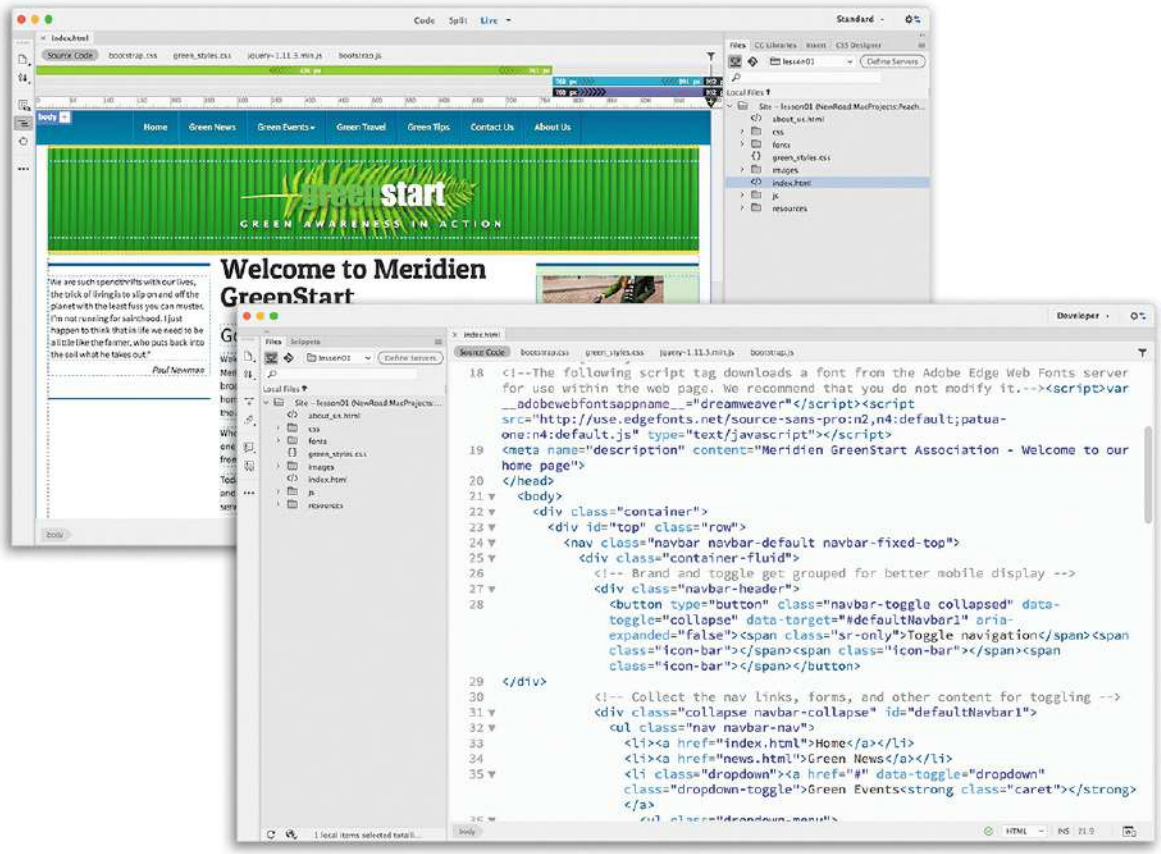
2 Html Basics

Lesson overview

In this lesson, you'll familiarize yourself with HTML and learn:

- What HTML is and where it came from
- Frequently used HTML tags
- How to insert special characters
- What semantic web design is and why it's important
- New features and capabilities in HTML

This lesson will take about 45 minutes to complete. This lesson does not have support files.



HTML is the backbone of the web, the skeleton of your webpage. It is the structure and substance of the Internet, although it is usually unseen except by the web designer. Without it, the web would not exist. Dreamweaver has many features that help you access, create, and edit HTML code quickly and effectively.

What is HTML?

“What other programs can open a Dreamweaver file?” asked a student in my Dreamweaver class. Although it might seem obvious to an experienced developer, it illustrates a basic problem in teaching and learning web design. Most people confuse the *program* with the *technology*. Some may assume that the extension .htm or .html belongs to Dreamweaver or Adobe. This isn’t as unusual as it seems. Print designers are used to working with files ending with extensions, such as .ai, .psd, .indd, and so on; it’s just a part of their jobs. They have learned over time that opening these file formats in a different program may produce unacceptable results or even damage the file.

On the other hand, the goal of the web designer is to create a webpage for display in a browser. The power or functionality of the originating program has little bearing on the resulting browser display, because the display is all contingent on the HTML code and how the browser interprets it. Although a program may write good or bad code, it’s the browser that does all the hard work.

The web is based primarily on HyperText Markup Language (HTML). The language and the file

format don't belong to any individual program or company. In fact, it is a *non*-proprietary, plain-text language that can be edited in any text editor, on any operating system, on any computer. Dreamweaver is, in part, an HTML editor, although it is also much more than this. But to maximize the potential of Dreamweaver, it's vital that you have a good understanding of what HTML is and what it can (and can't) do. This lesson is intended as a concise primer on HTML and its capabilities. It will be a helpful foundation for understanding Dreamweaver.

Where did HTML begin?

HTML and the first browser were invented in 1989 by Tim Berners-Lee, a computer scientist working at the CERN (Conseil Européen pour la Recherche Nucléaire, which is French for European Council for Nuclear Research) particle physics laboratory in Geneva, Switzerland. He intended the technology as a means for sharing technical papers and information via the fledgling Internet that existed at the time. He shared his HTML and browser inventions openly as an attempt to get the scientific community at large and others to adopt them and engage in the development themselves. The fact that he did not copyright or try to sell his work started a trend for openness and camaraderie on the web that continues to this day.

The Internet before HTML looked more like MS DOS or the macOS Terminal application. There was no formatting, no graphics, and no user-definable color.

The language that Berners-Lee created over 25 years ago was a much simpler construct of what we use now, but HTML is still surprisingly easy to learn and master. At the time of this writing, HTML is now at version 5, officially adopted as of October 2014. It consists of over 120 *tags*, such as `html`, `head`, `body`, `h1`, `p`, and so on.

The tag is inserted between less-than (<) and greater-than (>) angle brackets, as in `<p>`, `<h1>`, and `<table>`. These tags are used to identify, or *mark up*, text and graphics to signal the browser to display them in a particular way. HTML code is considered properly *balanced* when

the markup features both an opening (`<...>`) and a closing (`</...>`) tag, such as `<h1>...</h1>`.

When two matching tags appear this way, they are referred to as an *element*; an element encompasses any contents contained within the two tags, as well. Empty, or void, elements, like the horizontal rule, can be written in an abbreviated fashion using only one tag, such as `<hr/>`, essentially opening and closing the tag at the same time. In HTML5, empty elements can also be validly expressed without the closing slash, such as `<hr>`. Some web applications require the closing slash, so it's a good idea to check before using one form over the other.

Some elements are used to create page structures, others to structure and format text, and yet others to enable interactivity and programmability. Even though Dreamweaver obviates the need for writing most of the code manually, the ability to read and interpret HTML code is still a recommended skill for any burgeoning web designer. Sometimes it's the only way to find an error in your webpage. The ability to read and understand code may also become an essential skill in other fields as more information and content is created and disseminated via mobile devices and internet-based resources.

Basic HTML code structure

Here you see the basic structure of a webpage:

Note

Go to the book's online resources at peachpit.com for bonus hands-on exercises to gain some vital skills and experience writing and editing HTML code. See the "Getting Started" section at the beginning of the book for more details.

You may be surprised to learn that the only text from all this code that displays in the web browser is "Welcome to my first webpage." The rest of the code creates the page structure and text formatting. Like an iceberg, most of the content of the actual webpage remains out of sight.

Frequently used HTML elements

HTML code elements serve specific purposes. Tags can create distinct objects, apply formatting, identify content semantically, or generate interactivity. Tags that make their own space on the screen and stand alone are known as *block* elements; the ones that perform their duties within the flow of another tag are known as *inline* elements. Some elements can also be used to create *structural* relationships within a page, like stacking content in vertical columns or collecting several elements together in logical groupings. Structural elements can behave like block or inline elements or do their work entirely invisible to the user.

HTML tags

Table 2.1 shows some of the most frequently used HTML tags. To get the most out of Dreamweaver and your webpages, it helps to understand the nature of these elements and how they are used. Remember that some tags can serve multiple purposes.

Table 2.1 Frequently used HTML tags

TAG	DESCRIPTION
`<!--...-->`	Comment. Designates an HTML comment. Allows you to add notes within the HTML code (represented by ... in the tag) that are not displayed within the browser.
`<a>`	Anchor. The basic building block for a hyperlink.
`<blockquote>`	Quotation. Creates a standalone, indented paragraph identifying content quoted from another source.
`<body>`	Body. Designates the document body. Contains the visible portions of the webpage content.
` `	Break. Inserts a visual line break without creating a new paragraph.
`<div>`	Division. Used to divide webpage content into discernible sections.
`<em>`	Emphasis. Adds semantic emphasis. Displays as italics by default in most browsers and readers.
`<form>`	Form. Designates an HTML form. Used for collecting data from visitors.
`<h1>` to `<h6>`	Headings. Creates headings. Default formatting is bold.
`<head>`	Head. Designates the document head. Contains code that performs background functions, such as meta tags, scripts, styling, links, and other information not overtly visible to site visitors that may provide instructions on how to display the page or its contents.
`<hr>`	Horizontal rule. Empty element that generates a horizontal line.
`<html>`	Root element of most webpages. Contains the entire webpage, except in certain instances where server-based code must load before the opening `<html>` tag.
`<iframe>`	Inline frame. A structural element that can contain another document or load content from another website.
`<img>`	Image. Provides the source reference to display an image.
`<input>`	Input. An input element for a form such as a text field.
`<li>`	List item. An element within an HTML list.
`<link>`	Link. Designates the relationship between a document and an external resource.
`<meta>`	Metadata. Additional information provided for search engines or other applications.
`<ol>`	Ordered list. Defines a numbered list. List items display in an alpha, numeric, or roman numeral sequence.

<p>	Paragraph. Designates a standalone paragraph.
<script>	Script. Contains scripting elements or points to an internal or external script.
	Span. Designates a section within an element. Provides a means to apply special formatting or emphasis to a portion of an element.
	Strong. Adds semantic emphasis. Displays as bold by default in most browsers and readers.
<style>	Style. Embedded or inline element or attribute containing CSS styling.
<table>	Table. Designates an HTML table.
<td>	Table data. Designates a table cell.
<textarea>	Text area. Designates a multi-line text input element for a form.
<th>	Table header. Identifies a table cell containing a header.
<title>	Title. Contains the metadata title reference for the current page.
<tr>	Table row. Structural element that delineates one row of a table from another.
	Unordered list. Defines a bulleted list. List items display with bullets by default.

HTML character entities

Text content is normally entered via a computer keyboard. But many characters don't appear on a typical 101-key input device. If a symbol can't be entered directly from the keyboard, it can be inserted within the HTML code by typing the name or numeric value referred to as an *entity*. Entities exist for every letter and character that can be displayed. Some popular entities are listed in **Table 2.2**.

Note

Some entities can be created using either a name or a number, as in the copyright symbol, but named entities may not work in all browsers or applications. So either stick to numbered entities or test the specific named entities before you use them.

Table 2.2 HTML character entities

CHARACTER	DESCRIPTION	NAME	NUMBER
©	Copyright	©	©
®	Registered trademark	®	®
™	Trademark		™
•	Bullet		•
–	En dash		–
—	Em dash		—
	Nonbreaking space		

Go to www.w3schools.com/html/html_entities.asp to see a complete list and description of entities.

What's new in HTML5

Every new version of HTML has made changes to both the number and the purpose of the tags that make up the language. HTML 4.01 consisted of approximately 90 tags. HTML5 has removed some HTML 4 tags from its specification altogether, and some new ones have been adopted or proposed.

Changes to the list usually revolve around supporting new technologies or different types of content models, as well as removing features that were bad ideas or ones infrequently used. Some changes simply reflect customs or techniques that have been popularized within the developer community over time. Other changes have been made to simplify the way code is created, to make it easier to write and faster to disseminate.

HTML5 tags

Table 2.3 shows some of the important new tags in HTML5. The specification features nearly 50 new tags in total, while at least 30 old tags were deprecated. As we move through the exercises of this book, you will learn how to use many of these new HTML5 tags, as appropriate, to help you understand their intended role on the web. Take a few moments to familiarize yourself with these tags and their descriptions.

Go to www.w3schools.com/tags/default.asp to see the complete list of HTML5 elements.

Table 2.3 Important new HTML5 tags

TAG	DESCRIPTION
`<article>`	Article. Designates independent, self-contained content that can be distributed independently from the rest of the page or site.
`<aside>`	Aside. Designates sidebar content that is related to the surrounding content.
`<audio>`	Audio. Designates multimedia content, sounds, music, or other audio streams.
`<canvas>`	Canvas. Designates graphics content created using a script.
`<figure>`	Figure. Designates a section of standalone content containing an illustration, image, or video.
`<figcaption>`	Figure caption. Designates a caption for a `<figure>` element.
`<footer>`	Footer. Designates a footer of a document or section.
`<header>`	Header. Designates a section of the content that introduces a document or specific topic area.
`<hgroup>`	Heading group. Designates a set of `<h1>` to `<h6>` elements when a heading has multiple levels.
`<main>`	Main. Designates the unique content of a page. A page may have only one `main` element.
`<nav>`	Navigation. Designates a section containing a navigation menu or hyperlink group.
`<picture>`	Picture. Designates one or more resources for a webpage image to support the various resolutions available on smartphones and other mobile devices. This is a new tag that may not be supported in older browsers or devices.
`<section>`	Section. Designates a section within the content of a document.
`<source>`	Source. Designates media resources for video or audio elements. Multiple sources can be defined for browsers that do not support the default file type.
`<video>`	Video. Designates video content, such as a movie clip or other video streams.

Semantic web design

Many of the changes to HTML were made to support the concept of *semantic web design*. This movement has important ramifications for the future of HTML, its usability, and the interoperability of websites on the Internet. At the moment, each webpage stands alone on the web. The content may link to other pages and sites, but there's really no way to combine or collect the information available on multiple pages or multiple sites in a coherent manner. Search engines do their best to index the content that appears on every site, but much of it is lost due to the nature and structure of old HTML code.

HTML was initially designed as a presentation language. In other words, it was intended to display technical documents in a browser in a readable and predictable manner. If you look carefully at the original specifications of HTML, it looks like a list of items you would put in a college term paper: headings, paragraphs, quotations, tables, numbered and bulleted lists, and so on.

The element list in the first version of HTML basically identified how the content would be displayed. These tags did not convey any intrinsic meaning or significance. For example, using a heading tag displayed a particular line of text in bold, but it didn't tell you what relationship or importance the heading had to the following text or to the story as a whole. Is it a title or merely a subheading?

HTML5 has added a significant number of new tags to help us add semantic meaning to our markup. Tags such as `<header>`, `<footer>`, `<article>`, and `<section>` allow you for the first time to identify specific content without having to resort to additional attributes. The result is simpler code and less of it. But most of all, the addition of semantic meaning to your code allows you and other developers to connect the content from one page to another in new and exciting ways—many of which haven't even been invented yet. It's truly a work in progress.

New techniques and technology

HTML5 has also revisited the basic nature of the language to take back some of the functions that over the years have been increasingly handled by third-party plug-in applications and programming.

If you are new to web design, this transition will be painless because you have nothing to relearn, no bad habits to break. If you already have experience building webpages and applications, this book will guide you safely through some of these waters and introduce the new technologies and techniques in a logical and straightforward method. But either way, you don't have to trash all your old sites and rebuild everything from scratch.

Valid HTML 4 code will remain valid for the foreseeable future. HTML5 was intended to make web design easier by allowing you to do more with less work. So let's get started!

See www.w3.org/TR/2014/WD-html5-20140617 to learn more about HTML5.

See www.w3.org to learn more about W3C.

Review questions

1. What programs can open HTML files?
2. What does a markup language do?
3. HTML is composed of how many code elements?
4. What is the difference between block and inline elements?
5. What is the current version of HTML?

Review answers

1. HTML is a plain-text language that can be opened and edited in any text editor and viewed in any web browser.
2. A markup language places tags contained within brackets <> around plain-text content to pass information concerning meaning, structure, and formatting from one application to another.
3. HTML5 contains over 100 tags.
4. A block element creates a standalone element. An inline element can exist within another element.
5. HTML5 was formally adopted at the end of 2014. However, full support may take several more years. And as with HTML 4, some browsers and devices may support the specification in differing ways.

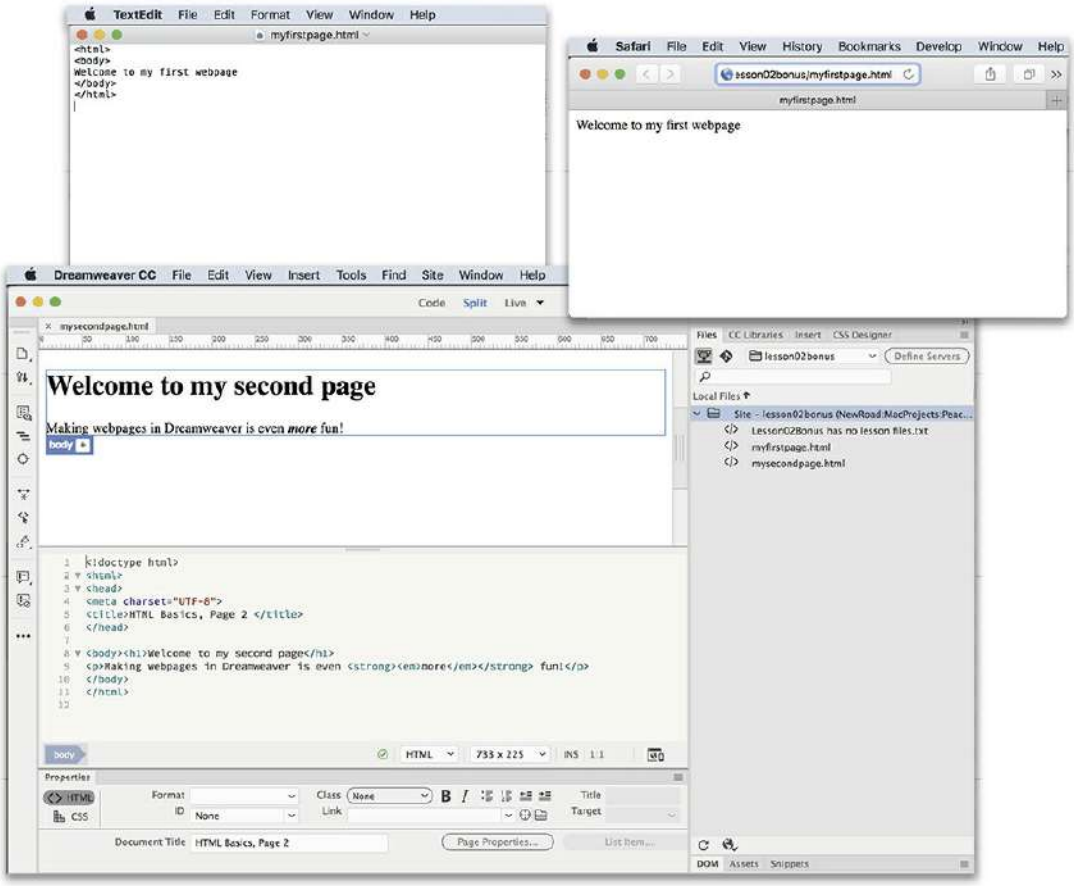
2 Html Basics Bonus

Lesson overview

In this lesson, you will gain valuable hands-on skills and experience in writing HTML code and learn the following:

- How to write HTML code in a text editor
- How to construct the basic structure of a webpage, including the root, head, and body elements
- How to write code by hand in Dreamweaver
- How to use code hinting and other productivity enhancements in Dreamweaver
- How to preview HTML in a browser and in Dreamweaver's Live view

This lesson will take about 45 minutes to complete. If you have not already done so, please log in to your account on peachpit.com to download the project files for this lesson as described in the "Getting Started" section at the beginning of this book and follow the instructions under "Accessing the Lesson Files and Web Edition." Define a site based on the lesson02bonus folder.



Dreamweaver provides many productivity enhancements for writing and proofing HTML code.

Writing your own HTML code

Writing code may sound difficult or at least tedious, but creating a webpage is actually much easier than you think. In the next few exercises, you will learn how HTML works by creating a basic webpage and adding and formatting simple text content.

Note

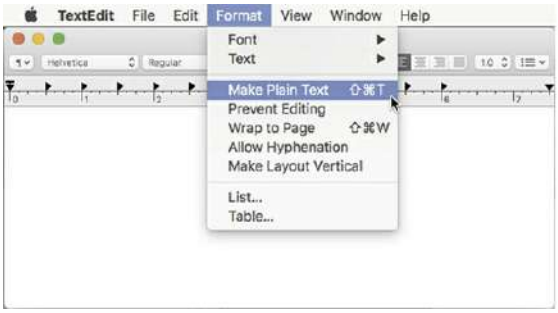
Feel free to use any text editor for these exercises, but be sure to save your files as plain text or text only.

Note

TextEdit may default to saving the file as rich text (.rtf); in this case, you need to choose Format > Make Plain Text before you can save the file as HTML (.html).

- Launch Notepad (Windows) or TextEdit (Mac).
- Enter the following code in the empty document window:

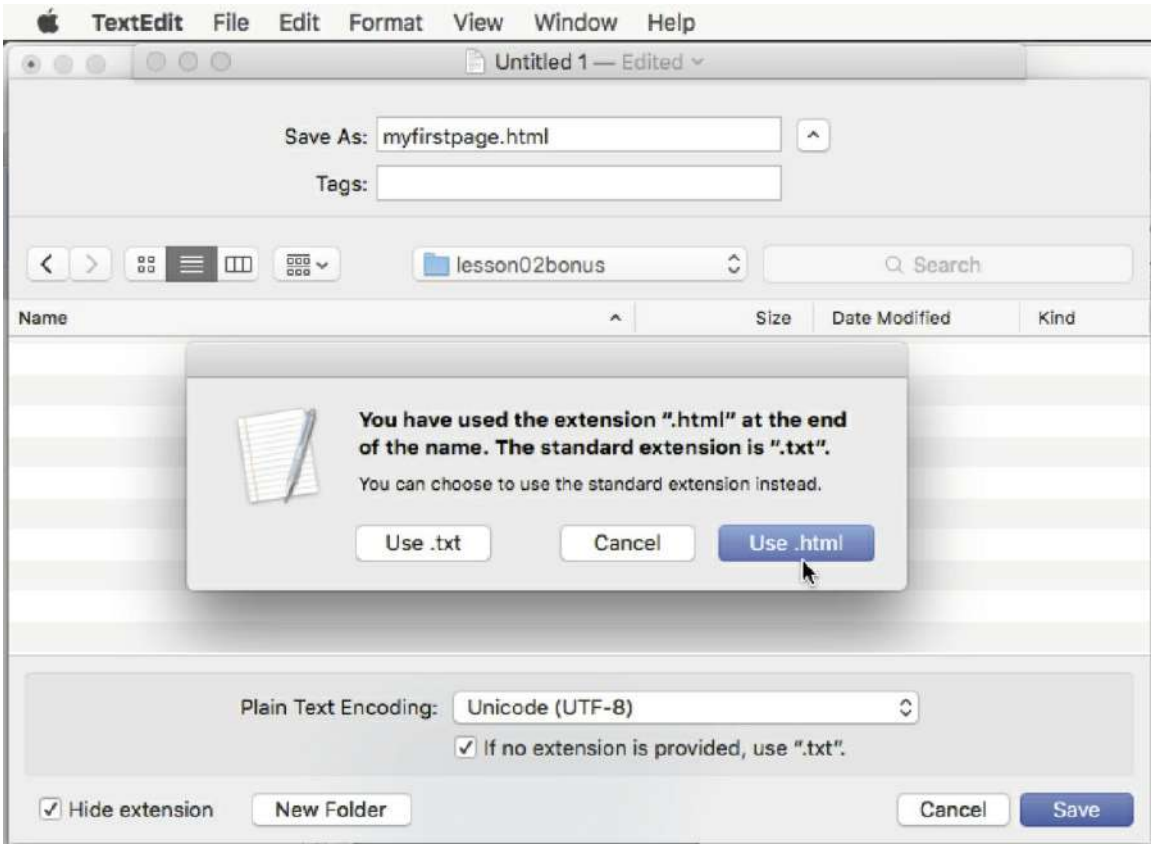
```
<html>
<body>
Welcome to my first webpage
</body>
</html>
```



- Save the file to the lesson02bonus folder as **myfirstpage.html**.

Note

In text editors, you will have to type the .html extension yourself. Some text editors may try to change the .html extension or prompt you to confirm the choice.



. Launch Chrome, Firefox, Internet Explorer, Safari, or another installed web browser.

. Open **myfirstpage.html** in the browser.

Arrange the two programs so you can see the code in the text editor side by side with the display in the browser.

Congratulations! You just created your first webpage. As you can see, it doesn't take much code to create a serviceable webpage.

Understanding HTML syntax

Next, you'll add content to your new webpage to learn some important aspects of HTML code syntax.

. Switch back to the text editor, but don't close the browser.

- Insert your cursor at the end of the text "`Welcome to my first webpage`" and press Enter/Return to insert a paragraph return.
- Type **`Making webpages is fun`** on the new line.

  Press the spacebar five times to insert five spaces.

  Finish by typing **`and easy!`** on the same line.
- Save the file.
- Switch to the browser.

  Although you saved the changes, you'll notice that the new text doesn't appear in the browser. That's because you never see a webpage "live" on the Internet. It must first be downloaded to your computer and saved, or *cached*, on the hard drive. The browser is actually displaying the page that it downloaded originally. To see the latest version of the webpage, you'll have to reload it.

  This is important to remember as a web designer. People frequently miss changes in a website because they are looking at the cached versions of a page instead of the most current version.
- Refresh the window to load the updated page.

> **Tip**
>
> In most browsers, you can press Ctrl+R/Cmd+R to refresh the page view.

As you can see, the browser is displaying the new text, but it's ignoring the paragraph return between the two lines as well as the extra spaces. In fact, you could add hundreds of paragraph returns between the lines and dozens of spaces between each word, and the browser display would be no different. That's because the browser is programmed to ignore extra whitespace and honor only HTML code elements. By inserting a tag here and there, you can easily create the desired text display.

Inserting HTML code

In this exercise, you will insert HTML tags to produce the correct text display.

- Switch back to the text editor.

- Add the highlighted tags to the text as follows:

  Click here to view code image

  ```
  <p>Making webpages is fun   and easy!</p>
  ```

  To add extra spacing or other special characters within a line of text, HTML provides code elements known as *entities*. Entities are entered into the code differently than tags. For example, the method for inserting a nonbreaking space is to type the ` ` entity.

  > **Tip**
  >
  > Another method for creating a nonbreaking space is by typing the numbered entity ` ` (which will generate the nonbreaking space in the browser). In most cases, the two entities are identical in use and performance; however, ` ` is not a valid entity for some applications, such as EPUB 2.0. Before you use a specific entity, make sure it is compatible with your workflow.

- Replace the five spaces in the text with five nonbreaking spaces so the code looks like the following sample:

  Click here to view code image

  ```
  <p>Making webpages is fun     
  and easy!</p>
  ```

  > **Tip**
  >
  > Feel free to copy and paste the entity code to save time.

- Save the file. Switch to the browser and reload or refresh the page display.

  The browser is now showing the paragraph return and the desired spacing.

Because you added the tags and entities, the browser can display the paragraph structure and spacing exactly as desired.

Although line breaks, extra spacing, and even indentation are ignored by the browser, web designers and coders frequently add such whitespace to make the code easier to read and edit.

But don’t go crazy. Although whitespace doesn’t affect the display of a page in a browser, it can contribute to the time it takes to download and render a webpage overall. Whitespace and extraneous code contribute to what web developers call the overall *weight* of a page. When a page has too much weight, it downloads, renders, and operates more slowly than it should.

Formatting text with HTML

Tags often serve multiple purposes. Besides creating paragraph structures and creating whitespace, as demonstrated earlier, tags can impart basic text formatting, as well as identify the relative importance of the page content.

For example, HTML provides six heading tags (`<h1>` to `<h6>`) that you can use to set headings off from normal paragraph text. Not only do the tags format the heading text differently than the paragraph text, but they also impart additional meaning. Heading tags are automatically formatted in bold and often at a larger relative size. The number of the heading (1 through 6) also plays a role. Using the `<h1>` tag identifies the heading as being the highest in importance for search engines and other applications. The other headings are given less importance in descending order.

In this exercise, you will add a heading tag to the first line.

- Switch back to the text editor.
- Add the highlighted tags to the text as follows:

  Click here to view code image

```
<h1>Welcome to my first webpage</h1>
```

- Save the file. Switch to the browser and reload or refresh the page display.

Note how the text display changed. It is now larger and formatted in boldface.

Web designers use heading tags to identify the importance of specific content and to improve their site rankings on Google, Yahoo!, and other search engines. Headings also help individuals who use screen readers and other assistive devices that allow them to access web-based content.

Applying inline formatting

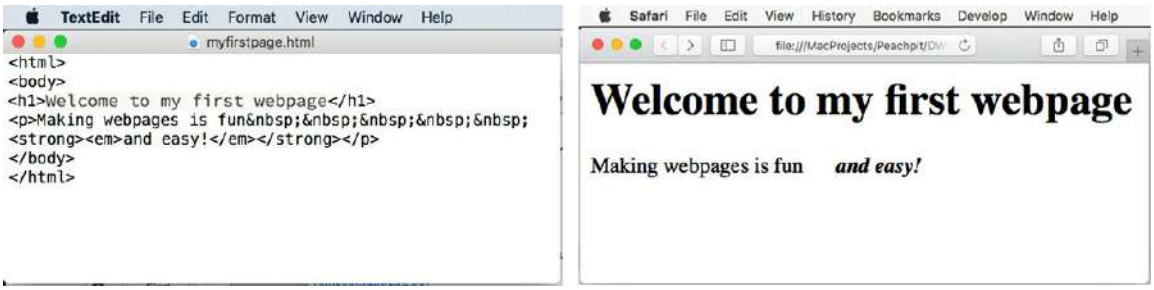
So far, all the tags you have used work as paragraph or standalone elements. These are referred

to as *block* elements. HTML also provides the ability to apply formatting and structure to content that's contained within the flow of another tag, or *inline*. A typical use of inline code would be to apply bold or italic styling to a word or a portion of a paragraph. In this exercise, you will learn how to apply inline formatting.

- Switch back to the text editor.
- Add the highlighted tags to the text as follows:

  Click here to view code image

```
<p>Making webpages is fun     
<strong><em> and easy!</em></strong></p>
```



- Save the file. Switch to the browser and reload or refresh the page display.

Note

The <b> and <i> tags were resurrected in HTML5 to provide bold and italic styling when no semantic value is needed.

Strong and *em* (emphasis) are used in place of the tags for bold (<b>) and italic (<i>) because they provide enhanced semantic meaning for visitors with disabilities or visual impairment, but the visual result is basically identical.

Most formatting, both inline and otherwise, is properly applied using cascading style sheets (CSS). The and tags are among the few still acceptable ways to apply inline formatting using specific HTML code elements. Technically speaking, these elements are intended more to add semantic meaning to text content than to add styling, but the result is the same. Today, modern browsers still display and as bold and italics, respectively.

However, this may change in the near future. There has been a move to separate the content from its presentation, or formatting, over the last decade. Although most browsers and HTML readers currently apply default formatting based on specific tags, this may not always be the case. See Lesson 3, "CSS Basics," for a full explanation of the strategy and application of CSS in standards-based web design.

Adding structure

Most webpages feature at least three fundamental elements: root (typically `<html>`), `<head>`, and `<body>`. These elements create the essential underlying structure of the webpage. The root element contains all the code and content, with the exception of any dynamic code that must load before the page content itself. This dynamic code is usually written in web programming languages such as ASP, Cold Fusion, or PHP. The `<head>` element holds code that performs vital background tasks, including styling, external links, and other metadata. The `<body>` element holds all the visible content for the webpage, such as text, tables, images, movies, and so on.

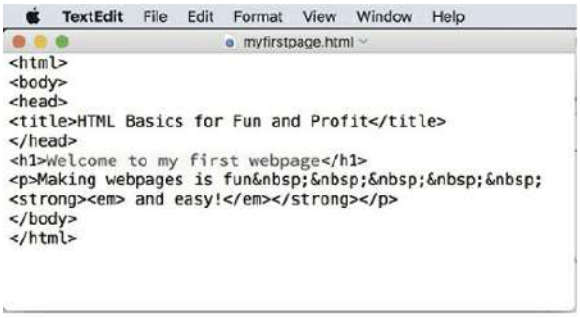
The sample page you created doesn't have a `<head>` element. A webpage can exist without this section, but adding any advanced functionality to this page without one would be difficult. In this exercise, you will add `<head>` and `<title>` elements to your webpage.

- Switch back to the text editor.
- Add the highlighted tags and content as shown here:

  Click here to view code image

```
<html>
<head>
<title>HTML Basics for Fun and Profit</title>
</head>
<body>
```

- Save the file. Switch to the browser and reload or refresh the page display.



Note

In some instances, tabs may be turned off by default when single pages are displayed. If you don't see the page title, check the view settings in the browser.

Did you notice what changed? It may not be obvious at first. Look at the title bar or window tab of the browser. The words "HTML Basics for Fun and Profit" now magically appear in the tab or title bar above your webpage. By adding the `<title>` element, you have created this display

yourself. But it's not just a cool trick; it's good for your business too.

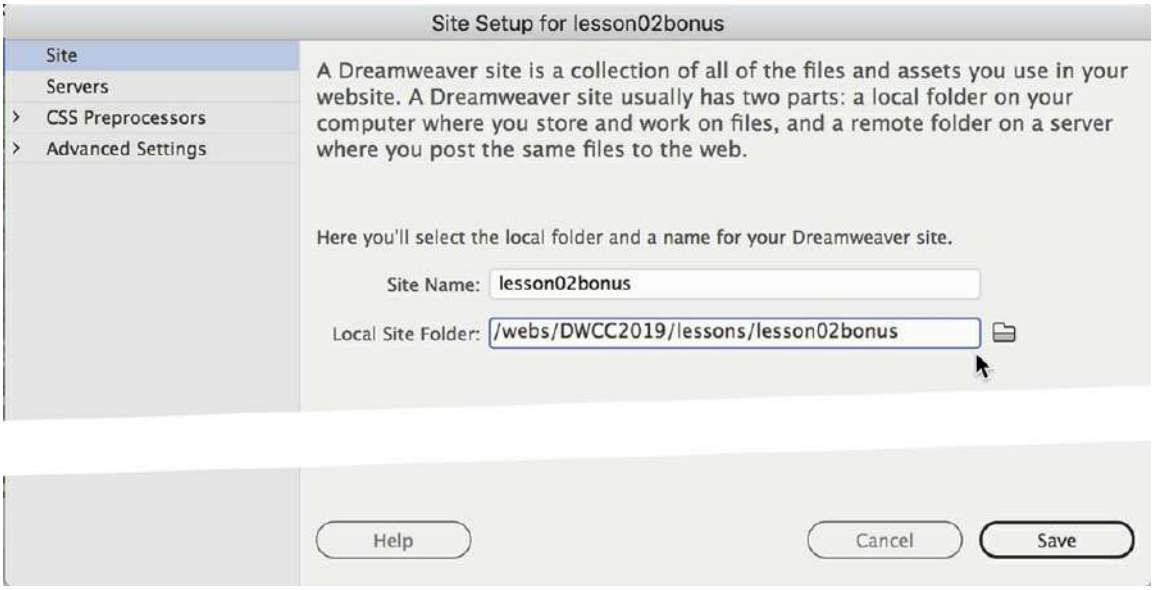
Google, Yahoo!, and the other search engines catalog the `<title>` element of each page and use it, among other criteria, to index and rank webpages. The content of the title is one of the items typically displayed within the results of a search. It also appears automatically when you create a bookmark for that page in your browser.

A well-titled page could be ranked higher than one with a bad title or one with none at all. Keep your titles short but meaningful. For example, the title "ABC Home Page" doesn't really convey any useful information. A better title might be "Welcome to the Home Page of ABC Corporation." Check out other websites (especially peers or competitors) to see how they title their own pages.

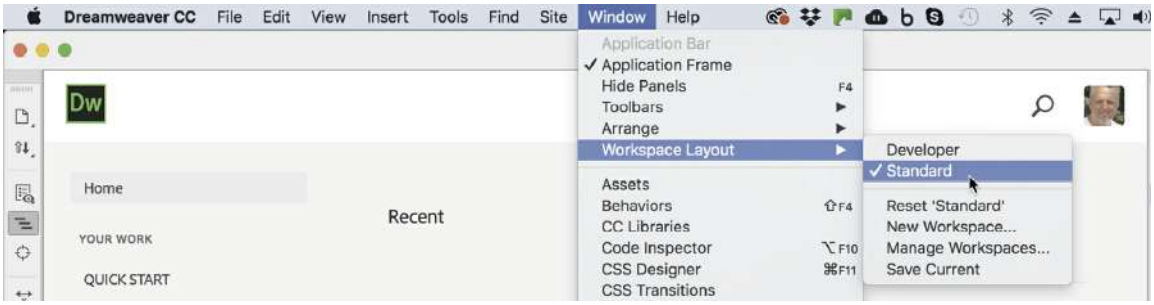
Writing HTML in Dreamweaver

So the inevitable question is, "If I can write HTML in any text editor, why do I need to use Dreamweaver?" Although a complete answer awaits you in the 13 lessons in the book and the seven online bonus lessons, the question begs a quick demonstration. In this exercise, you will re-create the same sample webpage using Dreamweaver.

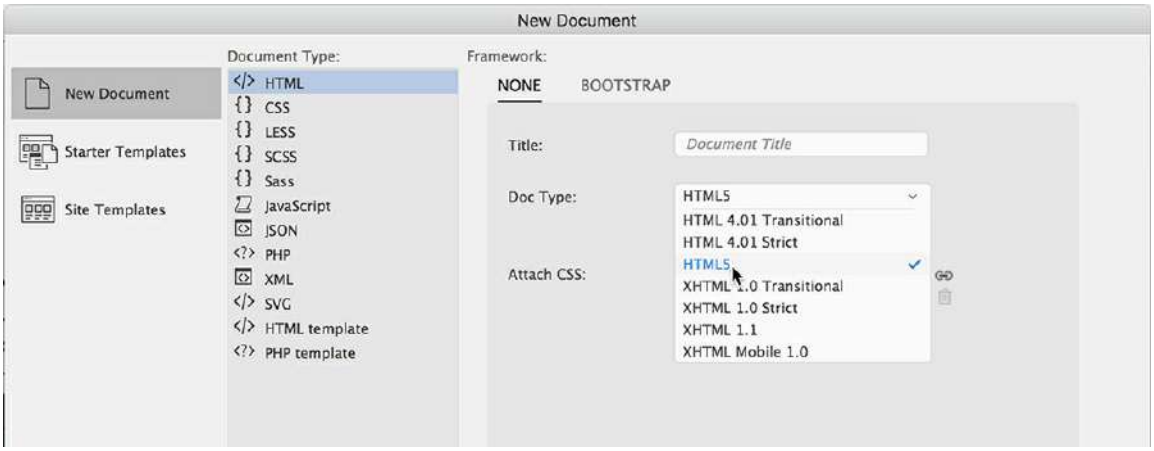
- Launch Dreamweaver CC (2019 release) or later.
- Define a new site based on the **lesson02bonus** folder as described in the "Getting Started" section at the beginning of the book.



- If necessary, select the Standard workspace from the Workspace menu, or choose Window > Workspace Layout > Standard.



- Choose File > New.
- In the New Document dialog, select the New Document category.
- Select HTML from the Document Type column, if necessary.
- In the Framework section, choose the None tab.
- If necessary, in the Doc Type menu, select HTML5.



- Click Create.

A new empty HTML document appears in Dreamweaver. The document window may default to one of four displays: Live view, Code view, Design view, or Split view.

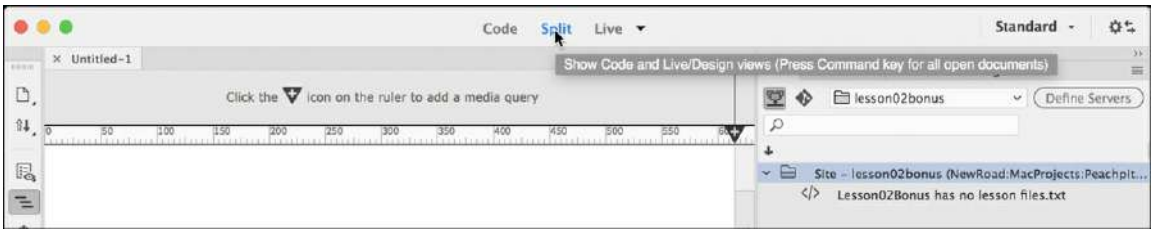
- If it’s not already selected, click Code view at the top of the document window.

The first thing you should notice in Code view is that Dreamweaver gives you a huge head

start over the text editor. The basic structure of the page is already in place, including the root, `head`, `body`, and `title` elements, among others.

Another advantage is evident when you need to view the results of your coding efforts. The text editor required the use of a separate application to preview the HTML code. Dreamweaver provides a built-in method.

. Click the Split view button.



In Split view, the interface is divided into two windows. One will show the HTML code; the other can be used to provide an accurate preview of the finished webpage. This preview alone can save you hours of time loading and previewing pages in a separate application.

. If necessary, activate Live view in Split view.

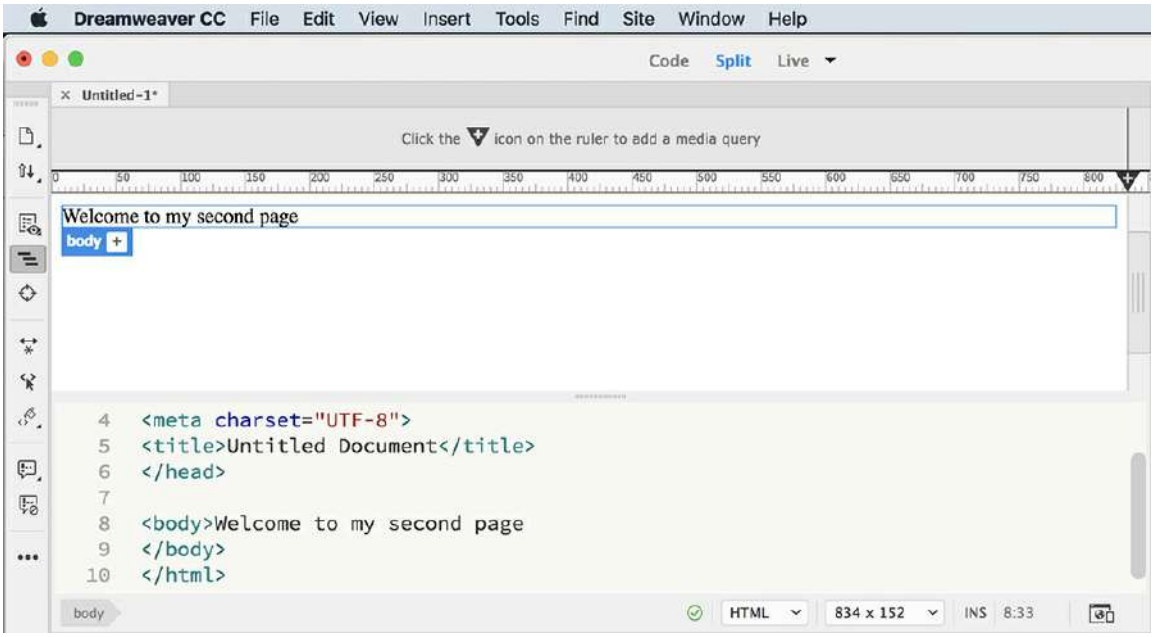
The program interface should now be divided in half, displaying Code view in one window and Live view in another. Dreamweaver also makes it easy to write HTML code.

. In the Code view window, insert the cursor after the `<body>` tag and enter the following text:

```
Welcome to my second page
```

Note

Split view may use either Live view or Design view to display side by side or top to bottom with the Code window. For simple pages like this, either one will do. See Lesson 1, "Customizing Your Workspace," for more information on how you can tailor your workspace to fit your needs.



Live view renders the HTML code as it would appear in a web browser. The current version is based on the same engine as Apple's Safari. After you enter the text, it may appear immediately in the Live view window. However, sometimes you may need to refresh the preview to see the changes.

. Click in the Live view window to refresh the preview, if necessary.

The text appears in Live view without any special formatting.

. In the Code view window, insert the cursor at the beginning of the text

"Welcome to my second page."

. Type **<**

A drop-down menu appears next to the cursor. This is Dreamweaver's *code-hinting* feature displaying a list of compatible HTML, CSS, JavaScript, and other supported code elements.

. Type **h**

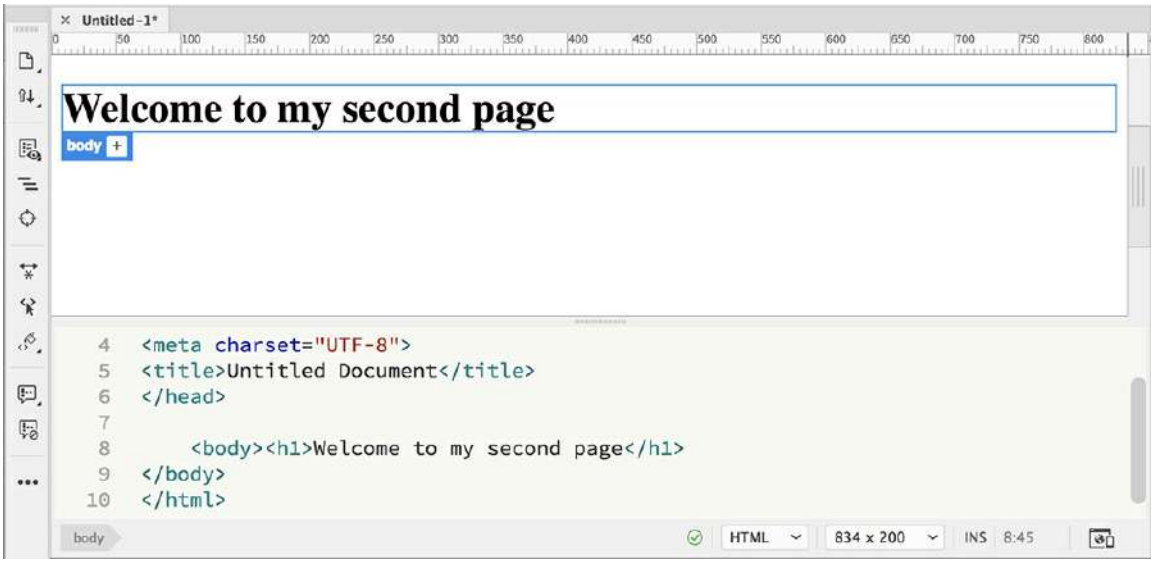
The list filters as you type, showing only elements that match the entered characters. You can continue typing the tag name manually or select it using the mouse or keyboard.

. Double-click `h1` from the list to insert it in the code. Type **>** to close the opening tag, if necessary.

Note

Depending on your preference settings, Dreamweaver may create only the opening tag or the entire element at once. The following steps assume that only the opening tag is created. Feel free to adjust the code-completion preferences to your liking.

. Move the cursor to the end of the text. Type **</** at the end of the sentence.



Note that Dreamweaver closes the `<h1>` element automatically. In this case, you entered the `<h1>` tags after the fact, but many web designers add the tags as they write.

. In Code view, press Enter/Return to insert a line break.

Type **<p** and press Enter/Return.

Type **>** to close the tag.

Note

If the <h1> and <p> elements are closed automatically in step 18 and 21, you may need to move the closing tags to the end of the text manually.

. Type **Making webpages in Dreamweaver is even more fun!** and then type **</** to close the <p> element.

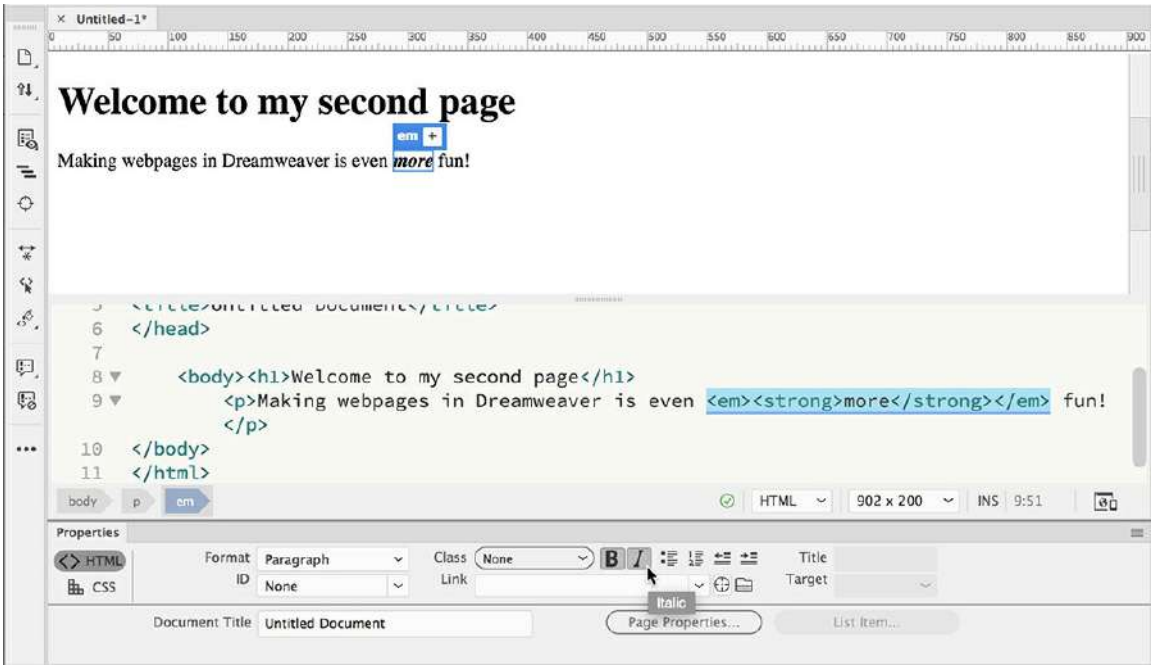
Tired of hand-coding yet? Dreamweaver offers multiple ways to write code automatically.

. Select the word “more.”

. Choose Window > Properties to display the Property inspector, if necessary.

This panel is an important component to many workflows in Dreamweaver. If it appears as a floating panel, you can dock it to the bottom of the document window so that it will be handy when you need it. See Lesson 1 for more information on how to customize the Dreamweaver interface.

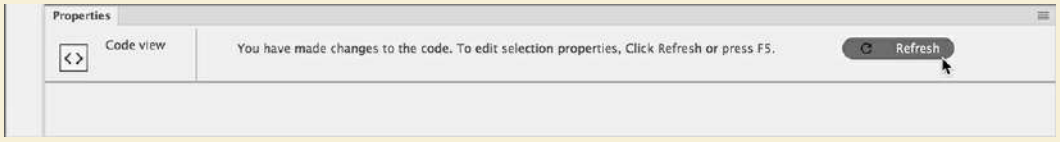
. In the HTML tab of the Property inspector, click the **B** button and the **I** button to apply and tags to the selected text.



These tags produce the appearance of bold and italic formatting on the selected text.

Something missing?

When you reached for the B and I buttons in step 24, were they missing? When you make changes in Code view, the Property inspector occasionally needs to be refreshed before you can access the formatting commands and metadata fields featured there. Simply click the Refresh button to make the formatting commands and other tools reappear.



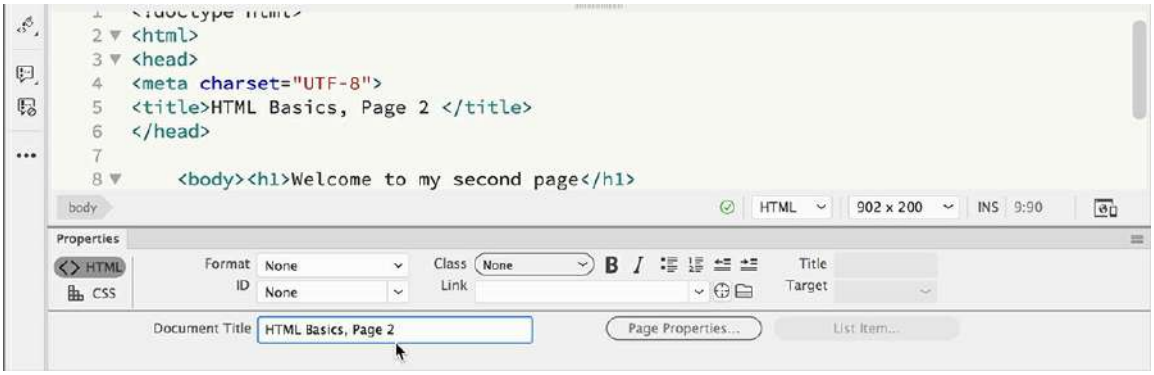
If you do not see the Property inspector, you can display it by choosing Window > Properties.

Only two more tasks remain before your new page is complete. Note that Dreamweaver created the `<title>` element and inserted the text “Untitled Document” within it. You could select the text within the code window and enter a new title, or you could change it using another built-in feature.

. Locate the Document Title field in the Property inspector, and select the `Untitled Document` placeholder text.

- Enter **HTML Basics, Page 2** in the Document Title field.

  Press Enter/Return to complete the title.

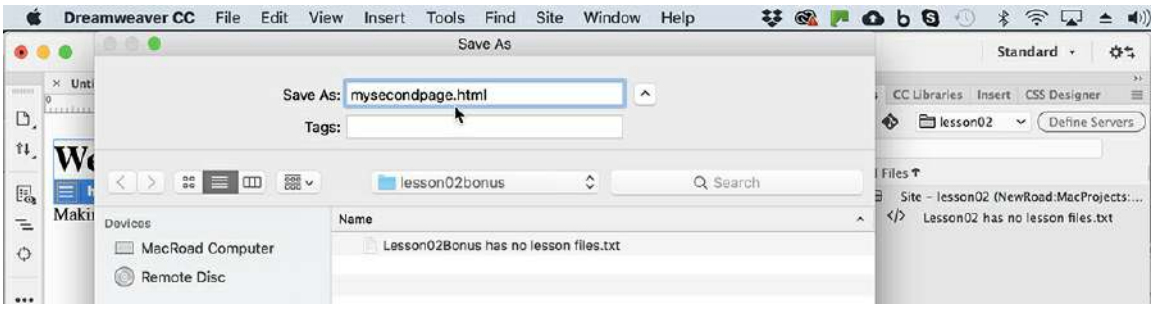


  The new title text appears in the code, replacing the original content. It's time to save the file and preview it in the browser.

- Choose File > Save.

  Navigate to the folder lesson02bonus.

  Name the file **mysecondpage** and click Save.

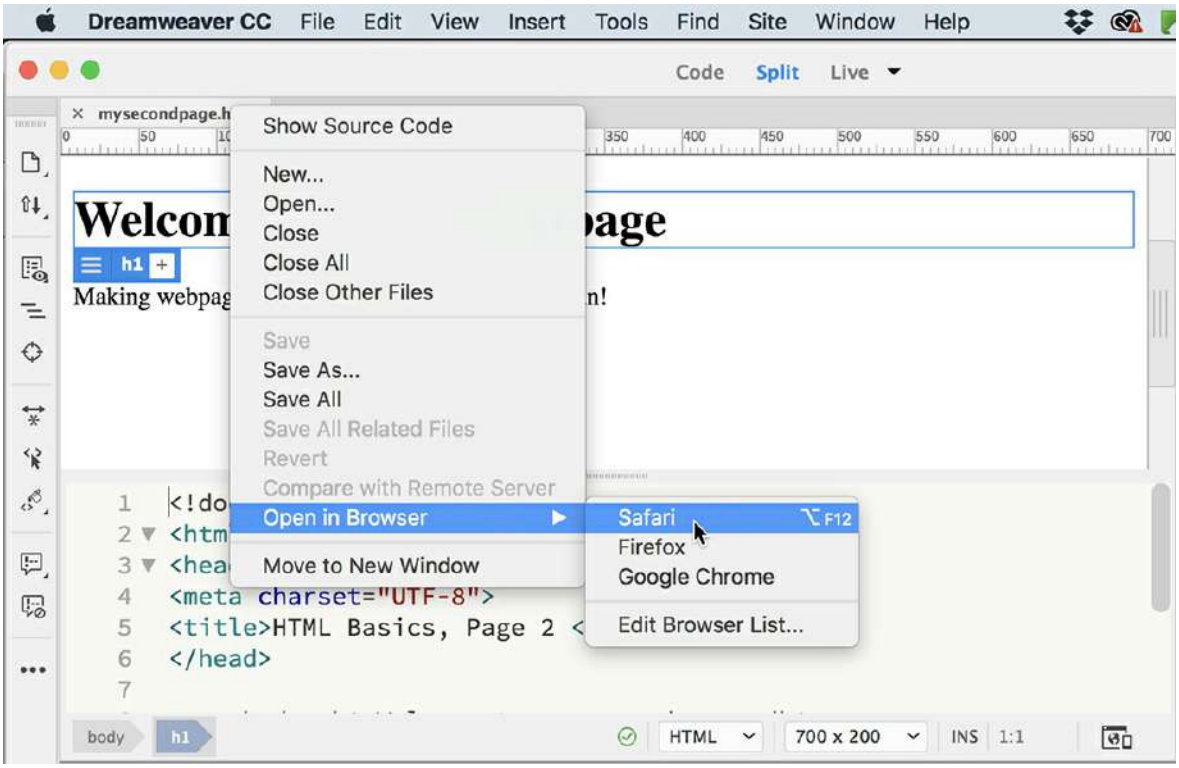


  Dreamweaver adds the proper extension (.html) automatically.

Note

Dreamweaver uses the browsers already installed on your computer. You may install additional, alternate browsers and configure their use in the Dreamweaver Preferences dialog.

- Right-click the document tab displaying the name of the file. Select Open In Browser from the context menu, and select your favorite browser.



The completed page appears in the browser window.

Using Dreamweaver you completed the task in a fraction of the time it took you to do it manually in a text editor.

You have just completed two webpages—one by hand and the other using Dream-weaver. In both cases, you can see how HTML played a central role in the whole process. To learn more about this technology, go to the website of the W3 Consortium, www.w3.org, or check out any of the following books:

Introducing HTML5, 2nd Edition (Peachpit Press, 2012), ISBN: 978-0-321-78442-1

HTML and CSS: 8th Edition (Peachpit Press, 2014), ISBN: 978-0-321-92883-2

HTML5 Pocket Reference, 5th Edition (O'Reilly, 2013), ISBN: 978-1-449-36335-2

3 CSS Basics

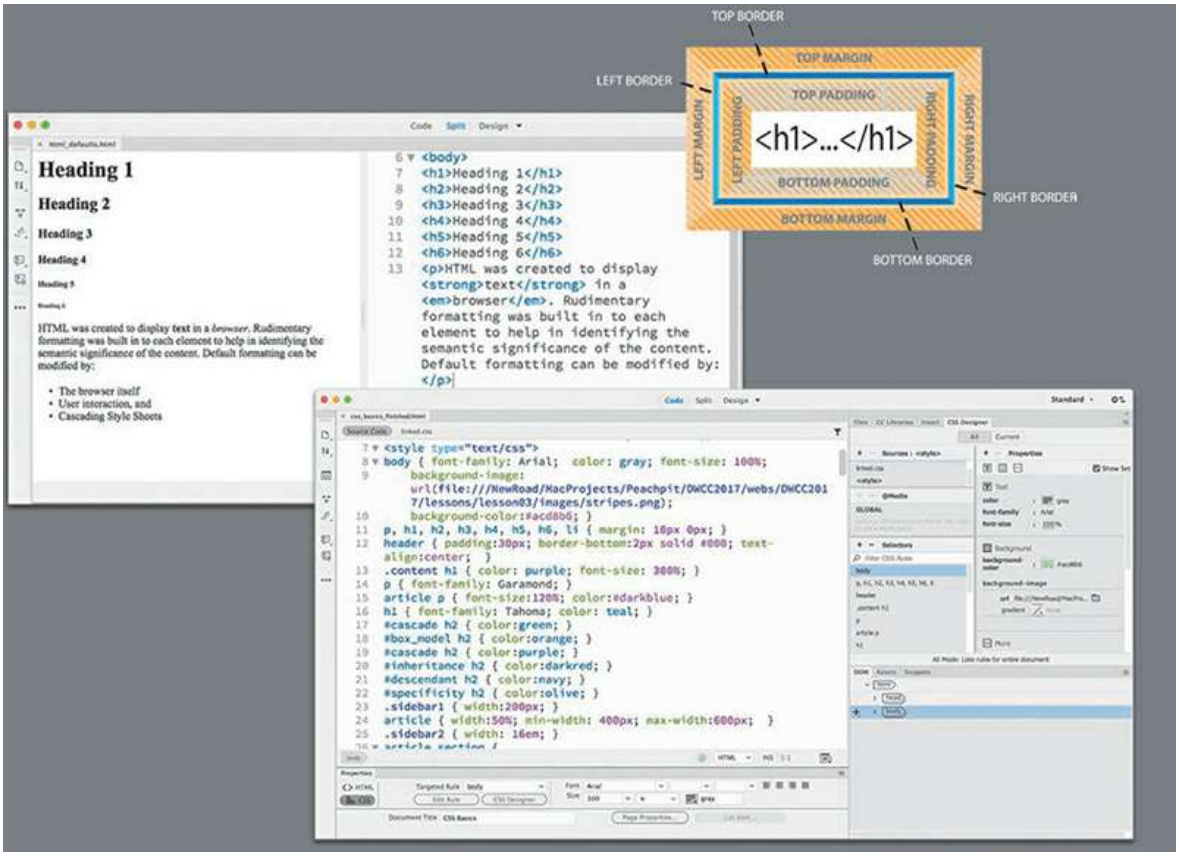
Lesson overview

In this lesson, you'll familiarize yourself with CSS and learn:

- CSS (cascading style sheets) terms and terminology
- The difference between HTML and CSS formatting
- Different methods for writing CSS rules and markup
- How the cascade, inheritance, descendant, and specificity theories affect the way browsers apply CSS formatting
- New features and capabilities of CSS3

This lesson will take about 1 hour and 15 minutes to complete. Please log in to your account on peachpit.com to download the files for this lesson, or go to the "Getting Started" section at the beginning of this book and follow the instructions under "Accessing the Lesson Files and Web Edition." Store the files on your computer in a convenient location. Define a site based on the lesson03 folder.

Your Account page is also where you'll find any updates to the lesson files. Look on the Lesson & Update Files tab to access the most current content.



Cascading style sheets control the look and feel of a webpage. The language and syntax of CSS are complex, powerful, and endlessly adaptable. CSS takes time and dedication to learn and years to master, but a modern web designer can't live without it.

What is CSS?

HTML was never intended to be a design medium. Other than allowing for bold and italic, version 1 lacked a standardized way to load fonts or even format text. Formatting commands were added along the way—up to version 3 of HTML—to address these limitations, but these changes still weren't enough. Designers resorted to various tricks to produce the desired results. For example, they used HTML tables to simulate multicolumn and complex layouts for text and graphics, and they used images when they wanted to display typefaces other than Times or Helvetica.

HTML-based formatting was so misguided a concept that it was deprecated from the language less than a year after it was formally adopted in favor of cascading style sheets (CSS). CSS avoids all the problems of HTML formatting while saving time and money too. Using CSS lets you strip the HTML code down to its essential content and structure and then apply the formatting separately so that you can more easily tailor the webpage to specific devices and applications.

By adding cell padding and margins to the table structure in Dreamweaver (left), you can see how this webpage relies on tables and images to produce the final design (right).

Note

We removed many of the hands-on exercises and moved them to an online bonus lesson. Go to the book's online resources at peachpit.com for bonus hands-on exercises to gain some vital skills and experience writing and editing CSS code. See the "Getting Started" section at the beginning of the book for more details.

HTML vs. CSS formatting

When comparing HTML-based formatting to CSS-based formatting, it's easy to see how CSS produces vast efficiencies in time and effort. In the following exercise, you'll explore the power and efficacy of CSS by editing two webpages, one formatted by HTML and the other by CSS.

Note

To save ink, screen shots in this and all subsequent lessons were taken using the lightest UI and the Classic code-coloring theme. You are free to use the default dark UI and code theme if you prefer it, or any custom setting of your own choosing. The program and lessons will perform identically in any UI color settings.

- Launch Dreamweaver CC 2019 or later, if it's not currently running.
- Create a new site based on the lesson03 folder, using the instructions in the "Getting Started" section at the beginning of the book. Name the site **lesson03**.
- Choose File > Open.
- Navigate to the lesson03 folder, and open **html_formatting.html**.

- Click the Split view button. If necessary, choose View > Split > Split Vertically to split Code and Live view windows vertically, side by side.

  Note how each element of the content is formatted individually using the deprecated `<font>` tag. Note the attribute `color="blue"` in each `<h1>` and `<p>` element.

Note

Deprecated means that the tag has been formally removed from future support in HTML but may still be honored by current browsers and HTML readers.

Note

Code and Live view windows can be swapped top to bottom and left to right by selecting the option under the View menu. See Lesson 1, "Customizing Your Workspace," for more information.

- Replace the word `"blue"` with **`"green"`** in each line in which it appears. If necessary, click the mouse cursor in the Live view window to update the display.

The text displays in green now in each line where you changed the color value. As you can see, formatting using the obsolete `<font>` tag is not only slow but also prone to error. Make a mistake, like typing **`greeen`** or **`geen`**, and the browser will ignore the color formatting entirely.

- Open **css_formatting.html** from the lesson03 folder.
- If it's not currently selected, click the Split view button.

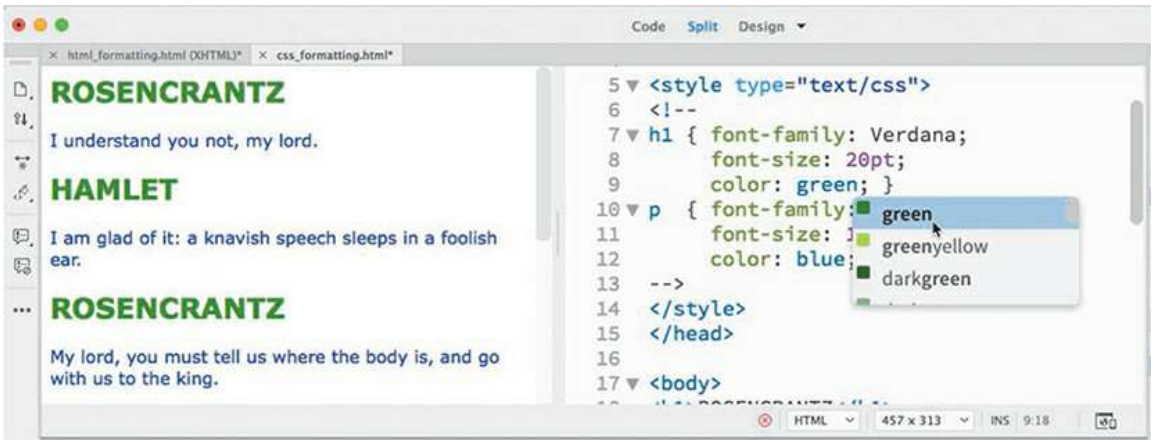
  The content of the file is identical to the previous document, except that it's formatted using CSS. The code that formats the HTML elements appears in the `<head>` section of this file.

Note that the code contains only two `color:blue;` attributes.

. In the code `h1 { color: blue; }` select the word `blue` and type **`green`** to replace it. If necessary, click in the Live view window to update the display.

Note

Dreamweaver usually defaults to Live view when you open or create a new page. If not, you may select it from the Document toolbar using the Live/Design drop-down menu.



In Live view, all the heading elements display in green. The paragraph elements remain blue.

. Select the word `blue` in the code `p { color: blue; }` and type **`green`** to replace it. Click in the Live view window to update the display.

In Live view, all the paragraph elements have changed to green.

. Close all files and do not save the changes.

Note

To get a fuller appreciation of the power and capabilities of CSS, check out the hands-on online CSS bonus lesson included with the book's lesson files. See the "Getting Started" section for details on how to download the bonus lessons.

In this exercise, CSS accomplished the color change with two simple edits, whereas using the HTML `<font>` tag required you to edit every line individually. Now think how tedious it would be to go through thousands of lines of code and hundreds of pages on a site to make such a change. Is it any wonder that the W3C, the web standards organization that establishes Internet specifications and protocols, deprecated the `<font>` tag and developed cascading style sheets?

This exercise highlights just a small sample of the formatting power and productivity enhancements offered by CSS, unmatched by HTML alone.

HTML defaults

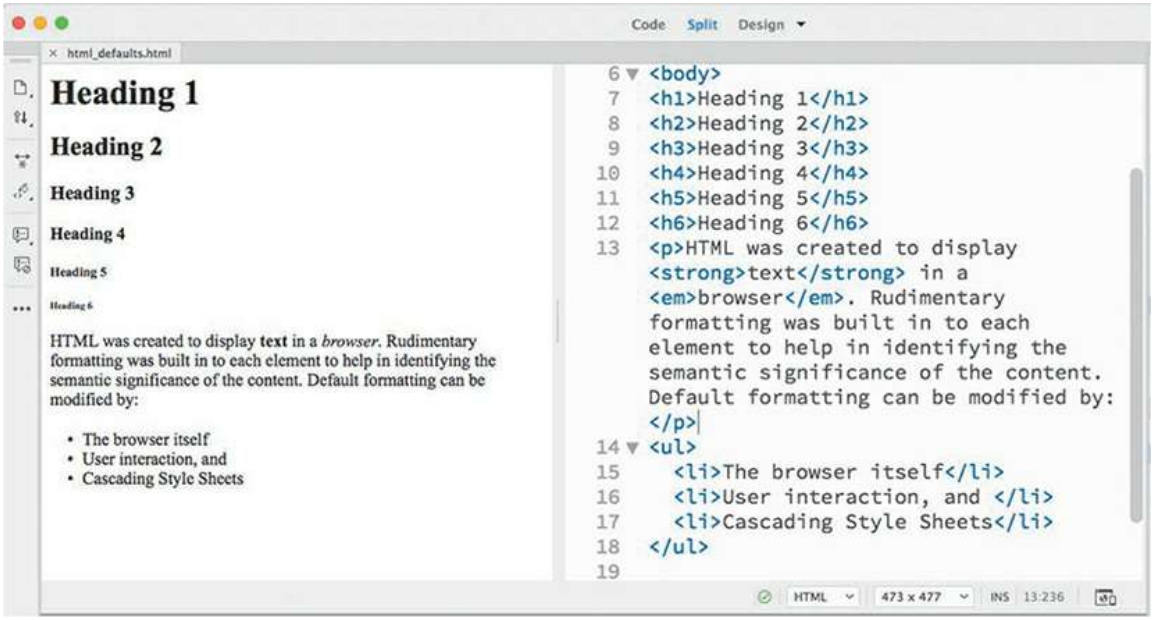
Since the very beginning, HTML tags came right out of the box with one or more default formats, characteristics, or behaviors. So even if you did nothing, much of your text would already be formatted in a certain way in most browsers. One of the essential tasks in mastering CSS is learning and understanding these defaults and how they may affect your content. Let's take a look.

. Open **html_defaults.html** from the lesson03 folder.

  If necessary, select Live view to preview the contents of the file.

  The file contains a range of HTML headings and text elements. Each element visually exhibits basic styling for traits such as size, font, and spacing, among others.

. Switch to Split view.



. In the Code view window, locate the `<head>` section, and try to identify any code that may be formatting the HTML elements.

  A quick look will tell you that there is no overt styling information in the file, yet the text still displays different kinds of formatting. So where does the formatting come from? And, more importantly, what are the settings being used?

. Close **html_defaults.html** and do not save any changes.

The answer is: It depends. In the past, HTML 4 elements drew characteristics from multiple sources. The first place to look is the W3C. It created a default style sheet, which you can find at

www.w3.org/TR/CSS21/sample.html. The style sheet defines the standard formatting and behaviors of all HTML elements. The browser vendors used this style sheet on which to base their default rendering of HTML elements. But that was before HTML5.

HTML5 defaults?

The last decade has seen a consistent movement on the web to separate "content" from its "styling." At the time of this writing, the concept of "default" formatting in HTML seems to be dead. According to specifications adopted by the W3C in 2014, there are no default styling standards for HTML5 elements. If you look for a default style sheet for HTML5 on w3.org—like the one noted for HTML 4—you won't find one. At the moment, there are no public moves to change this relationship, and browser manufacturers are still honoring and applying HTML 4 default styling to HTML5-based webpages. Confused? Join the club.

The ramifications of this trend could be dramatic and wide reaching. Someday, in the not-too-distant future, HTML elements may not display any formatting at all by default. That means that understanding how elements are currently formatted is more important than ever so that you will be ready to develop your own standards if or when the need arises.

To save time and give you a bit of a head start, I pulled together **Table 3.1**, with some of the most common defaults.

Note

If the current trends continue, the lack of an HTML5 default style sheet makes the development of your own site standards even more important.

Table 3.1 Common HTML defaults

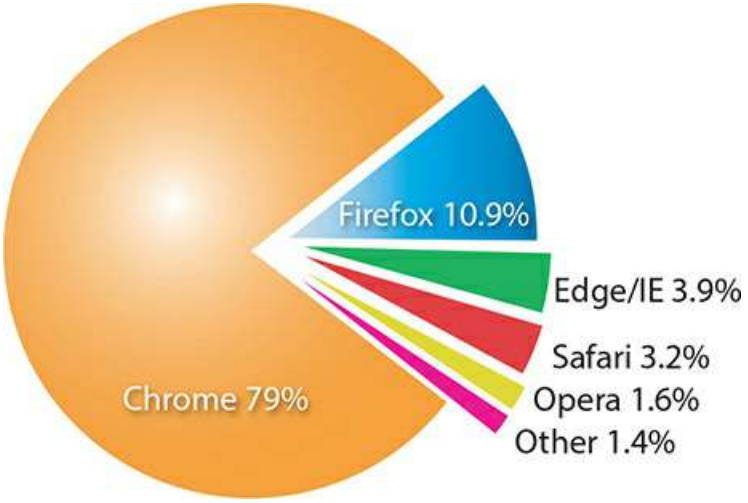
ITEM	DESCRIPTION
Background	In most browsers, the page background color is white. The background of the elements `<div>`, `<table>`, `<td>`, `<th>` and most other tags is transparent.
Headings	Headings `<h1>` through `<h6>` are bold and align to the left. The six heading tags apply differing font size attributes, with `<h1>` the largest and `<h6>` the smallest. Apparent sizes may vary between browsers. Headings and other text elements may also display additional spacing (margins) above or below.
Body text	Outside of a table cell, paragraphs—`<p>`, `<li>`, `<dd>`, `<dt>`—align to the left and start at the top of the page.
Table cell text	Text within table cells, `<td>`, aligns horizontally to the left and vertically to the center.
Table header	Text within header cells, `<th>`, aligns horizontally and vertically to the center (this is not standard across all browsers).
Fonts	Text color is black. Default typeface and font are specified and supplied by the browser, which in turn can be overridden by the user using the preference settings in the browser itself.
Margins	Spacing external to the element border/boundary is handled by margins. Many HTML elements feature some form of margin spacing. Margins are often used to insert additional space between paragraphs and to

	indent text, as in lists and blockquotes.
Padding	Spacing within the box border is handled by padding. According to the default HTML 4 style sheet, no elements feature default padding.

Browser antics

The next task in developing your own styling standards is to identify the browser (and its version) that is displaying the HTML. That's because browsers frequently differ (sometimes dramatically) in the way they interpret, or render, HTML elements and CSS formatting. Unfortunately, even different versions of the same browser can produce wide variations from identical code.

Web design best practices dictate that you build and test your webpages to make sure they work properly in the browsers employed by the majority of web users in general—but especially the browsers preferred by your own visitors. The breakdown of browsers used by your own visitors can differ quite a bit from the norm. They also change over time—especially now, as more and more people abandon desktop computers in favor of tablets and smartphones. In May 2018, the W3C published the following statistics identifying the most popular browsers from the 50 million visitors they receive each year on their website:



Although it's nice to know which browsers are the most popular among the general public, it's crucial that before you build and test your pages you identify the browsers your target audience uses.

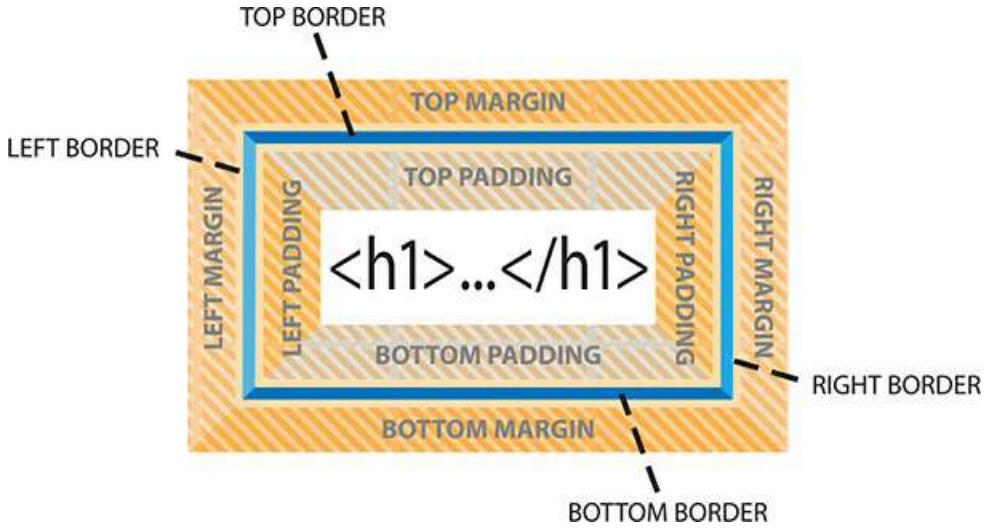
Although this chart shows the basic breakdown in the browser world, it obscures the fact that multiple versions of each browser are still being used. This is important to know because older browser versions are less likely to support the latest HTML and CSS features and effects. To make matters more complicated, these statistics show trends for the Internet overall, but the statistics for your own site may vary wildly.

As HTML5 becomes more widely supported, the inconsistencies will fade, although they may never go away. Some aspects of HTML 4 and CSS 1 and 2 are still not universally agreed upon to this day. It's vital that any styling or structure be tested carefully. Occasionally, you will find

that you must create custom rules to contend with issues in one or more browsers.

CSS box model

Browsers normally read the HTML code, interpret its structure and formatting, and then display the webpage. CSS does its work by stepping between HTML and the browser, redefining how each element should be rendered. It imposes an imaginary box around each element and then enables you to format almost every aspect of how that box and its contents are displayed.



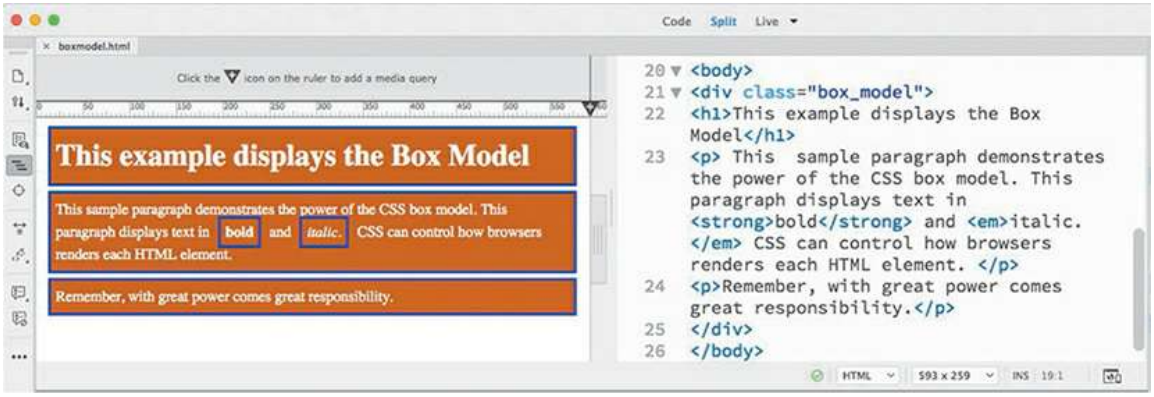
The box model is a programmatic construct imposed by HTML and CSS that enables you to format, or redefine, the default settings of any HTML element.

CSS permits you to specify fonts, line spacing, colors, borders, background shading and graphics, margins, and padding, among other things. Most of the time these boxes are invisible, and although CSS gives you the ability to format them, it doesn't require you to do so.

- Launch Dreamweaver CC 2019 or later, if necessary.

  Open **boxmodel.html** from the lesson03 folder.

- If necessary, switch to Split view.

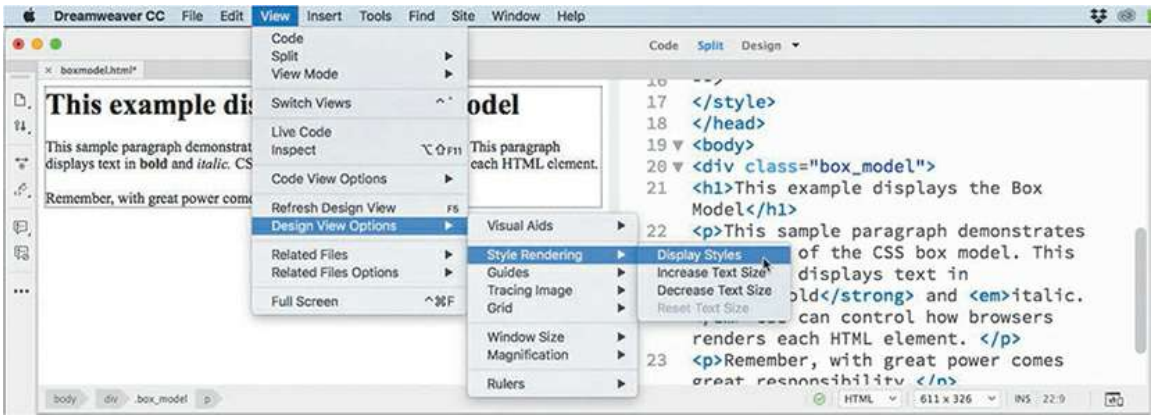


The file's sample HTML code contains a heading and two paragraphs with sample text formatted to illustrate some of the properties of the CSS box model. The text displays visible borders, background colors, margins, and padding.

To see the real power of CSS, sometimes it's helpful to see what the page would look like without CSS.

. Switch to Design view.

Choose View > Design View Options > Style Rendering > Display Styles to disable style rendering.



Dreamweaver now displays the page without any applied styling. A basic tenet in web standards today is the separation of the *content* (text, images, lists, and so on) from its *presentation* (formatting). Although the text now is not wholly unformatted, it's easy to see the power of CSS to transform HTML code. Whether the text is formatted or not, this illustrates the importance of the structure and *quality* of your content. Will people still be enthralled by your website if all the wonderful formatting were pulled away?

Note

The Style Rendering command is available only in Design view.

. Choose View > Design View Options > Style Rendering > Display Styles to enable the CSS rendering in Dreamweaver again.

. Close all files, and do not save changes.

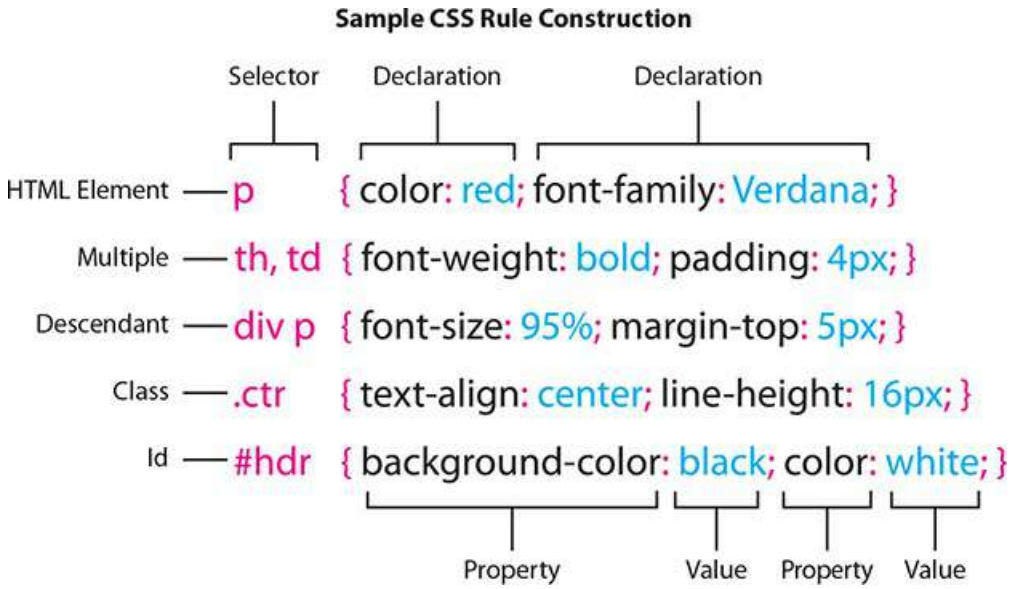
The working specifications found at www.w3.org/TR/css3-box describe how the box model is supposed to render documents in various media.

Applying CSS styling

You can apply CSS formatting in three ways: *inline* (on the element itself), *embedded* (in an

internal style sheet), or *linked* (via an external style sheet). A CSS formatting instruction is known as a *rule*. A rule consists of two parts—a *selector* and one or more *declarations*. The selector specifies what element, or combination of elements, is to be formatted; declarations contain the styling information. CSS rules can redefine any existing HTML element, as well as define two custom element modifiers, named “class” and “id.”

A rule can also combine selectors to target multiple elements at once or to target specific instances within a page where elements appear in unique ways, such as when one element is nested within another.



These sample rules demonstrate some typical constructions used in selectors and declarations. The way the selector is written determines how the styling is applied and how the rules interact with one another.

Applying a CSS rule is not a simple matter of selecting some text and applying a paragraph or character style, as in Adobe InDesign or Adobe Illustrator. CSS rules can affect single words, paragraphs of text, or combinations of text and objects. A single rule can affect an entire webpage, a single paragraph, or just a few words or letters. Basically, anything that has an HTML tag on it can be styled, and there is even an HTML tag specifically intended to style content that has no tag.

Many factors come into play in how a CSS rule performs its job. To help you better understand how it all works, the following sections illustrate four main CSS concepts, which I’ll refer to as theories: cascade, inheritance, descendant, and specificity.

Cascade theory

The cascade theory describes how the order and placement of rules in the style sheet or on the page affects the application of styling. In other words, if two rules conflict, which one wins out?

Take a look at the following rules that might appear in a style sheet:

```
p { color: red; }
p { color: blue; }
```

Both rules apply text color to the paragraph <p> tag. Since they style the same element, they both cannot win. According to the cascade theory, the rule declared last, or *closest* to the HTML code, wins. That means, in this case, the text would appear in blue.

CSS rule syntax: write or wrong

CSS is a powerful adjunct to HTML. It has the power to style and format any HTML element, but the language is sensitive to even the smallest typo or syntax error. Miss a period, comma, or semicolon and you may as well have left the code out of your page entirely. Even worse, an error in one rule may cancel all the styling in subsequent rules or the entire style sheet.

For example, take the following simple rule:

```
p { padding: 1px;
    margin: 10px; }
```

It applies both padding and margins to the paragraph <p> element.

This rule can also be written properly without spacing as:

```
p{padding:1px;margin:10px;}
```

The spaces and line breaks used in the first example are unnecessary, merely accommodations for the humans who may write and read the code. Removing excess spacing is known as *minification* and is often used to optimize style sheets. Browsers and other applications processing the code do not need this extra space, but the same cannot be said of the various punctuation marks sprinkled throughout the CSS.

Use parentheses `( )` or brackets `[ ]` instead of braces `{ }`, and the rule (and perhaps your entire style sheet) is useless. The same goes for the use of colons "`:`" and semicolons "`;`" in the code.

Can you catch the error in each of the following sample rules?

Click here to view code image

```
p { padding; 1px: margin; 10px: }
p { padding: 1px; margin: 10px; ]
p { padding 1px, margin 10px, }
```

Similar problems can arise in the construction of compound selectors too. For example, putting a space in the wrong place can change the meaning of a selector entirely.

The rule `article.content { color: #F00 }` formats the `<article>` element and all its children in this code structure:

```
<article class="content"><p>...</p></article>
```

On the other hand, the rule `article .content { color: #F00 }` would ignore the previous HTML structure altogether, and format only the <p> element in the following code:

Click here to view code image

```
<article class="content"><p class="content">...</p>
</article>
```

A tiny error can have dramatic and far-reaching repercussions. To keep their CSS and HTML functioning properly, good web designers keep their eyes peeled for any little error, misplaced space, or punctuation mark. As you work through the following exercises, keep a careful eye on all the code for any similar errors. As mentioned in the "Getting Started" section, some instructions in this book may omit an expected period or other punctuation in a sentence on purpose when including it might cause confusion or possible code errors.

When you try to determine which CSS rule will be honored and which formatting will be applied, browsers typically honor the following order of hierarchy, with number 4 being the most powerful:

. Browser defaults.
. External or embedded style sheets. If both are present, the one declared last supersedes the earlier entry in conflicts.
. Inline styles (within the HTML element itself).
. Styles with the value attribute `!important` applied.

Inheritance theory

The inheritance theory describes how an element can be affected by one or more rules at the same time. Inheritance can affect rules of the same name as well as rules that format *parent* elements—ones that contain other elements. Take a look at the following code:

Click here to view code image

```
<article>
  <h1>Pellentesque habitant</h1>
  <p>Vestibulum tortor quam</p>
  <h2>Aenean ultricies mi vitae</h2>
  <p>Mauris placerat eleifend leo.</p>
  <h3>Aliquam erat volutpat</h3>
  <p>Praesent dapibus, neque id cursus.</p>
</article>
```

The code contains various headings and paragraph elements and one parent element

<article> that contains them all. If you wanted to apply blue to all the text, you could use the following set of CSS rules:

Click here to view code image

```
h1 { color: blue;}
h2 { color: blue;}
h3 { color: blue;}
p { color: blue;}
```

That's a lot of code all saying the same thing, something most web designers typically want to avoid. This is where inheritance comes into play to save time and effort. Using inheritance you can replace all four lines of code with:

```
article { color: blue;}
```

That's because all the headings and paragraphs are children of the article element; they each inherit the styling applied to their parent, as long as there are no other rules overriding it. Inheritance can be of real assistance in economizing the amount of code you have to write to style your pages. But it's a double-edged sword. As much as you can use it to style elements intentionally, you also have to keep an eye out for unintentional effects.

Descendant theory

Inheritance provides a means to apply styling to multiple elements at once, but CSS also provides the means to target styling to specific elements.

The descendant theory describes how formatting can target specific elements based on their position relative to other elements. This technique involves the creation of a selector name that identifies a specific element, or elements, by combining multiple tags and, in some cases, id and class attributes.

Take a look at the following code:

Click here to view code image

```
<section><p>The sky is blue</p></section>
<div><p>The forest is green.</p></div>
```

Notice how both paragraphs contain no intrinsic formatting or special attributes, although they do appear in different parent elements. Let's say you wanted to apply blue to the first line and green to the second. You would not be able to do this using a single rule targeting the <p> tag alone. But it's a simple matter using descendant selectors, like these:

```
section p { color: blue;}
div p { color: green;}
```

See how two tags are combined in each selector? The selectors identify a specific kind of

element structure, or hierarchy, to format. One targets `p` tags that are children of `section` tags, the other `p` tags that are children of `div` tags. It's not unusual to combine multiple tags within a selector to tightly control how the styling is applied.

In recent years, a set of special characters has been developed to hone this technique to a fine edge. For example, use a plus (+) sign like this **`section+p`** to target only the first paragraph that appears after a `<section>` tag. Use the tilde (~) like this **`h3~ul`** to target unordered lists that are preceded by an `<h3>` tag. Check out www.w3schools.com/cssref/css_selectors.asp to see the full set of special selector characters and wild cards and how to use them. But be careful using these special characters. Many of them were added only in the last few years and still have limited support.

Specificity theory

Conflicts between two or more rules are the bane of most web designers' existence and can waste hours of time in troubleshooting CSS formatting errors. In the past, designers would have to spend hours manually scanning style sheets and rules one by one, trying to track down the source of styling errors.

Specificity describes how browsers determine what formatting to apply when two or more rules conflict. Some refer to this as *weight*—giving certain rules higher priority, or more weight, based on order (cascade), proximity, inheritance, and descendant relationships. One way to make it easier to see a selector's weight is by giving numeric values to each component in the name.

For example, each HTML tag gets 1 point, each class gets 10 points, each id gets 100 points, and inline style attributes get 1000 points. By adding up the component values within each selector, its specificity can be calculated and compared to another, and the higher specific weight wins.

Calculating specificity

Can you do the math? Look at the following list of selectors and see how they add up. Look through the list of rules appearing in the sample files in this lesson. Can you determine the weight of each of those selectors and figure out which rule is more specific on sight?

Click here to view code image

```
* (wildcard) { } 0 + 0 + 0 + 0    = 0    points
h1          { } 0 + 0 + 0 + 1    = 1    point
ul li       { } 0 + 0 + 0 + 2    = 2    points
.class      { } 0 + 0 + 10 + 0   = 10   points
.class h1   { } 0 + 0 + 10 + 1   = 11   points
a:hover     { } 0 + 0 + 10 + 1   = 11   points
#id         { } 0 + 100 + 0 + 0  = 100  points
#id.class   { } 0 + 100 + 10 + 0 = 110  points
#id.class h1 { } 0 + 100 + 10 + 1 = 111  points
style=" "   { } 1000 + 0 + 0 + 0 = 1000 points
```

As you have learned in this lesson, CSS rules often don't work alone. They may style more than one HTML element at a time and may overlap or inherit styling from one another. Each of the theories described so far has a role to play in how CSS styling is applied through your webpage and across your site. When the style sheet is loaded, the browser will use the following hierarchy—with number 4 being the most powerful—to determine how the styles are applied, especially when rules conflict:

. Cascade
. Inheritance
. Descendant structure
. Specificity

Of course, knowing this hierarchy doesn't help much when you are faced with a CSS conflict on a page with dozens or perhaps hundreds of rules and multiple style sheets. Luckily, Dreamweaver has two powerful tools that can help you in this endeavor. The first one we'll look at is named Code Navigator.

Code Navigator

Code Navigator is a tool within Dreamweaver that allows you to instantly inspect an HTML element and assess its CSS-based formatting. When activated, it displays all the embedded and externally linked CSS rules that have some role in formatting a selected element, and it lists them in the order of their cascade application and specificity. Code Navigator works in all Dreamweaver-based document views.

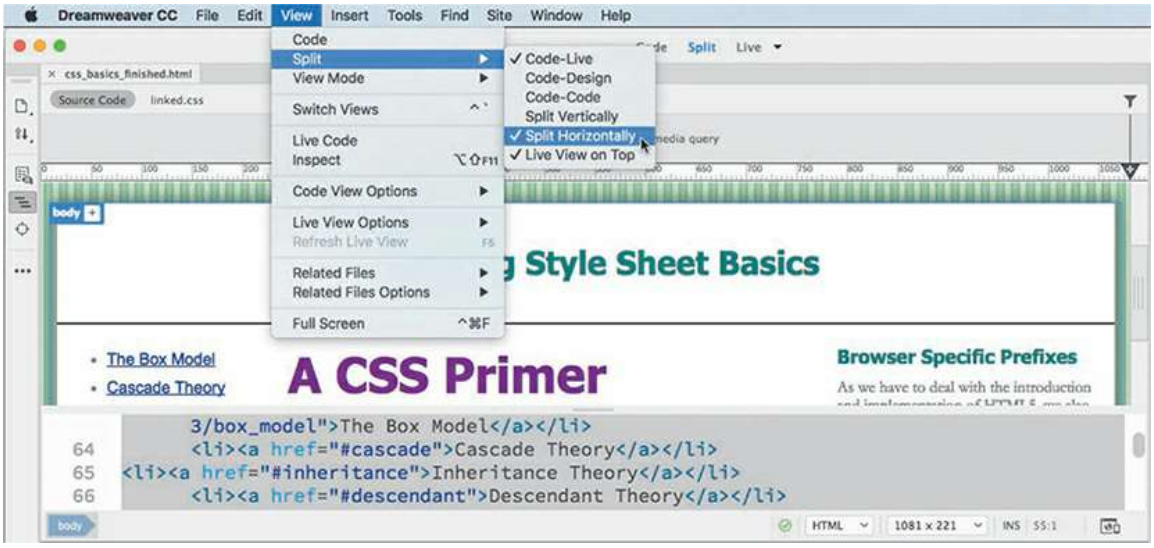
. Open **css_basics_finished.html** from the lesson03 folder.

  Since you were using Split view with the previous webpage, it should still be selected when the new file opens. One window shows Code view and the other shows Design view.

. Select Live view in the Document toolbar.

  Depending on the size of your computer display, you may want to split the screen horizontally to see the entire page width at once.

. Select View > Split > Split Horizontally.

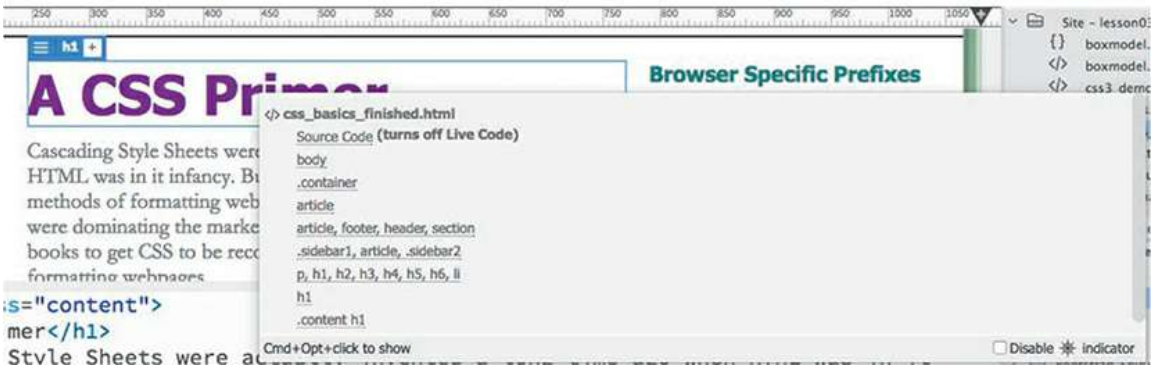


The screen shot shows the Live view window on top.

- In Split view, observe the CSS code and the structure of the HTML content. Then, note the appearance of the text in the Live view window.

  The page contains headings, paragraphs, and lists in various HTML5 structural elements, such as `article`, `section`, and `aside`, styled by CSS rules appearing in the `<head>` section of the code.

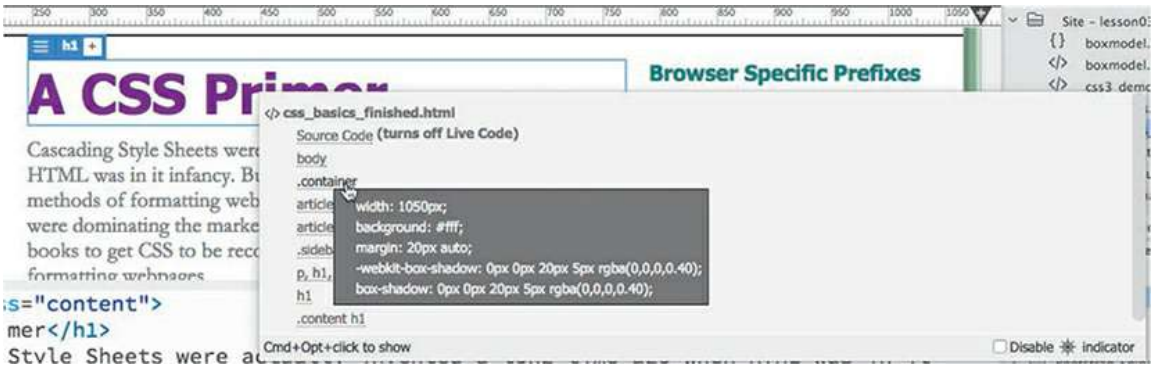
- In Live view, insert the cursor into the heading “A CSS Primer.” Press Ctrl+Alt+N/Cmd+Opt+N.



A small window appears, displaying a list of eight CSS rules that apply to this heading.

This is how you access Code Navigator in Live view. You can also right-click any element and select Code Navigator from the context menu.

If you position the pointer over each rule in turn, Dreamweaver displays any properties formatted by the rule and their values. The rule with the highest specificity (most powerful) is at the bottom of the list.



Unfortunately, Code Navigator doesn't show styling applied via inline styles, so you'll have to check for these types of properties separately and calculate the effect of inline styles in your head. Otherwise, the sequence of rules in the list indicates both their cascade order and their specificity.

When rules conflict, rules farther down in the list override rules that are higher up. Remember that elements may inherit styling from one or more rules, and default styling—that which is not overridden—may still play a role in the final presentation. Unfortunately, Code Navigator doesn't show what, if any, default styling characteristics may still be in effect. You have to figure that out for yourself.

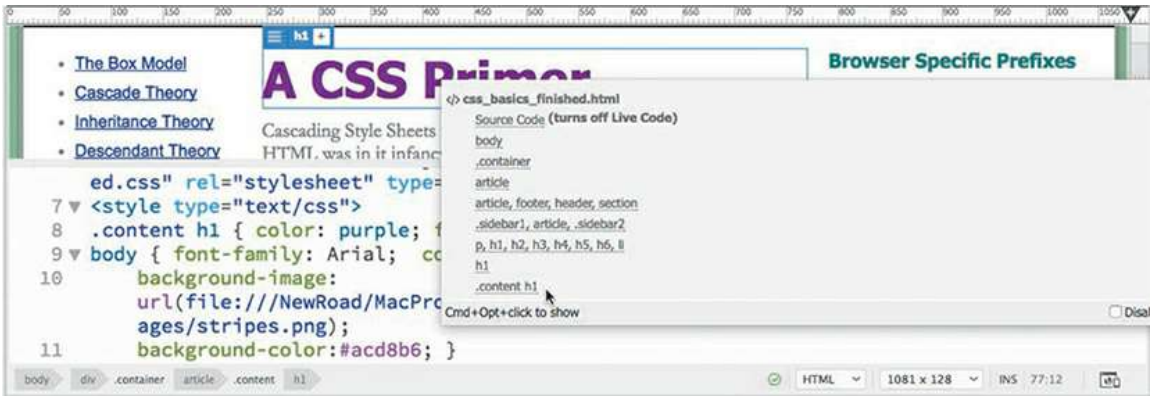
In this case, the `.content h1` rule appears at the bottom of the Code Navigator window, indicating that its specifications are the most powerful ones styling this element. But many factors can influence which of the rules may win. Sometimes the specificity of two rules is identical; then, it's simply the order (cascade) in which rules are declared in the style sheet that determines which one is actually applied.

As described earlier, changing the order of rules can often affect how the rules work. There's a simple exercise you can perform to determine whether a rule is winning because of cascade or specificity.

- In the Code view window, locate the `.content h1` rule (around line 13) and click the line number.

  Clicking the line number selects all the code on that line.
- Press Ctrl+X/Cmd+X to cut the line.
- Insert the cursor at the beginning of the style sheet (line 8). Press Ctrl+V/Cmd+V to paste the line at the top of style sheet.
- Click in the Live view window to refresh the display, if necessary.

  The styling did not change.
- Click the text of the heading "A CSS Primer" to select it and activate Code Navigator, as you did in step 5.



Tip

Code Navigator may be disabled by default. To have it display automatically, deselect the Disable option in the Code Navigator window when it's visible.

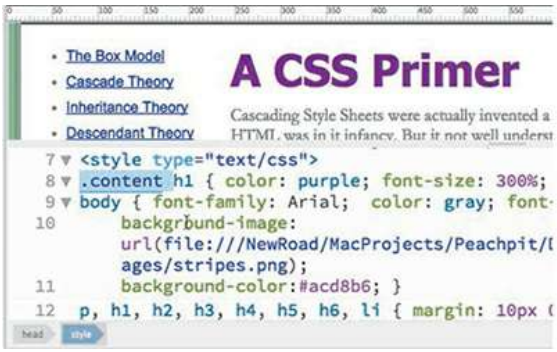
Although the rule was moved to the top of the style sheet—the weakest position—the order of the rules in Code Navigator did not change. In this case, cascade was not responsible for the power of the rule. The `.content h1` selector has a specificity higher than either the `body` or `h1` selectors. In this instance, it would win no matter where it was placed in the code. But you can change its specificity by simply modifying the selector.

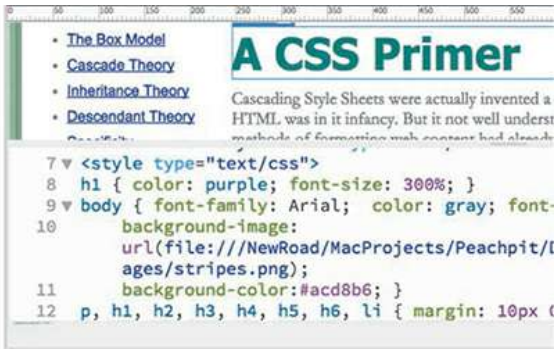
Note

Don't forget to delete the leading period indicating the class name.

- Select and delete the .content class notation from the `.content h1` selector.
- Click in the Live view window to refresh the display, if necessary.

  Did you notice how the styling changed? The "A CSS Primer" heading reverted to the color teal, and the other `h1` headings scaled to 300 percent. Do you know why this happened?
- Click the heading "A CSS Primer" to select it and activate Code Navigator.





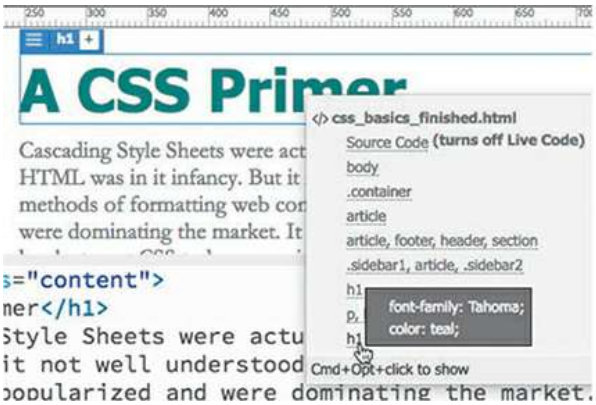
Because you removed the class notation from its selector, it now has equal value to the other `h1` rule, but since it is the first one declared, it loses precedence by virtue of its cascade position.

. Using Code Navigator, examine and compare the rules applied to the headings “A CSS Primer” and “Creating CSS Menus.”

Note

Code Navigator doesn’t display inline CSS rules. Since most CSS styling is not applied this way, it’s not much of a limitation, but you should still be aware of this blind spot as you work with Code Navigator.

Code Navigator shows the same rules applied to both.



Because the `.content` class was removed from the selector, the rule no longer targets only `h1` headings in the `<article class="content">` element; it’s now styling all `h1` elements on the page.

. Choose Edit > Undo to restore the `.content` class to the `h1` selector.

Refresh the Live view display.

All the headings return to their previous styling.

- Insert the pointer in the heading “Creating CSS Menus” and activate Code Navigator.

  The heading is no longer styled by the `.content h1` rule.

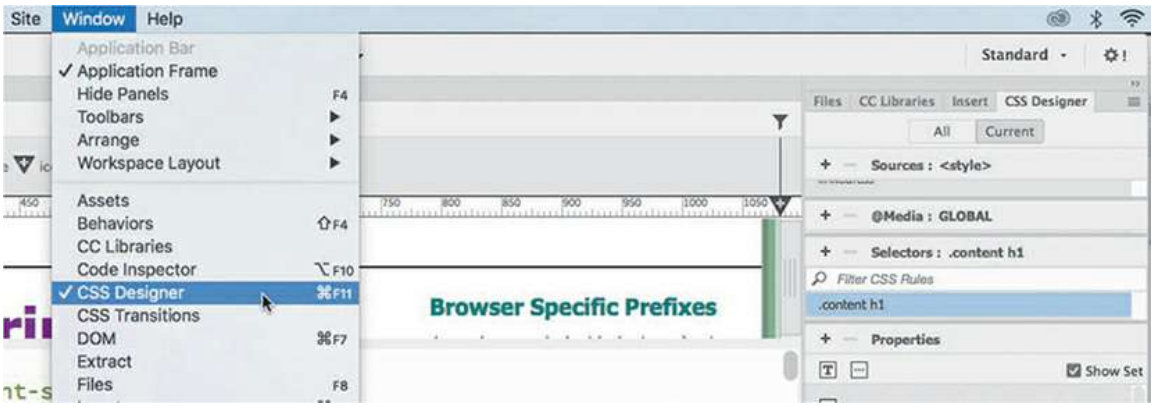
Is it starting to make more sense? Don’t worry, it will—over time. Until that time, just remember that the rule appearing last in Code Navigator has the most influence on any particular element.

CSS Designer

Code Navigator was introduced a while ago and has been an invaluable aid for troubleshooting CSS formatting. Yet a newer tool in Dreamweaver’s CSS arsenal is much more than a good troubleshooting tool. CSS Designer not only displays all the rules that pertain to any selected element but also allows you to create and edit CSS rules at the same time.

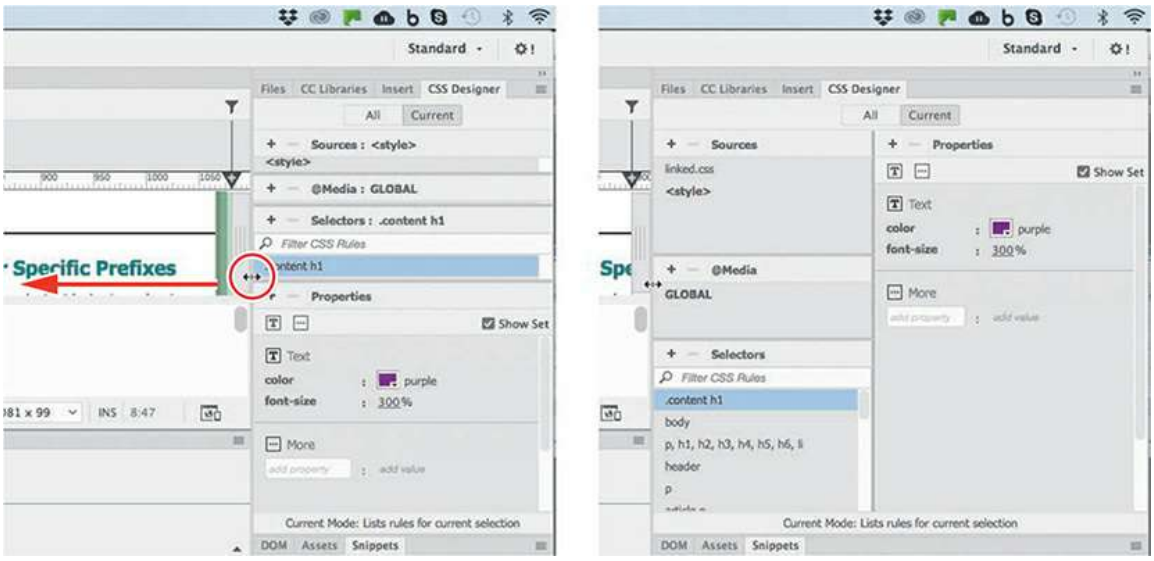
When you use Code Navigator, it shows you the relative importance of each rule, but you still have to access and assess the effect of all the rules to determine the final effect. Since some elements can be affected by a dozen or more rules, this can be a daunting task for even a veteran web coder. CSS Designer eliminates this pressure altogether by computing the final CSS display for you. And best of all, unlike Code Navigator, CSS Designer can even compute the effects of inline styles too.

- If necessary, open **css_basics_finished.html** in Split view.
- If you do not see the panel, choose Window > CSS Designer to display the panel.



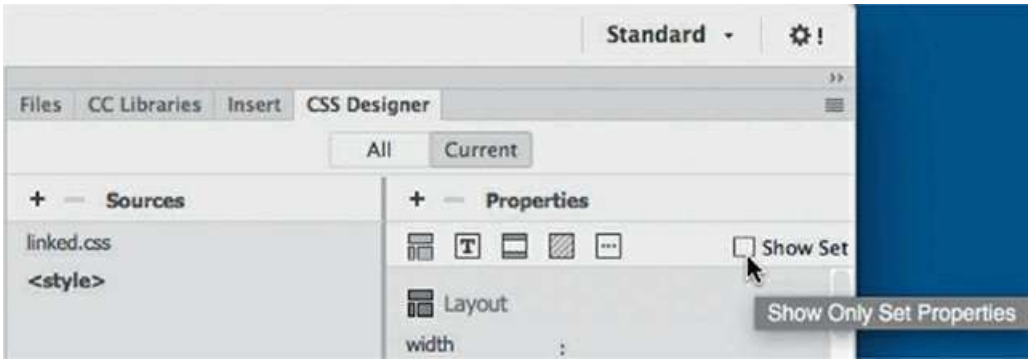
The CSS Designer panel features four panes: Sources, @Media, Selectors, and Properties. Feel free to adjust the heights and widths of the panes as needed. The panel is also responsive—it will take advantage of any extra screen space by splitting into two columns if you increase the panel’s width.

- If you do not see two columns in the CSS Designer, drag the left edge of the panel to the left to increase its width.



The CSS Designer will split into two columns, displaying the Sources, @Media, and Selectors panes on the left and the Properties pane on the right.

. If necessary, deselect the Show Set checkbox in the CSS Designer.



Show Set is disabled by default when Dreamweaver is installed, and if you are a beginner with CSS you may want leave it disabled until you become more comfortable with the language.

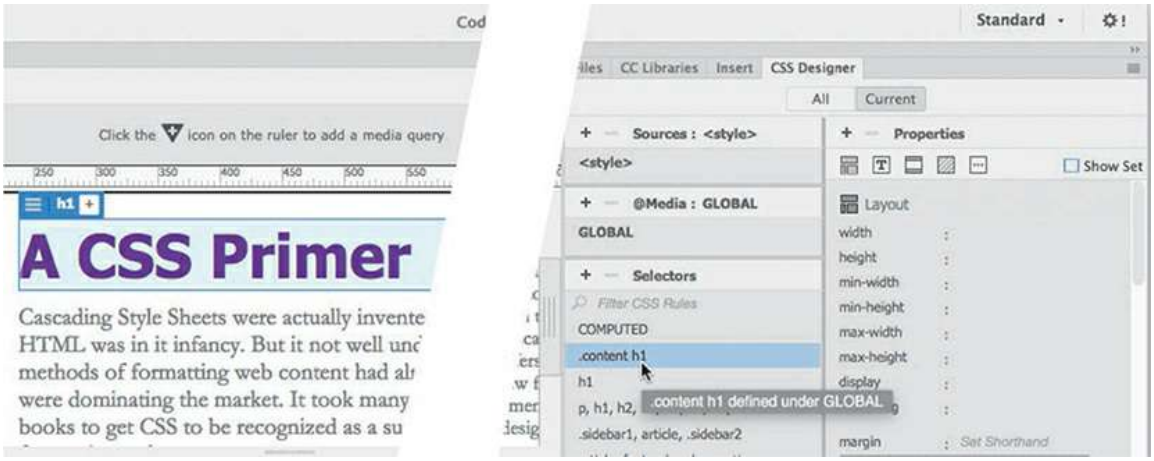
. Select the heading “A CSS Primer” in Live view.

CSS Designer has two basic modes: *All* and *Current*. When All mode is engaged, the panel allows you to review and edit all existing CSS rules and create new rules. In Current mode, the panel allows you to identify and edit the rules and styling already applied to a selected element.

. If necessary, click the Current button in the CSS Designer panel.

When Current mode is active, the panel displays the CSS rules that are affecting the selected heading. In CSS Designer, the most powerful rules appear at the top of the Selectors window, the opposite of Code Navigator.

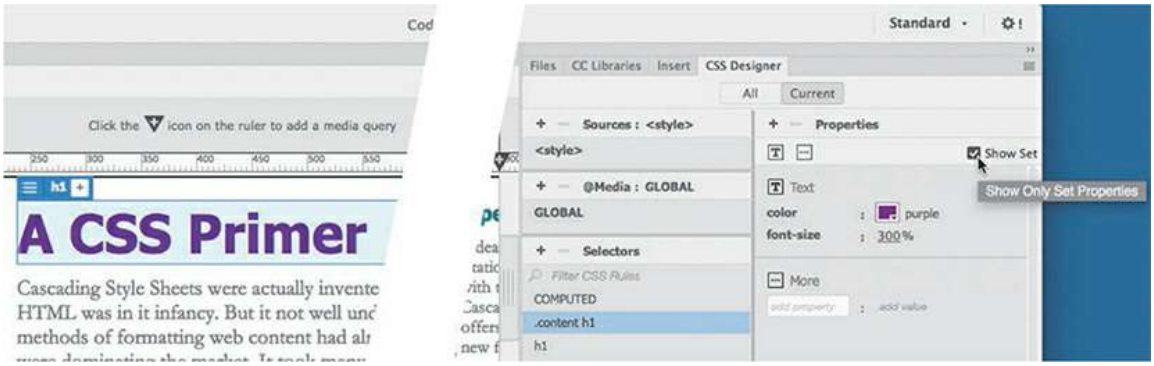
. Click the rule `.content h1` in the Selectors panel.



By default, the Properties window of CSS Designer displays a list of properties that you can style for this element. The list is not exhaustive, but it contains most of the properties you will need.

Showing a seemingly endless list of properties can be confusing as well as inefficient. For one thing, it makes it difficult to differentiate the properties that are assigned from those that aren't. Luckily, CSS Designer allows you to limit the display to only the properties currently applied to the selected element.

- Click the Show Set option in the CSS Designer panel menu to enable it.



When Show Set is enabled, the Properties panel shows only the items that have been set in that rule.

- Select each rule that appears in the Selectors window, and observe the properties of each.

Some rules may set the same properties, whereas others will set different properties. To weed out the conflicts and see the expected result of all the rules combined, select the COMPUTED option.

The COMPUTED option analyzes all the CSS rules affecting the element and generates a list of properties that should be displayed by browsers or HTML readers. By displaying a list of pertinent CSS rules and then computing how the CSS should render, CSS Designer does Code Navigator one step better. But it doesn't stop there.

Although Code Navigator allows you to select a rule and then edit it in Code view, CSS

Designer lets you edit the CSS properties right inside the panel itself. Best of all, CSS Designer can even compute *and* edit *inline* styles.

. Select COMPUTED in the Selectors window.

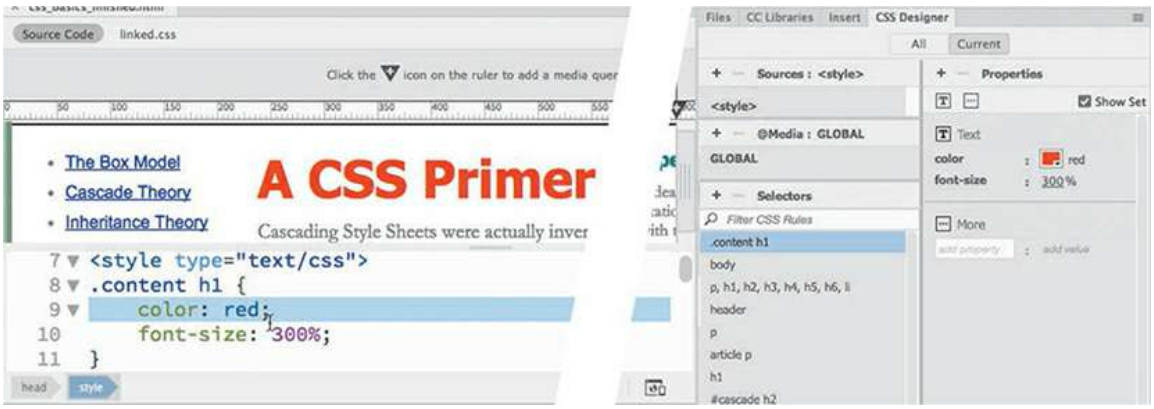
In the Properties window, select the `color` property `purple`.

Enter **`red`** in the field, and press Enter/Return to complete the change.

The heading displays in red. What you may not have noticed is that the change you made was actually entered directly in the rule that originally contributed the styling in the first place.

. In the Code view window, scroll to the embedded style sheet, and examine

the `.content h1` rule.



As you can see, the color was changed within the code and added to the proper rule.

. Close all files and do not save any changes.

Tip

Click to edit the text-based color name. You can also select colors by using the color picker.

In upcoming exercises, you'll get the chance to experience all aspects of CSS Designer as you learn more about cascading style sheets.

Multiples, classes, and ids, oh my!

By taking advantage of the cascade, inheritance, descendant, and specificity theories, you can target formatting to almost any element anywhere on a webpage. But CSS offers a few more ways to optimize and customize the formatting and increase your productivity even further.

Applying formatting to multiple elements

To speed things up, CSS allows you to apply formatting to multiple elements at once by listing each in the selector, separated by commas. For example, the formatting in these rules:

Click here to view code image

```
h1 { font-family:Verdana; color:gray; }
h2 { font-family:Verdana; color:gray; }
h3 { font-family:Verdana; color:gray; }
```

can also be expressed like this:

```
h1, h2, h3 { font-family:Verdana; color:gray; }
```

Using CSS shorthand

Although Dreamweaver will write most of the CSS rules and properties for you, at times you will want, or need, to write your own. All properties can be written out fully, but many can also be written using a shorthand method. Shorthand does more than make the job of the web designer easier; it reduces the total amount of code that has to be downloaded and processed. For example, when all properties of margins or padding are identical, such as:

Click here to view code image

```
margin-top:10px;
margin-right:10px;
margin-bottom:10px;
margin-left:10px;
```

the rule can be shortened to `margin:10px;`

When the top and bottom and left and right margins or padding are identical, like this:

Click here to view code image

```
margin-top:0px;
margin-right:10px;
```

```
margin-bottom:0px;
margin-left:10px;
```

it can be shortened to **`margin:0px 10px;`**

But even when all four properties are different, like this:

Click here to view code image

```
margin-top:20px;
margin-right:15px;
margin-bottom:10px;
margin-left:5px;
```

they can still be shortened to **`margin:20px 15px 10px 5px;`**

In these three examples, you can see clearly how much code can be saved using shorthand. There are far too many references and shorthand techniques to cover here. Check out **tinyurl.com/shorten-CSS** to get a full description.

Throughout the book I'll use common shorthand expressions wherever possible; see if you can identify them as we go.

Creating class attributes

So far, you've learned that you can create CSS rules that format specific HTML elements and ones that can target specific HTML element structures or relationships. In some instances, you may want to apply unique formatting to an element that is already formatted by one or more existing rules. To accomplish this, CSS allows you to make your own custom attributes named *class* and *id*.

Class attributes may be applied to any number of elements on a page, whereas id attributes can appear only once per page. If you are a print designer, think of classes as being similar to a combination of Adobe InDesign's paragraph, character, table, and object styles all rolled into one. Class and id names can be a single word, an abbreviation, any combination of letters and numbers, or almost anything, but they may not contain spaces. In HTML 4, ids could not start with a number. There doesn't seem to be any similar restrictions in HTML5. For backward compatibility, you should probably avoid starting class and id names with numbers.

Although there's no strict rule or guideline on how to create them, classes should be more general in nature, and ids should be more specific. Everyone seems to have an opinion, but at the moment there is no absolutely right or wrong answer. However, most agree that they should be descriptive, such as `"co-address"` or `"author-bio"` as opposed to `"left-column"` or `"big-text"`. This will especially help improve your site analytics. The more sense Google and other search engines can make of your site's structure and organization, the higher your site will rank in the search results.

To declare a CSS class selector, insert a period before the name within the style sheet, like this:

```
.content
.sidebar1
```

Then, apply the CSS class to an entire HTML element as an attribute, like this:

```
<p class="intro">Type intro text here.</p>
```

Or to individual characters or words using the `<span>` tag, like this:

Click here to view code image

```
<p>Here is <span class="copyright">some text formatted
differently</span>.</p>
```

Creating id attributes

HTML designates `id` as a unique attribute. Therefore, any id should be assigned to no more than one element per page. In the past, many web designers used id attributes to style or identify specific components within the page, such as the header, the footer, or specific articles. With the advent of HTML5 elements—`header`, `footer`, `aside`, `article`, and so on—the use of id and class attributes for this purpose became less necessary. But ids can still be used to identify specific text elements, images, and tables to assist you in building powerful hypertext navigation within your page and site. You will learn more about using ids this way in Lesson 9, "Working with Navigation."

To declare an id attribute in a CSS style sheet, insert a number sign, or hash mark, before the name, like this:

```
#cascade
#box_model
```

Here's how you apply the CSS `id` to an entire HTML element as an attribute:

Click here to view code image

```
<div id="cascade">Content goes here.</div>
<section id="box_model">Content goes here.</section>
```

Or to a portion of an element:

Click here to view code image

```
<p>Here is span id="copyright">some text</span> formatted
differently.</p>
```

CSS3 features and effects

CSS3 has over two dozen new features. Many have been implemented in all the modern

browsers and can be used today; others are still experimental and are supported less fully. Among the new features, you will find

- Rounded corners and border effects
- Box and text shadows
- Transparency and translucency
- Gradient fills
- Multicolumn text elements

You can implement all these features and more via Dreamweaver today. The program will even assist you in building vendor-specific markup when necessary. To give you a quick tour of some of the coolest features and effects brewing, I've provided a sample of CSS3 styling in a separate file.

. Open **css3_demo.html** from the lesson03 folder.

  Display the file in Split view and observe the CSS and HTML code.

  Some of the new effects can't be previewed directly in Design view. You'll need to use Live view or an actual browser to get the full effect.

. If necessary, activate Live view to preview all the CSS3 effects in the Live view window.

The file contains a hodgepodge of features and effects that may surprise and even delight you—but don't get too excited. Although many of these features are already supported in Dreamweaver and will work fine in modern browsers, there's still a lot of older hardware and software out there that can turn your dream site into a nightmare. And there's at least one additional twist.

Some of the new CSS3 features have not been standardized, and certain browsers may not recognize the default markup generated by Dreamweaver. In these instances, you may have to include specific vendor commands to make them work properly, such as `-ms-`, `-moz-`, and `-`

`webkit-`. If you look carefully in the code of the demo file, you'll be able to find examples of these within the CSS markup. Can you think of ways of using some of these effects in your own pages?

Note

When writing new CSS3 properties that still require vendor prefixes today, place standard properties last. That way, when the subject browser finally supports the standard specifications, their cascade position will allow them to supercede the other settings.

CSS3 overview and support

The Internet doesn't stand still for long. Technologies and standards are evolving and changing constantly. The members of the W3C have been working diligently to adapt the web to the latest realities, such as powerful mobile devices, large flat-panel displays, and HD images and video—all of which seem to get better and cheaper every day. This is the urgency that currently drives the development of HTML5 and CSS3.

Many of these new standards have not been officially defined yet, and browser vendors are implementing them in varying ways. But don't worry. This version of Dreamweaver, as always, has been updated to take advantage of the latest changes. This includes ample support for the current mix of HTML5 elements and CSS3 properties. As new features and capabilities are developed, you can count on Adobe to add them to the program as quickly as possible using Creative Cloud.

As you work through the lessons that follow, you will be introduced to and actually implement many of these new and exciting techniques in your own sample pages.

Additional CSS support

CSS formatting and application is so complex and powerful that this short lesson can't cover all aspects of the subject. For a full examination of CSS, check out the following books:

CSS3: The Missing Manual (4th Edition), David Sawyer McFarland (O'Reilly Media, 2015) ISBN: 978-1-491-91801-2

CSS Secrets: Better Solutions to Everyday Web Design Problems, Lea Verou (O'Reilly Media, 2015) ISBN: 978-1-449-37263-7

HTML5 & CSS3 for the Real World (2nd Edition), Alexis Goldstein, Louis Lazaris, and Estelle Weyl (SitePoint Pty. Ltd., 2015) ISBN: 978-0-987-46748-5

Stylin' with CSS: A Designer's Guide (3rd Edition), Charles Wyke-Smith (New Riders Press, 2012) ISBN: 978-0-321-85847-4

Review questions

1. Should you use HTML-based formatting?
2. What does CSS impose on each HTML element?
3. True or false? If you do nothing, HTML elements will feature no formatting or structure.
4. What four “theories” affect the application of CSS formatting?
5. True or false? All CSS3 features are experimental, and you shouldn’t use them at all.

Review answers

1. No. HTML-based formatting was deprecated in 1997, when HTML 4 was adopted. Industry best practices recommend using CSS-based formatting instead.
2. CSS imposes an imaginary box on each element. This box, and its content, can then be styled with borders, background colors and images, margins, padding, and other types of formatting.
3. False. Even if you do nothing, many HTML elements feature default formatting.
4. The four theories that affect CSS formatting are cascade, inheritance, descendant, and specificity.
5. False. Many CSS3 features are already supported by modern browsers and can be used right now.

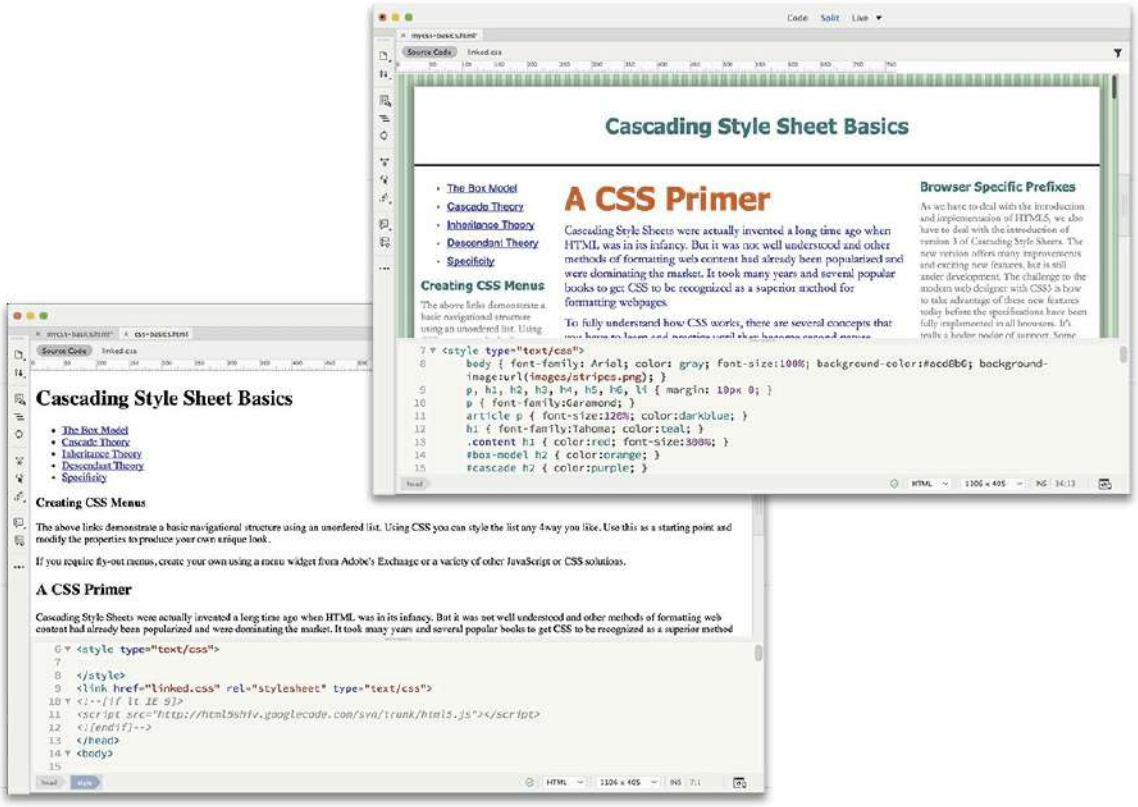
3 CSS Basics Bonus

Lesson overview

In this lesson, you will gain valuable skills working with cascading style sheets and learn the following:

- How to write CSS rules by hand
- How to write different types of selectors based on cascade, inheritance, and descendant theories
- How to format your HTML text and structural elements

This lesson will take about 2 hours to complete. If you have not already done so, please log in to your account on peachpit.com to download the project files for this lesson as described in the “Getting Started” section at the beginning of this book and follow the instructions under “Accessing the Lesson Files and Web Edition.” Define a site based on the lesson03bonus folder.



Key to any understanding of website design is knowledge of and experience with cascading style sheets (CSS).

Previewing the completed file

The best way to learn CSS is by creating your own style sheets. In this online bonus lesson, you'll learn how CSS works by creating a complete style sheet for a sample HTML file like the one you will view in this exercise.

- Define a new site based on the lesson03bonus folder as described in the " Getting Started" section at the beginning of the book.
- Select the Standard workspace in the Workspace menu, if necessary.
- Open the **css-basics-finished.html** file from the finished folder in lesson03bonus.
- If necessary, switch to Split view using Code view and Live view.

  Observe the content and code in the two windows.

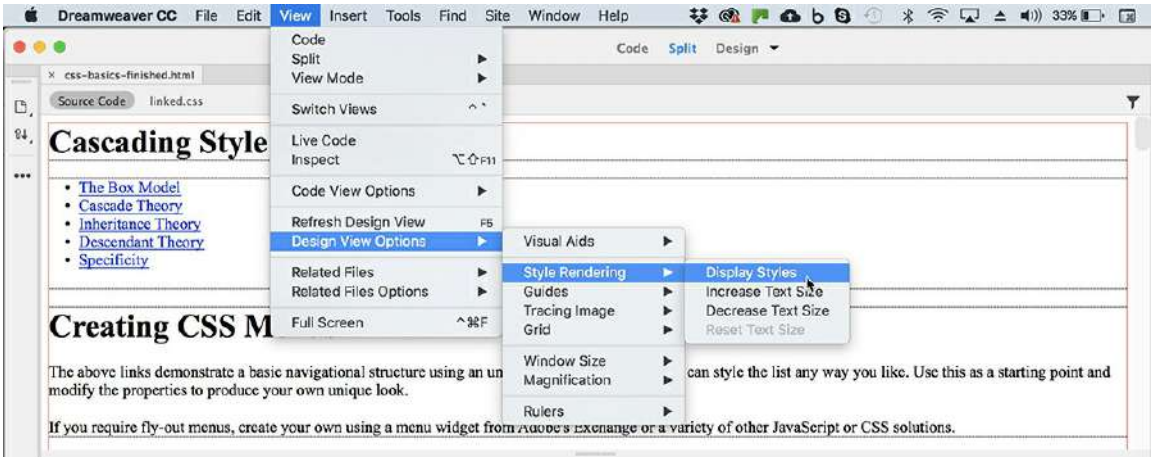
The file contains a complete HTML page with a variety of elements, including headings, paragraphs, lists, and links, all fully formatted by CSS. Note the text styling, as well as the colors and borders assigned to each element. Typically, files will open in Live view or with Live view active in Split view. This is important because some styling may not display properly in Design view alone.

. If necessary, switch to Live view only.

The entire webpage is displayed in one window. Live view displays CSS styling more accurately than does Design view. For example, in Live view you should see a drop shadow around the main content section. As before, the best way to see the true power of CSS is by shutting it off.

. Switch to Design view.

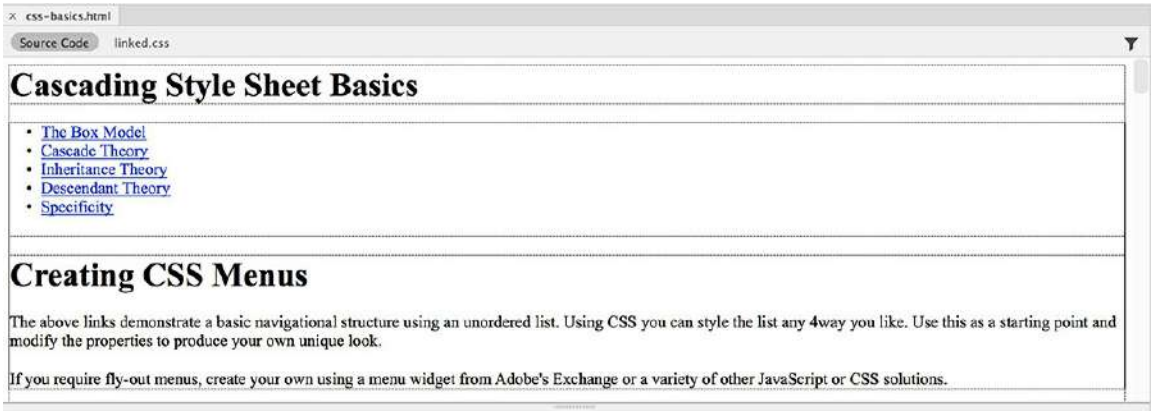
. Choose View > Design View Options > Style Rendering > Display Styles to toggle off the CSS rendering in Dreamweaver.



The page is no longer formatted by CSS and displays only the default styling you would expect on standard HTML elements. The text content now stacks vertically in a single column, with no colors, borders, backgrounds, or shadows.

Note

The Style Rendering option is visible only in Design view.



. Choose View > Design View Options > Style Rendering > Display Styles to toggle on the CSS rendering in Dreamweaver again.

  The CSS styling is turned back on.

. Select Live view.

  Dreamweaver previews the page in Live view again.

. Choose File > Close.

As you can see, CSS can style all aspects of a webpage with amazing variety and detail. First let’s take a look at how CSS can format text.

Formatting text

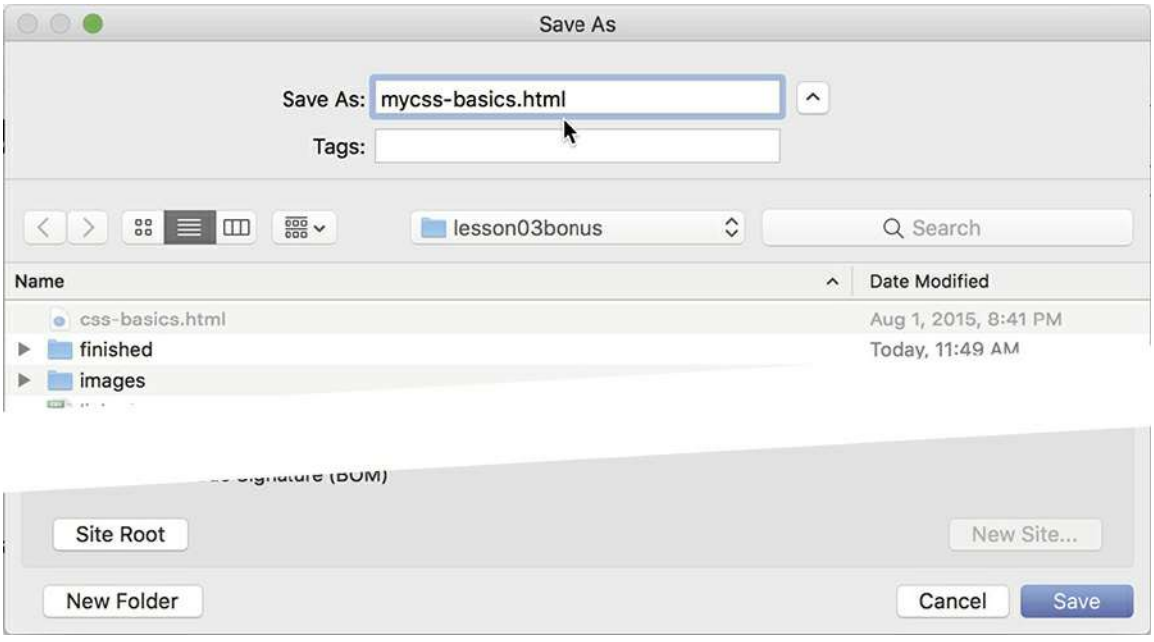
> **Tip**
>
> Dreamweaver CC (2019 release) will open most files in Live view by default. In the following exercises, be aware that certain commands will function only in Design view. Be prepared to switch between Live view and Design view as necessary.

- Open **css-basics.html** from the lesson03bonus folder.
- Switch to Split view, if necessary.

  The file contains an exact copy of the content you reviewed in the finished file in the preview exercise. The page contains various elements, headings, paragraphs, lists, and links that are formatted only by default HTML styling.
- Select File > Save As.

  Name the file **mycss-basics.html.**

  Save the file in the site root folder.



Dreamweaver creates a copy of the file using the new name and displays two tabs at the top of the document window. Since the original file is still open, it could cause confusion during the following exercises. Let's close the original file.

- Select the document tab for the **css-basics.html** file.

  Choose File > Close.

The **mycss-basics.html** file should be the only one open.

. Observe the `<head>` section in the Code view window.

Note that the code contains a `<style>` section but no CSS rules.

. Insert the cursor between the opening `<style>` and closing `</style>` tags (line 7).

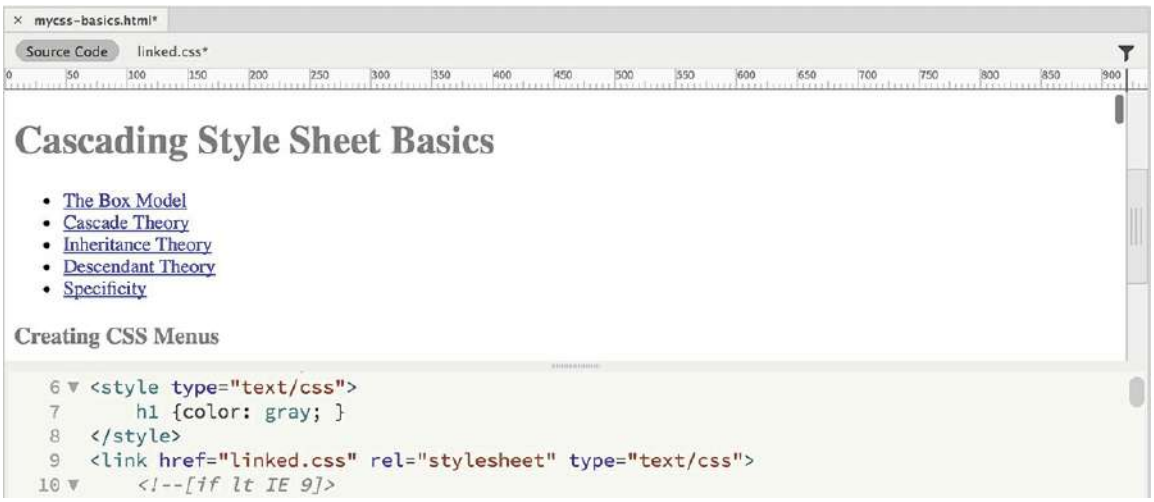
Note

As you type the rule markup, Dreamweaver provides code hints, as it did with the HTML code in bonus Lesson 2, "HTML Basics Bonus." Feel free to use these hints to speed up your typing, or simply ignore them and continue typing.

Tip

Live view should automatically refresh the display when you edit content in the Code window. But if you don't see the display change as described, you may need to refresh the display manually by clicking in the Live view window.

. Type `h1 { color:gray; }` and, if necessary, click in the Live view window to refresh the display.



. Save the file.

  The `h1` headings throughout the page now display in gray. The rest of the text still displays the default formatting. Congratulations! You wrote your first CSS rule.

Cascade theory in action

Once you start writing CSS rules, you will encounter various types of conflicts. Putting your knowledge of CSS theory to the test will help you better understand how to use cascading style sheets and how to troubleshoot the conflicts.

In this exercise, you will learn how cascade theory functions firsthand.

Note

CSS does not require line breaks between rules, but they do make the code easier to read. Feel free to add them between rules or properties at any time.

. Open **mycss-basics.html** in Split view, if necessary.

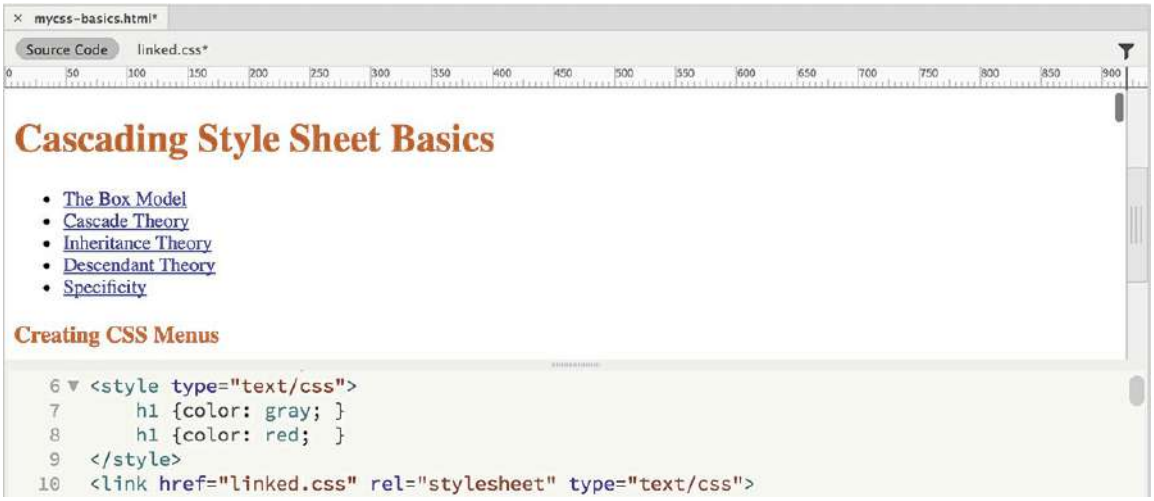
. In the Code window, insert the cursor at the end of the rule created in step 7 of the previous exercise. Press Enter/Return to create a new line.

Note

Remember to insert the new line after the curly brace.

. Type **`h1 { color:red; }`** and click in the Live view window, if necessary, to refresh the

display.



The `h1` headings now display in red. The styling of the new rule supersedes the formatting applied by the first rule. It's important to understand that the two rules are identical except that they apply different colors: red or gray. Both rules want to format the same elements, but only one will be honored.

It's clear the second rule won, but why? In this case, the second rule is the *last* one declared, making it the *closest* one to the actual content. Whether intentional or not, a style applied by one rule may be overridden by declarations in one or more subsequent rules.

. Click the line number containing the rule

`h1 { color: red; }` (line 8).

The entire line is selected. Dreamweaver allows you to drag selections within the code window.

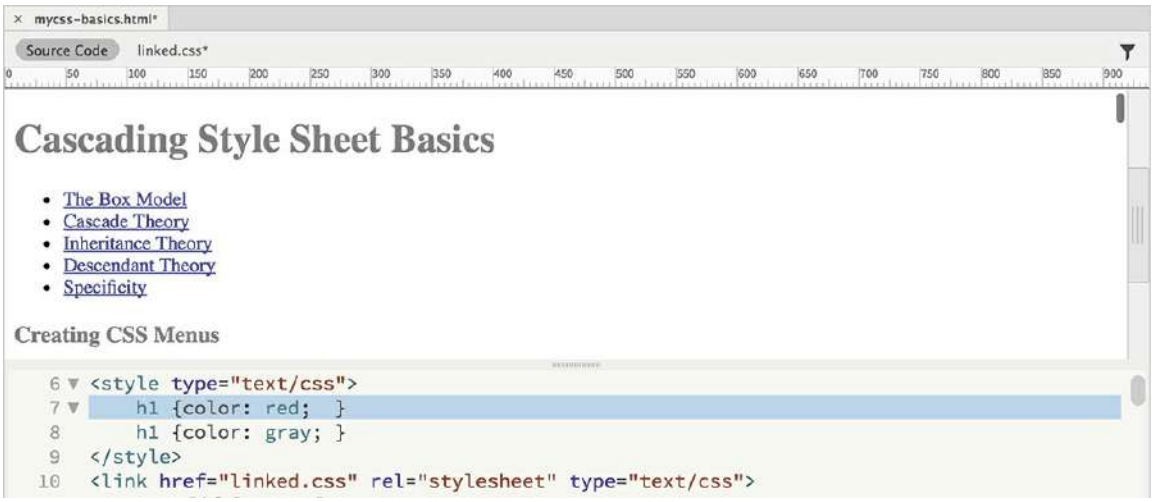
Note

It may take you some practice to learn how to drag and drop code properly. If you make a mistake, select Edit > Undo and try again.

. Drag the selected code to insert it before the markup `h1 { color: gray; }`.

The two rules swap position. The rule applying the gray color now appears last. You have switched the order of the rules.

. If necessary, click in the Live view window to refresh the preview display.



The `h1` headings display in gray again.

Cascade applies to styles whether the rules are embedded in the webpage or located in a separate external, linked style sheet.

. Select **linked.css** in the Related Files interface.

This is an external CSS file linked to the webpage. When you select the name of the referenced file, the contents of that file appear in the Code view window. If the file is stored on a local hard drive, Dreamweaver allows you to edit the contents without actually opening the file. At the moment, the file contains only a character set attribute.

. Insert the cursor in line 2.

Type **`h1 { color:orange; }`** and press Ctrl+S/Cmd+S to save the file.

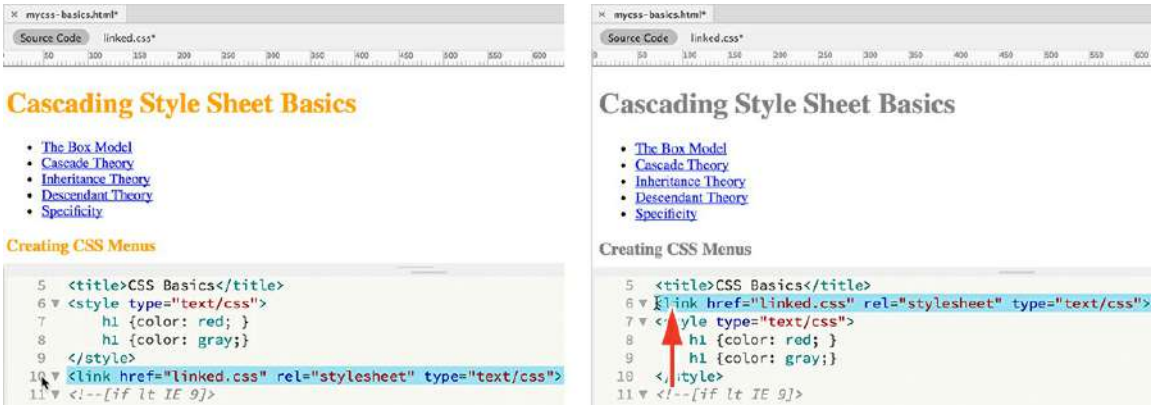
Click in the Live view window to refresh the display, if necessary.

. Select Source Code in the Related Files interface. Locate the `<link>` reference for **linked.css** in the `<head>` section (around line 10).

The `<link>` element appears *after* the closing `</style>` tag. Based strictly on cascade, this

means any rule that appears in the linked file will supersede duplicate rules in the embedded sheet.

- Click the line number for the external CSS `<link>` reference to select the entire reference. Drag the entire `<link>` reference above the `<style>` element.
- Click in the Live view window to refresh the display, if necessary.



The headings revert to gray.

- Choose File > Save All.

As you can clearly see, the order, or *proximity*, of the rules within the markup is a powerful factor in how CSS is applied.

Inheritance theory in action

Inheritance theory speaks to the lineage or parental origins of an element. In other words, if your parent has blue eyes and blond hair, the odds of you also having blue eyes and blond hair are very high.

In this exercise, you will learn how inheritance theory functions firsthand.

- If necessary, open **mycss-basics.html** in Split view.
- Insert the cursor after the rule `h1 { color: gray; }` in the embedded style sheet. Press Enter/Return to insert a new line.
- Type **`h1 { font-family:Arial; }`** and click in the Live view window to refresh the display.

The `h1` elements appear in Arial and gray. The other content remains formatted by default styling.

Now there are four CSS rules that format `<h1>` elements. Can you tell, by looking at the Live view window, which rule, if any, are formatting the `<h1>` text now? If you said *two* of them, you're the winner.

At first glance, you may think that the rules formatting `<h1>` elements are separate from each other. And technically, that's true. But if you look closer, you'll see that the new rule doesn't contradict the others. It's not resetting the `color` attribute, as you did in the previous exercises; it's declaring a new, additional attribute. In other words, since both rules do something different, both will be honored, or inherited, by the `h1` element. All `<h1>` elements will be formatted as gray *and* Arial. Even if the font specification was added to a competing rule, it would still be inherited whether or not the other settings were applied.

Far from being a mistake or an unintended consequence, the ability to build rich and elaborate formatting using multiple rules is one of the most powerful and complex aspects of cascading style sheets.

. Insert the cursor after the last `h1` rule.

Insert a new line in the code.

. Type **`h2 { font-family:Arial; color:gray; }`** and click in the Live view window to refresh the display.

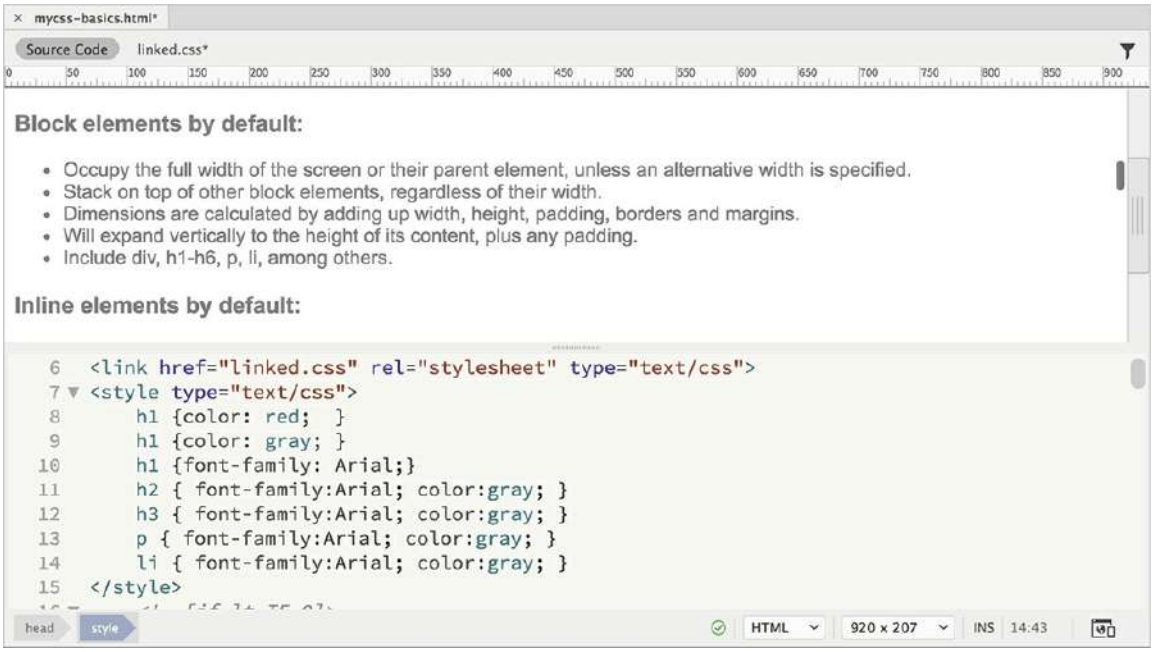
The `h2` elements originally displayed in a serif font in black; now they appear in Arial and gray.

. After the `h2` rule, type the following code:

Click here to view code image

```
h3 { font-family:Arial; color:gray; }
p { font-family:Arial; color:gray; }
```

```
li { font-family:Arial; color:gray; }
```



- If necessary, refresh the Live view window display by clicking in it or by clicking the Refresh button in the Properties panel.

  All the elements now display the same styling, but you used six rules to format the entire page.

  Although CSS styling is far more efficient than the obsolete HTML-based method, inheritance can help you optimize your styling chores even more. For example, all the rules include the statement `{ font-family:Arial; color:gray; }`. Redundant code like this should be avoided whenever possible. It adds to the amount of code in each webpage and adds to the time it takes to download and process it. By using inheritance, sometimes you can create the same effect with a single rule. One way to make your styling more efficient is to apply formatting to a *parent* element instead of to the individual elements themselves.

- Create a new line in the `<style>` section.

  Type the following code:

  Click here to view code image

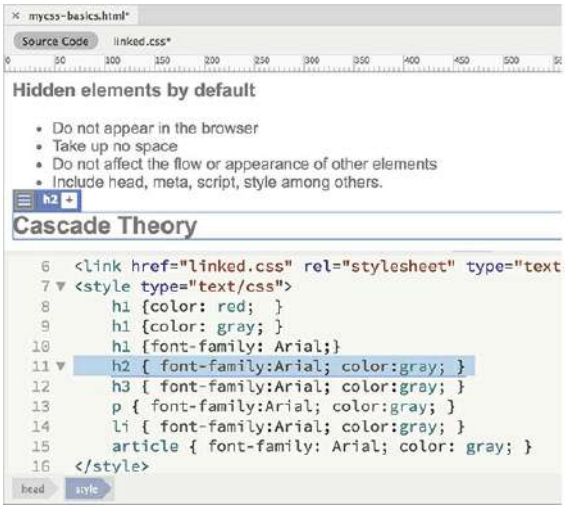
  ```
  article { font-family: Arial; color: gray; }
  ```

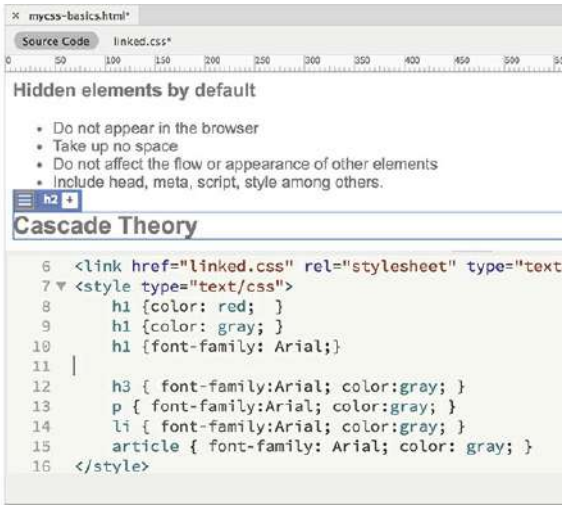
  If you look through the code, you will see that the `<article>` tag contains much but not all of the webpage content. Let's see what happens if you delete some of your CSS rules.

- Select and delete the following rule:

  Click here to view code image

  ```
  h2 { font-family: Arial; color: gray; }
  ```





. Refresh the Live view window display.

The `h2` elements appearing within the `<article>` element remain formatted as gray Arial. The other `h2` element down the page and outside the `<article>` now appears in HTML default styling.

Note

There is no requirement to create rules in any specific order or hierarchy. You may order them any way you please, as long as you keep in mind how the application of styling is governed by cascade and inheritance.

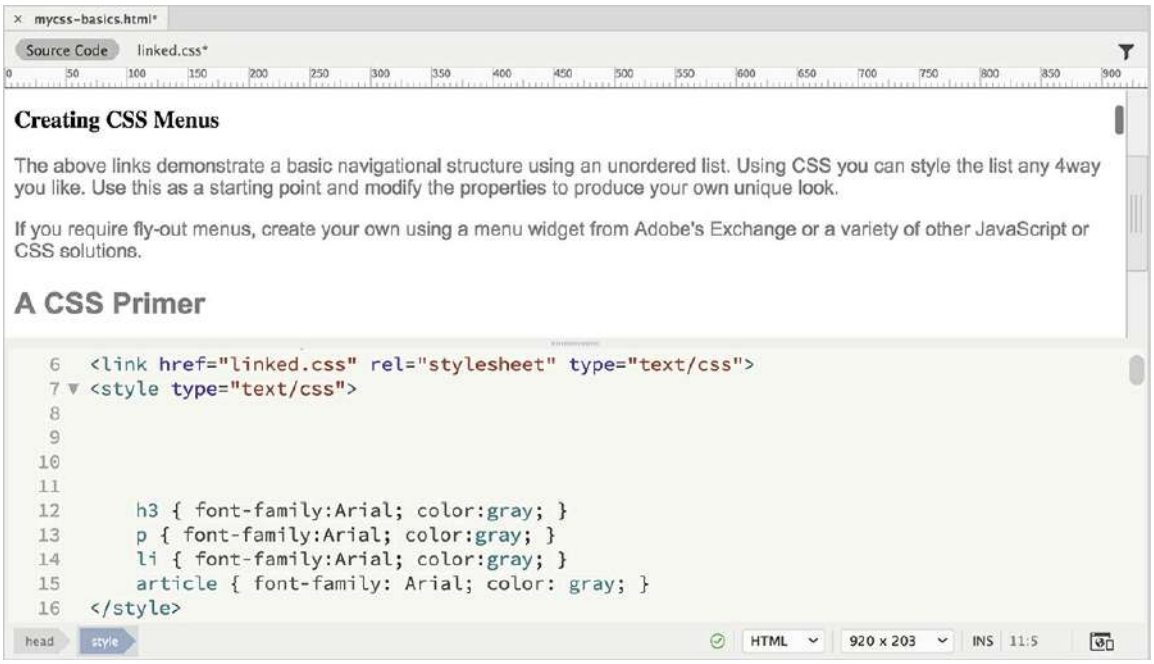
Tip

Rules typically contain multiple property declarations.

. Select and delete all `h1` rules.

Don’t forget the one in the **linked.css** file.

Refresh the Live view display.



The `h1` elements contained within the `<article>` element continue to be styled. Those outside the `<article>` element have reverted to the HTML defaults. Since the new rule targets only the `<article>` element, only the elements contained within it are styled.

. Select and delete the rules formatting `h3`, `p`, and `li`.

Refresh the Live view display.

As in step 11, any content contained in the `<article>` tag remains formatted, while content elsewhere has reverted.

This is the way inheritance works. You could simply re-create the rules to format the other content, but there's a simpler alternative. Instead of adding additional CSS rules, can you figure out a way to use inheritance to format all the content on the page the same way? Hint: Look carefully at the entire structure of the webpage.

Did you choose the `<body>` element? If so, you win again. The `<body>` element contains all the visible content on the webpage and therefore is the parent of all of it.

- Change the rule selector from `article` to **body** and delete any blank lines.

Tip

In this particular page, you could also use the `<div>` element to achieve the same result. But since `<div>` is a frequently used element, it might pose unpredictable conflicts in the future. Since web-pages have only one `<body>` element, it is definitely the preferred target.

- Choose File > Save All.

  Refresh the Live view display, if necessary.

Once again, all the text displays in gray Arial. By using inheritance, all the content is formatted using *one* rule instead of six. You'll find that the `<body>` element is a popular target for setting various default styles.

Note

Dream-weaver frequently edits externally linked files. You will see asterisks by the affected filenames in the Related Files interface. Use Save All whenever changes on your page may affect multiple files in your site.

Descendant theory in action

Although inheritance affects elements automatically, the randomness with which it functions is often frustrating. Styling elements when their parent is formatted is not always predictable or reliable. Descendant theory allows you to target the child element directly and specifically. Although inheritance can affect any child element! —headings, paragraphs, and list items—descendant theory can limit the styling to a specific child element, if desired.

> **Note**
>
> You will examine the role of specificity later in this lesson.

In this exercise, you will learn how descendant theory functions firsthand.

. If necessary, open **mycss-basics.html** in Split view and observe the structure of the HTML content.

  The page contains headings and paragraphs in various HTML5 structural elements, such as `article`, `section`, and `aside`. The rule `body {font-family:Arial; color: gray; }` applies a default font and color to the entire page. In this exercise, you will learn how to create descendant CSS rules to target styling to specific elements in context.

. In Code view, insert the cursor at the end of the `body` rule.

  Press Enter/Return to insert a new line.

. Type **`p { font-family:Garamond; }`** and refresh the Live view display.

> **Note**
>
> Step 3 assumes you have Garamond installed on your computer. If you do not, select another serif font, such as Times.

All `p` elements on the page now display in Garamond. The rest of the page continues to be formatted in Arial.

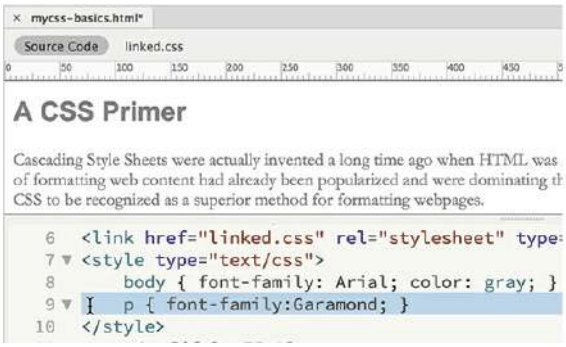
Because you created a selector using the `p` tag, the font formatting applied by the `body` rule has been overridden for all `p` elements no matter where they appear. You may be thinking that since the `p` rule appears *after* the `body` rule, this type of styling simply relates to the cascade order. Let's try an experiment to see whether that's true.

> **Note**
>
> Dragging code can be a difficult skill to master. Feel free to simply cut and paste the code.

. Click the line number for the `p` rule.

  Drag the `p` rule above the `body` rule.

  Refresh the Live view display.



The `p` rule now appears above the `body` rule, but the styling of the paragraphs did not change.

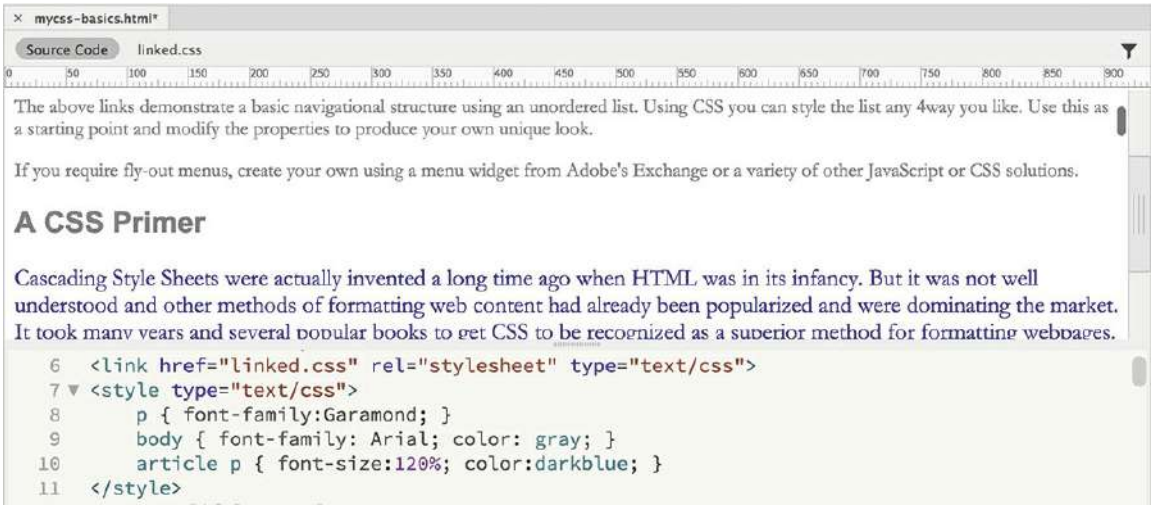
If the styling of `p` elements were determined simply by cascade, you would expect the headings to revert to gray Arial. Yet here, the styling is unaffected by changing the order of the rules. Instead, because you used a more specific tag name in the selector, the new `p` rule becomes more powerful than the generic `body` rule. By properly combining two or more tags in the selector, you can craft the CSS styling on the page in even more sophisticated ways.

. Create a new line after the `body` rule.

On the new line, type the following:

Click here to view code image

```
article p { font-size:120%; color:darkblue; }
```



By adding a `p` tag immediately after `article` in the selector, you are telling the browser to format only `p` elements that are children, or *descendants*, of `article` elements. Remember that a *child* element is one that is contained or nested within another element.

. If necessary, refresh the Live view display.

All paragraphs appearing within the `<article>` element now display in dark blue and 120 percent larger than the other paragraph text on the page.

. Choose File > Save All.

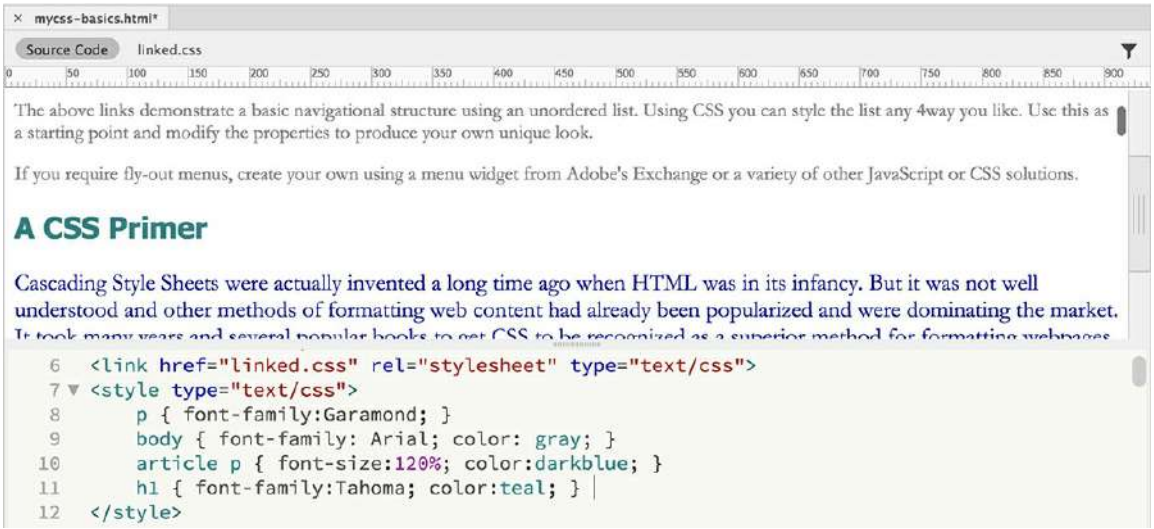
Although it may be hard to understand at this moment, the styling in other rules—both `body` and `p`—is still being inherited by the newly formatted paragraphs. But wherever two or more rules, or portions of a rule, conflict, a descendant selector will win over any less specific styling.

Styling using class and id selectors

In **mycss-basics.html**, all `h1` elements are formatted identically regardless of where they appear in the layout. In the following exercise, you'll use classes and ids to differentiate the styling among the headings.

. Insert a new line after the rule `article p`.

Type **`h1 { font-family:Tahoma; color:teal; }`** and refresh the display.



All `h1` headings now display in the color teal and the font Tahoma.

Although it's tagged identically to the other `h1` headings, "A CSS Primer" is the main heading in the `<article>` element. To make it stand out from the other headings, you can use the class attribute assigned to its parent to target it for special formatting.

Note

Unless otherwise specified, you can add rules anywhere in the style sheet.

. Create the following new rule:

Click here to view code image

```
.content h1 { color:red; font-size:300%; }
```

The heading “A CSS Primer” displays in red and 300 percent larger than the body text.

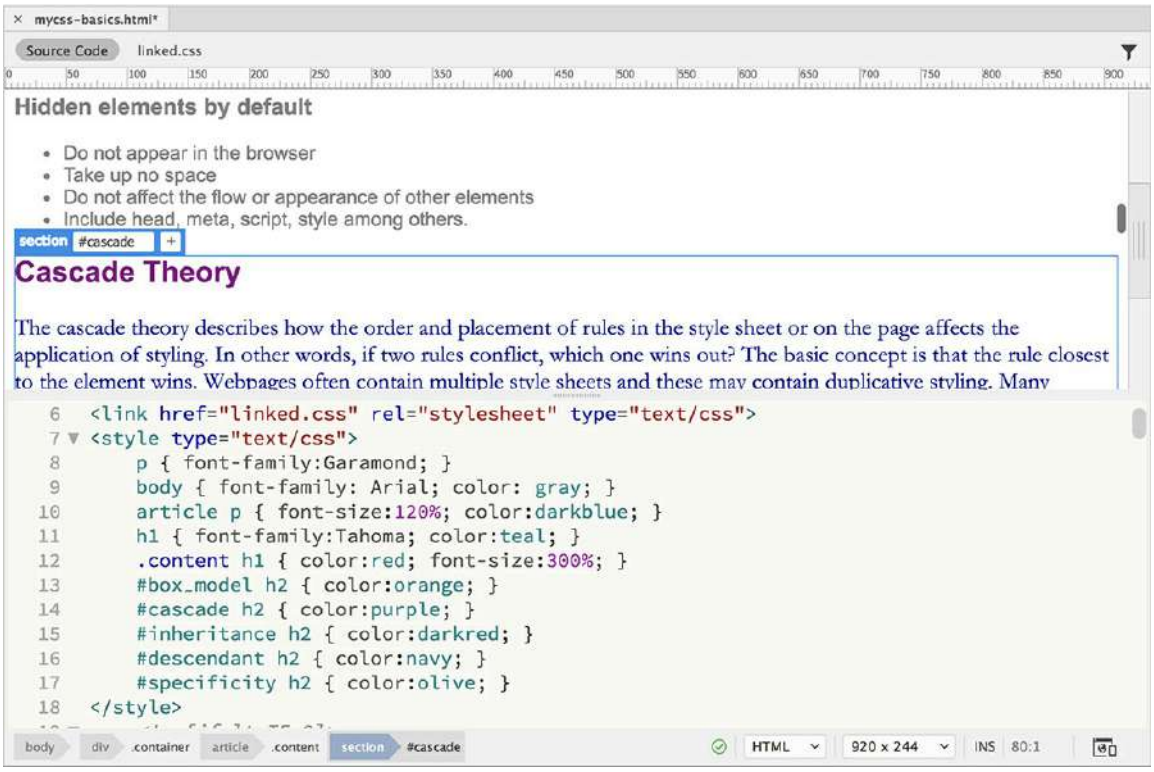
In CSS syntax, the period (`.`) refers to a *class* attribute, and the hash (#) means *id*. By adding `.content` to the selector, you have targeted the styling only to `h1` elements in `<article class="content">`.

In the same way, you can assign custom styling to each of the subheadings (`h2`) by using the id attributes assigned to each of `<section>` elements in the code.

. Create the following rules:

Click here to view code image

```
#box_model h2 { color:orange; }
#cascade h2 { color:purple; }
#inheritance h2 { color:darkred; }
#descendant h2 { color:navy; }
#specificity h2 { color:olive; }
```



. Choose File > Save All.

If necessary, refresh the display.

The `h2` headings targeted by the new rules now display unique colors. What's important to understand here is that all the formatting you see has been applied without adding a single attribute to any of the headings. They are being formatted based solely on their position within the structure of the code.

Understanding descendant styling

CSS formatting can be confusing for designers coming from the print world. Print designers are accustomed to applying styles directly to text and objects, one at a time. In some cases, styles can be based on one another, but this relationship is intentional. In print-based styling, it's impossible for one paragraph or character style to affect another *un*intentionally. On the other hand, in CSS, the chance of one element's formatting overlapping or influencing another's happens all the time.

It may be helpful to think of it as if the elements were formatting themselves. When you use CSS properly, the formatting is intrinsic not to the element but to the entire page and to the way the code is structured.

The ability to separate the content from its presentation is an important concept in modern web design. It allows you great freedom in moving content from page to page and structure to structure without worrying about the effects of residual or latent formatting. Since the formatting doesn't reside with the element itself, the content is free to adapt instantly to its new

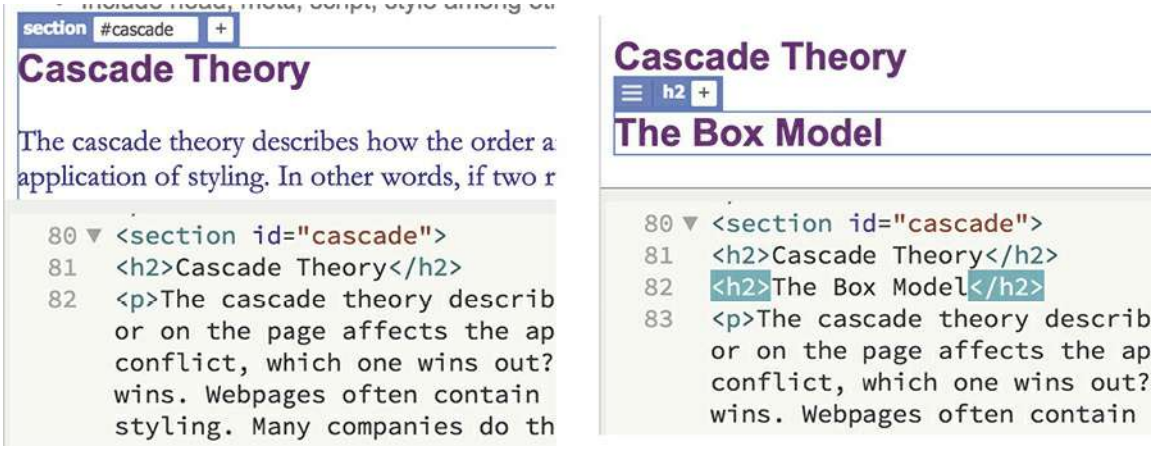
surroundings.

In this exercise, you'll move one of the uniquely styled elements to a different location in the page to see how its position dictates how it is styled.

- In Code view, click the line number for the heading "The Box Model" (around line 50).

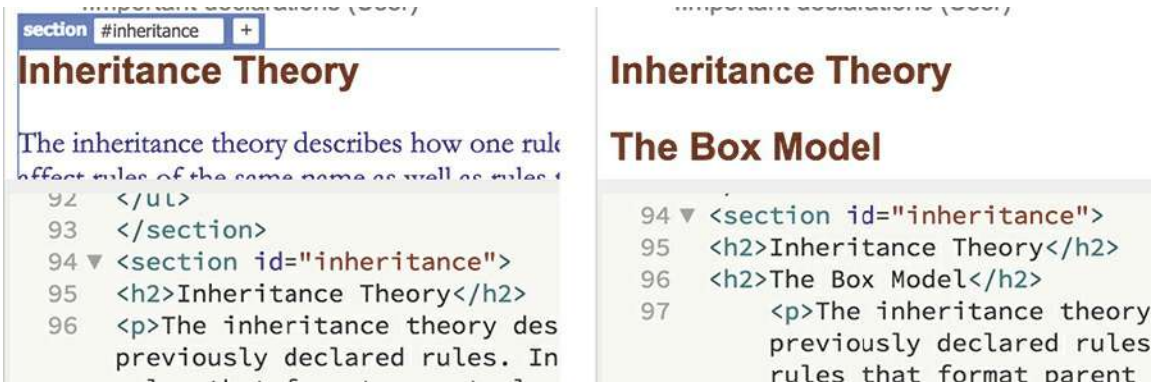
  This should highlight the entire element. Note that the heading is orange.
- Choose Edit > Copy or press Ctrl+C/Cmd+C.
- Insert the cursor at the end of the `h2` element "Cascade Theory" (around line 81) and press Enter/Return to create a new line.
- Click to select the line number for the new blank line.

  Choose Edit > Paste or press Ctrl+V/Cmd+V to replace the blank line.
- Refresh the display, if necessary.



The heading "The Box Model" follows and is formatted identically to the heading "Cascade Theory."

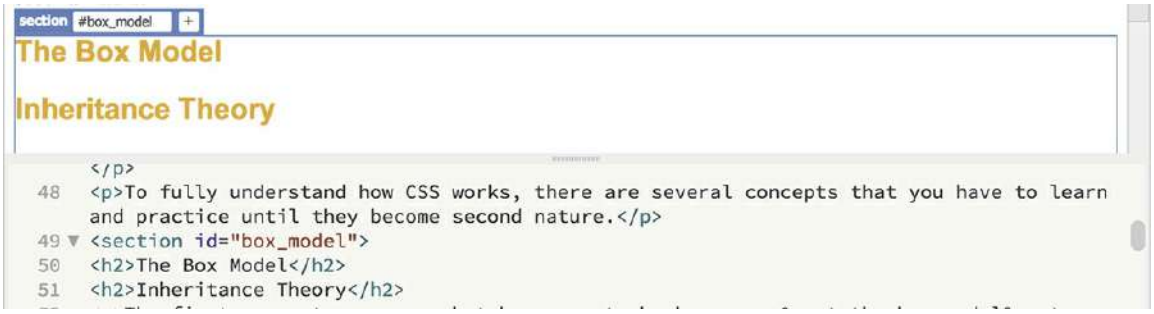
- Insert the cursor at the end of the "Inheritance Theory" heading and create a new line.
- Press Ctrl+V/Cmd+V to paste the heading again.



The pasted heading is dark red and styled identically to the heading " Inheritance Theory."

As you can see, the formatting of the heading in the original instance does not travel with the text pasted in a new location. That's the point of separating content from presentation—you can insert the content anywhere and it will adopt the formatting native to that position. It even works in reverse.

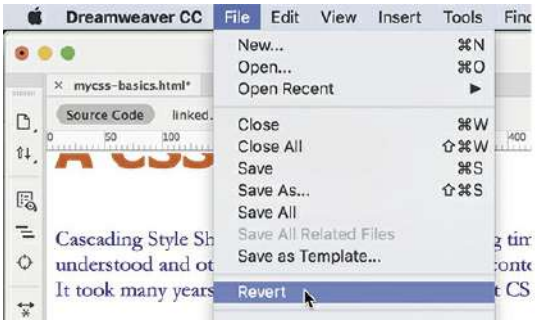
- In Code view, select and copy the "Inheritance Theory" heading.
- Insert the cursor after the original "The Box Model" heading and paste the text on a new line.

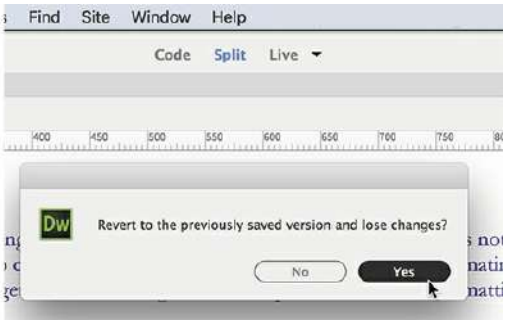


The heading appears and adopts the same styling as "The Box Model."

Once again, the pasted text matches the formatting applied to the other `h2` element within the `<article>`, ignoring its original styling altogether. Now that you've seen how descendant theory works, there's no need for the extra headings.

- Choose File > Revert. Click Yes to revert the file to the previously saved version.





The file reverts to the last saved version. All of the duplicate headings are gone.

Specificity theory in action

As explained in Lesson 3, "CSS Basics," specificity is the means by which brows-ers and other web-compatible applications determine how CSS styling is applied, especially when two or more rules conflict. You could compare rules manually and calculate which rule or rules should win, but Dreamweaver provides a couple of built-in tools that can do that work for you: Code Navigator and CSS Designer.

Using Code Navigator to identify specificity

Code Navigator puts a CSS specificity checker a mouse-click away.

- In Live view, insert the cursor in the heading “A CSS Primer.”
- Right-click the heading and select Code Navigator from the context menu.

  A pop-up window appears listing all the rules formatting the heading. Notice that the display shows three rules (`body`, `h1`, and `.content h1`) in descending order. The order is important because Code Navigator puts the most specific rule at the bottom. In this case, that’s `.content h1`. But that’s not all it does; it can also show you what styling is being applied by each rule.
- Position the cursor over each rule in the pop-up window, but do not click them.

  When the cursor hovers over the rule, another window opens showing the properties and values assigned to that rule. Note that all three rules apply a color specification, but `.content h1` wins. Code Navigator also enables you to edit the CSS rules.
- Click the rule `.content h1`.

The Code view window focuses on the CSS rule `.content h1`, enabling you to edit the CSS if so desired. If the rule were in an external CSS file, the Code view window would load the contents of that file.

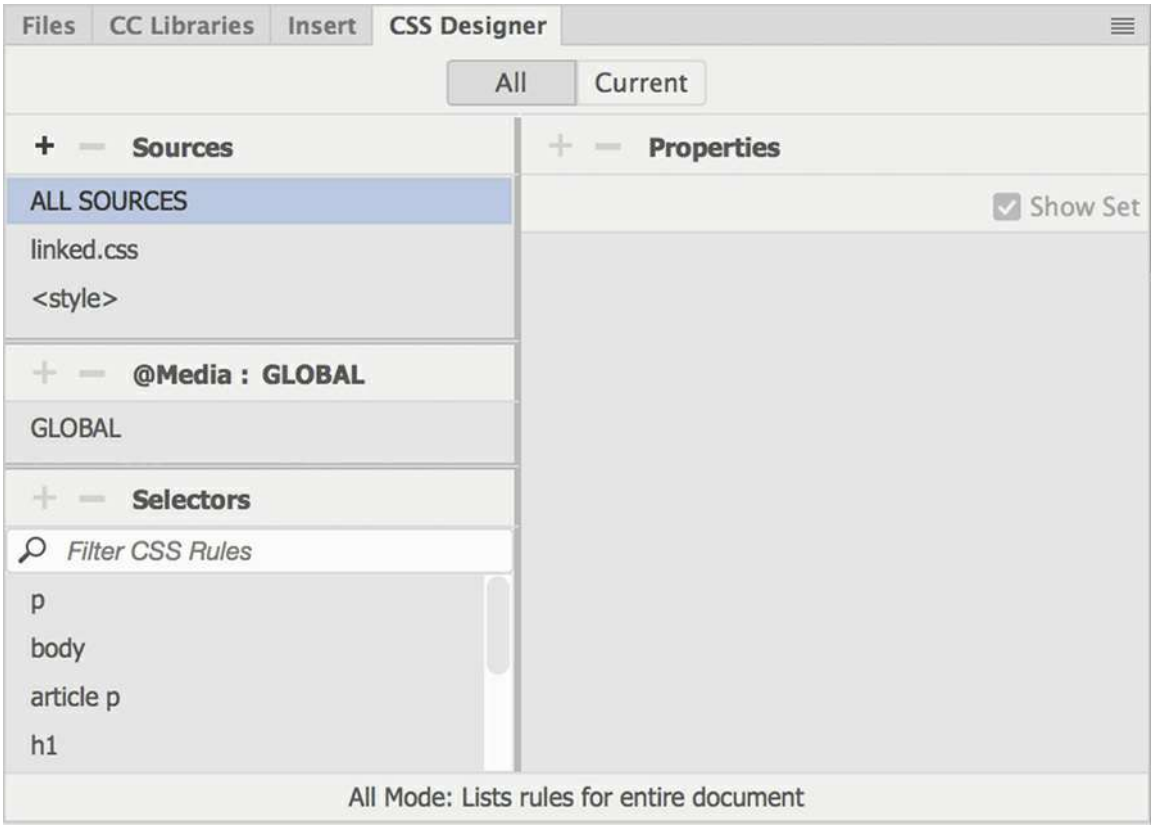
Another tool that provides similar features gives you even more power to troubleshoot and edit CSS styling.

Using CSS Designer to identify specificity

The CSS Designer is the main tool in Dreamweaver for creating, editing, and troubleshooting CSS styling. You will use it extensively throughout this lesson and the rest of the book to write CSS rules and format content.

- If necessary, choose Window > CSS Designer to display the panel.

  The CSS Designer should be a permanent fixture of the Standard workspace, but if you see it appear as a floating panel, feel free to dock it to the right side of the interface.
- Click the Sources tab to view the style sheets defined in the page.



When the All button is selected, the Source pane shows two style sheets defined in the page: `linked.css` and `<style>`. The first is obviously an external CSS file, while the notation `<style>` indicates that the second style sheet is embedded within the webpage. CSS Designer can work with both types, as well as with inline CSS styles.

Tip

To obtain the two-column display, drag the left edge of the CSS Designer to increase its width.

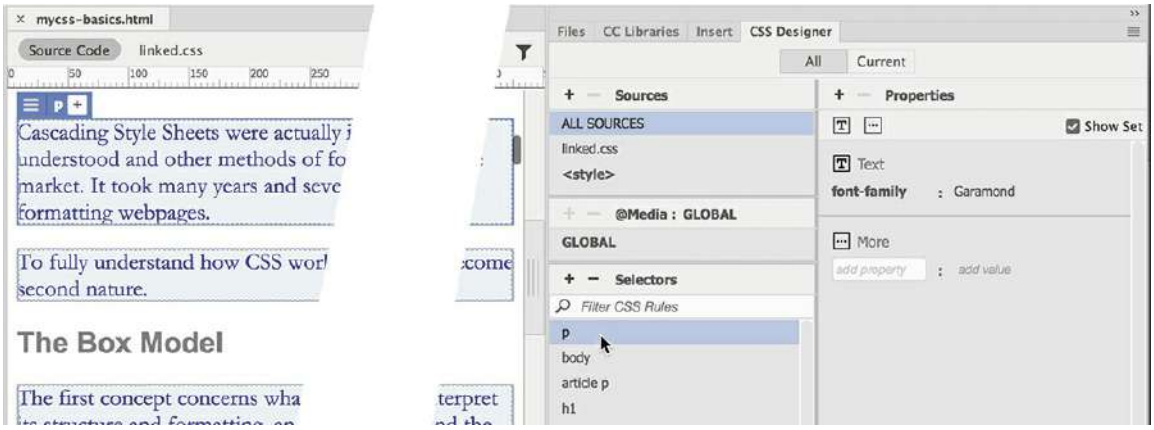
. Click All Sources.

This option shows all rules in all style sheets.

. Click the All button at the top of the CSS Designer.

CSS Designer displays all CSS rules defined in the webpage and its associated style sheets. The Selectors pane lists the names of the rules. In All mode, rules are listed in the order they are defined in the style sheets. The first in the pane should be the `p` rule.

. Click the `p` rule. If necessary, select the option Show Set.

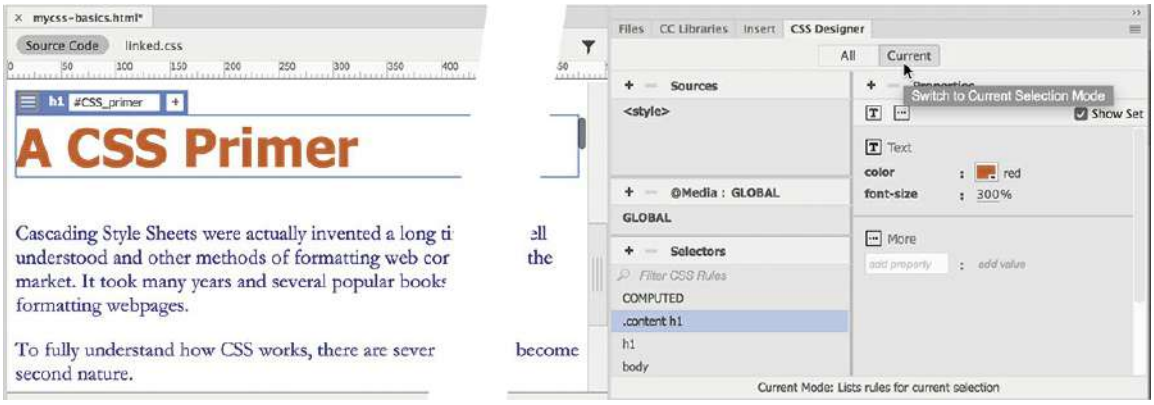


The Properties pane displays the settings assigned to the rule, in this case `font-family: Garamond`. Note that all the `<p>` elements in the Live view document window are highlighted in blue. This is one of the best features of CSS Designer, which allows you to see graphically what elements are being styled by a selected rule. The CSS Designer also works the other way.

- Click the heading "A CSS Primer" in the Live view window.

The Element Display appears around the heading.

- Click the Current button in the CSS Designer.

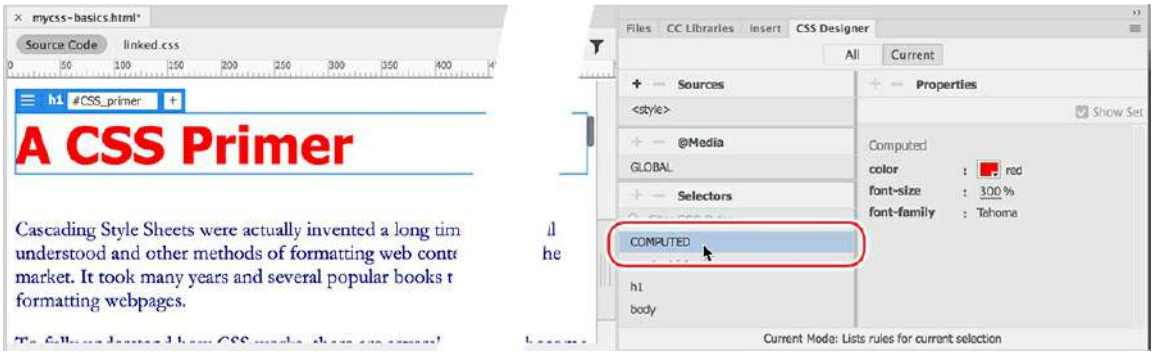


The CSS Designer now displays the list of three rules that supply styling to the selected element. The most specific rule appears at the top. Although `.content h1` is the most specific, this does not mean the other rules are not contributing to the current styling in some fashion.

- Click each rule listed in the Selectors pane, and observe the properties defined in it.

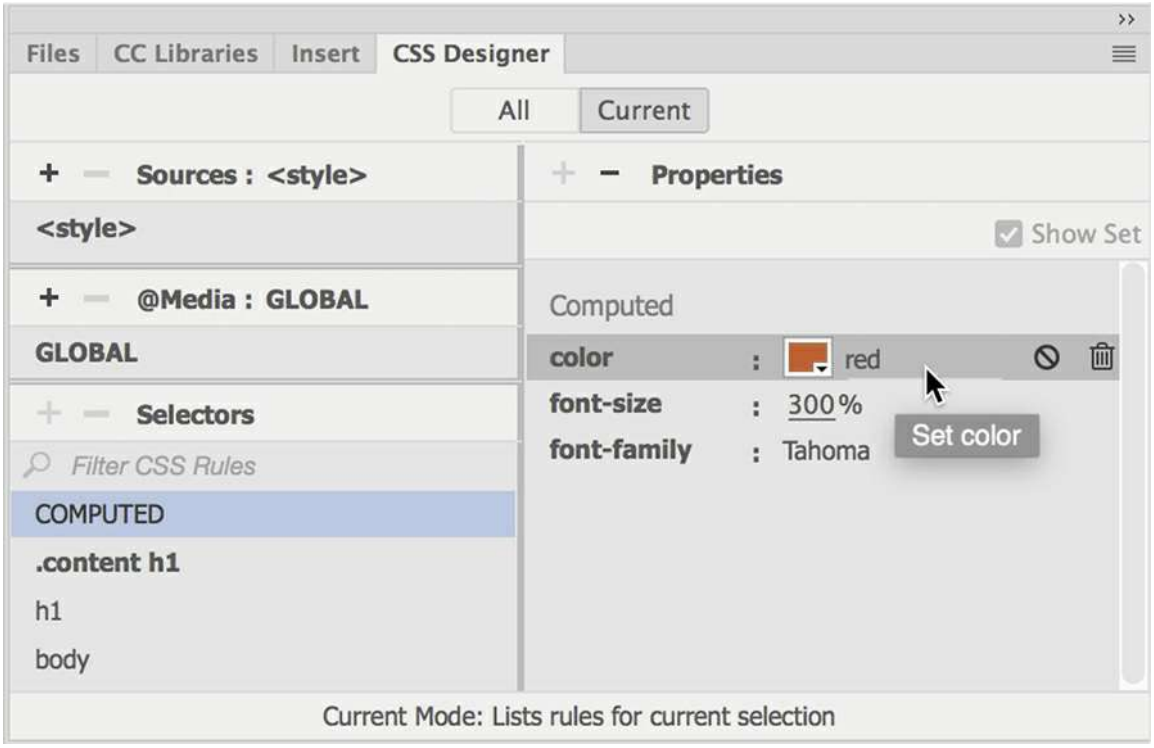
All three apply a font color, two apply a font family, and one sets the font size. In the past, manually checking the specifications was the only way to find out whether more than one rule was styling an element. Today, CSS Designer can do that tedious work with a single click of the mouse.

- Click COMPUTED in the Selectors pane.



The CSS Designer compares all the competing rules and specifications and calculates their specificity. It then displays the properties actually applied. You can even use the display to identify the source of the styling.

. Click the `color: Red` property.



Note that the `<style>` notation in Sources and the `.content h1` selector in Selectors are highlighted in bold. The bold tells you that the property is contained in the `.content h1` rule in the embedded style sheet. Keep an eye on the bold notations to identify the source of formatting as you work through the other exercises.

Formatting objects

After the last few exercises, you may be thinking that styling text with CSS is perplexing and difficult to understand. But hold onto your hat. The next concept you'll explore is even more

complex and sometimes even controversial: object formatting. Consider object formatting as specifications directed at modifying an element's size, background, borders, margins, padding, and positioning. Since CSS can redefine any HTML element, object formatting can basically be applied to any tag, although it's most commonly directed at HTML container elements, such as `<div>`, `<header>`, `<article>`, and `<section>`, among others.

By default, all elements start at the top of the browser window and appear consecutively, one after the other, from left to right and top to bottom. Block elements generate their own line or paragraph breaks, inline elements appear at the point of insertion, and hidden elements take up no space on the screen at all. CSS can control all these default constraints and enables you to size, style, and position elements almost any way you want them.

Size is the most basic specification and is the least problematic for an HTML element. CSS can control both the width and the height of an element, with varying degrees of success. All specifications can be expressed in absolute terms (pixels, inches, points, centimeters, and so on) or in relative terms (percentages, ems, or exes).

Width

As you should already be aware, all HTML block elements take up the entire width of the screen by default, but there are a variety of reasons why you'd want to set the width of an element to something less than that. For example, studies have shown that text is easier to read, and more understandable, if the length of a line of type is between 35 and 50 characters. On a normal computer screen, a line of type stretching across 1000 or more pixels could easily include 120 characters or more. That's why most websites today break up their layouts and display text into two or more columns.

CSS makes it simple to set the width of an element. In this exercise, we'll experiment by applying different types of measurements to the various content elements on the page.

. If necessary, open **mycss-basics.html** from the lesson03bonus folder.

. View the page in Split view, and observe the CSS code and HTML structure.

The file contains headings, paragraphs, and lists in several HTML5 semantic elements. The text is partially formatted by several CSS rules, but the structural elements display only default styling.

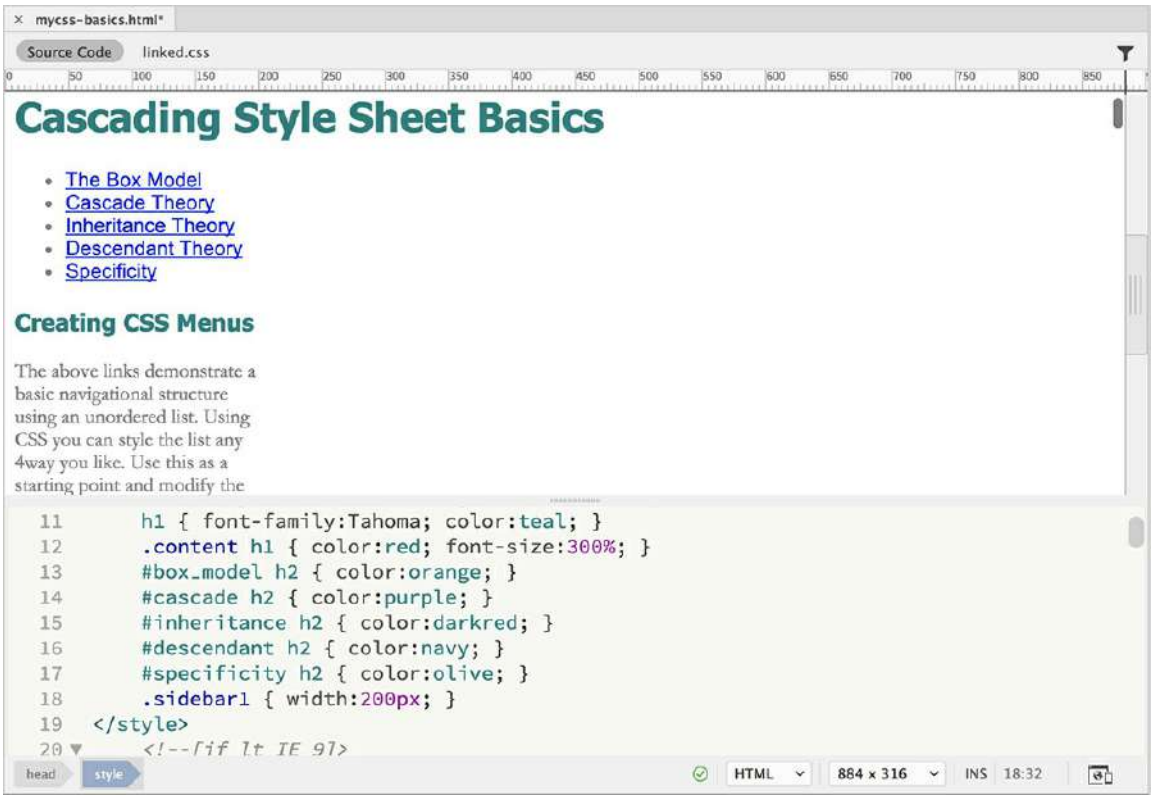
Fixed widths

The container elements, such as `<div>`, `<header>`, `<article>`, `<section>`, and `<aside>`, each currently occupy 100 percent of the width of the browser window or their parent element by default. CSS allows you to control the width by applying absolute (fixed) or relative (flexible) measurements.

. If necessary, open **mycss-basics.html** in Split view.

In Code view, insert a new line after the last rule in the `<style>` section.

. Type `.sidebar1 { width:200px; }` to create a new rule to style `sidebar1`.

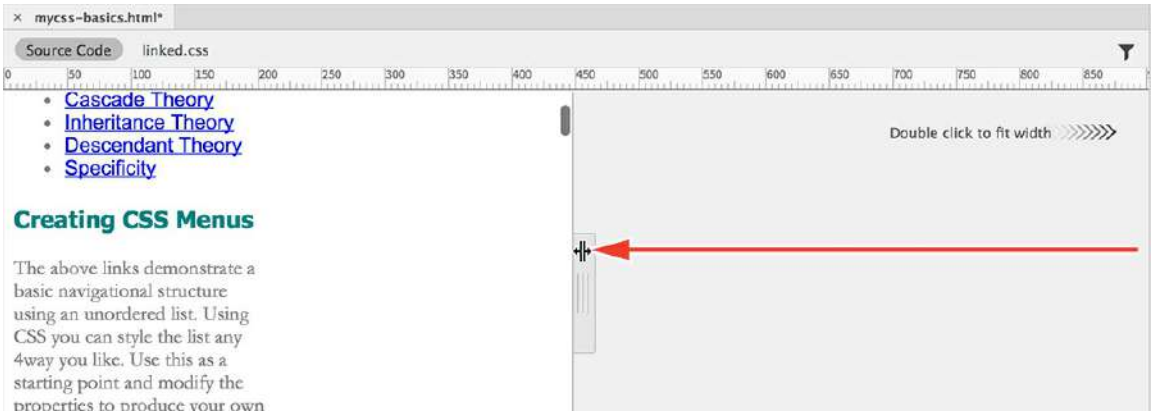


. Save the file and refresh the Live view display, if necessary.

The `sidebar1` element now occupies only 200 pixels in width; the other elements in the layout are unchanged.

By using pixels, you have set the width of this element, and its children, by an absolute, or *fixed*, measurement. This means `sidebar1` will maintain its width regardless of changes to the browser window or screen orientation.

. Select the Scrubber, drag it to the left and right, and observe how the different elements react.



As expected, the element `sidebar1` displays at 200 pixels in width no matter what size the

screen assumes. The other containers remain at full screen width.

. Drag the Scrubber fully to the right side of the document window.

Fixed widths are still popular all over the Internet, but in some cases, such as when designing for mobile devices, you'll want elements to change or adapt to the screen size or user interaction. CSS provides three methods for setting widths using relative, or flexible, measurements such as em, ex, and percentage (%).

Relative widths

Relative measurements set by percentage (%) are the easiest to define and understand. The width is set in relation to the size of the screen: 100% is the entire width of the screen, 50% is half, and so on. If the screen or browser window changes, so does the width of the element. Percentage-based designs are popular because they can adapt instantly to different displays and devices. But they are also problematic because changing the width of a page layout dramatically can also play havoc with your content and its layout.

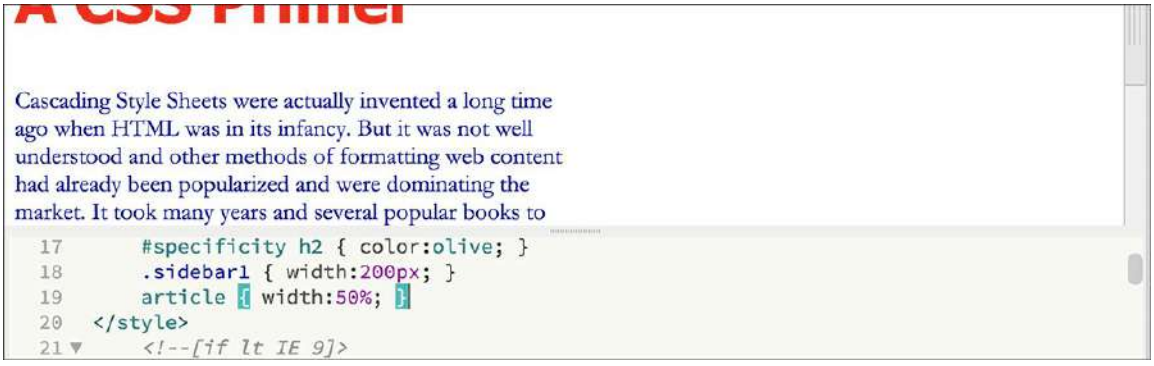
. If necessary, open **mycss-basics.html** in Split view.

Add a new rule: `article { width:50%; }`

. Save the file and refresh the Live view display.

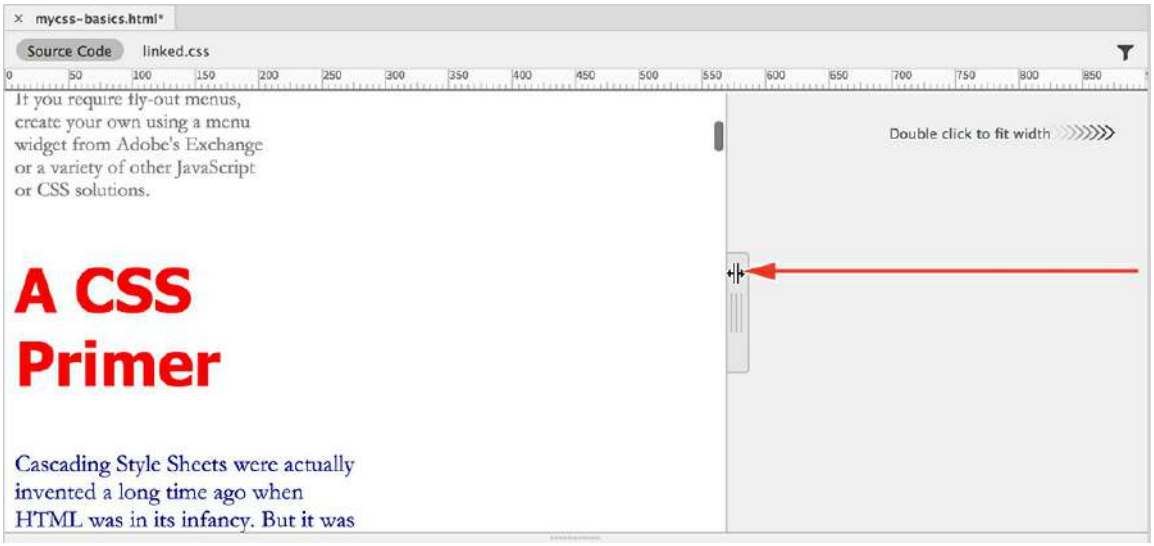
Tip

To test element widths, it may be easier to display Split view windows vertically. You can access this preference in the View menu.



The <article> element displays at 50 percent of the screen width. Widths set in percentage adapt automatically to any changes to the screen size.

. Drag the Scrubber to the left and right, and observe how the <article> element reacts.



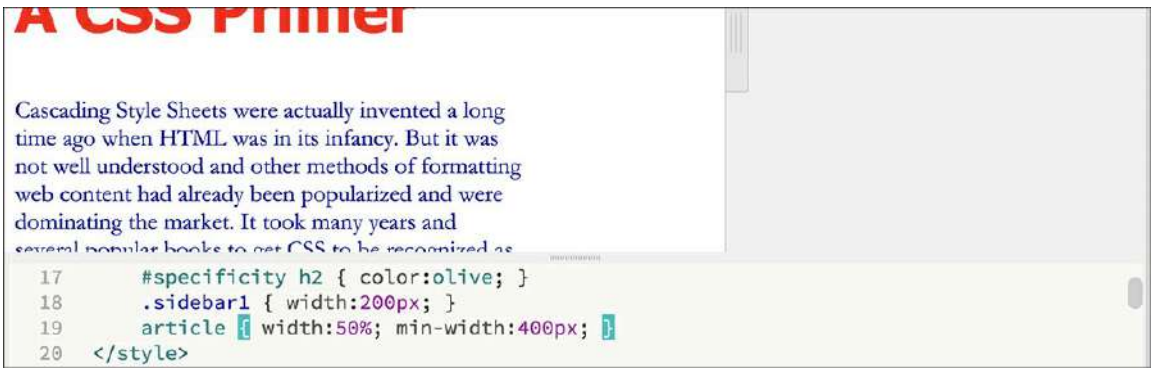
While `sidebar1` remains at a fixed width, the `article` element scales larger and smaller, continuing to occupy 50 percent of the width no matter what size the screen becomes.

Observe how the text wraps within the element as it changes in size. Note that it stops scaling when the frame shrinks to the size of the largest word and that the box model diagram juts out of the element below certain widths. Do you think these issues would affect the page's readability or usability?

Many designers forgo the use of percentage-based settings for these reasons. Although they like that the containers scale to fit the browser window, they'd prefer that it would stop scaling before it affects the content detrimentally. This is one of the reasons the properties `min-width` and `max-width` were created.

. Add the highlighted notation in the rule

`article { width:50%; min-width:400px; }` and save the file.



The `min-width` property prevents the element from scaling smaller than 400 pixels. Note that the `min-width` specification unit is defined in pixels. This is important, because when combining the `width` setting with `min-width` or `max-width`, you must use differing measurement units or only one of the specifications will be applied. To see the effect of the `min-width` specification, you need to use the entire screen.

- Switch to Live view and refresh the display, if necessary.

  Drag the Scrubber left and right.

  Observe how the `<article>` element reacts.

  The `<article>` displays at 50 percent of the screen width as you scale it smaller. When the screen becomes narrower than 800 pixels, the `<article>` stops scaling and remains at a fixed width of 400 pixels. To limit scaling at the upper end, you can also add the `max-width` property.

- Switch to Split view.

  Insert the highlighted notation as shown here:

  `article { width:50%; min-width:400px;` **`max-width:700px;`** `}` and save the file.

- Refresh the Live view display and drag the Scrubber left and right.

The `<article>` displays at 50 percent of the screen only between the widths of 800 pixels to 1400 pixels, where it stops scaling at the dimensions specified.

It’s all relative, or not

Ems and exes are kind of a hybrid cross between fixed and relative systems. The em is a measurement that is more familiar to print designers. It’s based on the size of the typeface and font being used. In other words, use a large font and the em gets bigger; use a small font and the em gets smaller. It even changes based on whether the font is a regular, condensed, or expanded face.

This type of measurement is typically used to build text-based components, such as navigation menus where you want the structure to adapt to user actions that may increase or decrease the font size on a site but where you don’t want the text to reflow.

- If necessary, open **mycss-basics.html** in Split view.

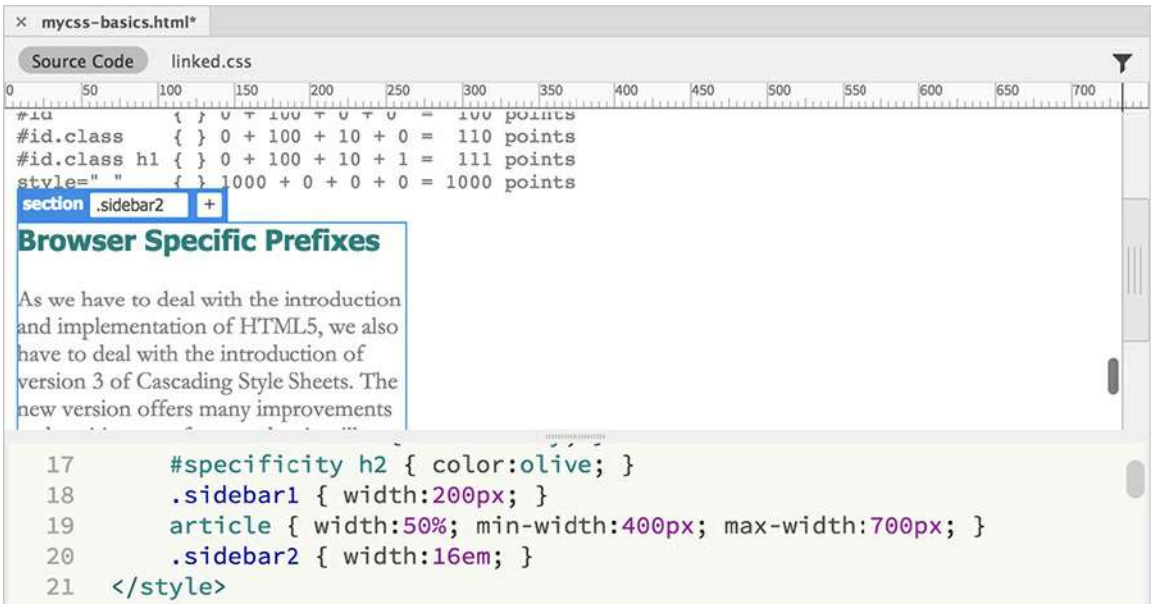
  Add the rule **`.sidebar2 { width:16em; }`** and save the page.

Refresh the Live view display, if necessary.

Although ems are considered a relative measure, they behave differently than widths set in percentages. Unfortunately, em measurements don't display properly in Design view; to see the exact relationship, you need to use Live view.

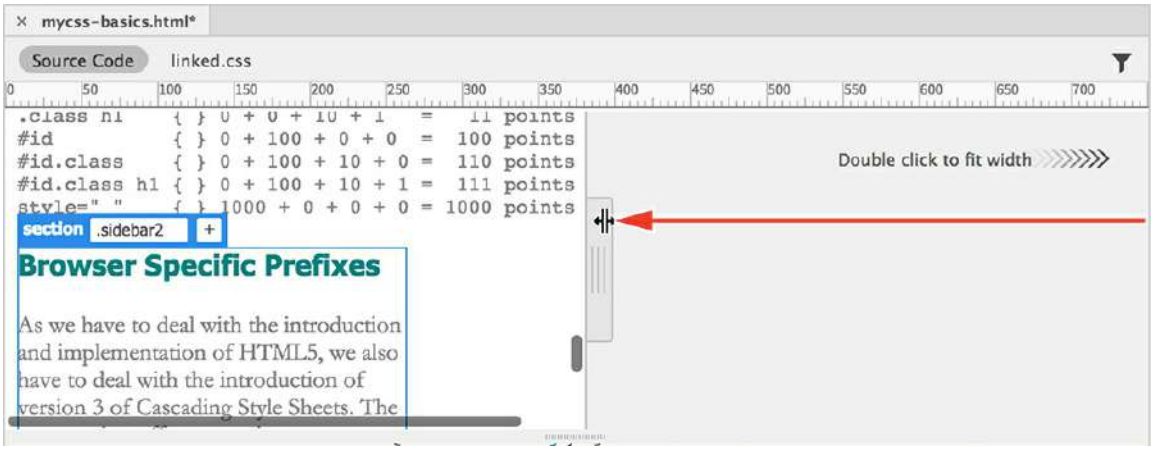
. Switch to Live view, if necessary.

Scroll down to view `sidebar2`.



The "Browser Specific Prefixes" heading should exactly fit the width of the element without breaking to two lines or leaving any extra space.

. Refresh the display if necessary and drag the Scrubber left and right.



The element seems to react like an element with a fixed measurement; it doesn't change size as you make the screen bigger and smaller. That's because the " relative" nature of ems is based not on screen size but on the font size.

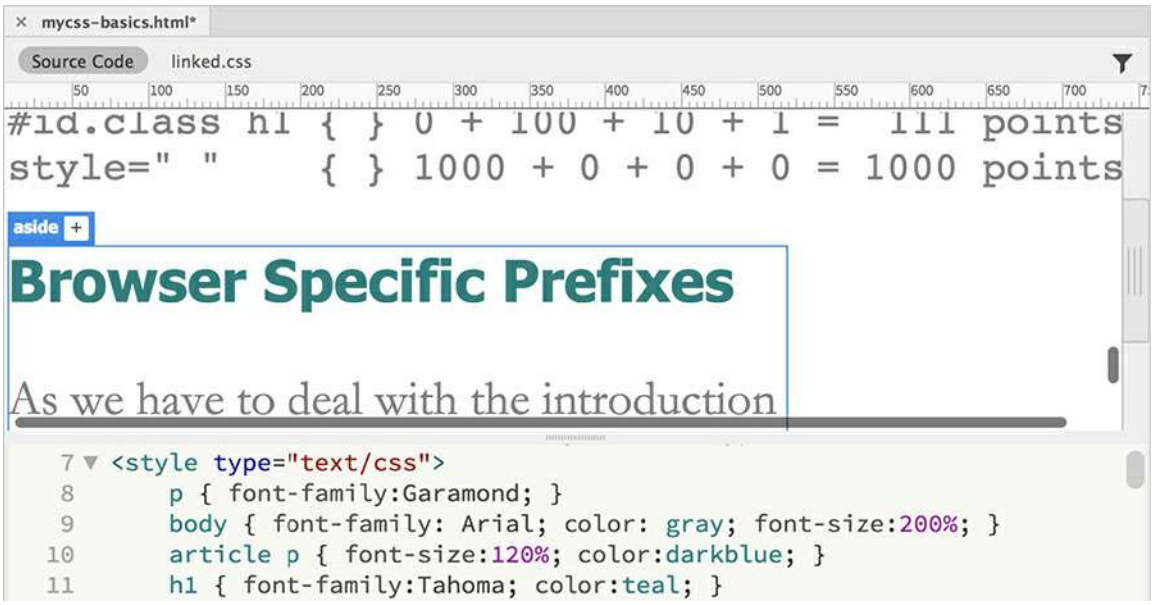
. Switch to Split view.

Add the highlighted code as shown here:

Click here to view code image

```
body { font-family:Arial; color:gray;
font-size:200%; }
```

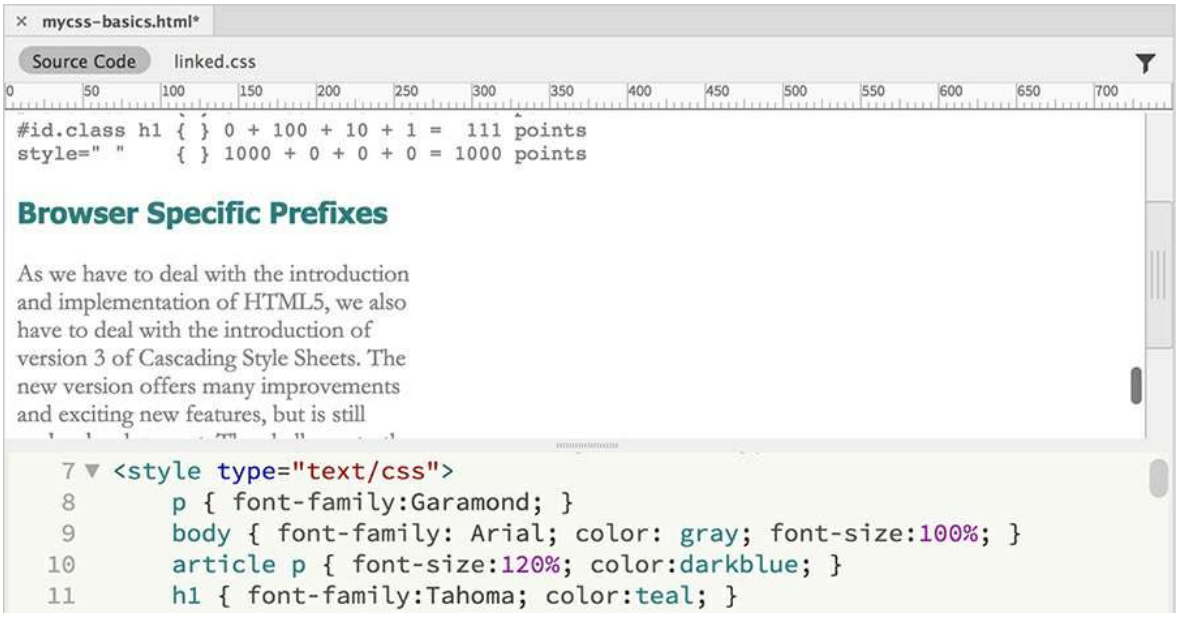
. Refresh the display.



All the text on the page scales 200 percent. The text in `sidebar1` and the `<article>` element have to wrap to fit within the container. On the other hand, `sidebar2` actually scales larger to accommodate the bigger text. Note how using ems also preserves the line endings in that element.

There's one small caveat when you use ems: The measurement is based on the base font size of the nearest parent element, which means it can change anytime the font size changes within the element's HTML structure. And you always need to remember that the relative size of the em is influenced by inheritance.

. Change the `font-size` property for `body` to **`100%`** and save the file.



The text in the page returns to its previous size.

By assigning various widths to the containers, you're setting up the basic structure for creating a multicolumn layout. The next step would be to start repositioning these containers on the page. But CSS positioning can be tricky; lots of factors can affect the display and interaction of these elements. Before you move the containers to their final positions, it may help you to understand these techniques better if you first apply some borders and background effects to make them easier to see.

Borders and backgrounds

Since every HTML element is a box, they can feature four individually formatted borders (top, bottom, left, and right). These are handy for creating boxes around paragraphs, headings, or containers, but there's no requirement to use all four borders on every element. For example, you can place them at the top or bottom (or both) of paragraphs in place of `<hr />` (horizontal rule) elements or to create custom bullet effects.

Borders

It's easy to create different border effects using CSS.

. If necessary, open **mycss-basics.html** in Split view and observe the CSS and HTML code.

You can assign borders to text or containers.

Note

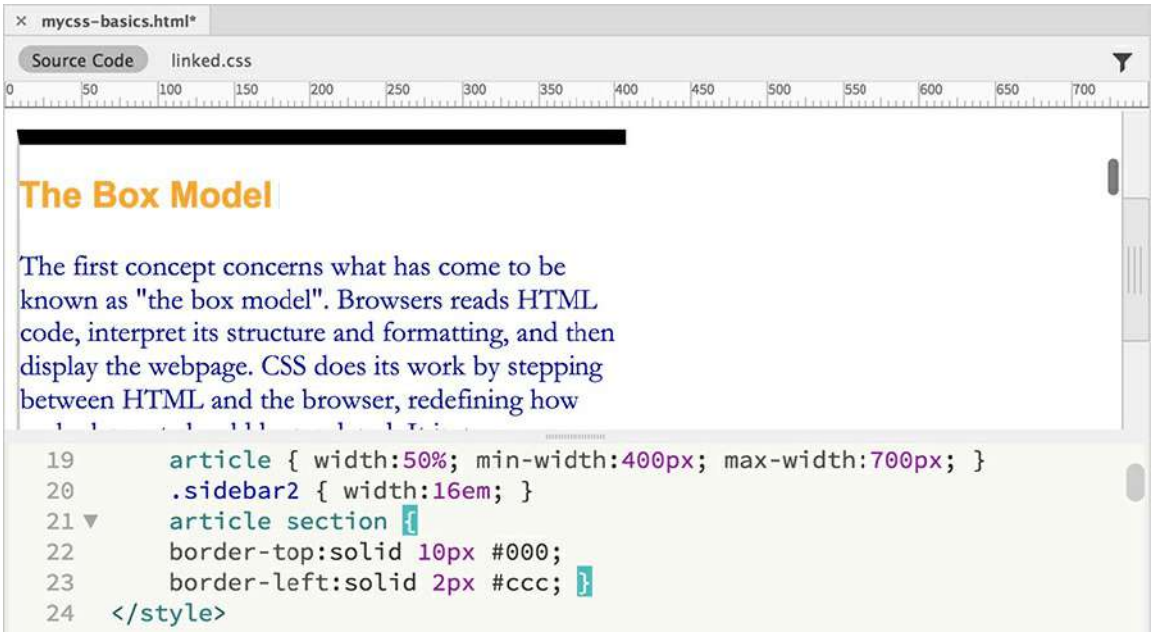
Remember that you can add the CSS rules with or without the line breaks and

indents.

. Add the following rule and properties to the `<style>` section:

```
article section {
border-top:solid 10px #000;
border-left:solid 2px #ccc; }
```

. Refresh the display, if necessary.



A custom border appears at the top and left sides of each `section` in the `<article>` element. The border gives a visible indication of the width and height of the HTML section. At the moment, the borders sit uncomfortably close to the text, but don't worry—we'll address this issue later. Let's apply borders to the other main containers now.

Note

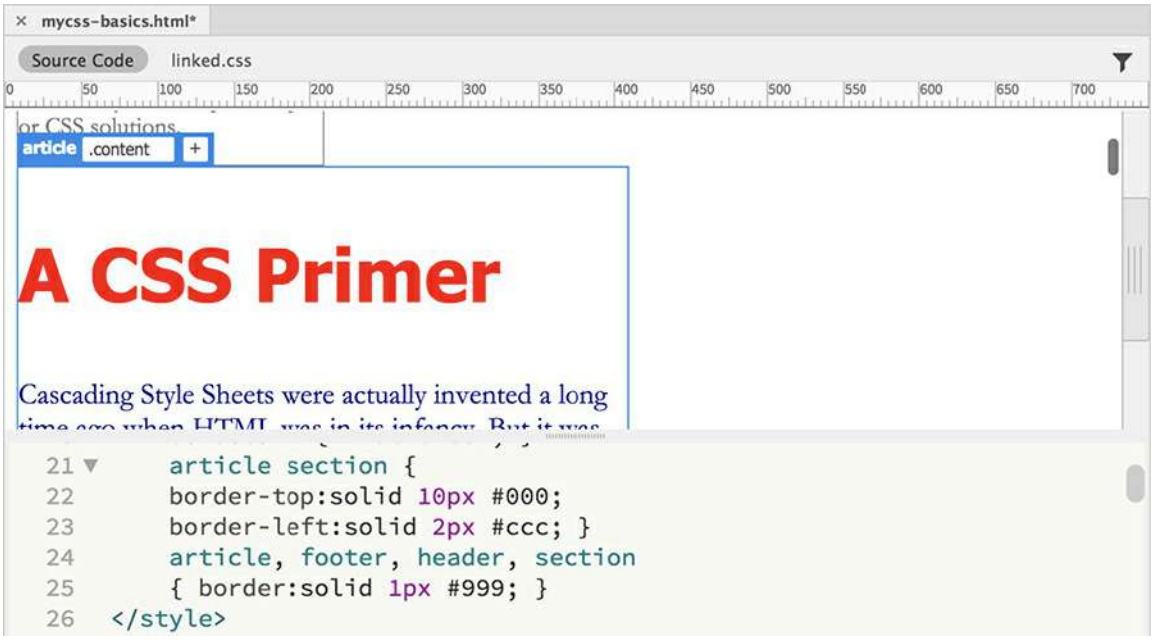
Don't forget the commas between the elements.

. Add the following rule and properties:

Click here to view code image

```
article, footer, header, section
{ border:solid 1px #999; }
```

. Refresh the display, if necessary.



Although it may be hard to see in the Live view display, a 1-pixel gray border appears on the `article`, `footer`, and `header` and on every `section` element on the page, including the ones already formatted in step 2.

Tip

When an element inherits properties from another rule that are undesirable, you may need to add a new rule (or new property in an existing rule) specifically to turn off that styling.

. Save the file.

As described earlier, it's not unusual for an element to be formatted by two or more rules. Even though the `<section>` elements nested in the `<article>` were styled with borders on the top and left by the first rule, they are now inheriting the 1-pixel border from the second rule, for the right and bottom sides.

It's also important to emphasize that there is no extraneous markup within the actual content; all the effects are generated by CSS code alone. That means you can quickly adjust content, turn on and off effects, and move the content easily without having to worry about graphical elements or extra code cluttering it up. You keep your code sleek and efficient.

Now that you can see the outer boundaries of each container, keep a wary eye on each to see how they react to the CSS styling created in the upcoming exercises.

Backgrounds

By default, all element backgrounds are transparent, but CSS lets you format them with colors, images, or both. If you use both, the image will appear above, or in front of, the color. This behavior allows you to use an image with a transparent or translucent background to create layered graphical effects. If you use an opaque image and it fills the visible space or is set to repeat like wallpaper, it may obscure the color entirely.

. Open **mycss-basics.html** from the lesson03bonus folder in Split view.

  Observe the CSS and HTML code.

  Backgrounds can be assigned to any visible block or inline element. If you want the background to appear behind the entire webpage, assign it to the `body` element.

. In the `body` rule, add **`background-color:#ccc;`** and refresh the display.

The background of the document window is now filled with light gray.

Some background colors may make content hard to read. Studies show that white is still the best color on which to read text. Let's fill the main text containers with a white background.

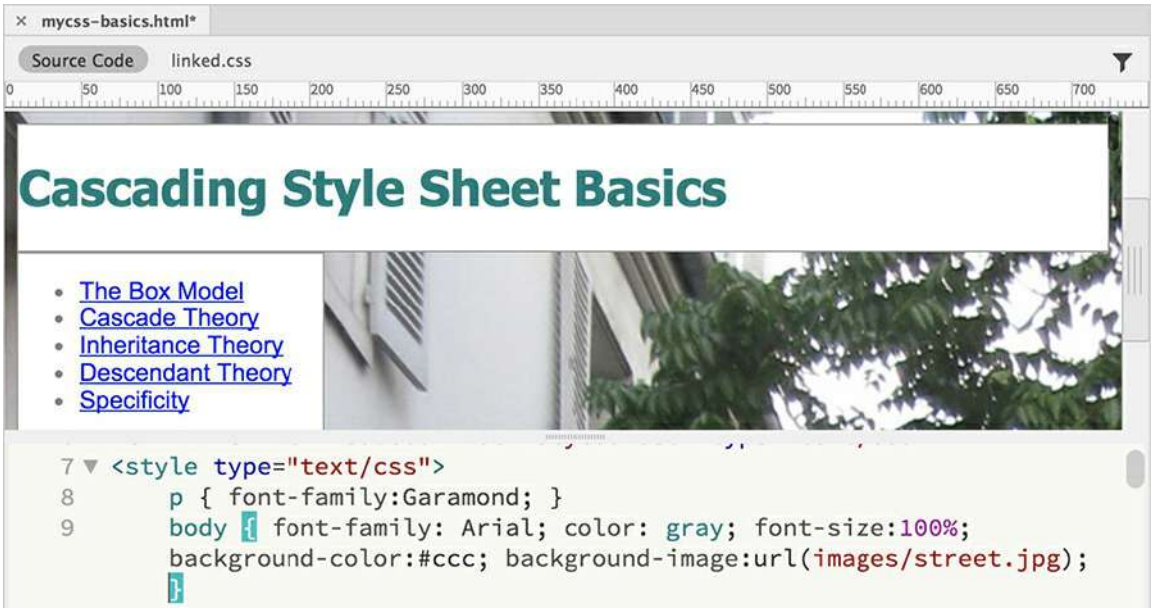
. In the `article, footer, header, section` rule, add the property **`background-color:#fff;`** and refresh the display.

Each of the targeted containers now displays a white background.

Websites have been using graphical backgrounds for many years. In the beginning, images or icons were used to create wallpaper effects. Since the connection speeds of the Internet were much slower in those days, those images were typically very small. Today, large images are becoming a popular way to apply a custom look to websites everywhere. Let's experiment with both types.

. In the `body` rule, add the property

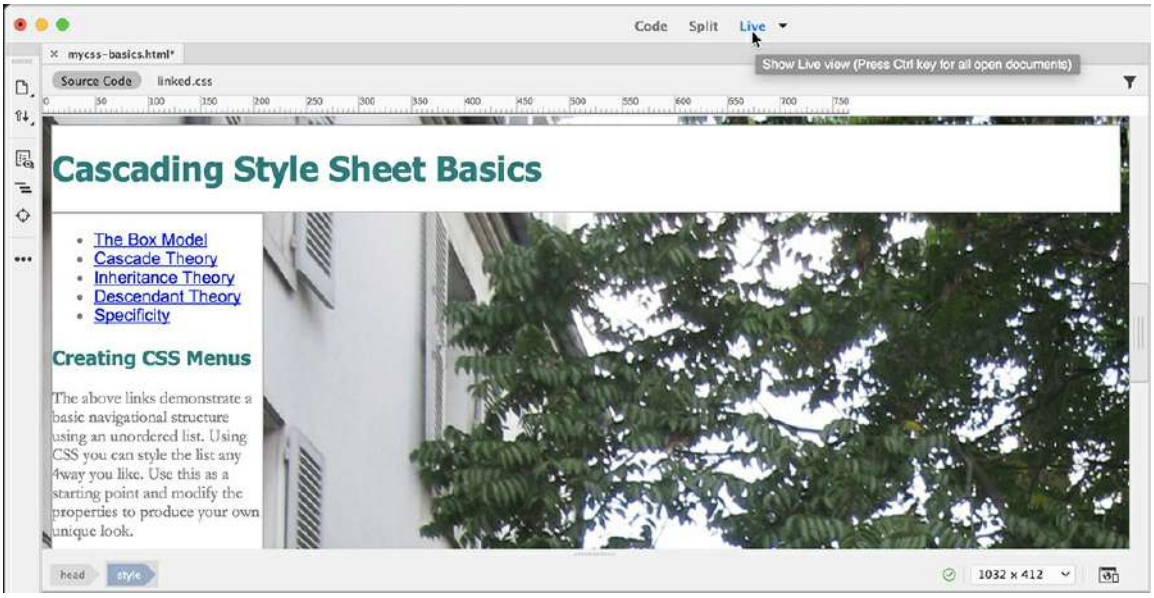
`background-image:url(images/street.jpg);` and refresh the display.



A photograph of a street scene appears in the background of the page, the exact composition of which is determined by the width and height of your document window.

By default, background images display at 100 percent of their original size and attach at the upper-left corner of the screen. If you check the dimensions of the image, you'll see that it's 1900 pixels by 2500 pixels and nearly 4 MB in size. That's big enough to fit almost any type of screen, but some aspects of a background image can't be seen properly in Design view.

. Switch to Live view to fill the entire document window with the webpage display.



Note

`Background` and `background-image` can both be declared at the same time. The `background-image` settings will appear above or in front of the `background` settings, but depending on the image, both settings may be visible at the same time.

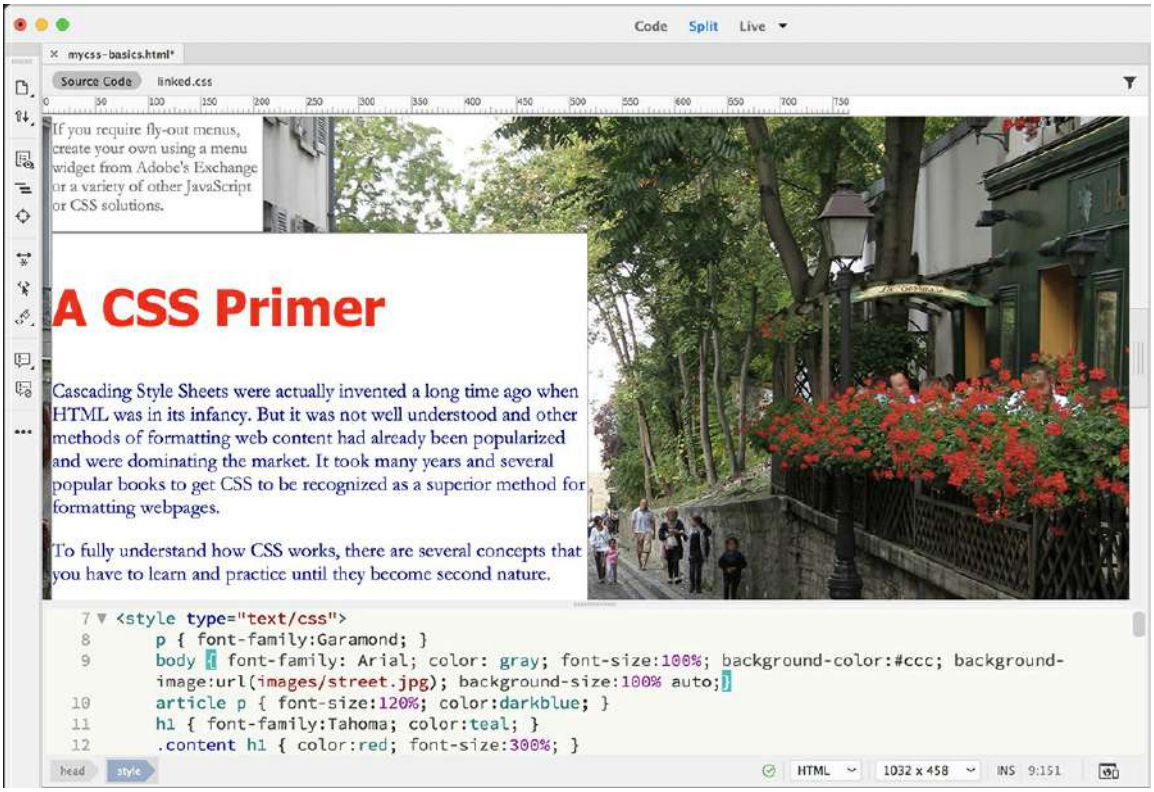
Live view renders web content for a browser-like environment.

. Drag the Scrubber left and right and observe how the background image reacts to the changing window size.

The background image does not respond to the changing window. As the window narrows, the right side of the image is hidden and off-center. It would look better if the image scaled along with the window.

. Switch to Split view.

Add the property **`background-size:100% auto;`** to the `body` rule.



. Refresh the display.

Note

`Background-size`, like many CSS properties, can be specified in fixed or relative measurements. The specification can be expressed in one or two values. When you use two values, the first applies the width, the second the height.

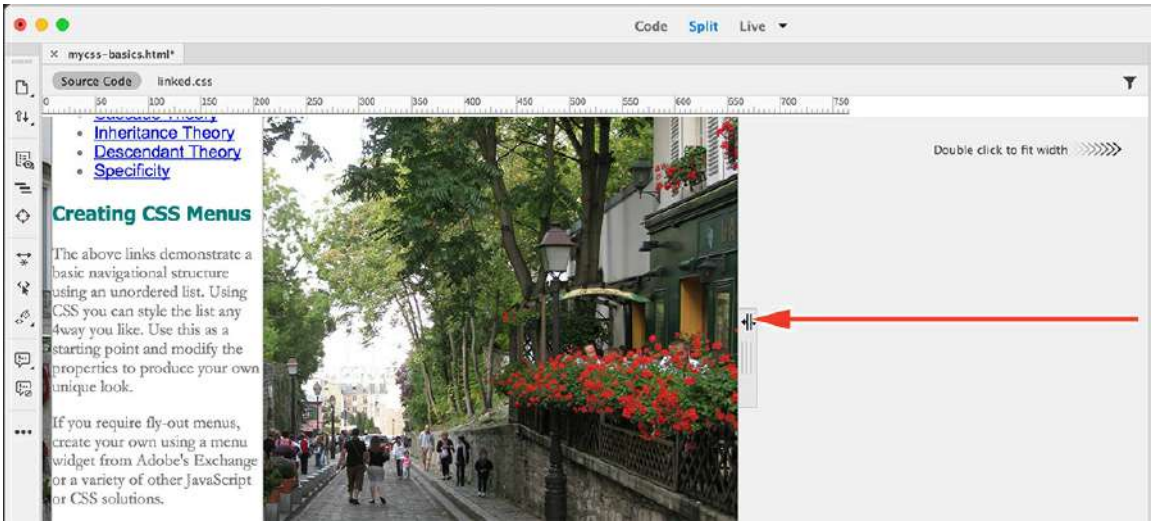
The background image scales automatically to fit the width of the browser screen. But if you scroll down the page, you will notice that the image repeats vertically once you get to the bottom of the image. CSS allows you to control many aspects of the background image to make it look and respond better on a variety of devices. By adding other properties you can make the image stay fixed in one location.

. Add the following properties to the `body` rule and switch to Live view:

Click here to view code image

```
background-position: center center;
background-attachment:fixed;
```

. Drag the Scrubber left and right and scroll the screen down to view the page content.



The background image no longer scrolls along with the content, and it remains centered regardless of how the window changes.

In the past, using such a large image on a webpage would have been avoided. But as more people access the Internet with high-speed connections, this type of large background image is becoming more popular. Although the user has to download an image that may be several megabytes, they have to do it only once, and for those who regularly visit the site or visit multiple pages, the image will be cached on the visitor's hard drive.

Yet for many designers an image of this size will never be acceptable. They know that large images can cause undesirable delays as webpages and resources download. Instead, these designers resort to a method that is still very popular: creating a background pattern or wallpaper using a simple, smaller graphic. Such graphics can be only a few kilobytes yet can produce beautiful, sophisticated effects.

. Switch to Code view.

Change the `body` rule as highlighted:

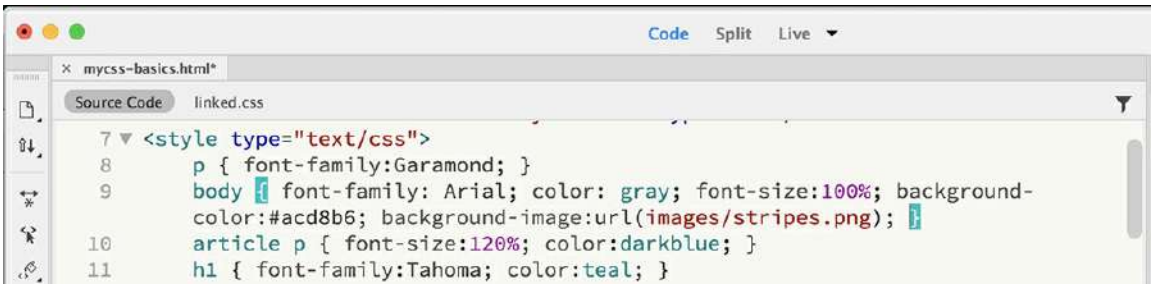
```
background-color:#acd8b6;

background-image:url(images/stripes.png);
```

This new graphic is 15 pixels by 100 pixels and only 2 KB. Even on a slow connection, this graphic will download almost instantly. However, its small size requires a few changes to the styling.

. Delete the following properties from the `body` rule:

Click here to view code image

```
background-size:100% auto; background-position:center center;
background-attachment:fixed;
```



By default, background images are intended to repeat vertically and horizontally. But before you allow this to happen, make the following changes to get a better idea of what's going to happen.

. Add the following property to the `body` rule:

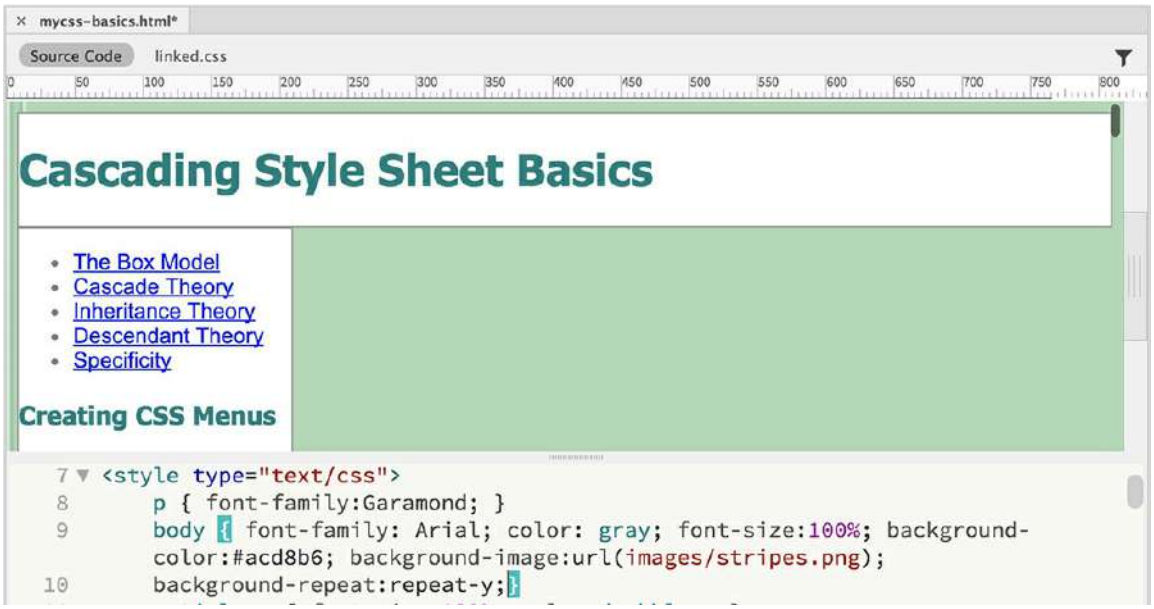
```
background-repeat:repeat-x;
```

. Switch to Split view.

You can see the graphic repeating horizontally along the x-axis at the top of the page. To make the graphic repeat vertically, you can make a simple change.

. Edit the highlighted property in the `body` rule:

`background-repeat:repeat-y;`



Although it's hard to see with the white background color, the graphic now repeats vertically along the left edge of the screen. Background graphics repeat horizontally and vertically by default. If you delete the repeat property altogether, the wallpaper effect would be seamless.

. Delete the `background-repeat:repeat-y;` property from the `body` rule and refresh the display.

Without the `repeat-y` property, the background repeats vertically *and* horizontally by default; the graphic is now repeating across the entire page.

. Save the file.

Combined with the right color or gradient, these types of backgrounds are both attractive and efficient in the use of resources. The choice is yours. No matter what type you choose, be sure to fully test any background treatments. In some applications, CSS background specifications are not fully supported or are supported inconsistently.

Positioning

As you have already learned, block elements generate their own line or paragraph breaks; inline elements appear at the point of insertion. CSS can break all these default constraints and let you place elements almost anywhere you want them to be.

As with other object formatting, positioning can be specified in relative terms (such as left, right, center, and so on) or by absolute coordinates measured in pixels, inches, centimeters, or other standard measurement systems. Using CSS, you can even layer one element above or below another to create amazing graphical effects. By using positioning commands carefully, you can create a variety of page layouts, including popular multicolumn designs.

. If necessary, open **mycss-basics.html** in Split view.

Make sure that Dreamweaver is maximized on your screen and that the document window is at least 1024 pixels in width.

. Observe the CSS and HTML code.

The file contains headings, paragraph text, and various HTML5 container elements partially formatted by CSS. The rules, created in previous exercises, set specific sizes on the main container's `sidebar1`, `sidebar2`, and `article` elements so that they no longer take up the entire width of the screen. In the same way—using only CSS—you can control the positioning of all these elements on the page. There are several ways to do this, but the *float* method is by far the most popular. The options for the `float` property are `left`, `right`, and `none` and can have a dramatic effect on the positioning of the targeted elements. If no property is actually set by CSS, the default styling is `none`.

. Create the following rule:

Click here to view code image

```
.sidebar1, article, .sidebar2 { float:right; }
```

. Save the file. If necessary, switch to Live view and refresh the display.

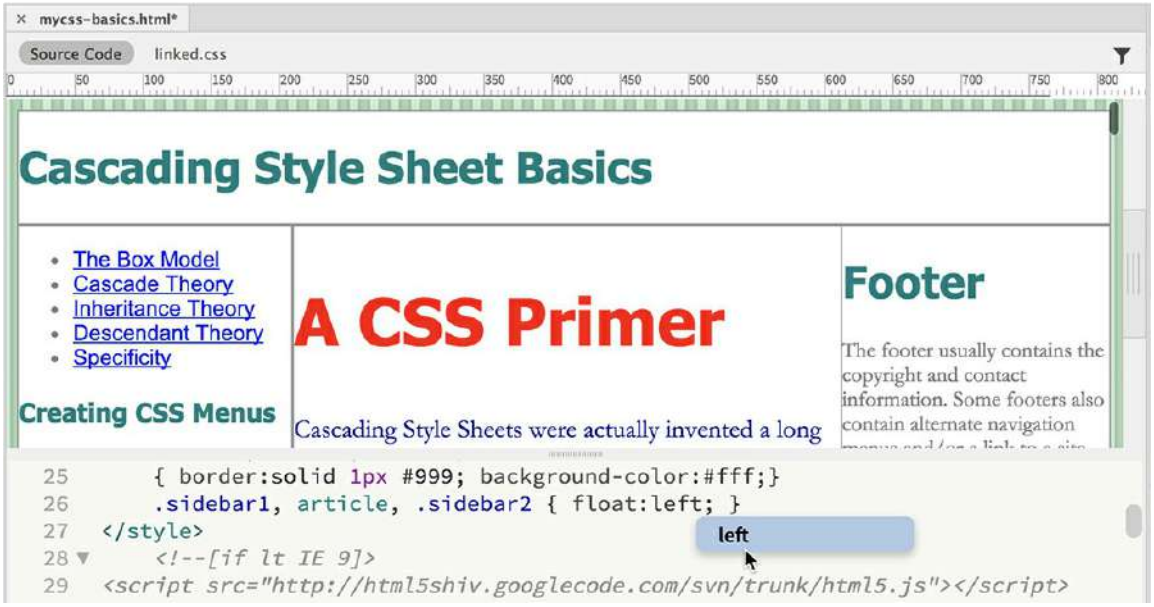
Maximize the program window to the full size of the computer display.

Depending on the width of your screen, the `aside` and `article` elements now display horizontally, side by side, from right to left in the document window. By using `float:right`, the elements display from right to left on the screen. Notice that `sidebar1` appears on the far right, followed by `article` and then by `sidebar2`. If you change the `float` value, you can produce the opposite effect.

Note

The `float` property can also be applied individually to the existing rules for these elements. By combining selectors, it is easier to update the property by simply editing one value.

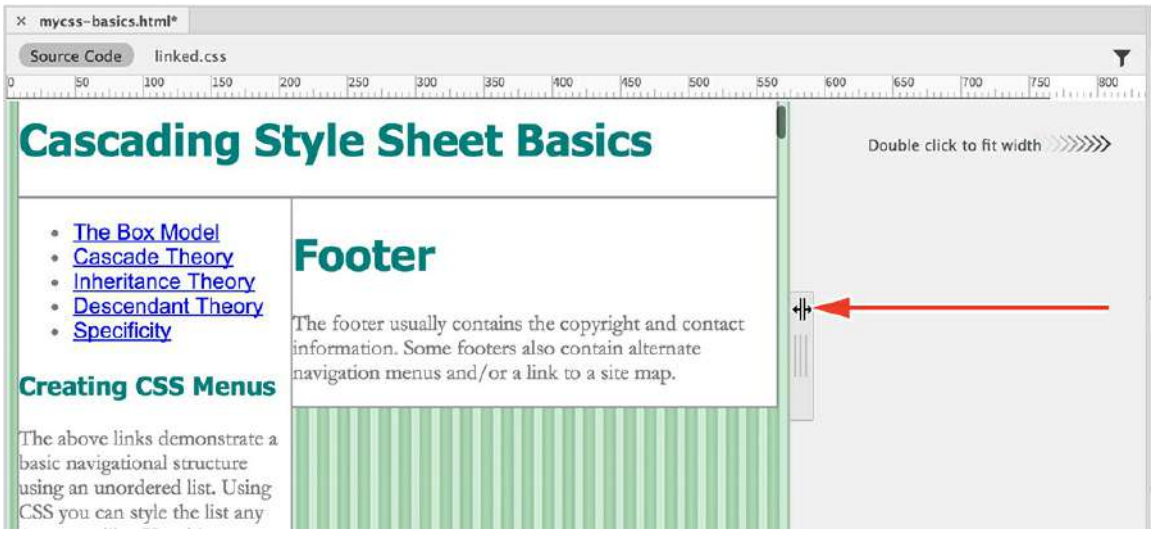
. Change the property `float:right` to **`float:left`** and refresh the display.



All three elements reverse direction, now starting on the left.

As you can see, the `float` property takes an element out of the normal HTML flow. By setting widths smaller than the default 100 percent on the sidebars and the article, `float` allows these block elements to behave in a totally different manner and *share* the space with each other. And `float` is also a dynamic property in that it reacts to the width of the document window.

. Drag the Scrubber left and right to change the width of the display.



As the window narrows, there's no longer enough space to accommodate the widths of the individual elements. They are forced to move down into the open space below. When the window gets wider, the elements move up to share the space again. This type of behavior allows websites to display rows of items that adapt automatically to any type of screen, no matter how big or small. As you make the window larger and smaller, you may notice that the footer element slips up into the row with the other elements if your screen is wide enough.

You may be saying, "But the footer isn't floated!" And you'd be right. This is one of the consequences of using `float`: The first subsequent *non*floated element will share the space with any floated ones if it doesn't have a specific width or other property that prevents it. The first nonfloated block element will honor all the width, margin, and padding settings of any floated element and then occupy 100 percent of the space left over. At times you may take advantage of this behavior to create some multicolumn layouts. However, for this layout, you want the footer to stay at the bottom.

You can force an element to move, or position, itself differently by simply setting a specific width to the parent container, or to the children themselves, that will preclude them from sharing the available space. At the moment, the combined width of the three floated elements is less than the width of the whole screen, which allows the footer to sneak in if the screen is wide enough. To prevent this from happening, you need to limit the amount of space the floated elements can use.

. Add a new rule `.container { width:1050px; }` and refresh the display. If necessary, switch to Live view.

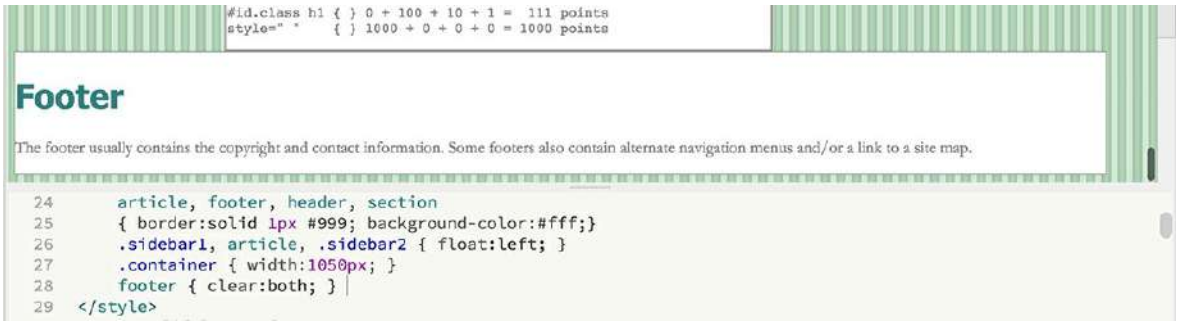
The `div` element displays at a width of 1050 pixels.

Now there's not enough room for the elements to float all the way across the screen, but the footer is still trying to sneak into the layout at the bottom of `sidebar2`. There is a CSS property specifically designed to keep this from happening.

. Create a new rule **`footer { clear:both; }`** and refresh the display.

Note

For this current layout, the footer could suffice with `clear:left`. But the footer is an element that should clear all potential content elements on the page; therefore, I chose to use `clear:both`.



The footer moves down to the bottom of the page again. The `clear` property controls the behavior of elements when they are floated or interacting with ones that are.

Although you fixed this situation with a single CSS property, it's not always that easy. Unfortunately, as powerful as CSS positioning seems to be, it is the one aspect of CSS that is most prone to misinterpretation by the browsers in use today. Commands and formatting that

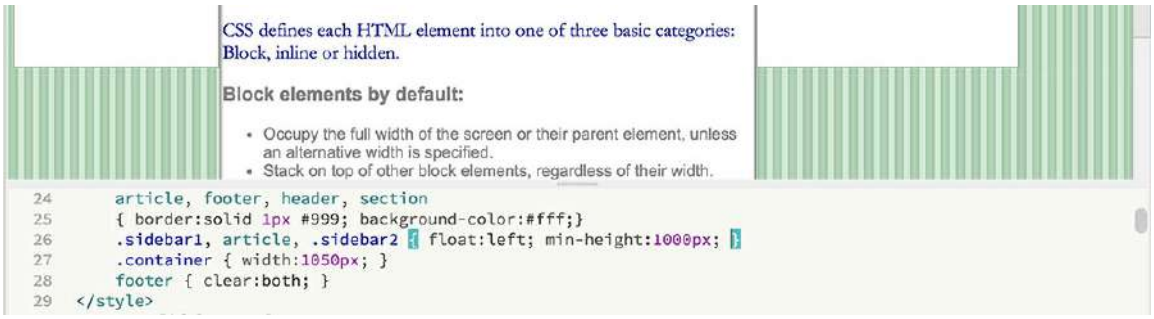
work fine in one browser can be translated differently or totally ignored—with tragic results—in another. In fact, specifications that work fine on one page of your website can even fail on another page containing a different mix of code elements.

Height

The `sidebar1`, `sidebar2`, and `article` elements contain different amounts of content and display at different heights. You could set a fixed height for all three that would work on this page, but what dimension would work on the other pages of the site?

Height is not specified as frequently on the web as width is. That's mainly because the height of an element or component is usually determined by the content contained within it, combined with any assigned margins and padding. Setting a fixed height can often result in undesirable effects, such as truncating, or clipping, text or pictures. If you must set a height, the safest way is to use the `min-height` property.

- In the `.sidebar1, article, .sidebar2` rule, add the **`min-height:1000px;`** property and refresh the display.



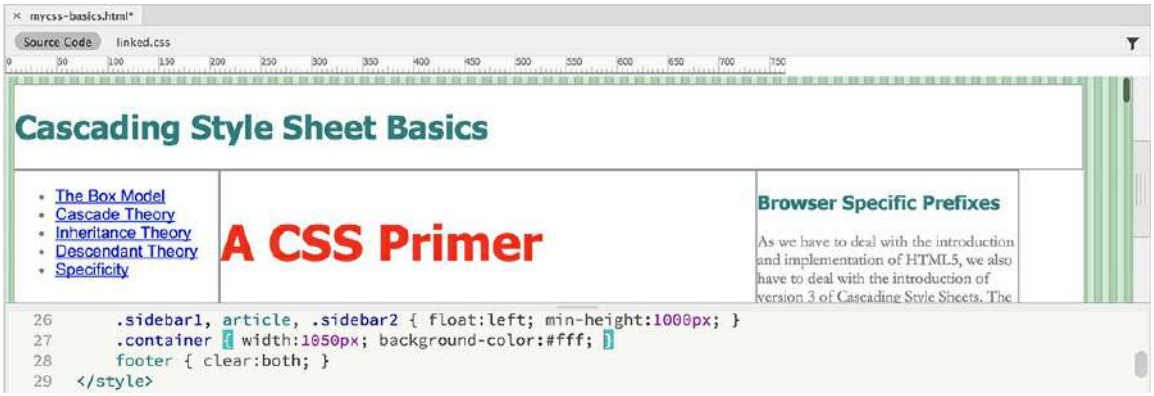
  Sidebars 1 and 2 now display at a minimum height of 1000 pixels but will grow as needed to match the length of their content.

  On a different page, setting a common element height might work, but with the amount of content in this article, a common element height isn't really a solution to the problem at hand. The main issue is the graphical background on the page. It makes it pretty obvious that sidebars 1 and 2 are shorter than the `article` element.

  One answer would be to ditch the page background graphic altogether and apply a background color that matches the one used in the elements. Or you could simply apply a matching background color to the `<div>` containing the layout itself.

- In the rule `.container` add the following property:

  `background-color:#fff;`



The background color for all the elements is now identical.

. Save the file.

For all intents and purposes, the heights of `sidebar1`, `sidebar2`, and the `article` element are irrelevant. If not for the gray borders applied to each, you'd have no idea how tall the elements are at all. Problem solved.

Margins and padding

Margins control the space outside the boundaries, or borders, of an element; padding controls the space inside an element, between its content and its border. It doesn't matter whether the borders are visible; the effective use of such spacing is vital in the overall design of your webpage.

Margins

Margins are used to separate one block element from another.

. If necessary, open **mycss-basics.html**.

Margins and padding don't always render properly in Design view.

. If necessary, switch to the full Live view display.

Observe the page layout and styling.

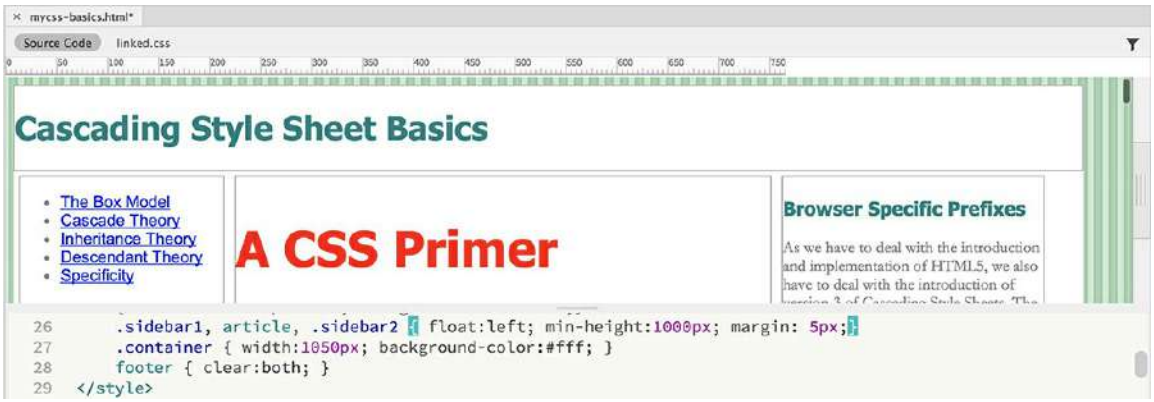
The page displays a header, three columns, and a footer. The column elements are touching each other, and the text within each column is touching the edges of each container.

. In the rule `.sidebar1, article, .sidebar2` add the property **`margin:5px;`** and refresh the display.

Note

In most cases, horizontal margins between two adjacent objects combine to increase the total spacing. On the other hand, only the larger of the two settings is

honored for vertical margins between two adjacent elements.

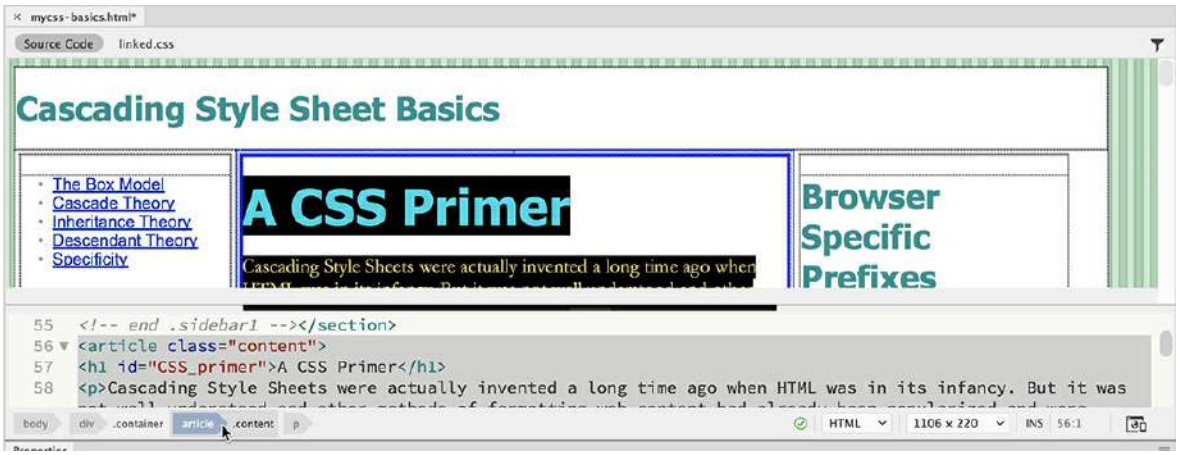


The new property adds 5 pixels of spacing outside the borders of each targeted element. Although Live view gives you a more accurate browser-like display, Design view still has a few tricks up its sleeve.

. Switch to Design view.

Click the heading “A CSS Primer.”

Select the `<article.content>` tag selector.



Design view highlights the element and displays a hashed pattern to provide a visual representation of the margin specifications. An equal amount of spacing has been added around the two sidebars and the center section. Although margins are used to apply space outside an element, they can also be used to center an element.

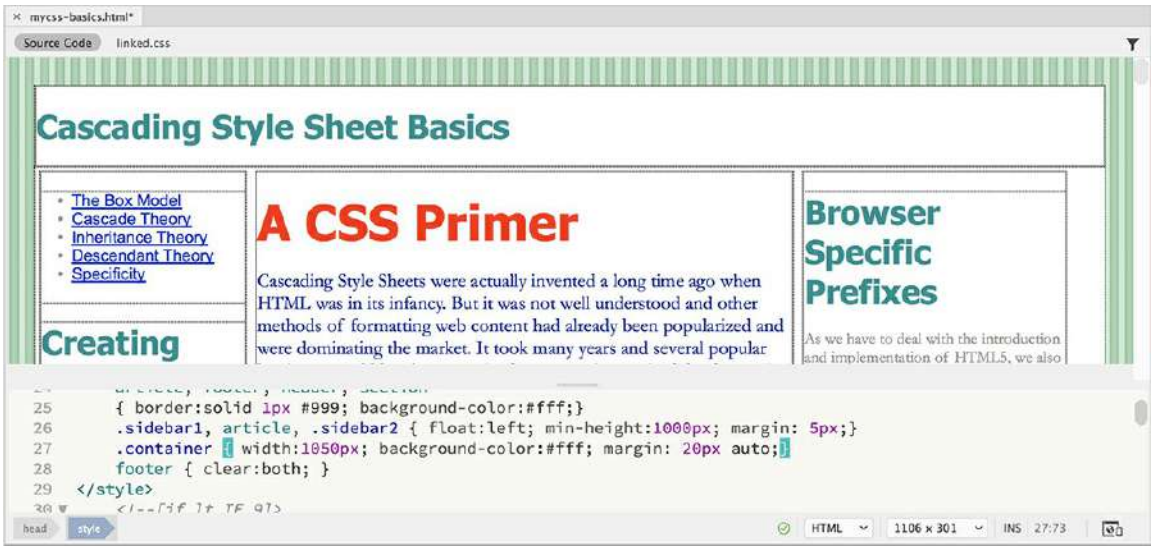
Centering elements

HTML block elements normally fill the screen edge to edge. In this page, you have set the various containers to a fixed size less than the width of the screen. When elements don’t stretch

across the screen, they will align to the left by default. In the finished layout, the content was centered. HTML 4 provided an element attribute (*align*) to allow you to position elements to the left, right, or center. This attribute was deprecated in HTML5, and CSS has no specific feature for centering block elements. Until a better method is developed, you can use margins.

. In the `.container` rule add the following property: **`margin: 20px auto;`**

. Refresh the display.



The element centers in the window. In this CSS shorthand notation, the `auto` value applies equal amounts of spacing to the left and right sides of the article.

. Save the file.

You've added spacing between the elements. Now, let's add some spacing inside the elements too.

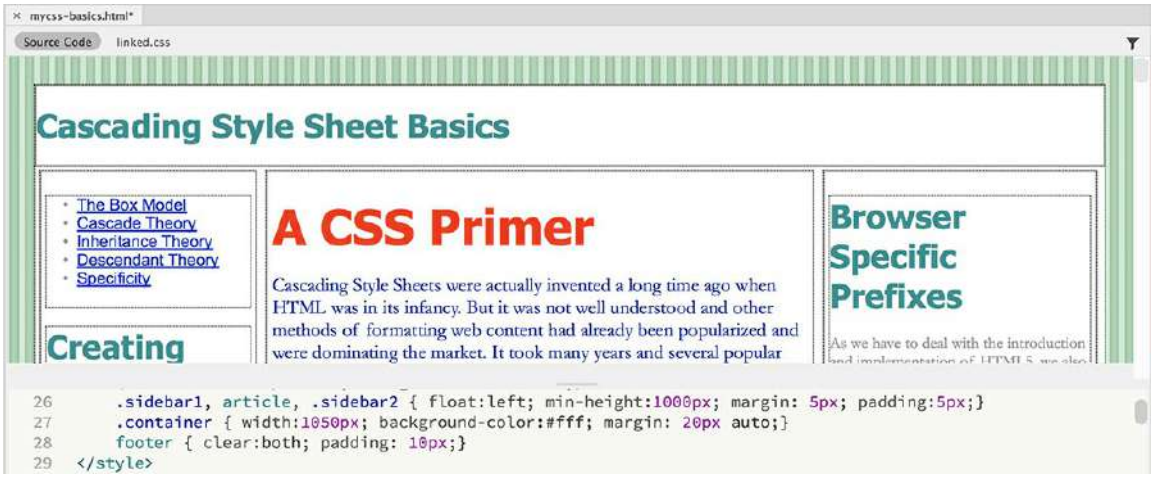
Padding

The text inside the layout is touching the borders within the containers. Padding puts spacing between the content and an element's border.

. In the rule `.sidebar1, article, .sidebar2` add the following property: **`padding: 5px;`**

. In the rule `footer`, add the following property: **`padding:10px;`**

. Refresh the display, if necessary.

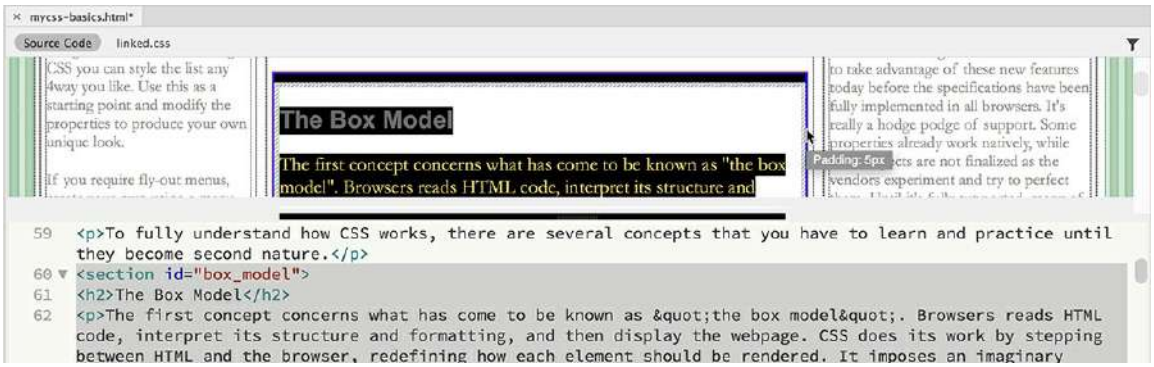


The text inside the targeted elements is now spaced away from all four element borders.

Did you notice that the subsections in the article element didn't inherit the padding themselves? The text is still touching the border of its element. This may be confusing, because earlier we discussed how styling is inherited from a parent element. Although it works for many properties, inheritance isn't a guarantee for several reasons.

Inheriting text formatting from a parent element makes a lot of sense if you think about it. You would want all the text to have the same font, size, and color. But is the same logic true for *structural* properties, such as width, height, padding, and margins? For example, if you set a width of 300 pixels on the container `div` element, would you want all the child elements, such as `sidebar1`, `sidebar2`, and `article`, to inherit the same width too? For this reason and others, padding and margins are a few of the properties that are not inherited via CSS.

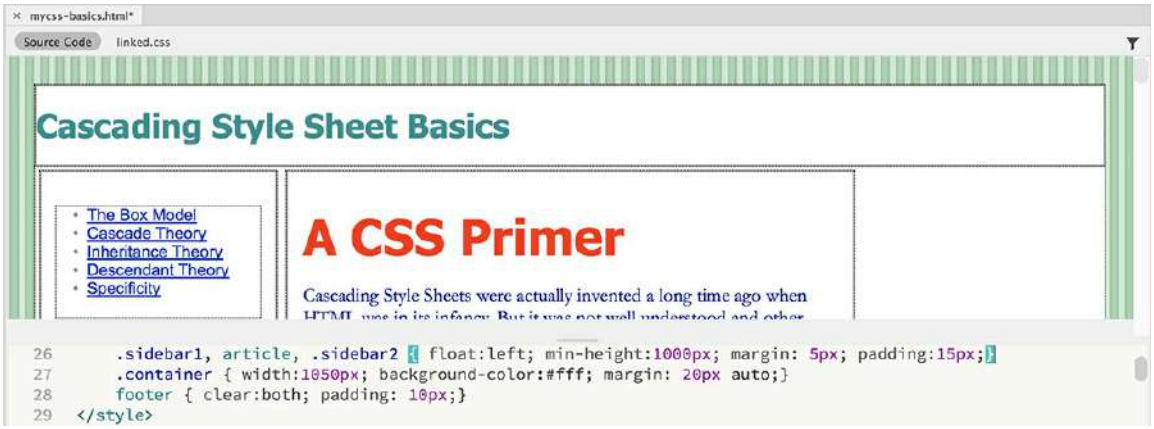
- In the rule `article section`, add the following property: **`padding: 5px;`**
- Click the edge of the box model `<section>` to select it.



Using Design view you can see the padding represented graphically. If you position the cursor over the element border, it even displays the amount of padding applied. Did you notice that the element grew slightly larger when you applied padding? Margins, padding, and even borders increase the width and height of an element. Add too much, and you might break your carefully constructed layout.

> **Tip**
>
> In Design view, you may need to click in the document window to force Dreamweaver to refresh the display.

. In the rule `.sidebar1, article, .sidebar2`, change the padding value to **`15px;`** and refresh the display.



Increasing the padding has broken the layout. Sidebar 2 no longer fits beside the `article` element. It will move down the page until it can find enough space. This type of conflict happens frequently in web design. The constant interplay between the elements and the CSS can produce undesirable results like this. Luckily, in this case, the solution is as simple as the cause.

. In the rule `.sidebar1, article, .sidebar2` change the padding value back to **`5px;`** and refresh the display.

Reducing the padding value has fixed the layout and allows `sidebar2` to display side by side with the other elements again.

Normalization

Now you know that margins and padding—among other things—affect the overall size of an element. You have to factor these specifications into the design of your page components. But apart from the properties applied directly by the style sheet, don't forget that some elements feature default margin specifications too. In fact, you can see these very settings in the extra space appearing above and below the headings and paragraphs on the current page.

Many designers abhor these default specifications, especially because they vary so much among browsers. Instead, they start off most projects by purposely removing, or resetting, these settings using a technique called *normalization*. In other words, they declare a list of common elements and reset their default specifications to more desirable, consistent settings.

. In the CSS section of the page code, move the `body` rule to the top of the style sheet.

  Since the styles in the `body` rule are inherited by all elements on the page, most designers place it high in the style sheet, if not in the first position. Next should come rules designed to normalize basic elements.

. After the `body` rule add the following rule:

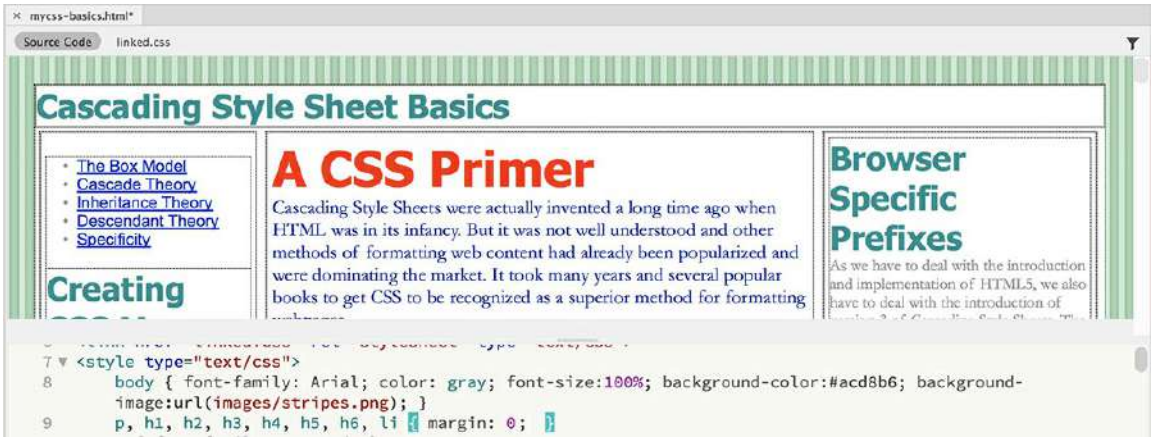
  Click here to view code image

```
p, h1, h2, h3, h4, h5, h6, li { margin: 0; }
```

. Save the file.

  As you learned in Lesson 3, the comma (`,`) means “and” in CSS syntax, indicating that you want to individually format all the tags listed. This rule resets the default margin settings for all the listed elements. It’s important that this rule be placed as high in the style sheet as possible, typically after the `body` rule, if one exists. That way, you can still add margins to specific instances of any of the targeted elements later in the style sheet without worrying about conflicts with this rule.

. Refresh the display.



  The text elements now display without the default spacing.

. Save the file.

Using zero margins may be a bit extreme for your tastes, but you get the picture. As you become more comfortable with CSS and webpage design, you can develop your own default specifications and implement them using CSS.

The page has come a long way from the beginning of this lesson. Let’s put some final tweaks on the design to match the original finished page.

Final touches

You’re nearly finished; the page needs only a few last touches to make it match the design you

saw at the beginning of the lesson.

- In the rule `article, footer, header, section` delete the property `border:solid 1px #999;`

  In Design view, the elements will display a light gray-colored border, but the black border displayed earlier is now gone.

- In the rule `p, h1, h2, h3, h4, h5, h6, li` add the highlighted value in the following property: `margin: 10px 0;`

- Create the following rule:

```
header { padding:30px;
border-bottom:2px solid #000;
text-align:center; }
```

The final style you need to add is a new CSS3 feature.

- Add the following properties to the `.container` rule:

Click here to view code image

```
-webkit-box-shadow: 0 0 20px 5px rgba(0,0,0,0.40);
box-shadow: 0px 0px 20px 5px rgba(0,0,0,0.40);
```

Advanced CSS properties cannot be seen in Design view.

- Save the file.

Switch to Live view and refresh the display.

In Live view, a drop shadow effect appears behind the main content frame. The sample page is complete. Congratulations! You successfully learned how to use CSS to style nearly all aspects of an entire webpage.

4 Web Design Basics

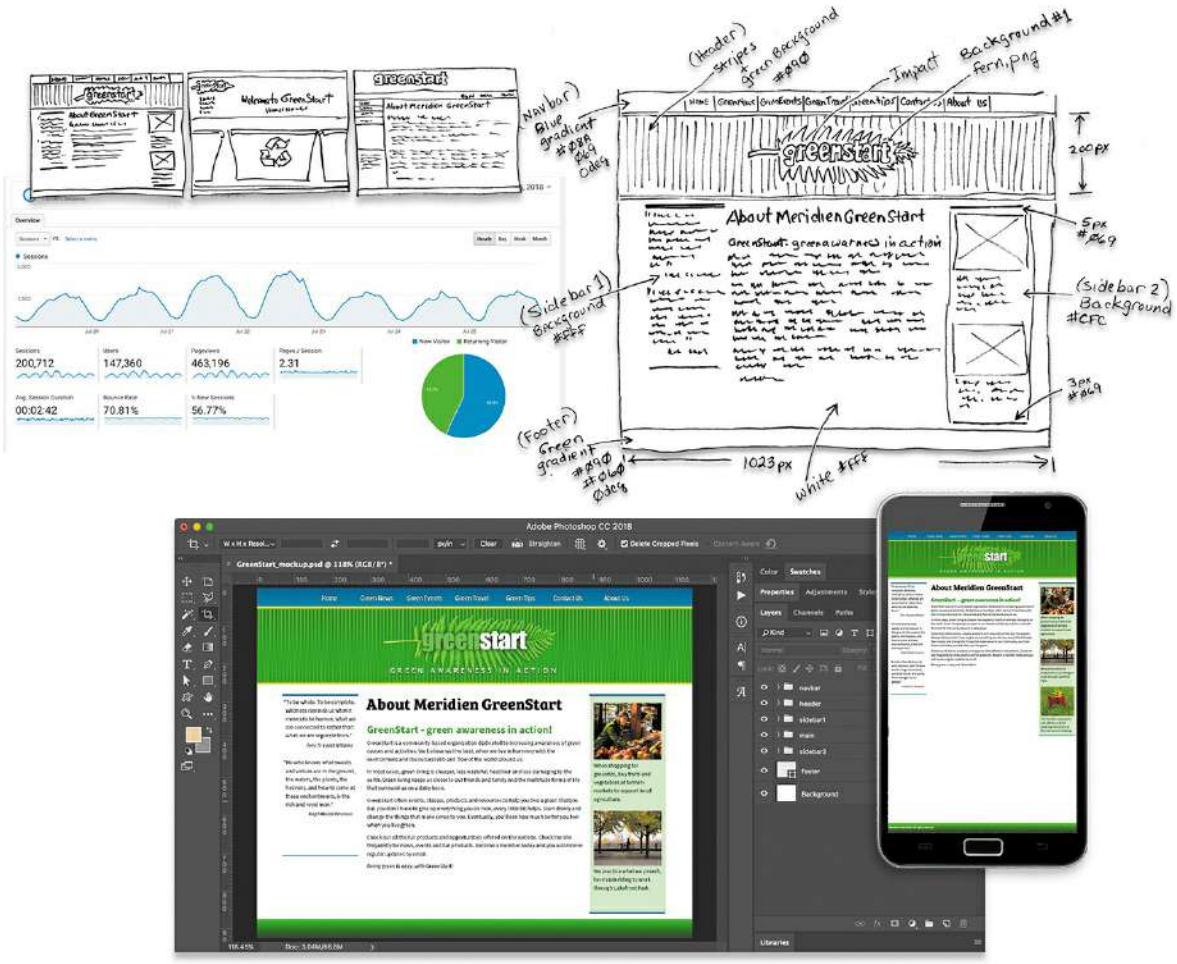
Lesson overview

In this lesson, you'll learn the following:

- The basics of webpage design
- How to create page thumbnails and wireframes
- How to use Adobe Photoshop to generate site image assets automatically

This lesson will take about 30 minutes to complete. Please log in to your account on peachpit.com to download the files for this lesson, or go to the "Getting Started" section at the beginning of this book and follow the instructions under "Accessing the Lesson Files and Web Edition." Store the files on your computer in a convenient location.

Your Account page is also where you'll find any updates to the lessons or to the lesson files. Look on the Lesson & Update Files tab to access the most current content.



Whether you use thumbnails and wireframes, Photoshop, or just a vivid imagination, Dreamweaver can quickly turn your design concepts into complete, standards-based CSS layouts.

Developing a new website

Before you begin any web design project for yourself or for a client, you need to answer three important questions:

What is the purpose of the website?

Who is the audience?

How do they get here?

What is the purpose of the website?

Will the website sell or support a product or service? Is your site for entertainment or games? Will you provide information or news? Will you need a shopping cart or database? Do you need to accept credit card payments or electronic transfers? Knowing the purpose of the website tells

you what type of content you'll be developing and working with and what types of technologies you'll need to incorporate.

Who is the audience?

Is the audience adults, children, seniors, professionals, hobbyists, men, women, everyone? Knowing *who* your audience will be is vital to the overall design and functionality of your site. A site intended for children probably needs more animation, interactivity, and bright, engaging colors. Adults will want serious content and in-depth analysis. Seniors may need larger type and other accessibility enhancements.

A good first step is to check out the competition. Is there an existing website performing the same service or selling the same product? Are they successful? You don't have to mimic others just because they're doing the same thing. Look at Google and Yahoo—they perform the same basic service, but their site designs couldn't be more different from one another.

How do they get here?

This sounds like an odd question when speaking of the Internet. But just as with a brick-and-mortar business, your online customers can come to you in a variety of ways. For example, are they accessing your site on a desktop computer, laptop, tablet, or smartphone? Are they using high-speed Internet, wireless, or dial-up service? What browser are they most likely to use, and what is the size and resolution of the display?

These answers will tell you a lot about what kind of experience your customers will expect. Dial-up and smartphone users may not want to see a lot of graphics or video, whereas users with large flat-panel displays and high-speed connections may demand as much bang and sizzle as you can send at them.

So where do you get this information? Some you'll have to get through painstaking research and demographic analysis. Some you'll get from educated guesses based on your own tastes and understanding of your market. But a lot of it is actually available on the Internet itself. W3Schools, for one, keeps track of tons of statistics regarding access and usage, all updated regularly:

- http://w3schools.com/browsers/default.asp provides information about browser statistics.
- http://w3schools.com/browsers/browsers_os.asp gives the breakdown on operating systems. In 2011, W3Schools started to track the usage of mobile devices on the Internet.
- http://w3schools.com/browsers/browsers_display.asp lets you find out the latest information on the resolution, or size, of screens using the Internet.

If you are redesigning an existing site, your web-hosting service itself may provide valuable statistics on historical traffic patterns and even the visitors themselves. If you host your own site, you can incorporate third-party tools, such as Google Analytics or Adobe Analytics, into your code to do the tracking for you for free or for a small fee.

As of the summer of 2018, Windows desktop computers still dominate the Internet (76 percent), with most users favoring Google Chrome (79 percent), followed by Firefox (10.9 percent), and with various versions of Edge/Internet Explorer (3.9 percent) a distant third. The vast majority of desktop browsers (98 percent) are set to a resolution higher than 1024 pixels by 768 pixels.

These statistics would be great news for most web designers and developers if it weren't for the rapid growth in usage of tablets and smartphones for accessing the Internet. But designing a website that can look good and work effectively on both flat-panel desktop displays and smartphones is a tall order.

Analytics provides comprehensive statistics on the visitors to your site. Google Analytics, pictured here, is a popular choice.

Responsive web design

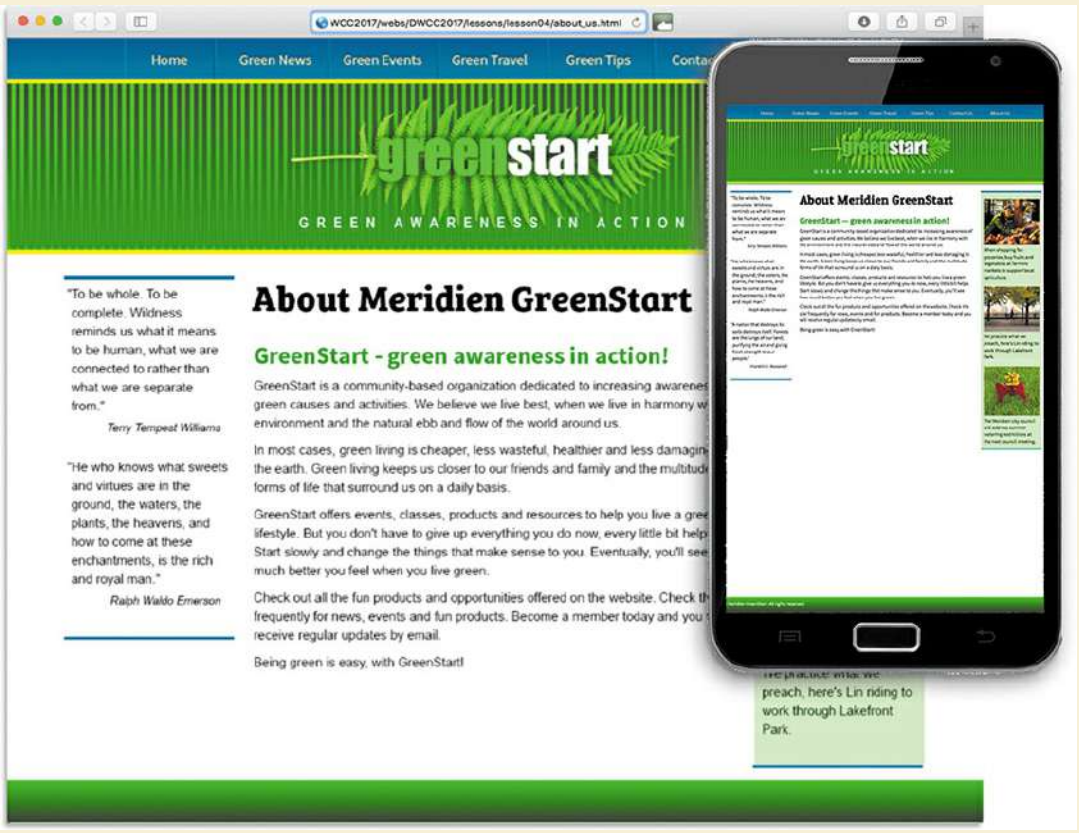
Each day, more people are using smartphones and other mobile devices to access the Internet. Some people may use them to access the Internet more frequently than they use desktop computers. This presents a few nagging challenges to web designers. For one thing, smartphone screens are a fraction of the size of even the smallest flat-panel display. How do you cram a two- or three-column page design into a meager 300 to 400 pixels?

Until the last five years or so, web design usually required that you target an optimum size (height and width in pixels) for a webpage and then build the entire site on these specifications. Today, that scenario is becoming a rare occurrence. Now, you are presented with the decision to build a site that either can scale to any size display (responsive) or can morph to support a few target display types for desktop and mobile users (adaptive).

Your own decision will be based in part on the content you want to provide and on the capabilities of the devices accessing your pages. Building an attractive website that supports video, audio, and other dynamic content is hard enough without throwing in a

panoply of different display sizes and device capabilities. The term *responsive web design* was coined, in a book of the same name (2011), by a Boston-based web developer named Ethan Marcotte. In the book, he describes the notion of designing pages that can adapt to multiple screen dimensions automatically. Along with more standard techniques, you will learn many techniques for responsive web design and implement them in your site and asset design later in this book.

Many of the concepts of print design are not applicable to the web, because you are not in control of the user's experience. For example, print designers know in advance the page size for which they are designing. The printed page doesn't change when you rotate it from portrait to landscape. On the other hand, a page carefully designed for a typical flat panel is basically useless on a smartphone.



Scenario

For the purposes of this book, you'll be working to develop a website for Meridien GreenStart, a fictitious community-based organization dedicated to green investment and action. This website will offer a variety of products and services and require a broad range of webpage types, including dynamic pages using technologies such as jQuery, which is a form of JavaScript.

Your customers come from a wide demographic that includes all ages and educational levels.

They are people who are concerned about environmental conditions and who are dedicated to conservation, recycling, and the reuse of natural and human resources.

Your marketing research indicates that most of your customers still use desktop computers or laptops, connecting via high-speed Internet services. You can expect to get 20 to 30 percent of your visitors exclusively via smartphone and other mobile devices, and much of the rest will be using mobile from time to time.

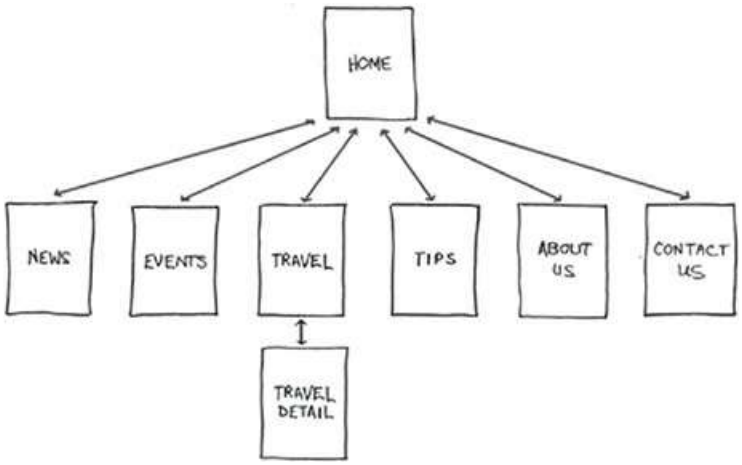
To simplify the process of learning Dreamweaver, we'll focus on creating a fixed-width desktop site design first. Later, you'll learn how to adapt your fixed-width design to work with smartphones and tablets.

Working with thumbnails and wireframes

After you have nailed down the answers to the three questions about your website purpose, customer demographic, and access model, the next step is to determine how many pages you'll need, what they will do, and what they will look like.

Creating thumbnails

Many web designers start by drawing thumbnails with pencil and paper. Think of thumbnails as a graphical shopping list of the pages you'll need to create for the website. Thumbnails can help you work out the basic navigation structure for the site. Draw lines between the thumbnails showing how the site navigation will connect them.

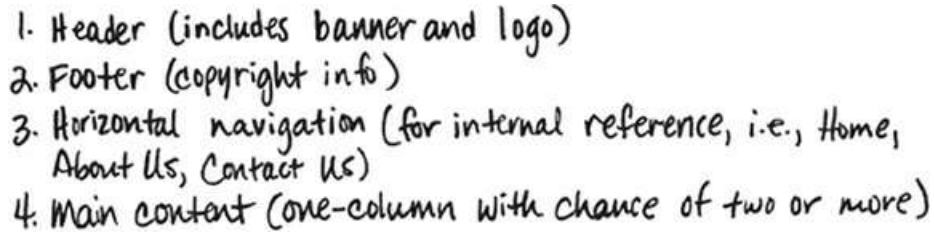


Thumbnails list the pages that need to be built and how they are connected to each other.

Most sites are divided into levels. Typically, the first level includes all the pages in your main navigation menu—the ones a visitor can reach directly from the home page. The second level includes pages you can reach only through specific actions or from specific locations, say from a shopping cart or product detail page.

Creating a page design

Once you've figured out what your site needs in terms of pages, products, and services, you can then turn to what those pages will look like. Make a list of components you want or need on each page, such as headers and footers, navigation, and areas for the main content and the sidebars (if any). Put aside any items that won't be needed on every page. What other factors do you need to consider? If mobile devices are going to be an important consideration of your design identity, will any of the components be required (as opposed to optional) for these devices? Although many components can simply be resized for mobile screens, some will have to be completely redesigned or reimagined.



Identifying the essential components for each page helps you create a page design and structure that will meet your needs.

Do you have a company logo, business identity, graphic imagery, or color scheme you want to match or complement? Do you have existing or proposed publications, brochures, or advertising campaigns you want to emulate? It helps to gather them all in one place so you can see everything all at once on a desk or conference table. If you're lucky, a theme will rise organically from this collection. In some cases, the print identity and publications will evolve from the web design.

Desktop or mobile

Once you've created your checklist of the components that you'll need on each page, sketch out several rough layouts that work for these components. Depending on your target visitor demographics, you may decide to focus on a design that's optimized for desktop computers or one that works best on tablets and smartphones.

Most designers settle on one basic page design that is a compromise between flexibility and sizzle. Some site designs may naturally lean toward using more than one basic layout. But resist the urge to design each page separately. Minimizing the number of page designs may sound like a major limitation, but it's key to producing a professional-looking site that's easy to manage. It's the reason why some professionals, such as doctors and airline pilots, wear uniforms. Using a consistent page design, or template, conveys a sense of professionalism and confidence to your visitor. While you're figuring out what your pages will look like, you'll have to address the size and placement of the basic components. Where you put a component can drastically affect its impact and usefulness.

In print, designers know that the upper-left corner of a layout is considered one of the "power positions," a place where you want to locate important aspects of a design, such as a logo or title. This is because in western culture we read from left to right, top to bottom. The second power

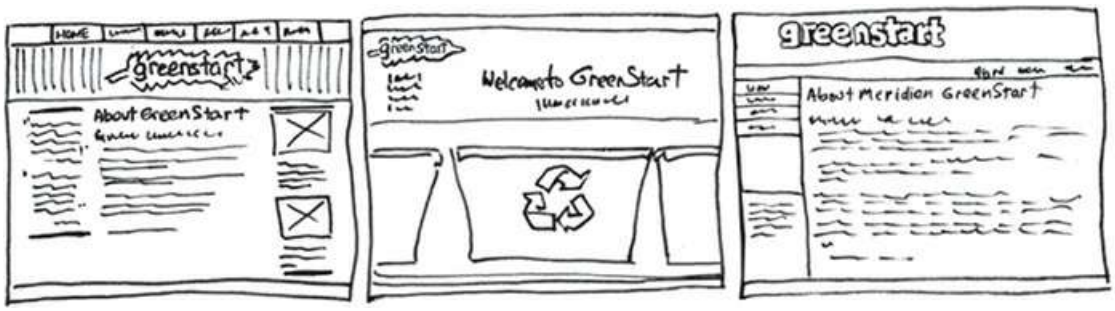
position is the lower-right corner, because this is the last thing your eyes will see when you're finished reading.

Unfortunately, in web design this theory doesn't hold up for one simple reason: You can never be certain how the user is seeing your design. Are they on a 20-inch flat panel or a 3-inch-wide smartphone?

In most instances, the only thing you can be certain of is that the user can see the upper-left corner of any page. Do you want to waste this position by slapping the company logo here? Or make the site more useful by slipping in a navigational menu? This is one of the key predicaments of the web designer. Do you go for design sizzle, workable utility, or something in between?

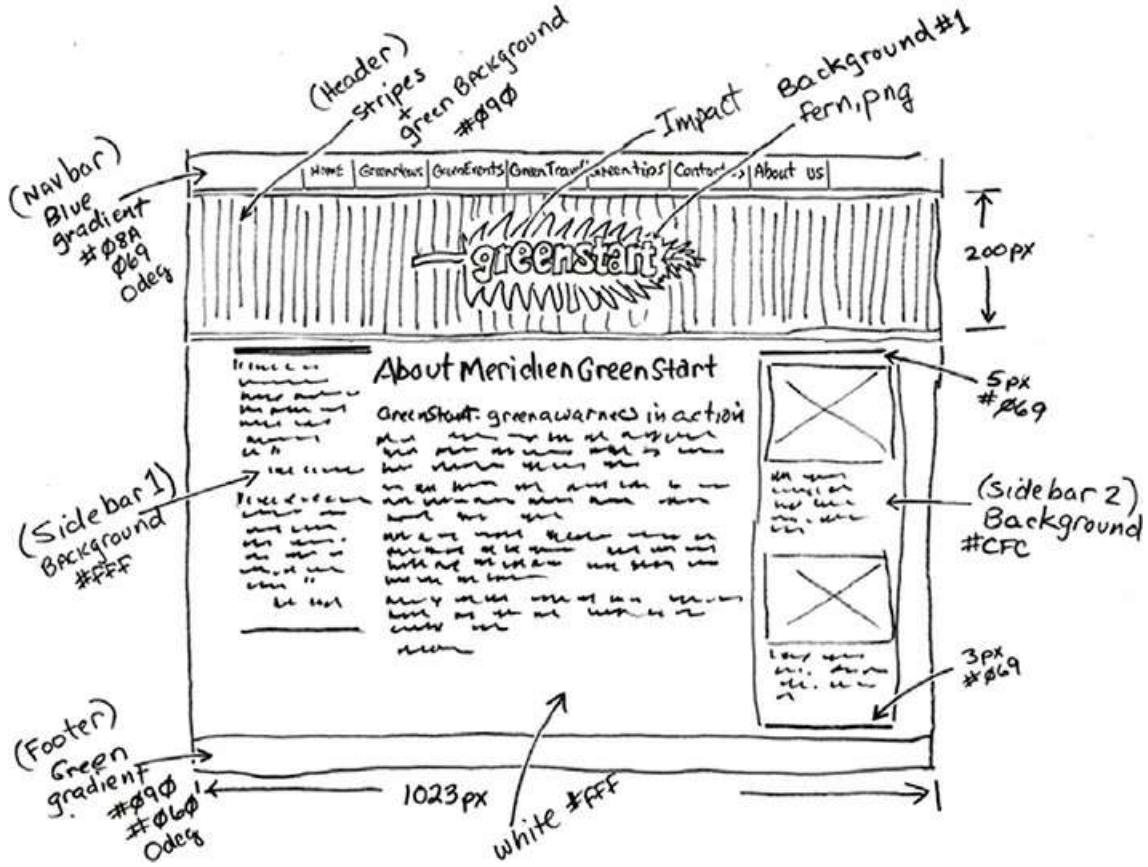
Creating wireframes

After you pick the winning design, wireframing is a fast way to work out the structure of each page in the site. A wireframe is like a thumbnail, but bigger, that sketches out each page and fills in more details about the components, such as actual link names and main headings, but with minimal design or styling. This step helps to anticipate problems before you smack into them when working in code. What might take you hours or days to produce digitally can be sketched out in minutes by hand.



Wireframes allow you to experiment with page designs quickly and easily without wasting time with code.

Once the basic concepts are worked out, many designers take an extra step and create a full-size mockup or "proof of concept" using a program like Photoshop or even Adobe Illustrator. It's a handy thing to do because you'll find that some clients just aren't comfortable giving approvals based only on pencil sketches. The advantage here is that all these programs allow you to export the results to full-size images (JPEG, GIF, or PNG) that can be viewed in a browser as if they were finished webpages. Such mockups are as good as seeing the real thing but may take only a fraction of the time to produce.



The wireframe for the final design should identify all components and include specific information about content, color, and dimensions.

Note

You should be able to open the sample file with any version of Photoshop CC or higher. Be aware that if you use a version different from the one pictured, the panels and menu options may appear different.

Note

The mockup uses fonts from Typekit, Adobe's online font service. To view the final design properly in Photoshop, you will need to download and install these fonts. Typekit fonts are included in your subscription to Creative Cloud.

To demonstrate how a graphics program could be used to build such a mockup, I created a sample webpage layout using Photoshop and saved it into the Lesson 4 resources folder. Let's take a look.

- Launch Photoshop CC or higher.
- Open **GreenStart_mockup.psd** from the lesson04/resources folder.

The Photoshop file contains a complete mockup of the GreenStart site design, which is composed of various vector-based design components as well as image assets stored in separate layers. Note the use of colors and gradients in the design. Feel free to experiment with the layers and various components to see how they were created.

In addition to creating graphical mockups, Photoshop has tricks geared specifically for web designers. So that you can see these features firsthand, I've provided bonus online Lesson 4, where you can learn how to use Photoshop to create your web image assets from this file. Check out the "Getting Started" section at the beginning of the book to learn how to access the bonus lessons.

Review questions

1. What three questions should you ask before starting any web design project?
2. What is the purpose of using thumbnails and wireframes?
3. Why is it important to create a design that takes into account smartphones and tablets?
4. What is responsive design, and why should Dreamweaver users be aware of it?
5. Why would you use Photoshop and Illustrator, or other programs, to create design mockups for a website?

Review answers

1. What is the purpose of the website? Who is the audience? How did they get here? These questions, and their answers, are essential in helping you develop the design, content, and strategy of your site.
2. Thumbnails and wireframes are quick techniques for roughing out the design and structure of your site without having to waste lots of time coding sample pages.
3. Mobile device users are one of the fastest-growing demographics on the web. Many visitors will use a mobile device to access your website on a regular basis or exclusively. Webpages designed for desktop computers often display poorly on mobile devices, making the websites difficult or impossible to use for these mobile visitors.
4. Responsive design is a method for making the most effective use of a webpage, and its content, by designing it to adapt to various types of displays and devices automatically.
5. Using Photoshop and Illustrator you can produce page designs and mockups much faster than when designing in code with Dreamweaver. Designs can even be exported as web-compatible graphics that can be viewed in a browser to get client approval.

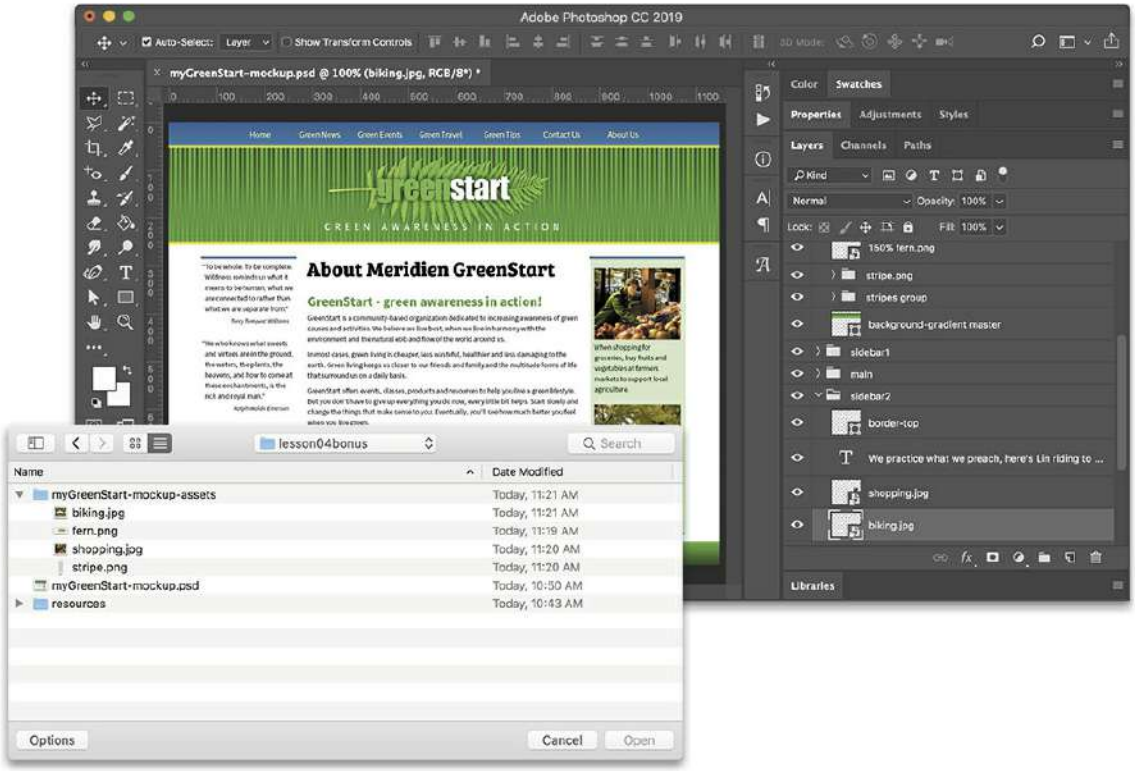
4 Creating Web Assets Using Photoshop Generator Bonus

Lesson overview

In this lesson, you will learn the following:

- How to access and set up Photoshop Generator
- How to create web assets using Photoshop Generator

This lesson will take about 30 minutes to complete. If you have not already done so, please log in to your account on peachpit.com to download the project files for this lesson as described in the “Getting Started” section at the beginning of this book and follow the instructions under “Accessing the Lesson Files and Web Edition.” Define a site based on the lesson04bonus folder.



There are tons of little-known productivity enhancements in Adobe apps—like Photoshop Generator—that can make mundane tasks like creating web image assets fast and easy.

Working with Adobe Photoshop Generator

Adobe Photoshop Generator is a web-oriented tool that allows you to export web assets from a Photoshop file in a variety of sizes, resolutions, and even file types. Best of all, this feature works in real time, exporting image assets from your file based on user-specified attributes added to the layer name. Let’s take a closer look at this great tool for web designers and developers.

Note

This exercise should work with any version of Photo-shop CC and the lesson04bonus lesson files. Be aware that if you use a different version than the one pictured, the dialog and menu options may vary.

Note

The sample file used in this exercise requires the font Bree Serif, which you can download and install for free from Adobe Type-kit. To access this font, and the

entire Typekit library, choose Type > Add Fonts from Typekit in Photoshop CC 2015 or higher.

In this exercise, you'll work with an Adobe Photoshop document to prepare assets for your web project.

- Launch Photoshop CC (2019 release).
- Open **GreenStart-mockup.psd** from the lesson04bonus/resources folder.

Tip

It's a good idea to save the file under a different name so you can refer back to the original assets should you make an error.

The Photoshop file contains a complete mockup of the GreenStart site design, which is composed of various vector-based design components as well as image assets stored in separate layers. Note the use of colors and gradients in the design.

- Choose File > Save As.

  Navigate to the lesson04bonus folder.

  Name the file **myGreenStart-mockup.psd** and save the file.

- If necessary, choose Window > Layers to display the Layers panel. Observe the names and contents of the layers and layer groups.

The layers and layer groups are named for the webpage components.

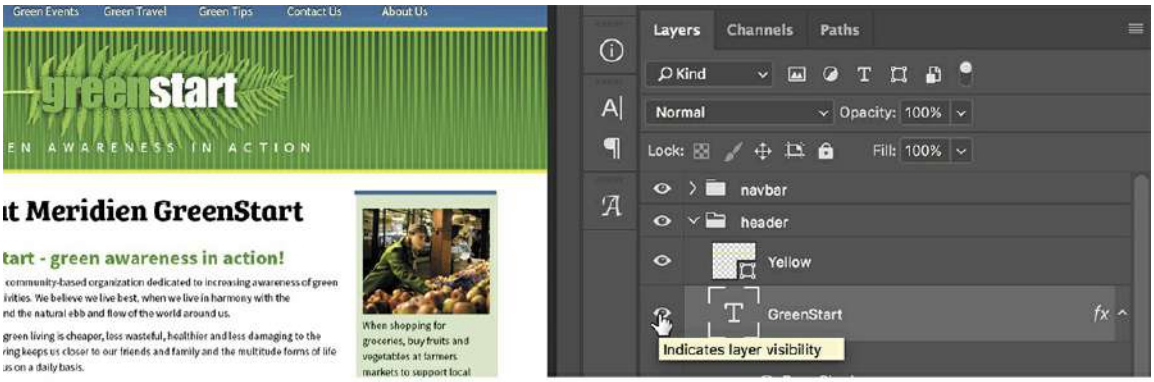
Note

The motto text uses the font Lucida Sans, which may not be installed on every computer and is not available from Typekit. Feel free to substitute Arial or Verdana for this font in this exercise.

. Open the header layer group and observe the contents.

The header group contains two text elements and four graphical elements. Often, it's difficult to understand how a graphic component is built or for what purpose it is intended by looking at the layer names alone.

. In the *GreenStart* text layer, click the eye icon to toggle the layer visibility off.



The text "greenstart" disappears from the layout.

. In the fern layer, click the eye icon to toggle the layer visibility off.

The image of the fern disappears. Using this method, you can identify each element of the header and see what role it plays in the creation of the overall design. The number and type of layers and the level of detail used here aren't necessary for a mockup that you merely want to use for client approval. This file was set up specifically to create many of the final assets for

the webpage design.

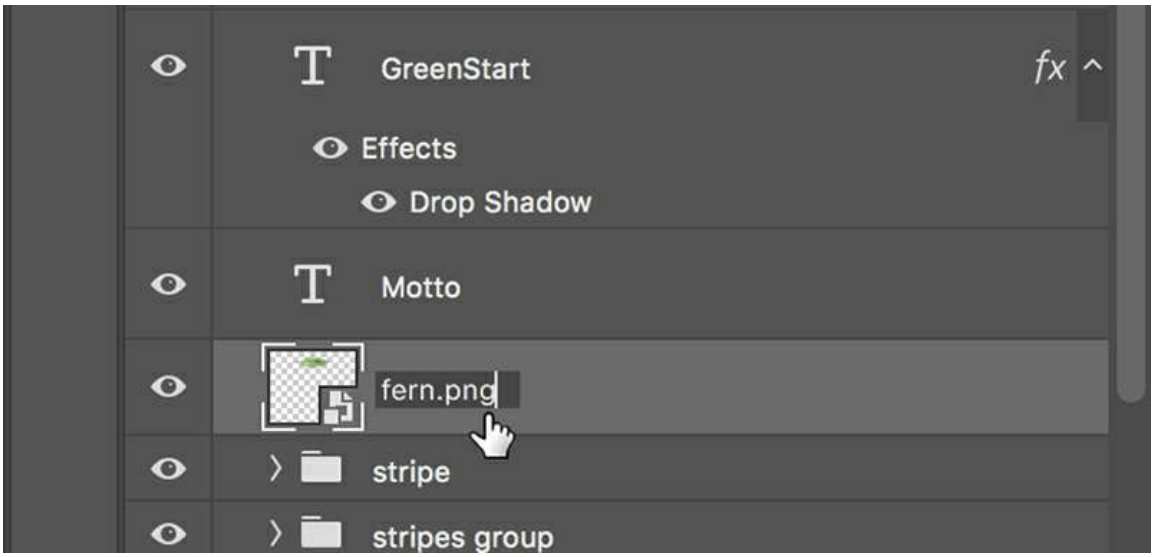
- Click the eye icon to toggle the *GreenStart* and *fern* layers back on.

There are several ways to create image assets from a Photoshop mockup; Generator is just one option.

Exporting assets from Photoshop

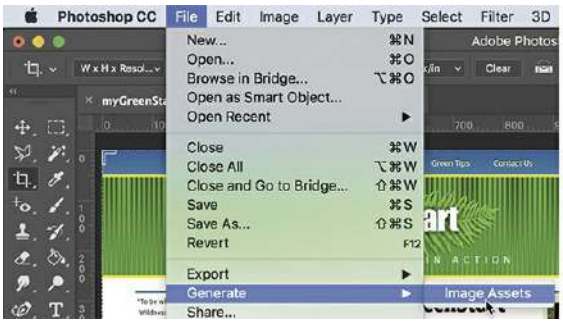
The *fern* layer will be used to create one component of the header background. Photoshop generates a web asset automatically if you add a file extension to the layer name.

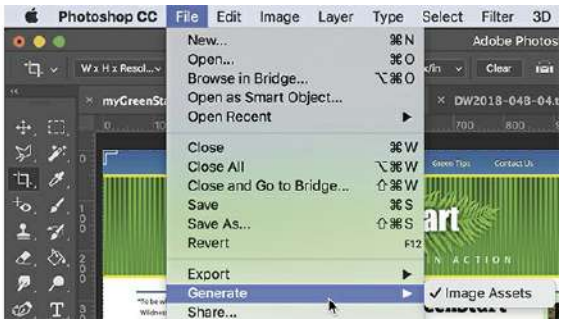
- Double-click the name of the *fern* layer; type **fern.png** and press Enter/Return to complete the name.



When activated, Generator works in the background, exporting assets in real time.

- Choose File > Generate > Image Assets.



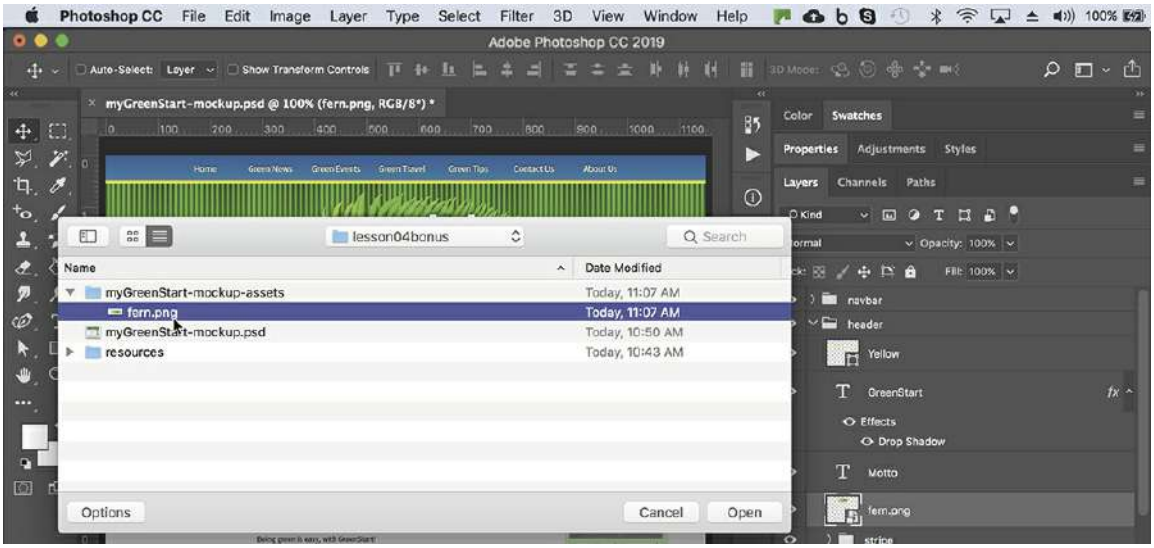


The next time you look at the Generate menu option, a checkmark will appear beside Image Assets, indicating that Generator is active.

- Choose File > Open. Navigate to the lesson04bonus > resources folder.

A new folder has been created and named **myGreenStart-mockup-assets** by Generator. Whenever you add file extensions to the layer names and enable Generator, it creates a folder and fills it with assets automatically, based on the layer names and settings.

- Navigate to the **myGreenStart-mockup-assets** folder created by Photoshop.



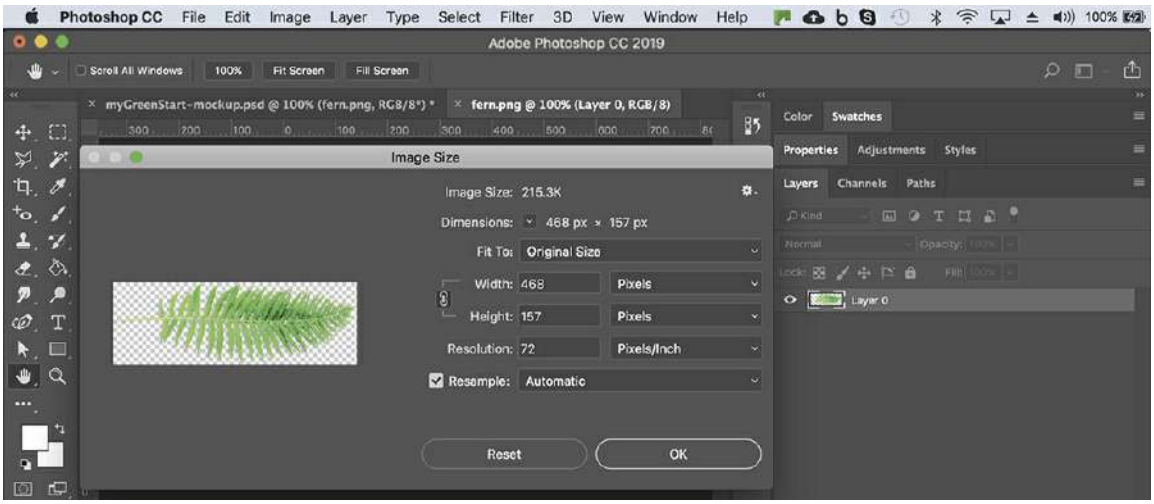
The folder contains a single image: **fern.png**.

- Open **fern.png**.

The file contains the fern with a transparent background. Note that the image displays a drop shadow. The shadow is a permanent part of the image, no longer created by a Photoshop effect.

- Choose Image > Image Size.

Note the dimensions and resolution of the image.



The image is 468 pixels by 157 pixels, at 72 pixels per inch (ppi).

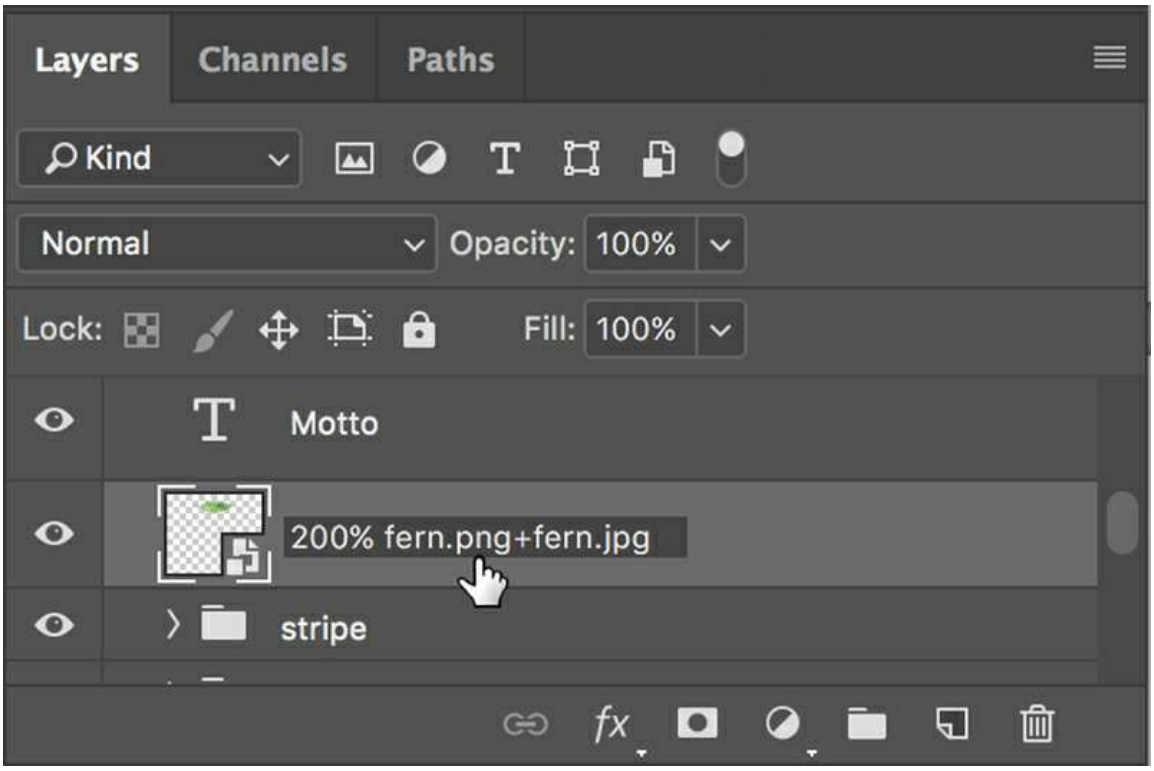
- Click OK to close the Image Size dialog. Close **fern.png**.

As you can see, Photoshop provides a powerful feature for creating web assets automatically simply based on the names you apply to image layers. But the feature doesn't stop there.

Creating multiple assets using Generator

Generator can also modify the default export specifications and even create multiple assets at multiple resolutions all at the same time.

- Change the layer name from *fern.png* to **200% fern.png+fern.jpg** and press Enter/Return to complete the name.



- Choose File > Open.

  If necessary, navigate to the **myGreenStart-mockup-assets** folder.

  The folder contains two images: **fern.png** and **fern.jpg**, both created by Generator. Older files with the same names are replaced automatically.

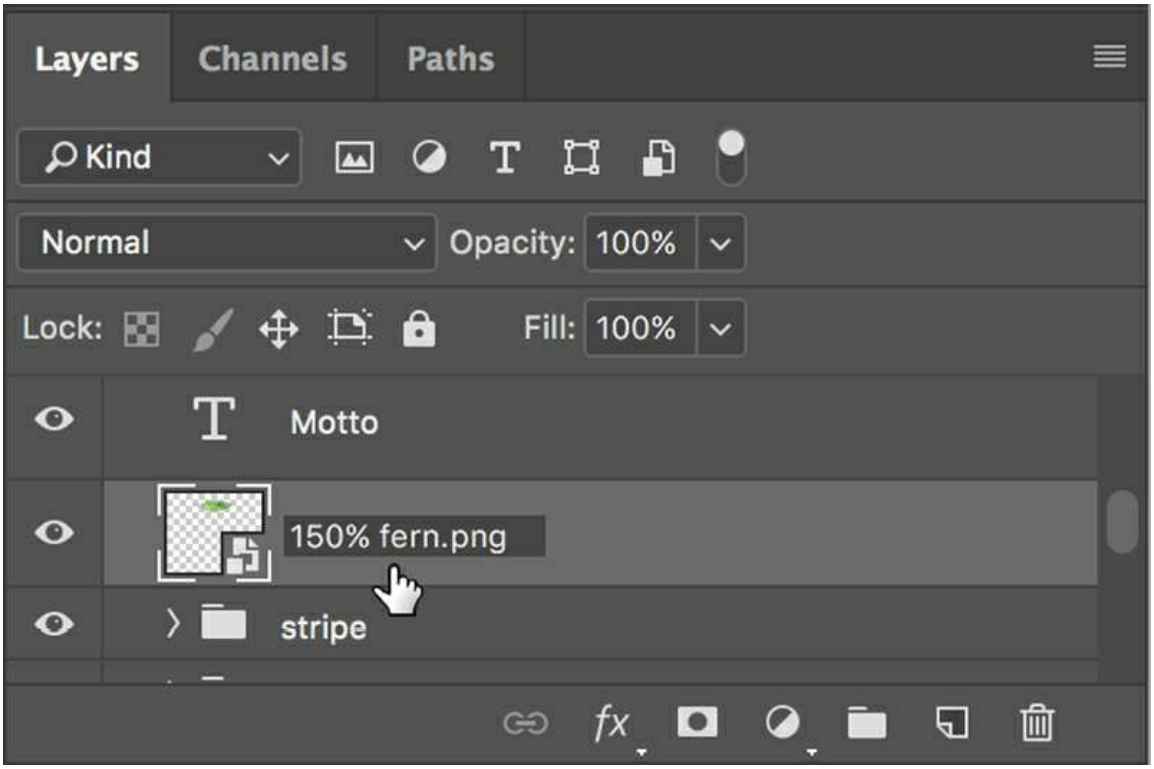
- Select **fern.png** and open the file.

  The file contains the fern, but it appears twice as large as the previous image.

- Open **fern.jpg**.

  The file contains the same fern image, but at the original size. JPEGs do not support transparency, so the background is white. There's no need for the JPEG version of the image. Generator can remove assets automatically too.

- Close **fern.jpg** and **fern.png**.
- In **myGreenStart-mockup.psd, c**hange the layer name from *200% fern.png+fern.jpg* to **150% fern.png** and press Enter/Return to complete the name.



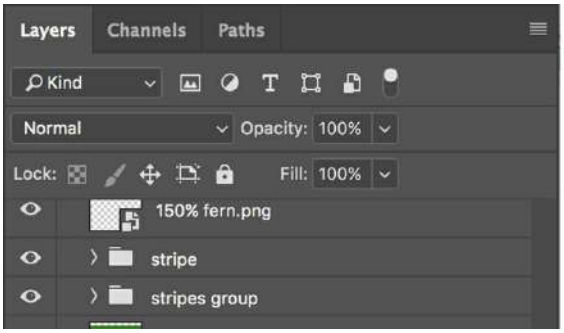
The specification creates a larger, higher-quality image that will display well in regular browsers and on new higher-resolution devices. Because you removed "fern.jpg" from the layer name, Generator automatically deleted the JPEG version of the file in the resources folder.

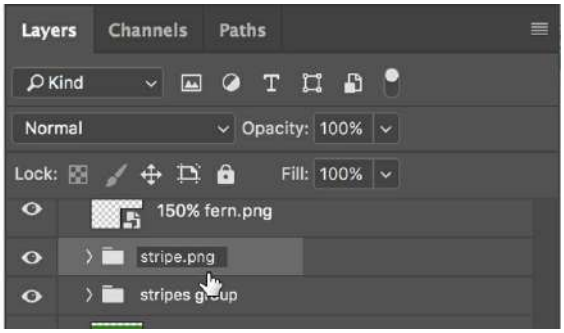
- Choose File > Open. Navigate to the myGreenStart-mockup-assets folder.

The JPEG version of the file is no longer visible in the folder. As you can see, Photoshop generates assets based on the name of the layer. You can create an entire set of images for your site design from the layers in this file.

- Press the Esc key or click Cancel to close the Save dialog.
- In the Layers panel, open the *header* layer group, if necessary.

Change the name of the *stripe* layer group to **stripe.png**.





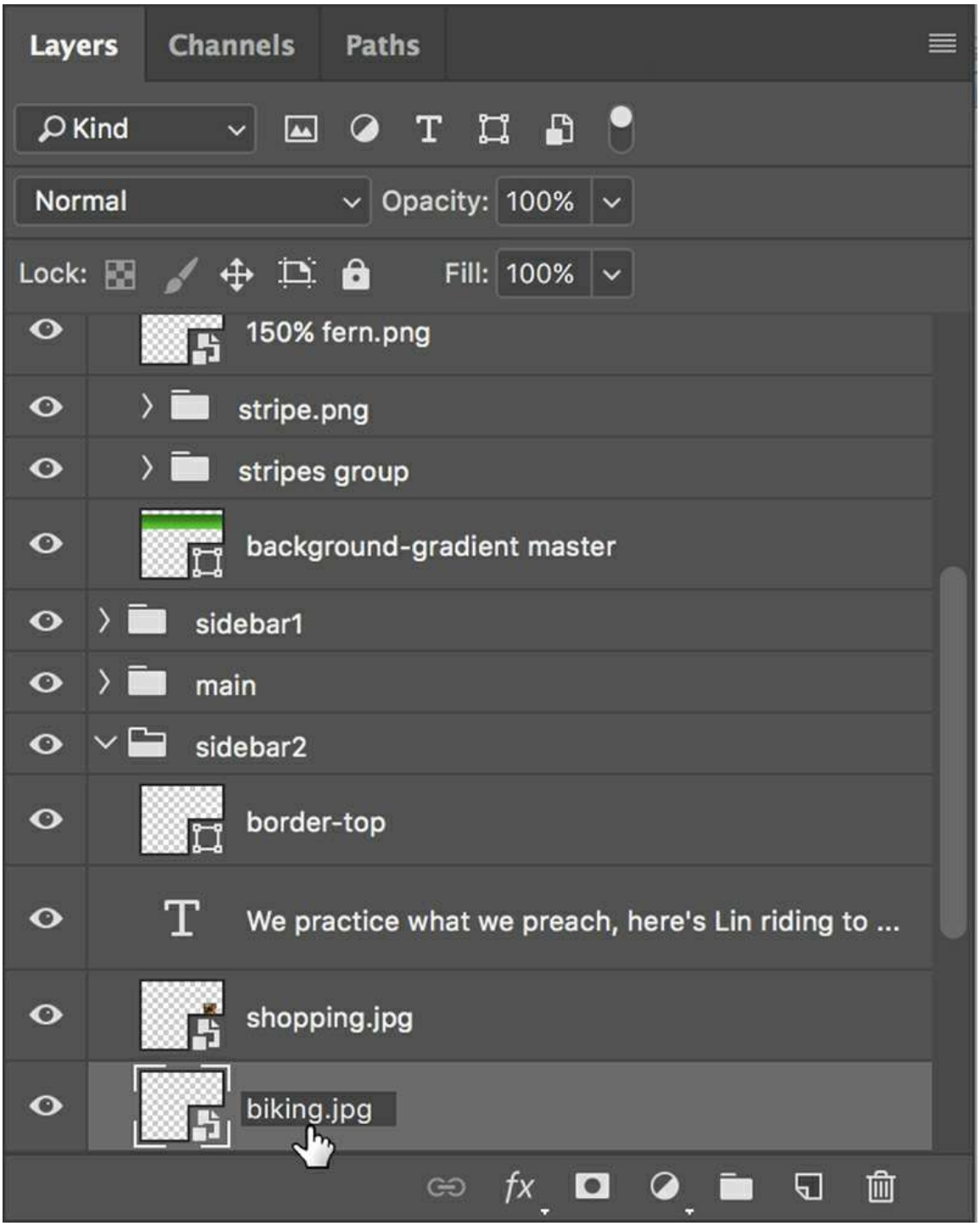
. Open the *sidebar2* layer group.

Change the layer *shopping* to **shopping.jpg**.

Change the layer *biking* to **biking.jpg**.

Note

In Photo-shop, the background is mocked up using two layers: *stripe* and *stripes* group. You are exporting the web background from the smaller stripe (singular) layer group.

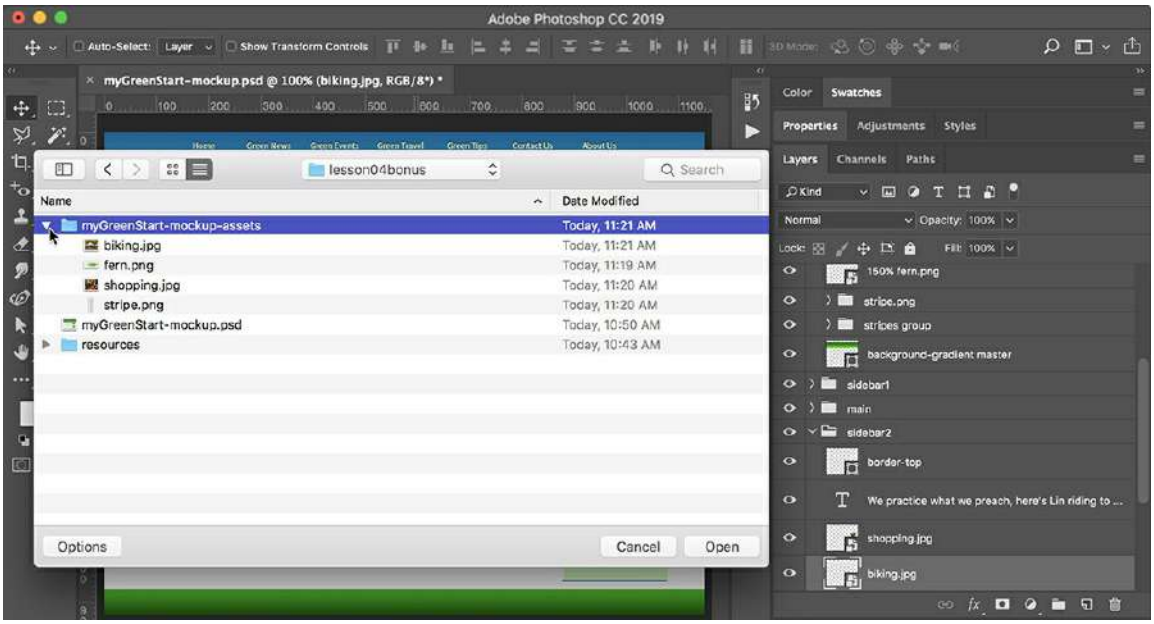


. Choose File > Open. Navigate to the myGreenStart-mockup-assets folder.

Note

If you do not see the images created as described, check to make sure that the command File > Generate > Image Assets is enabled as described in the previous

exercise.



The myGreenStart-mockup-assets folder now contains four image files. These files are identical to the ones you will use to build the site template and populate articles in upcoming lessons.

. Close the dialog. Close **myGreenStart-mockup.psd**.

Feel free to save your changes, if desired.

As you can see, Generator offers some handy tools for turning this mockup into real web design assets, but the tricks don't stop there. Photoshop and Dreamweaver have even more collaboration tools to offer, as you will see in Lesson 5, "Creating a Page Layout," and Lesson 8, "Working with Images."

Check out https://helpx.adobe.com/photoshop/using/generate-assets-layers.html to see a complete explanation of Adobe Generator and how to use it.

5 Creating a Page Layout

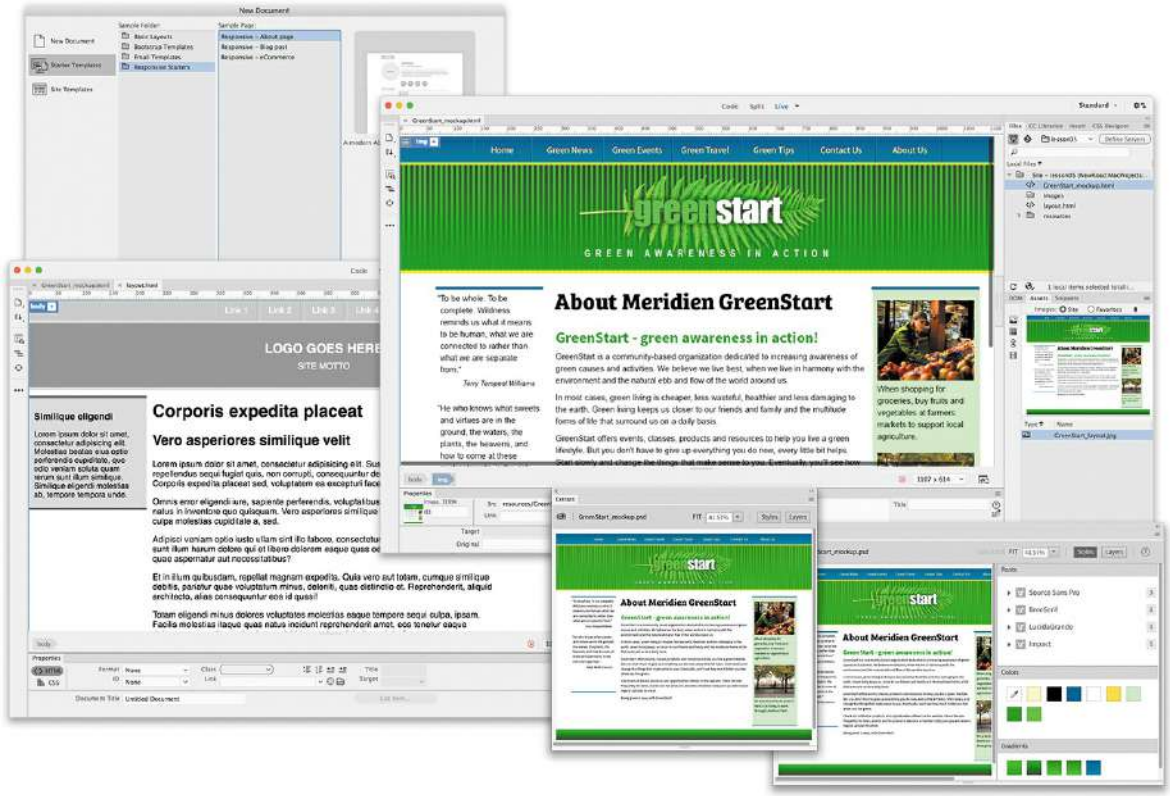
Lesson overview

In this lesson, you'll learn how to work faster, make updating easier, and be more productive. You'll learn how to do the following:

- Evaluate basic page structure from design mockups
- Upload a Photoshop mockup as a Creative Cloud asset
- Extract styling, text, and image assets from a Photoshop mockup
- Apply extracted styles, text, and image assets to an HTML page in Dreamweaver

This lesson will take about 1 hour and 15 minutes to complete. If you have not already done so, please log in to your account on peachpit.com to download the project files for this lesson as described in the "Getting Started" section at the beginning of this book and follow the instructions under "Accessing the Lesson Files and Web Edition." Define a site based on the lesson05 folder.

Your Account page is also where you'll find any updates to the lessons or to the lesson files. Look on the Lesson & Update Files tab to access the most current content.



Dreamweaver provides powerful tools with which to apply styling, text, and image assets created in other Adobe applications, such as Photoshop.

Evaluating page design options

In the previous lesson, you went through the process of identifying the pages, components, and structures you would need for a specific website. The selected design balances those needs against a variety of other factors, such as the types of visitors that may come to the site and their means of connecting to it. In this lesson, you will learn how to implement those structures and components in a basic layout.

Since there are almost unlimited ways to build a design, we'll concentrate on building a simple structure that uses the minimum number of HTML5 semantic elements. This will produce a page design that is easy to implement and maintain. Let's start by taking a look at the mockup introduced in Lesson 4, "Web Design Basics."

In Dreamweaver, open **GreenStart_mockup.html** from the lesson05 folder.

This HTML file contains an image depicting the final mockup of the GreenStart site design that you saw in Lesson 4. The design can be broken down into basic components: header, footer, navigation, and main and sidebar content elements. If you diagrammed this scheme over the mockup, it might look like the following figure.

Once you identify the basic page component scheme, you could then break down the diagram into basic HTML elements, like the following:

Although the element is perfectly acceptable and still in wide use as a page component, it is a holdover from HTML 4 and comes with some disadvantages. For example, it makes the underlying code more complex by requiring the use of class and/or id attributes to help delineate the various components within the design.

Today, web designers are instead using the new HTML5 elements to simplify their designs and to add semantic meaning to their code. If you substitute the elements with HTML5 structures, it's easy to see how simple the layout can be.

Once you diagram and break down a design into its components, you could start creating the basic structure right away. But before you spend any time creating the new layout by hand, Dreamweaver may offer a simpler alternative. Leave the mockup open and feel free to refer to it as needed as a reference.

Working with predefined layouts

Dreamweaver has always tried to offer the latest tools and workflows to all web designers, regardless of their skill level. For example, over the years, the program has provided a selection of predefined templates, various page components, and code snippets to make the task of building and populating webpages fast and easy.

Often, the first step of building a website was to see whether one of its predefined layouts matched your needs or whether your needs could be adapted to one of the available designs.

Dreamweaver CC (2019 release) continues this tradition by providing sample CSS layouts and frameworks that you can adapt to many popular types of projects. You can access these samples from the File menu.

. Choose File > New Document.

The New Document dialog appears. Dreamweaver allows you to build a wide spectrum of web-compatible documents besides those built using HTML, CSS, and JavaScript. The New Document dialog displays many of these document types, including PHP, XML, and SVG. Predefined layouts, templates, and frameworks can also be accessed from this dialog.

At the time of this writing, Dreamweaver CC (2019 release) offers three basic layouts, six Bootstrap templates, four email templates, and three responsive starter layouts. The exact number and features of these layouts may change over time through automatic updates via Creative Cloud. The changes to this list may occur without notice or fanfare, so keep your eyes peeled for new options in this dialog.

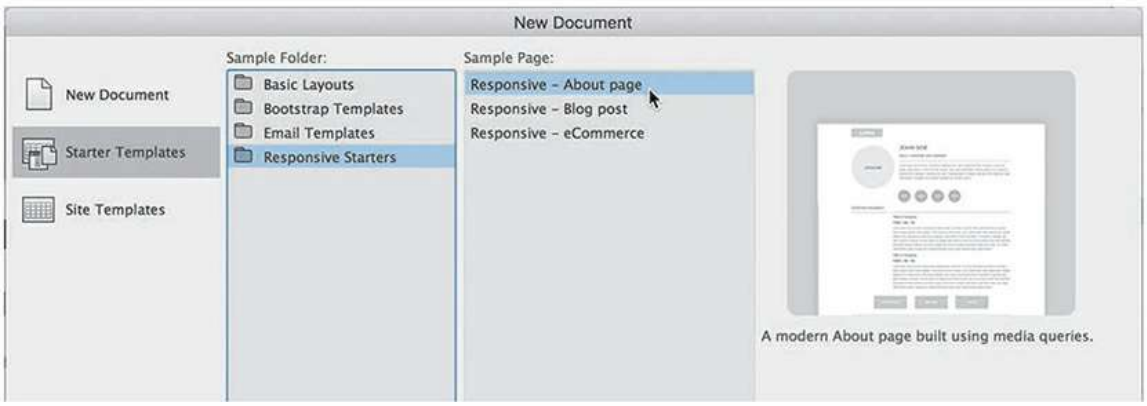
All the featured starter templates have responsive designs built using HTML5-compatible structures and will help you gain valuable experience with this evolving standard. Unless you need to support older browsers (such as IE5 and IE6), there's little to worry about when using these newer designs. Let's check out the options.

. In the New Document dialog, choose Starter Templates > Responsive Starters.

The Starter Templates window of the New Document dialog displays three choices: About Page, Blog Post, and eCommerce.

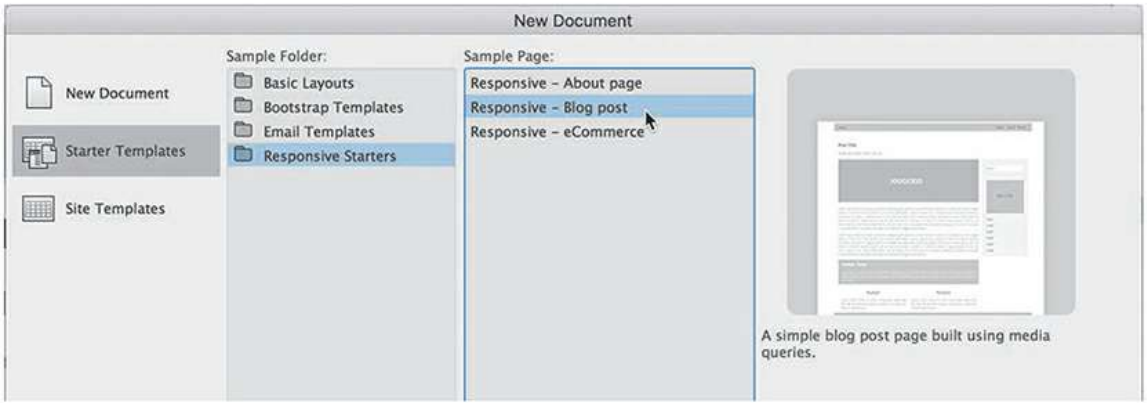
. Select **About Page**.

Observe the preview image in the dialog.



An image appears showing the design of a webpage that will adapt automatically to desktops, tablets, and smartphones.

. Select Blog Post.



The preview image changes to depict the new design.

. Select each of the design options in turn.

Observe the preview image in the dialog.

Each template offers a design appropriate for specific applications, but none of the templates is identical to, or even close to, our chosen design. Instead, I've provided a sample layout that will fit the need much better.

. Click Cancel to close the New Document dialog.

. Open **layout.html** from the lesson05 folder.

The file contains a three-column layout with navigation, header, and footer components. In the following exercises, you will learn how to adapt this layout to make the site template.

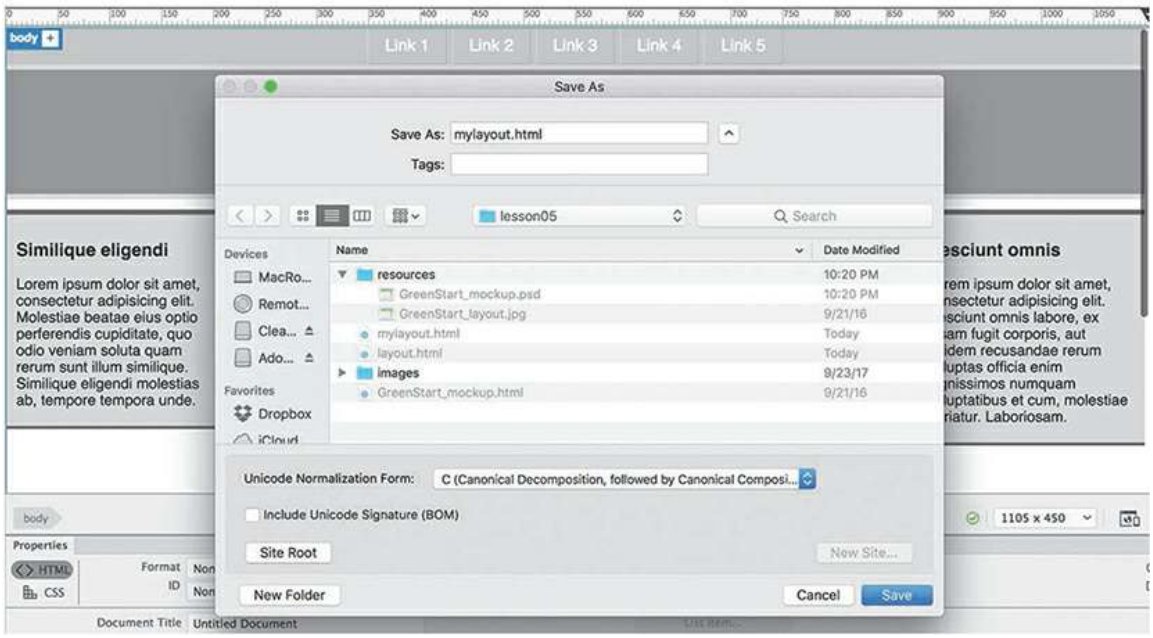
Styling an existing layout

Once you get the skills under your belt, it will be a simple thing to build a webpage layout. For

now, I've provided a sample HTML file that will jumpstart the process of building your site template.

- If necessary, open **layout.html** from the lesson05 folder.
- Select File > Save As.

  Name the file **mylayout.html** and click Save.



  Dreamweaver creates a new version of the layout. Notice that both versions are still open in the document window.

- Close **layout.html**.

The first step is to make this generic layout take on some of the personality of the proposed design. Normally, you would have to do that the old-fashioned way, by editing the CSS by hand. But since the layout was mocked up in Adobe Photoshop, Dreamweaver has a built-in feature called Extract that can use the site mockup to create the desired styling for you.

Extract is a recent addition to Dreamweaver. It is a feature hosted by Creative Cloud and accessed through a panel in the program.

Note

Before accessing the Extract panel, you must have the Creative Cloud desktop app running and be logged in to your account.

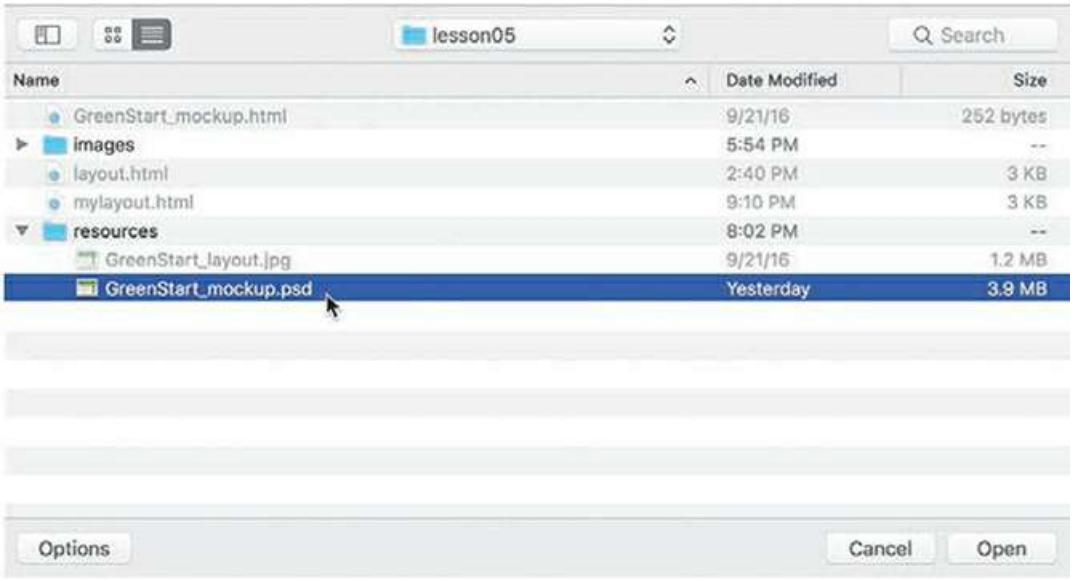
- Select Window > Extract.

The Extract panel appears. The panel connects to your Creative Cloud account and will display any Photoshop files in your assets. To use the site mockup, you first have to upload it to the Creative Cloud server.

. Click the option Upload PSD.

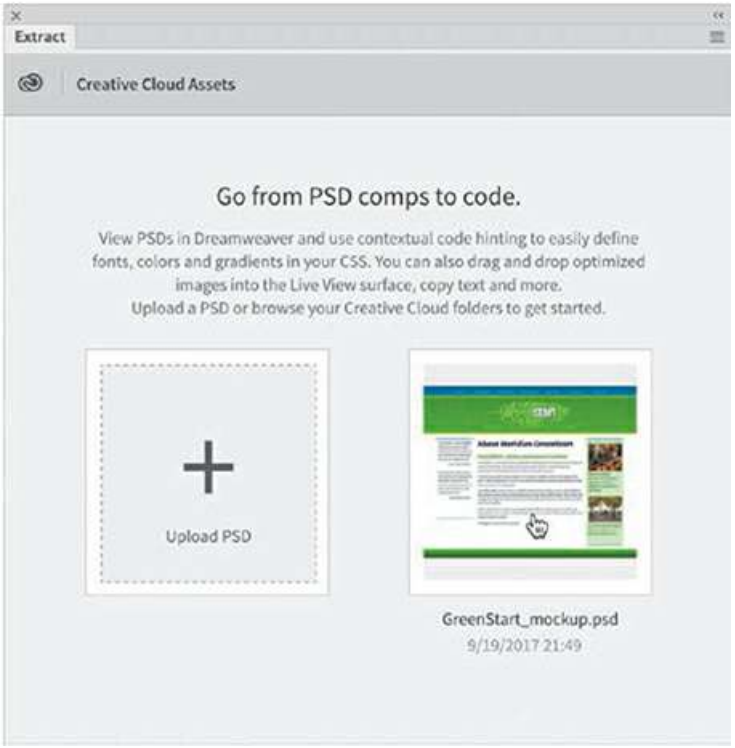
A file dialog appears.

. Select **GreenStart_mockup.psd** in the lesson05/resources folder and click Open.



The file is copied to your Creative Cloud Files folder on your computer, which is then synced to your Creative Cloud remote storage. Once the file is uploaded, it should be visible in the Extract panel.

. Click **GreenStart_mockup.psd** in the Extract panel.

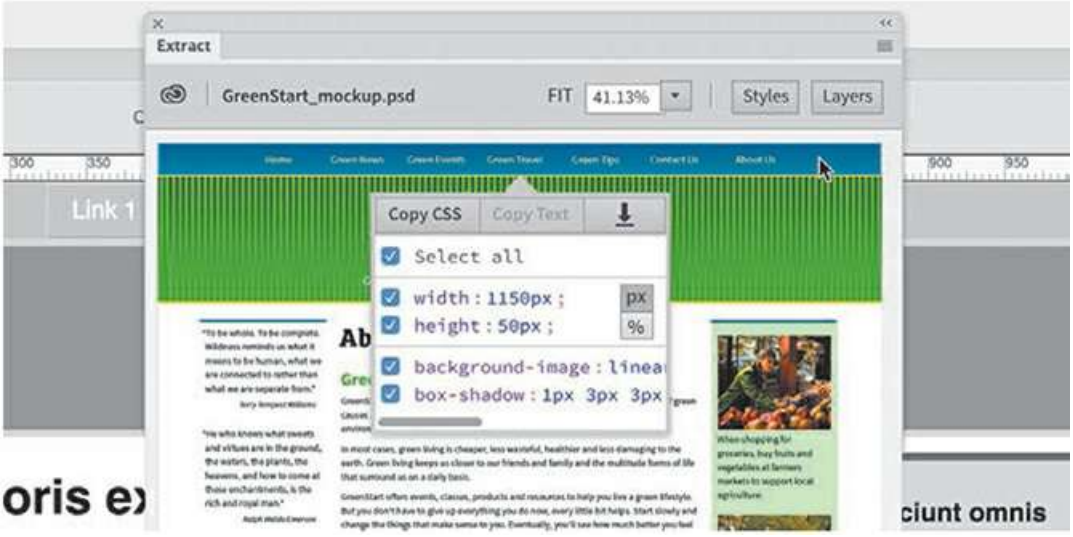


The mockup loads and fills the entire panel. Extract enables you to access and derive styling information, image assets, and even text from the mockup.

Styling elements using the Extract panel

In this exercise, we're interested in the styling data. Let's start at the top and work down the page.

. In the Extract panel, click the background of the top navigation menu.



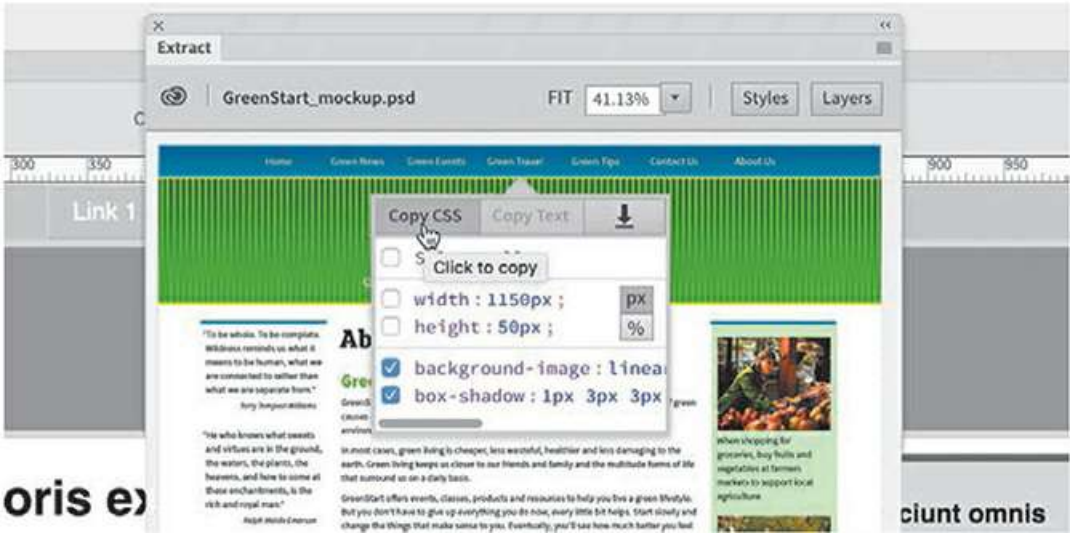
A pop-up window appears that allows you to select the data you want to obtain from the mockup. The buttons at the top of the window indicate what data is available from the selected component. The Copy CSS and Download arrow buttons are selectable, indicating that styling and image assets are available. The Copy Text option is grayed out, indicating that no text content is available to be downloaded.

The window displays the CSS styling as a list with checkboxes. When you select a checkbox, those specifications will be copied to program memory. The CSS styling that is displayed includes settings for width, height, background-image, and box-shadow. You can select all the settings or only the ones you want to use.

- If necessary, deselect **width** and **height**.

  Select **background-image** and **box-shadow**.

- Click the Copy CSS button.

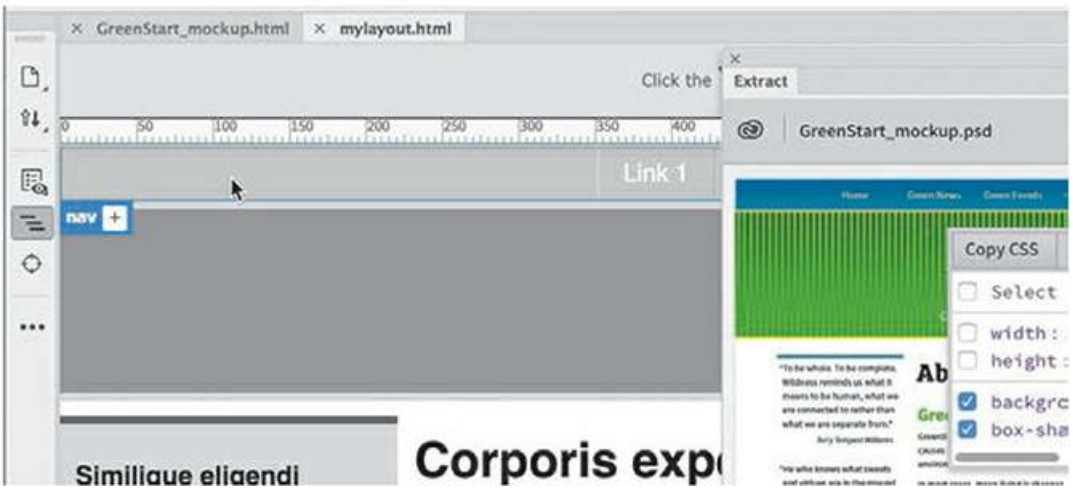


Once you've copied the settings, you can then apply them directly to the layout in Dreamweaver. The easiest way to use this data is via the CSS Designer.

- If necessary, select Window > CSS Designer to open or display the panel.

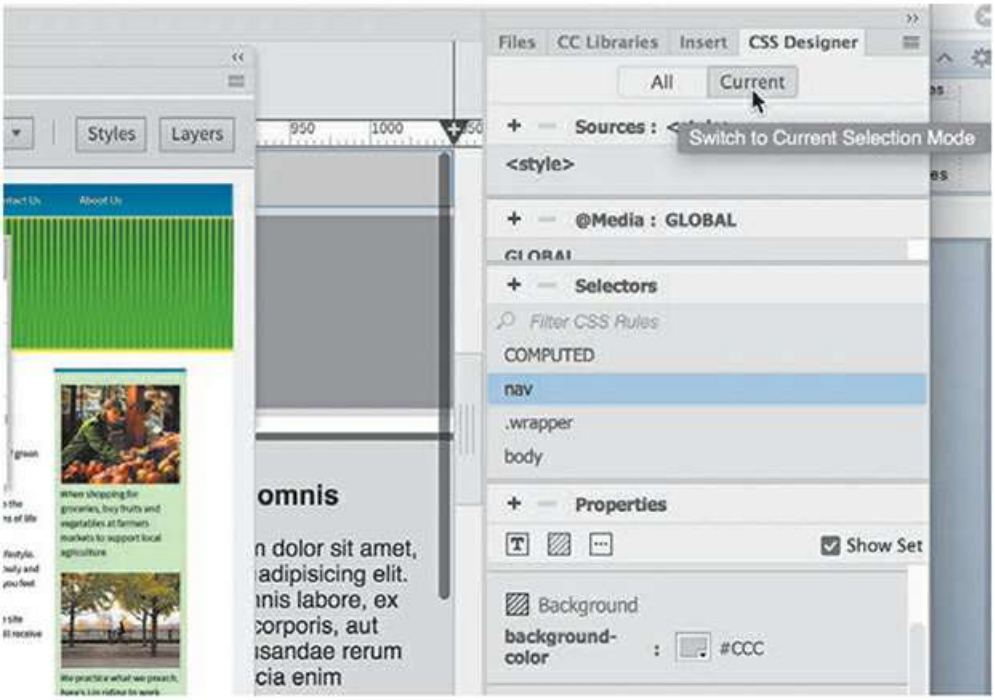
  We want to apply the specifications to the top navigation menu in the current layout. You can target the menu by selecting the appropriate rule in the Selectors pane or by selecting the actual element in Live view.

- In Live view, click to select the top navigation menu.



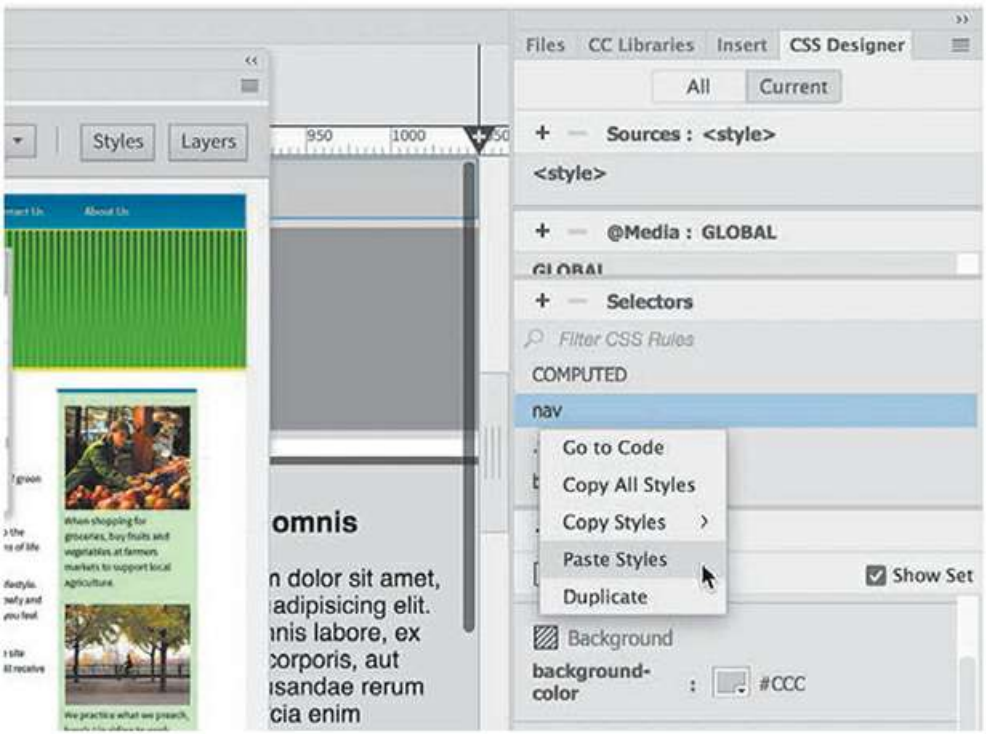
The Element Display will appear targeting the `<nav>` element.

. In the CSS Designer, click the Current button.



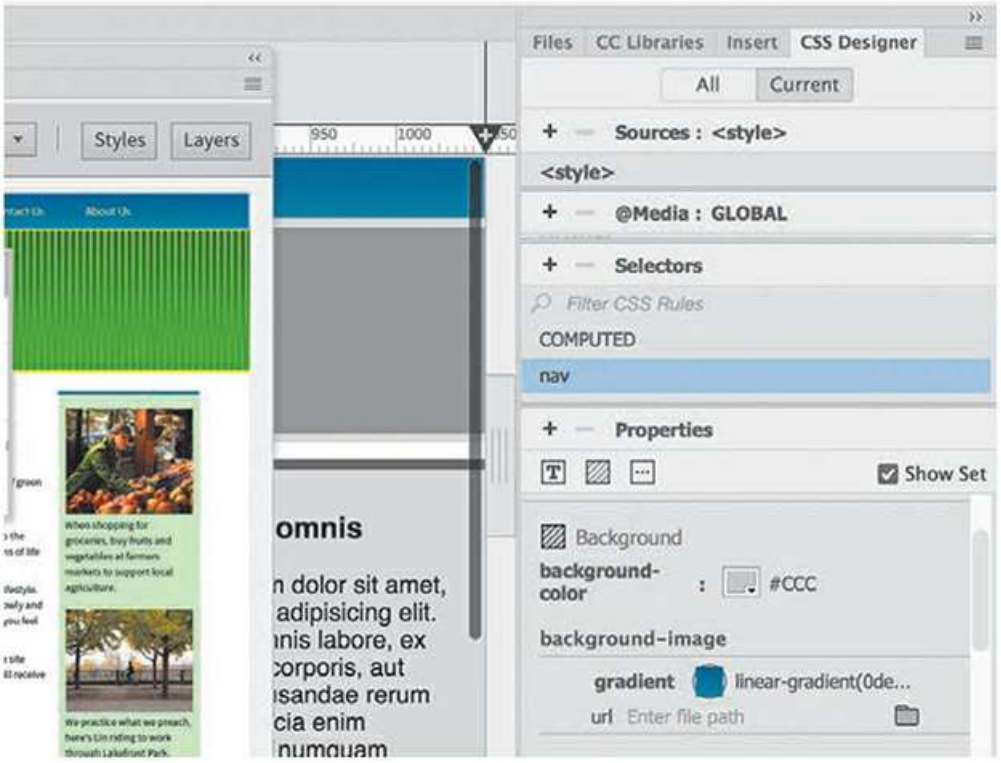
As you learned in Lesson 3, "CSS Basics," the Current button displays any styling set on the element selected in the layout. The Selectors pane in the CSS Designer displays the rules applied to the current navigation menu. The CSS rules listed are `nav`, `.wrapper`, and `body`. In this case, the styling from the mockup should be applied to the `nav` rule.

. If necessary, select the `nav` rule in the Selectors pane. Right-click the `nav` rule.



A context menu appears, providing options to interact with the rule by editing, copying, or pasting the CSS specifications. In this case, you want to *paste* the styles derived from the Extract panel.

. Select Paste Styles from the context menu.



The navigation bar in the layout reformats to match the styling shown in the mockup. You can see the new specifications in the Properties pane of the CSS Designer for the `nav` rule.

- Save **mylayout.html**.

You may have noticed that there is also text in the navigation menu. The text formatting was not brought over from the mockup, because it is separate from the background styling. The Extract panel also allows you to pick up the styling for the text.

Extracting text from a Photoshop mockup

The Extract panel enables you to pick up formatting for text as well as the text itself. In this exercise, you will pick up both from the mockup.

- If necessary, open **mylayout.html** from the lesson05 folder.
- If necessary, select Window > Extract to display the Extract panel.

  The mockup should still be displayed in the panel. If not, simply select it in the list of assets.
- Examine the top navigation menu in the mockup.

  The navigation bar has seven menu items: *Home*, *Green News*, *Green Events*, *Green Travel*, *Green Tips*, *Contact Us*, and *About Us*. The navigation bar in the current layout has only five items. Later in this exercise, you will learn how to add two items to your menu.

  The text and the styling have to be brought over separately.

- Select the first menu item: *Home*.

The pop-up window appears. Notice that all three buttons at the top of the window are active. This indicates that you can extract styling, text, and image assets from this selection.

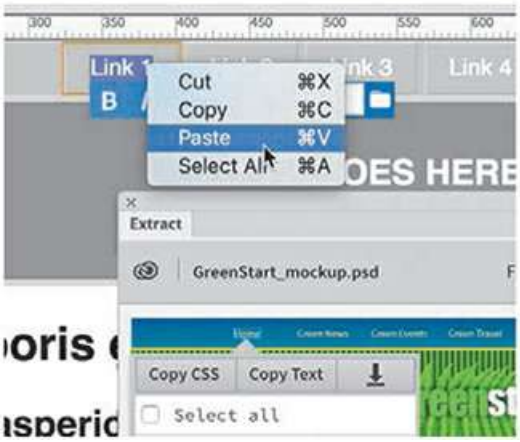
Tip

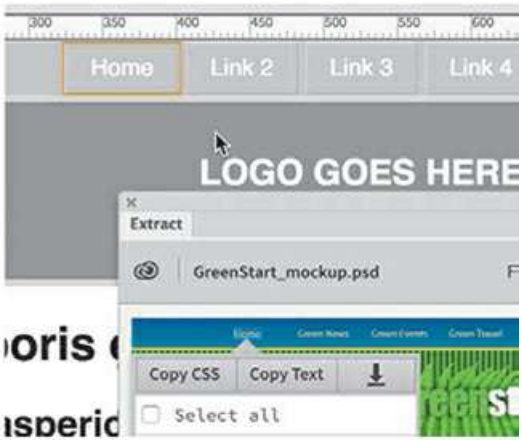
The Extract panel may obscure part of the page you need to work on. Feel free to reposition or dock the panel at any time.

- Click the Copy Text button.
- Double-click *Link 1* in **mylayout.html** in Live view.

  An orange box should appear around the text, indicating you are in text-editing mode.
- Select the text *Link 1*. Right-click the selected text.

  The context menu appears, giving you options for cutting, copying, and pasting the text.
- Click Paste.





The text *Home* replaces the text *Link 1*.

. Repeat steps 4 to 8 to replace Links 2 through 5.

You have now replaced all the text in Links 1 through 5 in **mylayout.html**. There are still two more items in the mockup that should be added to your layout. First, you should obtain the text for the next item.

Tip

The text extraction feature is really intended for longer passages of text. Feel free to type the menu items by hand if you prefer.

. Copy the text for the item *Contact Us*.

You will now learn how to add a new item to the navigation menu.

. Double-click the text *Green Tips* in **mylayout.html**.

Insert the cursor at the end of the text and press Enter/Return.

Pressing Enter/Return normally creates a new paragraph. In this case, you will be adding a new menu item.

. Choose Edit > Paste, or press Ctrl+V/Cmd+V, to paste the text copied in step 10.

The text *Contact Us* appears in the navigation bar, but it's not formatted like the other items. Sometimes styling is based on the structure of an HTML element, which may not be apparent on the surface. So let's dive into the code to see if we can ascertain the reason the new item looks different from the others.

Troubleshooting CSS styling

The new item you added to the navigation bar doesn't look like the others. In this exercise, you will examine the menu item and its code structure to see what the issue may be.

. Select the *Contact Us* menu item in **mylayout.html**.

The Element Display appears focused on the `<li>` element.

. If necessary, switch to Split view.

The document window divides between Live view and Code view. In the Code view window, you can see the text *Contact Us* as well as the other menu items. *Contact Us* should be selected in both views.

The menu is composed of an unordered list (`ul`) and six list items (`li`). Can you see the difference between *Contact Us* and the other menu items? *Contact Us* does not feature a hyperlink. Let's match the structure of the other menu items and see if that fixes the styling issue.

Note

Dreamweaver occasionally ignores your selection in Live view. If the element is not selected in Code view, you may need to repeat your selection in Live view.

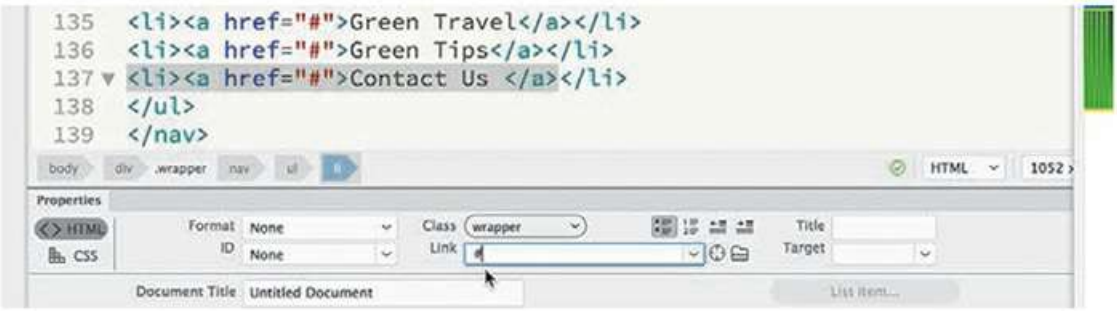
. If necessary, select Window > Properties to display the Property inspector.

Note

If this is your first time using the Property inspector, you may need to dock it at the bottom of the document window, as shown in the screen shots.

. If necessary, in the Property inspector, click the <>HTML button.

. Type **#** in the Link field in the Property inspector and press Enter/Return.



The *Contact Us* item now appears styled the same as the other links. It's clear that the styling was based on the application of a hyperlink to the text. Now let's create the final menu item.

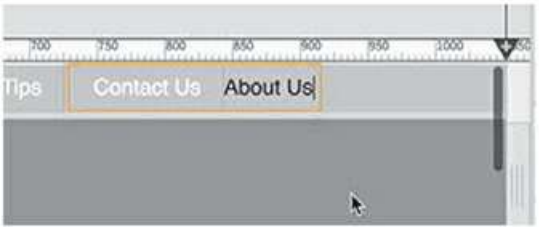
Note

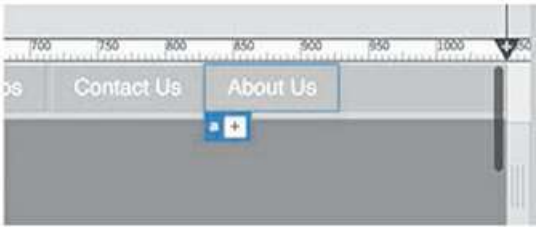
The hash symbol (#) is used as a placeholder element when you need to create a link but don't have a destination URL yet.

- Double-click the *Contact Us* menu item.

  Insert the cursor at the end of the text and press Enter/Return.

- Type `About Us`, and then select the text.
- Type # in the Link field in the Property inspector and press Enter/Return.





Dreamweaver adds a hyperlink to the menu item. The last link item is complete and styled as the others. But the styling in the layout does not match that of the mockup. You can use Extract to bring over the text styling from the mockup too.

Extracting text styling from a Photoshop mockup

You have learned how to extract styling for graphical elements. Extracting the styling for text works in the same way.

- In the Extract panel, select any of the text items in the navigation menu.

  The pop-up window appears, showing extraction options for the selection.

- If necessary, deselect width and height. Select all text styling specifications.

  Note the various settings. This will become important shortly.

- Click the Copy CSS button.

Once the CSS is copied, you have to identify the rules that format the text in the menu items. This can be tricky because text styling can be very complex. Often, several rules may affect a single piece of text. This doesn't mean you can't successfully format an item; it just means you have to be especially vigilant when applying the new styling.

Note

When selecting the text, make sure that Dreamweaver focuses on the `<a>` element. Often, it takes two or more clicks to get the focus on the correct element.

- In **mylayout.html**, select one of the text items in the navigation menu.

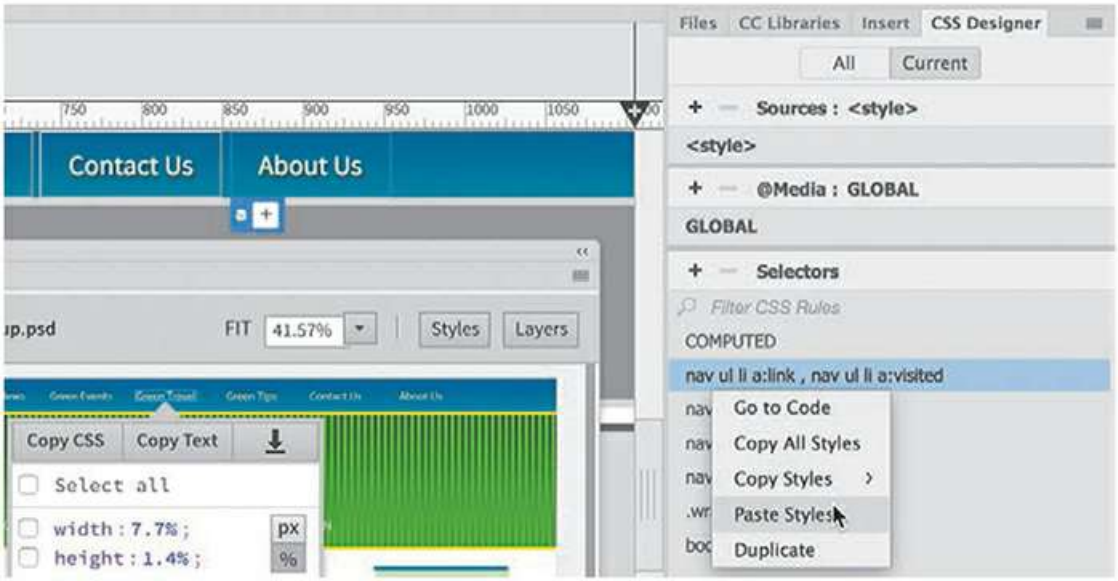
  The menu is composed of four HTML elements: `nav, ul, li,` and `a.` Text styling can be applied to any of these elements or even to all four at once. The goal is to have the styling from the mockup overwrite or override any existing settings.

- Click the Current button in the CSS Designer.

  The Selectors pane displays the CSS rules that affect the selected text. Remember that the rule at the top of the list is the most powerful. That's the one you usually want to target.

- Right-click and select Paste Styles on the rule

`nav ul li a:link, nav ul li a:visited`.



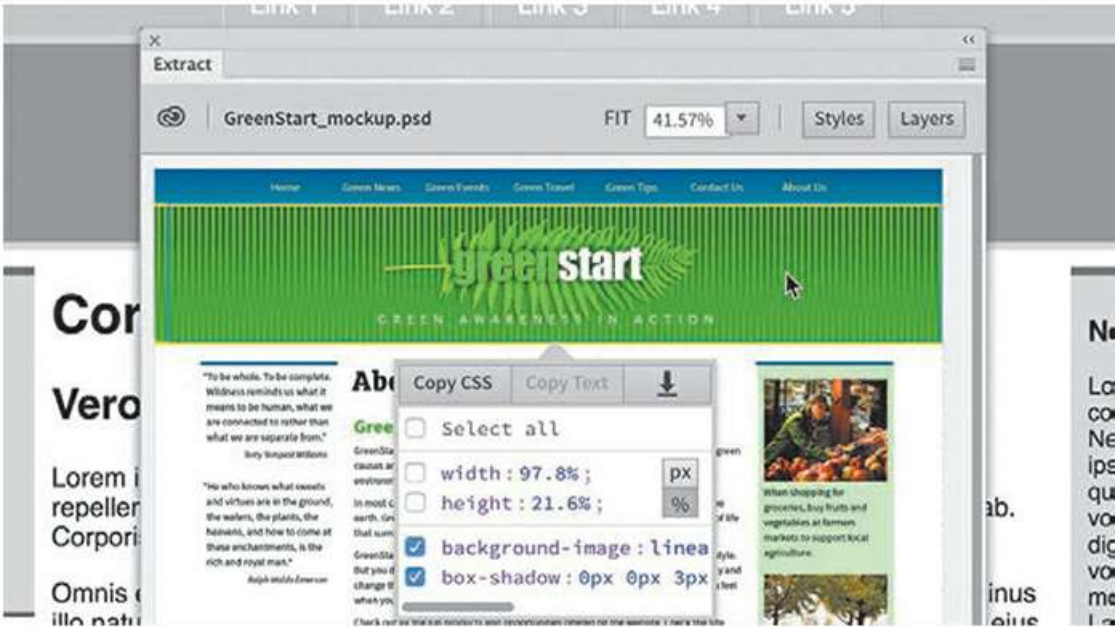
The text in the seven menu items now matches the styling in the mockup. The next element to format is the header.

Creating a gradient background using Extract

You might not be able to discern this from the tiny Extract panel preview, but the header element is composed of two separate text elements, two different images, and a color gradient. You would probably need to open the file in Photoshop to nail down exactly how the header is constructed, but the Extract panel can give you nearly everything you need to reconstruct it in this layout.

When reconstructing the header, you can build it from the bottom up or from the top down. Let's start at the bottom with the basic HTML element itself.

. Select the `header` in the Extract panel.



Be sure to target the graphical area of the `header` element and not the text. Once it is selected, the pop-up window displays the specifications for the `header`. You will see that the options for background-image and box-shadow are still selected, but not width or height. You won't need dimensional settings for most elements, but in this case, you will also want to pick up the height specification.

. Select the **height** option in the pop-up window.

If necessary, select the `px` option in the pop-up window.

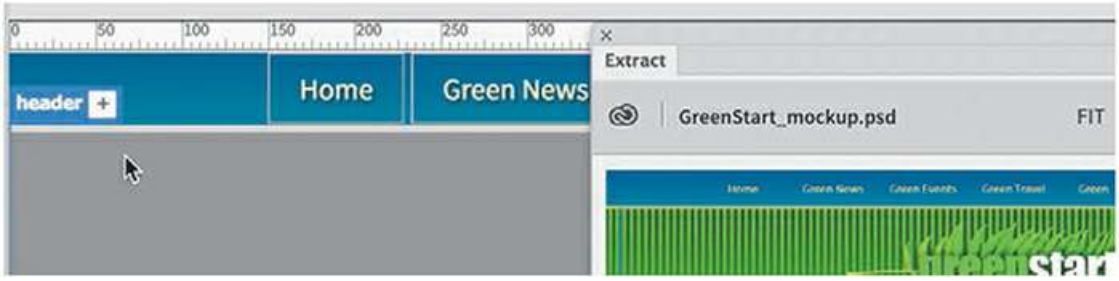
Dimensions can be specified in either pixels (`px`) or percentages (`%`). In this case, the header should be set to a fixed pixel dimension.

. Click Copy CSS.

Note

Be sure you select the `<header>` element and not the `<h2>` or `<p>`elements.

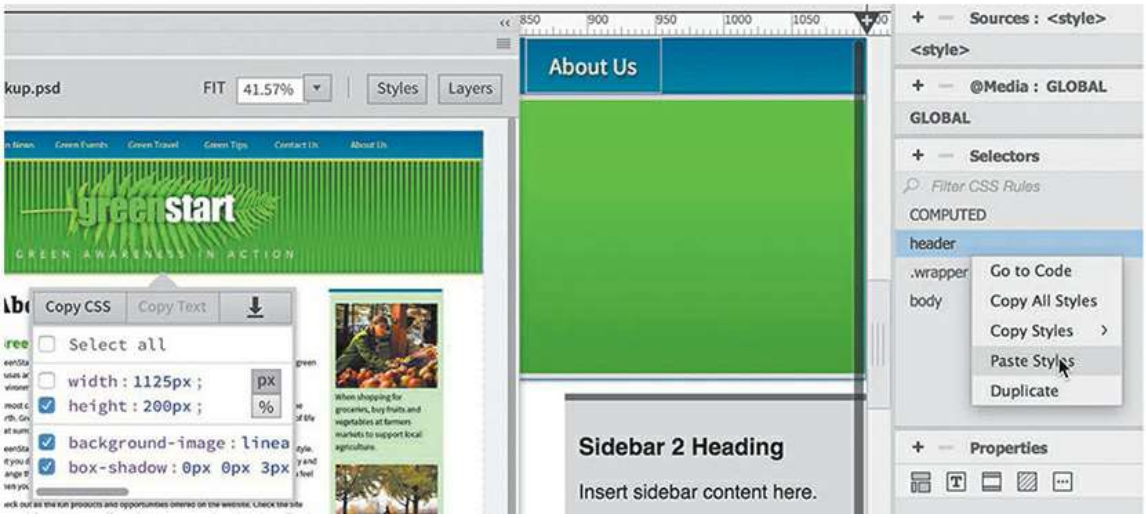
. In **mylayout.html**, select the `<header>` element.



The Element Display appears focused on the `header` element. Since Current mode is still selected, the CSS Designer automatically changes to display the rules and specifications formatting the header.

. Right-click the `header` rule in the Selectors pane.

Select Paste Styles in the context menu.



The `<header>` element now displays a gradient background that matches the color of the mockup's header and has an increased height, but it doesn't have the vertical stripes pattern. That's because the stripes are created by a separate image. Images are not automatically included when you copy the CSS. The Extract panel can also be used to download image assets. You will learn how to do that shortly, but there is one issue we need to address first concerning the gradient background.

The gradient in the mockup created by Photoshop shows dark green at the top, changing to a lighter green at the bottom. But if you look carefully at the layout, the direction of the gradient is reversed. Let's correct that error before we continue with the rest of the header.

Gradients

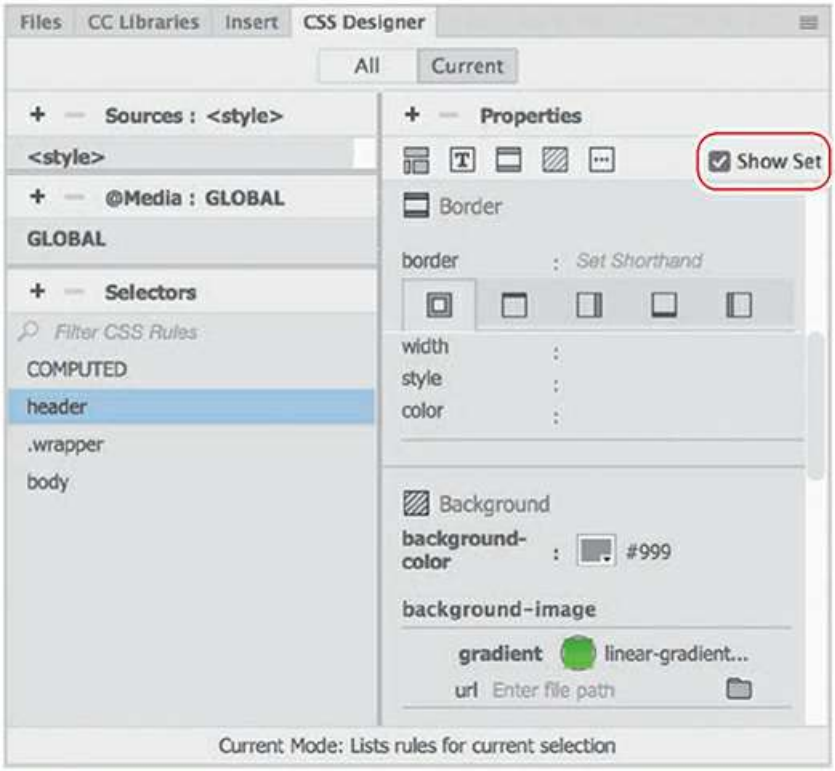
CSS is a living thing. It is changing all the time. New properties and values are being created; old ones are being changed or deprecated. The specification on gradients is still evolving. Different browser vendors are still developing their support, and as a result you need to insert vendor-specific settings in your style sheet along with the industry-standard ones when you add gradients. The good news is that Dreamweaver will do it for you automatically. This is one of the benefits of using Dreamweaver and the CSS Designer. Whenever you add a gradient to an HTML element, you will notice that Dreamweaver displays a message at the top of the document window.

The warning basically tells you that Dreamweaver has added vendor-specific syntaxes to your style sheet to cover browsers that still require them. You can recognize this code because it features a vendor prefix, such as –moz for Firefox or –webkit for Chrome and Safari. The process is automatic and will be updated by Adobe as needed when the specifications change and evolve further.

If you see the warning message at the top of the document, you can dismiss it by clicking the Close icon at the right side of the message. Once this special syntax is no longer needed, you can delete the code or leave it in place. The code is written in such a way that it will not affect your page display if the browser doesn't need it.

. In CSS Designer, choose Show Set, if necessary.

Click the rule `header` in the Selectors pane.



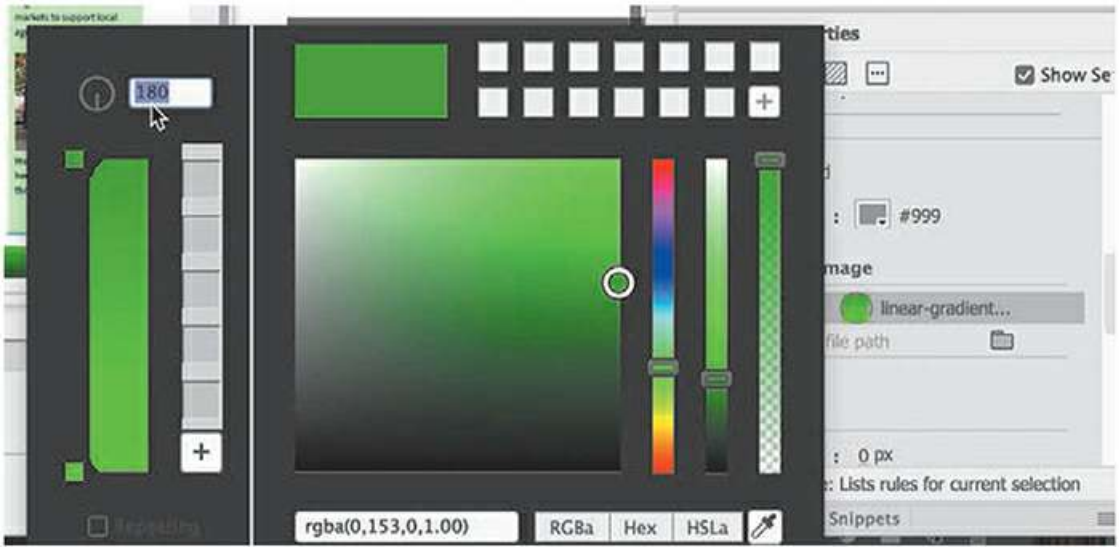
The Properties pane displays the CSS specifications set in the rule.

. In the Properties pane, locate and click the Gradient color picker.

On the left side of the Gradient color picker there are two color stops where you can choose the beginning and ending colors or add in-between colors. Above the color stops is the field that specifies the angle of the gradient. The current gradient is set at zero (`0`) degrees.

. Enter **`180`** in the Linear Gradient Angle field.

Press Enter/Return.



The gradient background in the header reverses direction.

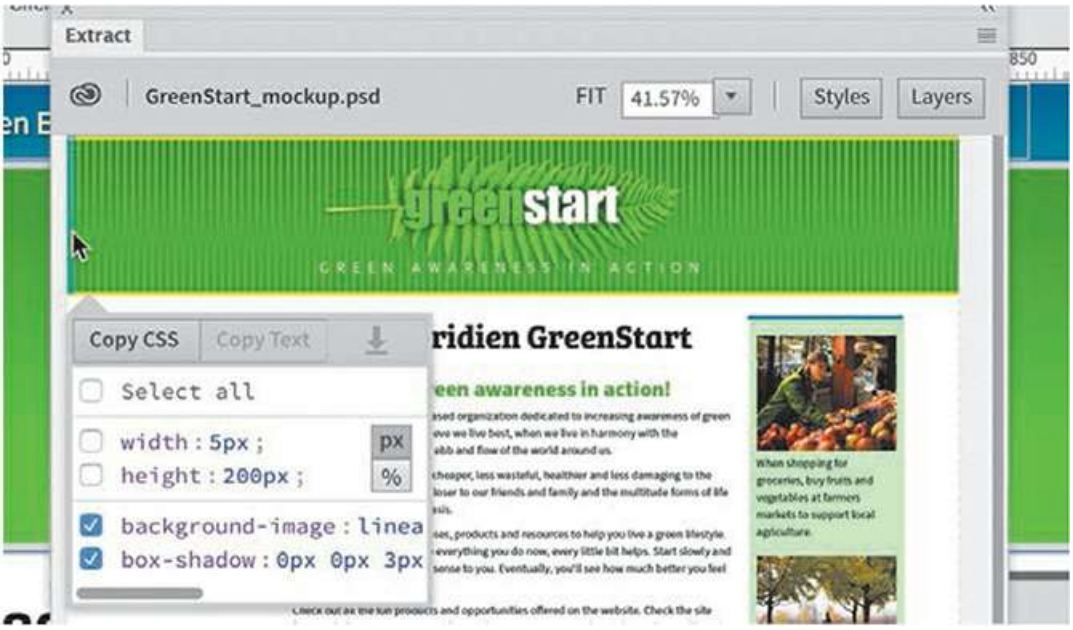
. Save **mylayout.html**.

Next, let's set up the striped background pattern. The mockup creates the stripe pattern by overlaying a narrow image containing a rectangle with a drop shadow and repeating it across the header. In Photoshop, one of those rectangles is set off in a separate layer. You are going to take the contents of that layer and create the stripes for the background.

Extracting image assets from a mockup

The background of the header in the mockup is composed of a repeated Photoshop vector shape. The shape has a gradient color fill and a shadow effect. But Dreamweaver doesn't support these vector shapes, so you first have to convert this Photoshop object into a web-compatible image.

. In the Extract panel, select the first stripe on the left side of the `header`.

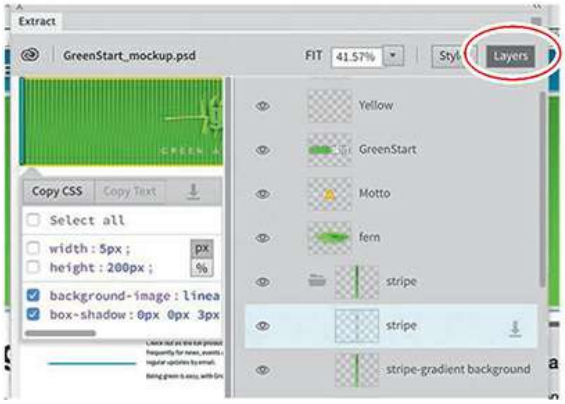


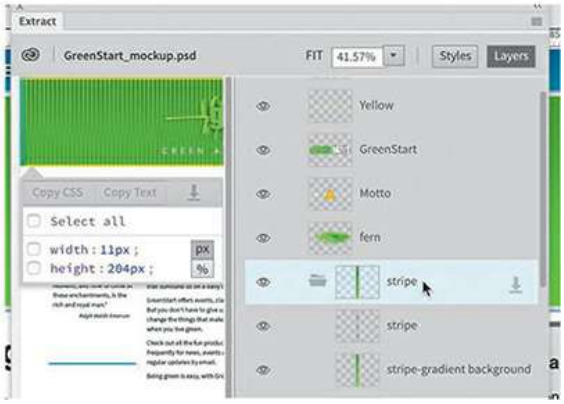
The stripe is composed of two objects. You have to make sure the entire layer is exported and not just the object selected.

. Click the Layers button at the top of the Extract panel.

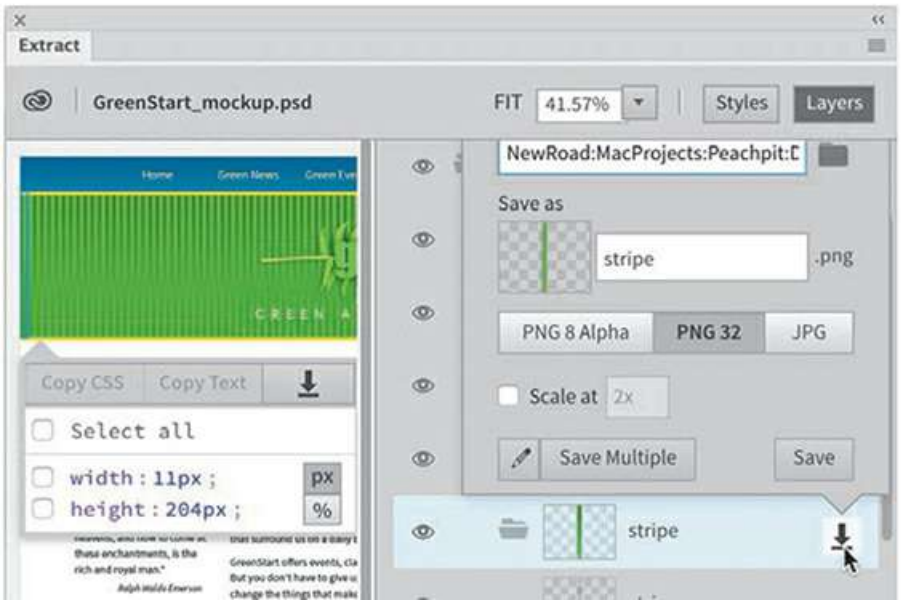
Extract can read and display the contents of the layers in the Photoshop file. Note the selected layer in the panel. To get the entire stripe image, you have to select the layer with the folder icon.

. If necessary, select the Stripe folder in the Extract layer display.





. Click the Extract Asset icon on the selected layer.



A pop-up window appears, allowing you to target the folder to which you want to download the image. It also has options for specifying the image type, the size, and other specifications for responsive design. For this image, we need only the basics.

Note

If you set up the default images folder in the advanced settings of the Site Definition dialog, the site image folder will already be targeted.

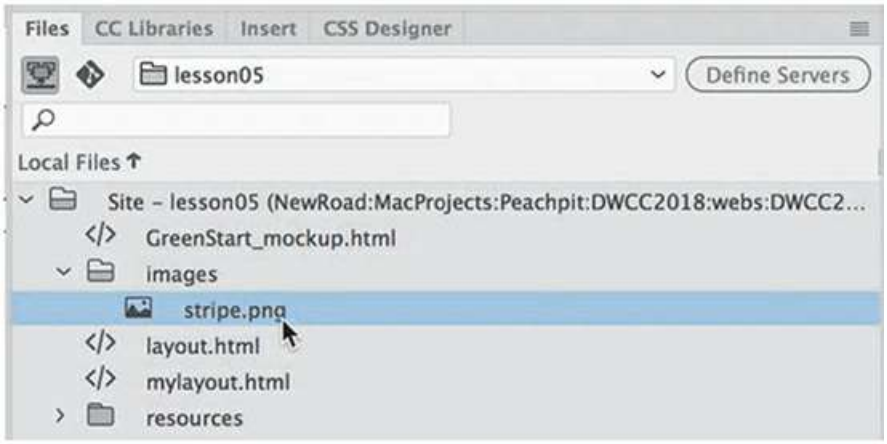
. If necessary, select PNG 32 as the image type. Click the Browse icon and ensure that the default site images folder is selected as the destination.

. Click the Save button.

The image is exported to the default site images folder or the folder you designated in the pop-up window.

. Select Window > Files to bring the Files panel to the top.

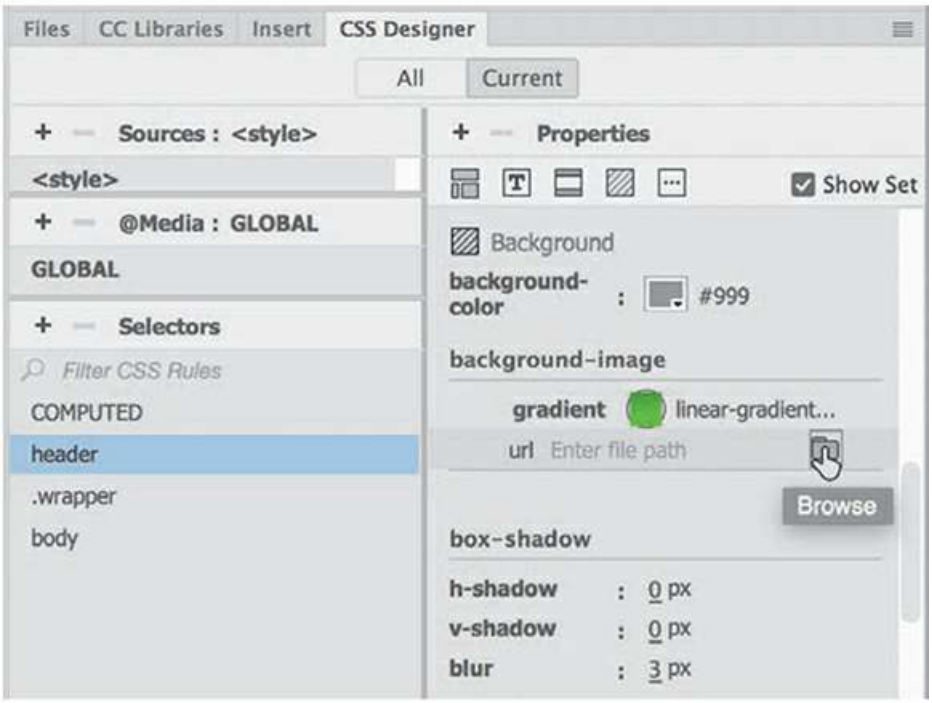
If necessary, reveal the contents of the images folder.



If you followed the steps properly, **stripe.png** appears in the images folder. Unlike with the other CSS settings, adding an image to the background of the `header` will have to be done manually.

. In the CSS Designer, select the `header` rule.

Examine the background properties applied to the rule.



In the Properties pane, you can see the gradient setting in the background-image section. Directly below it is a URL field that is intended for a

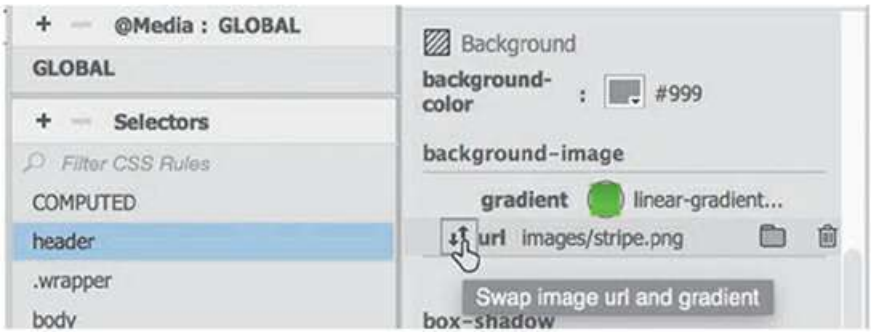
background image.

. Click the Browse icon. Navigate to the site images folder and select **stripe.png**.

Click Open.

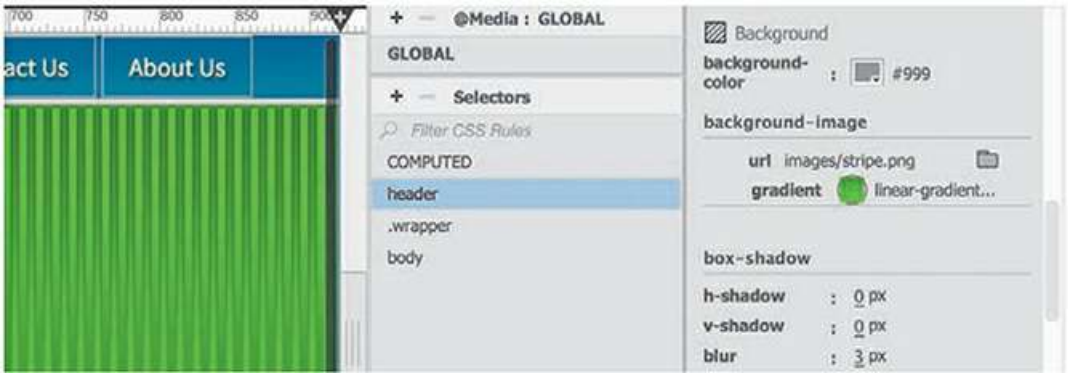
The image **stripe.png** appears in the URL field, but no stripes are visible in the header. That's because various CSS settings can interfere with each other. In this case, there are two background-image properties set for the header. The gradient setting is above the URL setting. That means the gradient is above the image, and since the color is opaque, you can't see the stripes. You will have to reverse the order of the specifications so that the stripes can be seen.

. Position the cursor over the left edge of the URL label.



The Swap icon appears under the cursor.

. Click the Swap icon.

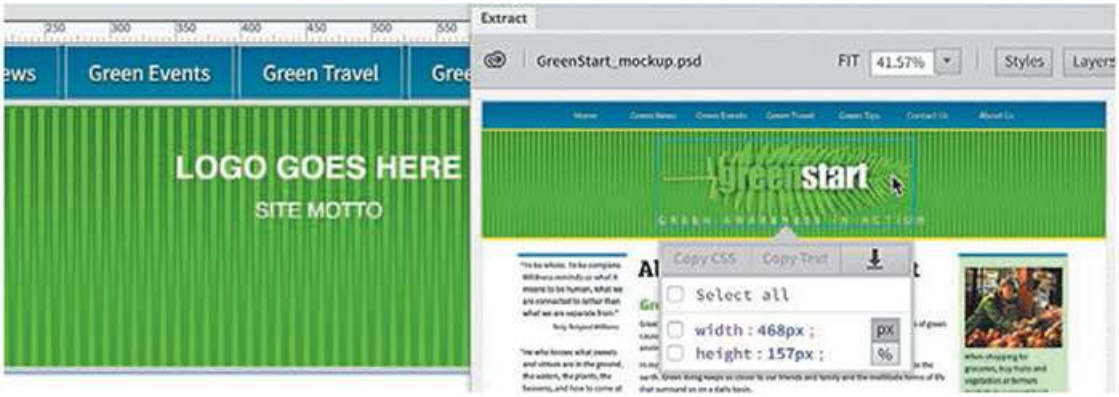


The two settings swap positions in the Properties pane. A line of stripes appears filling the header. But how is that possible? The image contained only a single stripe a few pixels wide. That's because the default setting for background images is to automatically repeat both vertically and horizontally. Background images are intended to fill the entire element like wallpaper, which is exactly the desired effect.

The header now has two background effects, but you're not finished yet. You still need to add

the fern image to the background.

- In the Extract panel, click the Layers button to close the layers display.
- Select the fern image in the mockup.



As with the stripe image, you can create a local copy of the fern for the website.

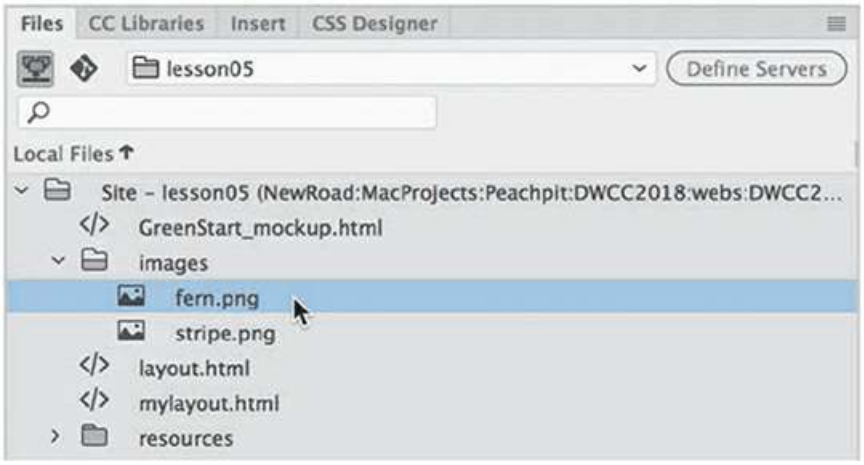
- Click the Extract Asset icon.

The pop-up window opens with image export options. Since the fern image has to float above the stripes, you will want to see the stripes behind the image. That means the image will have to be saved to support transparency. The two image formats that support transparency are the GIF and PNG file types. However, Extract supports only the PNG and JPEG formats, so the decision is made for you. You will learn more about web-compatible image formats in Lesson 8, "Working with Images."

- If necessary, select PNG 32 and confirm that the image will be saved to the site images folder.

. Click Save in the pop-up window.

. Select Window > Files. Examine the images folder.



The image **fern.png** appears in the images folder. The fern will be added to the background of the `header` element, but the CSS Designer in Dreamweaver supports only two background image settings. To add a third effect, you will have to do it manually.

Adding CSS background effects in code

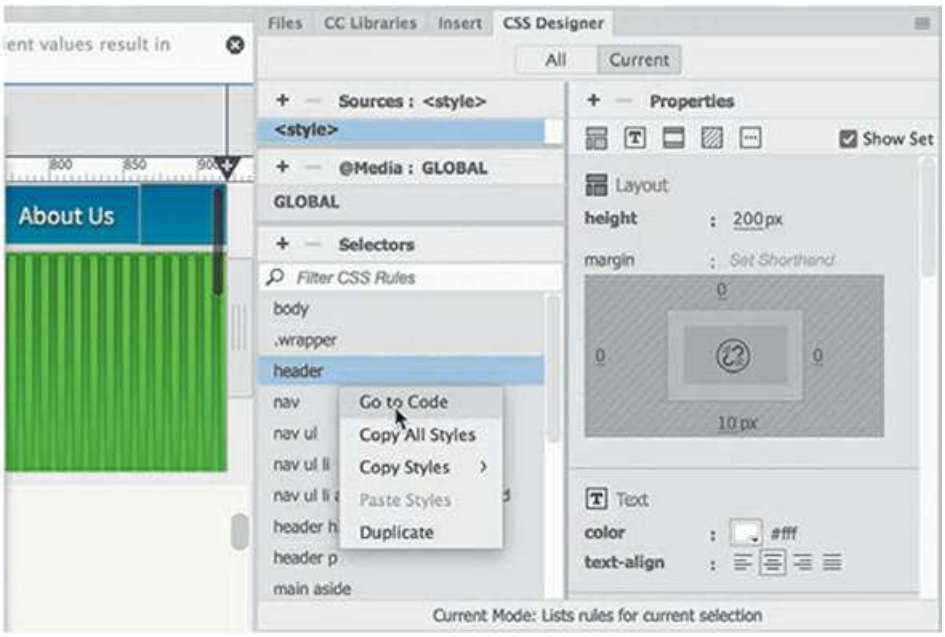
The `header` element in **mylayout.html** already has two background-image effects. The mockup calls for a third image to be added to the design. Normally, you could insert the fern image into the element itself. But the design requires text to be positioned over the fern too. By default, text and images cannot occupy the same position.

One option would be to reproduce the effect by using a single large image that combines all the elements at once. But as you learned earlier, you normally want to avoid using large images whenever possible. Also, by keeping the text in the final layout you can improve your search engine rankings.

In this exercise, you will learn how to add a CSS setting that isn't visible in Dreamweaver's interface. The CSS specifications describe numerous effects and capabilities, many more than you will find in the CSS Designer. Although you may not see every possible CSS specification in the panel, you can always enter these settings by hand.

. In CSS Designer, right-click the `header` rule.

  Choose **Go to Code** from the context menu.



In the document window, Code view displays the `<style>` section of **mylayout.html**. Dreamweaver will typically focus on the bottom of the selected rule.

. Examine the `header` rule specifications.

```
23      margin-bottom: 10px;
24      background-image: url(images/stripe.png), -webkit-linear-
        gradient(270deg,rgba(0,153,0,1.00) 0%,rgba(0,204,0,1.00) 100%);
25      background-image: url(images/stripe.png), -moz-linear-
        gradient(270deg,rgba(0,153,0,1.00) 0%,rgba(0,204,0,1.00) 100%);
26      background-image: url(images/stripe.png), linear-
        gradient(180deg,rgba(0,153,0,1.00) 0%,rgba(0,204,0,1.00) 100%);
27      -webkit-box-shadow: 0px 0px 3px 0px rgba(0, 0, 0, 0.55);
28      box-shadow: 0px 0px 3px 0px rgba(0, 0, 0, 0.55);
```

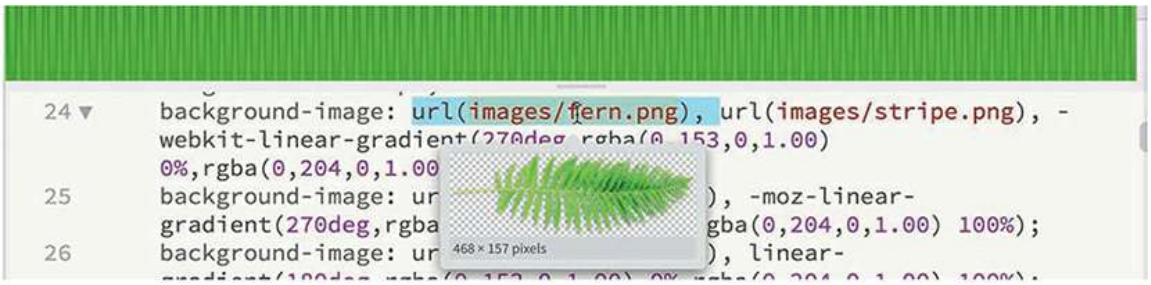
HTML 920 x 194 INS 24:23

The rule pictured above contains three background-image declarations: one standard setting and two with vendor prefixes. The background-image specification is still under development, so certain browsers may require their own declarations to honor the settings. Chrome, Safari, and Android use `-webkit`. Firefox uses `-moz`. Opera uses `-o`. To add the fern image properly, you have to add the image setting to any background-image declarations added by Dreamweaver.

Note

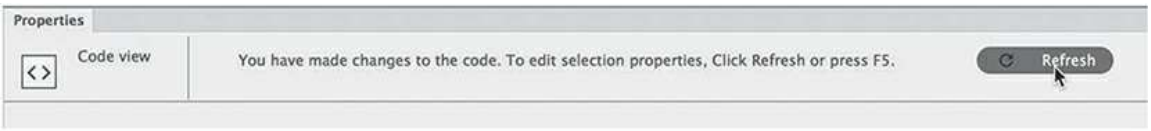
There may be a difference in your markup depending on whether you are in Windows or macOS and what build of Dreamweaver you are using.

- Insert the cursor between the properties `background-image:` and `url(images/stripe.png).`
- Type **`url(images/fern.png) ,`**



Don't miss the comma (`,`) at the end of the setting. This is essential to keep the three background images separated.

- Copy and paste the URL setting into the `-webkit`, `-moz`, and `-o` background-image linear-gradient properties as necessary.
- Click the Refresh button in the Property inspector.



Once the URL for the fern image is added to all three properties, you should see the fern image appear in the background of the header.

Like the stripe image, the fern background image repeats across the `header` element. But in the mockup, the fern appears only once in the center of the element. Although the repeating

behavior is the default setting, you can modify the behavior with another CSS setting.

- In the CSS Designer, examine the properties for the `header` rule.

  Although you added the fern and stripe images to the properties and you can see both images on the screen, the panel shows only **fern.png** and the gradient. The stripe image does not appear anywhere in the panel. That means any changes you need to make for this property will have to be done manually.

- In Code view, insert the cursor at the end of the last property in the `header` rule but before the closing brace (`}`).
- Press Enter/Return to create a new line.

  Type **`background-repeat: no-repeat;`**

- If necessary, click Refresh in the Property inspector.

The fern image should now appear only once in the `header` element, but so does the stripe image. Since you have three background effects and only one background-repeat command, it's applied to all the background images. To restore the stripe pattern, you'll have to add a second value to the background-repeat declaration.

- Insert the cursor before the semicolon in the `background-repeat` declaration.
- Add the following highlighted code to the declaration:

```
background-repeat: no-repeat , repeat-x;
```

This setting tells the browser to display the fern once but repeat the stripe horizontally. Next, you need to center the fern image as well as resize it.

Note

You'll notice you added only two settings to the `background-image` property, although there are three effects. In this case, the settings apply to the first two images. An item without its own settings takes the last setting declared.

- Insert the cursor after the semicolon in the `background-repeat` declaration and press Enter/Return to create a new line.
- Add the following properties to the `header` rule:

Click here to view code image

```
background-position: 45% center ,0 0;
background-size: auto 75%, auto auto;
```

These settings will center the fern and resize it but keep the stripes and gradient displaying properly. Be careful not to miss any punctuation or you may invalidate the rule or even the entire style sheet.

- Save **mylayout.html**.

Finishing up the layout

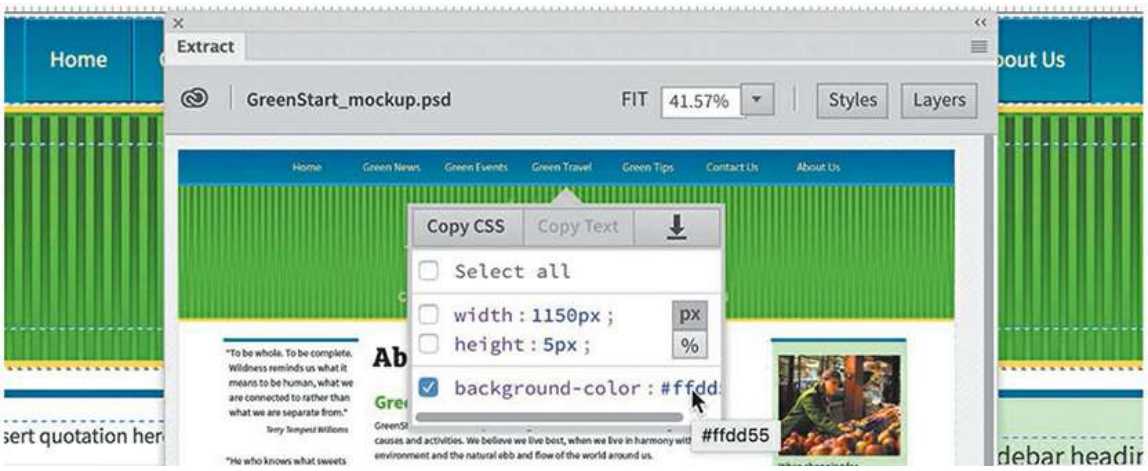
There are still several elements that need to be styled in this layout. The rest of the work should go pretty quickly. First, you'll finish the styling of the `header` element.

- If necessary, open **mylayout.html**.

  Select Window > Extract.

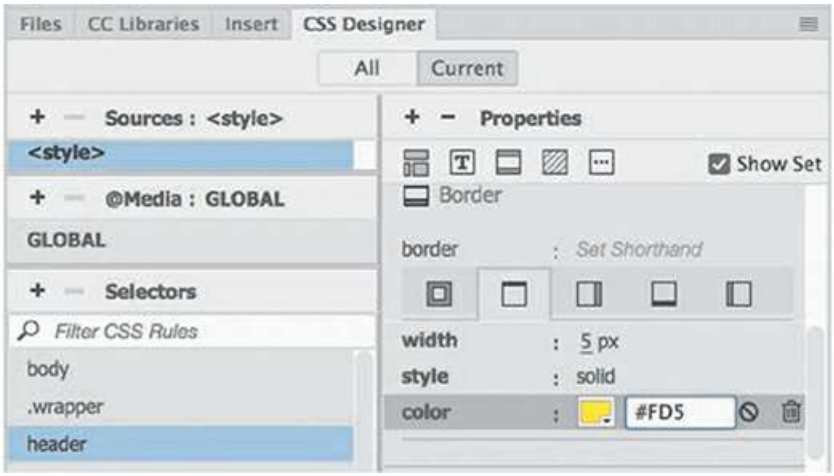
  You may have noticed that the header in the mockup has yellow borders at the top and bottom. Let's pick up the color and add that to the layout.

- In the Extract panel, select the top border of the `header`. Examine the color applied to the border.



The border in the mockup is actually a Photoshop vector rectangle. So there's actually no border style to copy and apply to the CSS. Instead, you'll identify the color used and then enter it manually into the appropriate setting. Once it's selected in the Extract panel, it's easy to see the hex color applied to the border element. The color is `#ffdd55`, which can be abbreviated to `#FD5` or `#fd5`. Various specifications can be written in either upper- or lowercase interchangeably without affecting how they function. You may notice that Dreamweaver may rewrite some specifications automatically.

- In the CSS Designer, select the `header` rule. In the `border-top` property, change the color to **`#FD5`**. Change the `border-bottom` color to **`#FD5`**

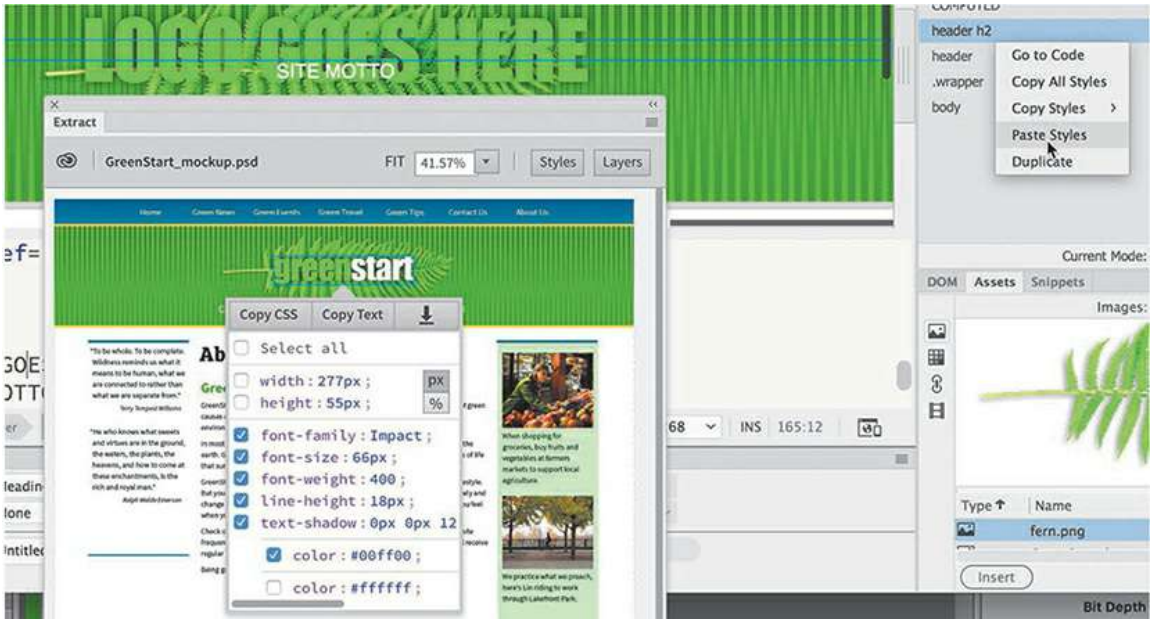


There are two text elements in the header. The top one features the name of the association; the bottom one, its motto. By now, you should probably already know how to pick up not only the styling but also the content of each element.

- Select the text *greenstart* in the Extract panel.

  Click Copy CSS.

- Select the text *LOGO GOES HERE* in **mylayout.html**. In the CSS Designer, paste the styles on the rule `header h2`.



Now, let’s bring over the text.

- In the Extract panel, click Copy Text.
- Double-click the text *LOGO GOES HERE* in **mylayout.html**.

  Select the placeholder text.

- Select Edit > Paste or press Ctrl+V/Cmd+V.

The text *greenstart* replaces the placeholder text. In the mockup, the word *start* is filled with white. Since *greenstart* is a single word, to apply a color to a portion of it requires you to add a `<span>` tag wrapping the text. As you learned in Lesson 2, "HTML Basics," the `span` tag is used to apply styling or other effects to a portion of text. There are several ways to add a `span` tag or other markup. Throughout the book you will be shown various methods to create the needed code or structures. Feel free to choose your favorite, or the most convenient one, and use it whenever needed.

- In Live view, double-click to edit the text *greenstart*. Select the letters *start*.

In Live view, many of the ways to add or create or apply classes are inaccessible. To create or apply a class to the selected text, you can use the Quick Tag Editor.

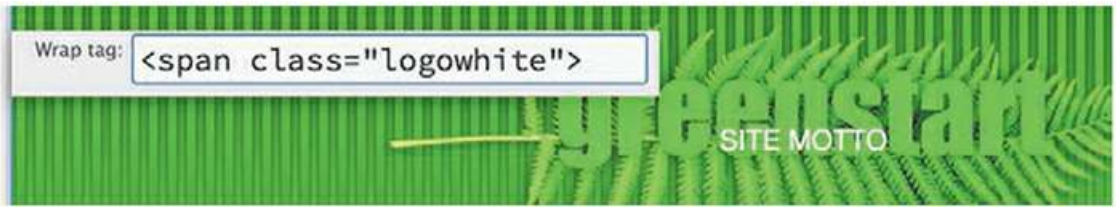
- Press Ctrl+T/Cmd+T.

The Quick Tag Editor appears in Wrap mode. The window says *Wrap Tag*, which is a bit misleading. There is no tag selected, but the result will be correct.

Tip

If the Quick Tag Editor appears in a different mode, press Ctrl+T/Cmd+T to switch to Wrap mode.

- Type: **`span class="logowhite"`** and press Enter/Return.



The selected text is now wrapped by a `span` tag and has a class of `logowhite`. You can now style this text by creating a rule in the CSS Designer.

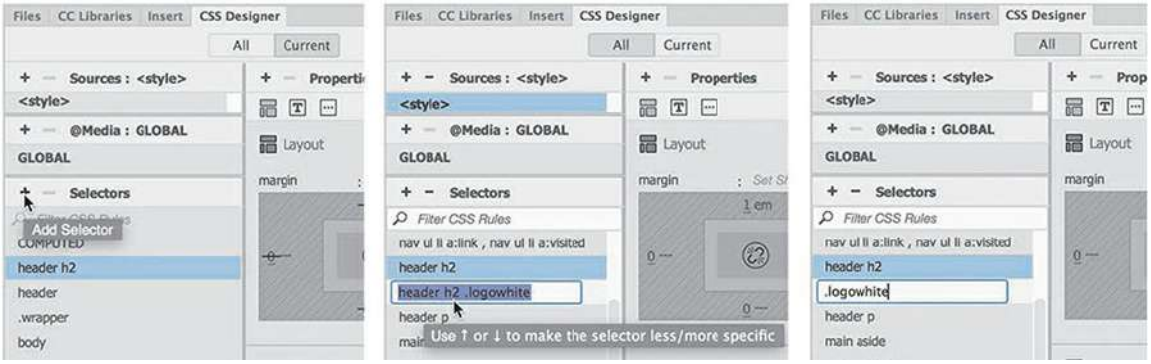
. In CSS Designer, select the rule `header h2`, if necessary. Click the Add Selector icon.

Dreamweaver creates a specific selector to target the selected element. The default selector is a bit too specific. The class `.logowhite` may be used in various places around the site, so a less specific selector would be appropriate.

Tip

If the Up and Down arrows do not modify the selector, you can edit the name manually.

. Press the up arrow key on your keyboard to simplify the selector to show only the class `.logowhite`.



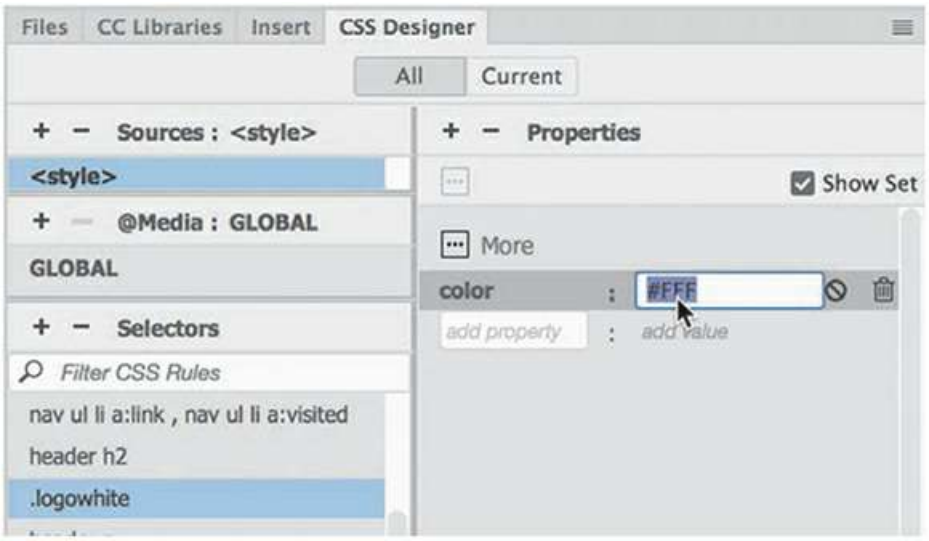
The Selector now targets the class `.logowhite` anywhere it may appear in the code.

. Press Enter/Return to create the new rule.

Tip

To learn how to add CSS properties using the CSS Designer, see Lesson 3, "CSS Basics Bonus."

. Add the following property to the `.logowhite` rule: **`color: #FFF`**



The letters *start* now display in white.

The logo is nearly complete, but it needs a little tweaking. The company name and motto are overlapping. Often, issues like this are caused by settings brought over from Photoshop that aren't ideal for HTML. For example, line spacing and font sizes will frequently need a little attention.

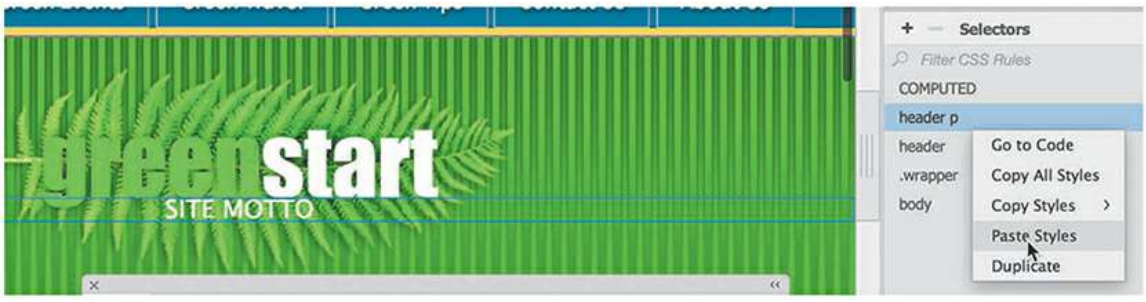
- Update the following property in the rule `header h2`: `line-height: 0.8em`

Let's finish the header by styling the motto text.

- In the Extract panel, select the motto text. Click Copy CSS.
- In **mylayout.html**, click the text *SITE MOTTO* in Live view.

The Element Display appears focused on the `<p>` tag.

- In the CSS Designer, paste the CSS on the rule `header p`.



Next, let's move the text itself over.

- In the Extract panel, click Copy Text.
- In **mylayout.html**, double-click the text *SITE MOTTO*. Select the text and press Ctrl+V/Cmd+V.

The text *GREEN AWARENESS IN ACTION* appears in the `header`, but the text styling doesn't match the mockup exactly. That's because some formatting in programs like Photoshop is handled differently than in HTML and CSS.

. Update the following property in the rule `header p`:

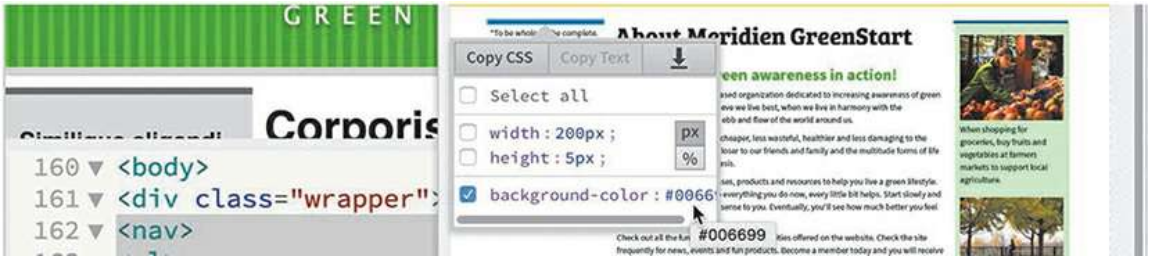
`line-height:`**`5em`**

. Add the following property to `header p`:

`letter-spacing: .5em`

The `header` is now complete. Let's move on to the sidebars. The color of the borders in both sidebars is the same.

. In the Extract panel, select the top border of the left sidebar and identify the color of the rule.



. In rule `main aside`, change the color of `border-top` and `border-bottom` to **`#069`**

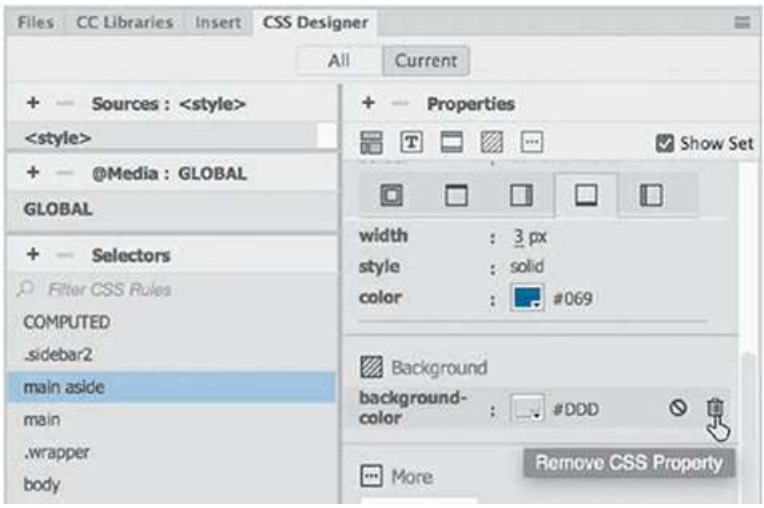
In the mockup, there are no headings in the left or right sidebars.

. In **mylayout.html**, select and delete the headings in both sidebars.

The left sidebar has no background color, but in the layout, both sidebars have the same

background. If you examine the CSS, you will notice that one rule applies the background color to both at once. The first step would be to remove the common background color.

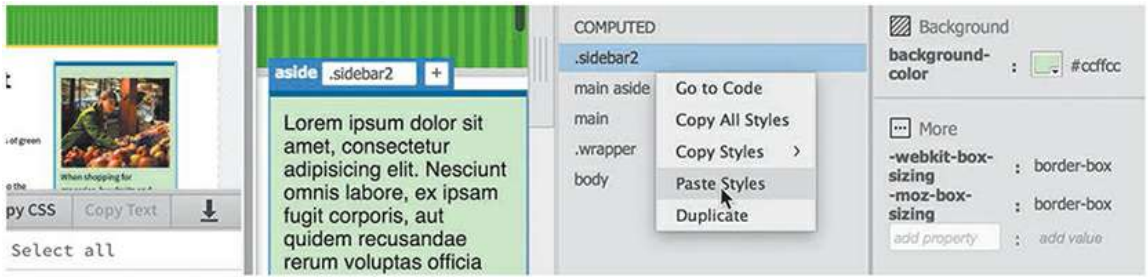
- In the rule `main aside`, delete the background-color property.



- In the Extract panel, select the background of the right sidebar. Click Copy CSS.

  Be sure you get the color of the background, not the text.

- Paste the styles on the rule `.sidebar2`.



The right sidebar background color matches the mockup. Let's grab the styling for the rest of the elements, but you can leave the text. You won't need the text, because in the next lesson you're going to add your own placeholder text.

- Copy the CSS for the left sidebar text.

  Paste the styles on the rule `.sidebar1 p`.

- Copy the CSS for the right sidebar text.

  Paste the styles on the rule `.sidebar2 p`.

- Copy the CSS for the main heading.

  Paste the styles on the rule `main section h1`.

- Copy the CSS for the secondary heading.

  Paste the styles on the rule `main section article h2`.

- Copy the CSS for the text in the main content.

  Paste the styles on the rule `main section article p`.

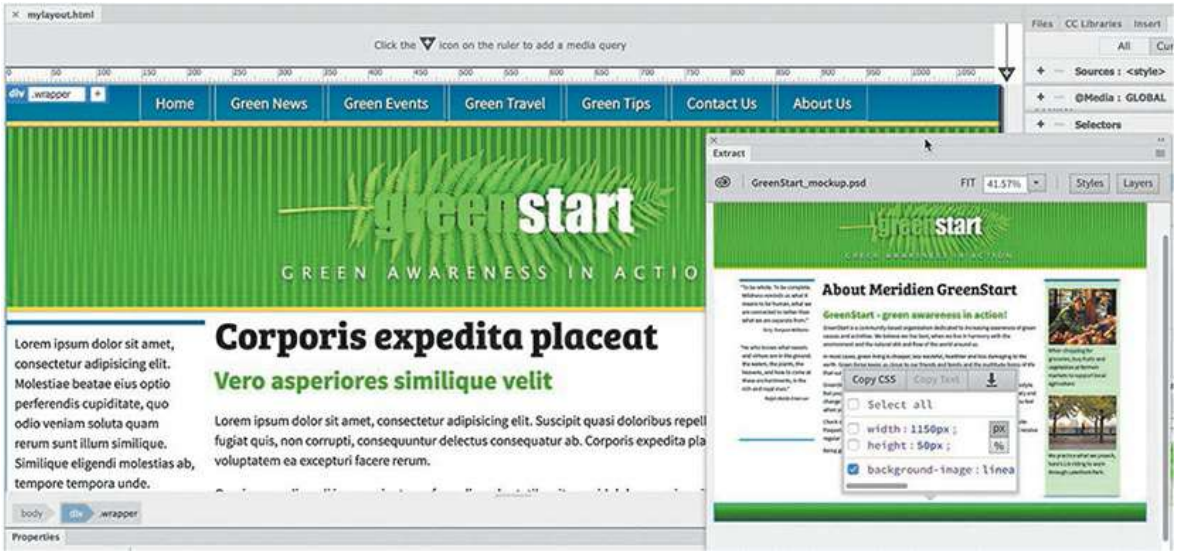
- Copy the CSS for the `footer` element.

  Paste the styles on the rule `footer`.

  In the mockup, there's no text to extract styling from. So you can go ahead and match the text color you used in the navigation menu.

- Add the following property to the rule `footer p`: **`color: #FFC`**

- Save **mylayout.html**.



Congratulations! You have learned how to extract styles from a Photoshop mockup and move them into this page. But the design is not finished. As you work through the upcoming lessons, you will continue to tweak and format the content and learn a variety of HTML and CSS tricks. In the next lesson, you will turn this basic HTML layout into your Dreamweaver site template.

Review questions

1. Does Dreamweaver provide any design assistance for beginners?
2. What advantages do you get from using a responsive starter layout?
3. What does the Extract panel enable you to do?
4. Does Extract enable you to download GIF image assets?
5. True or False. All the CSS properties generated by the Extract panel are accurate and are all you need to style a webpage and its content.
6. How many background images does Dreamweaver support?

Review answers

1. Dreamweaver CC (2019 release) provides three basic layouts, six Bootstrap templates, four Email templates and three responsive Starter layouts.
2. Responsive starter layouts help you jumpstart the design of a site or layout by providing a finished layout complete with predefined CSS and placeholder content.
3. The Extract panel enables you to derive CSS styling, text content, and even image assets from page mockups created in Adobe Photoshop and Adobe Illustrator.

4. No. Extract supports only the PNG and JPEG image formats.
5. False. Although many of the CSS properties are perfectly usable, styling in Photoshop and Illustrator is geared for print output and may not be entirely suitable for web applications.
6. Dreamweaver can support only two background images in the CSS Designer, but you can add as many as you desire by hand in Code view.

6 Working with Templates

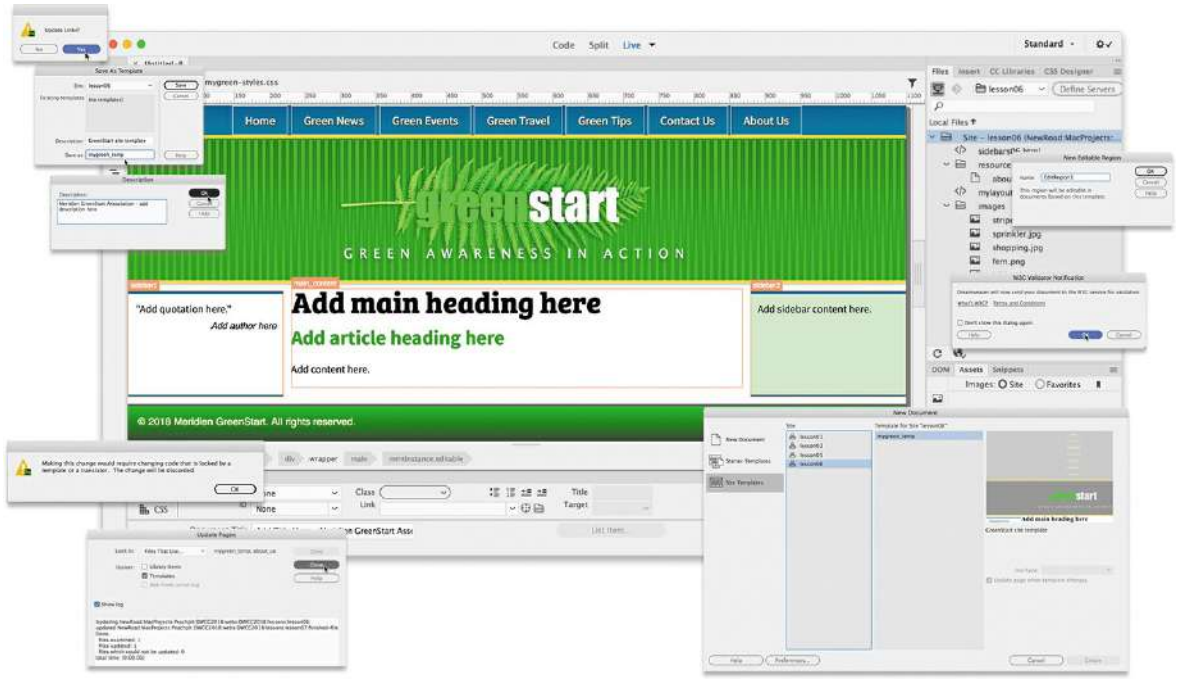
Lesson overview

In this lesson, you'll learn how to work faster, make updating easier, and be more productive. You'll learn how to do the following:

- Create a Dreamweaver template
- Insert editable regions
- Produce child pages
- Update templates and child pages
- Move embedded CSS styles to an external style sheet

This lesson will take about 2 hours and 15 minutes to complete. If you have not already done so, download the project files for this lesson from the Lesson & Update Files tab on your Account page at www.peachpit.com, store them on your computer in a convenient location, and define a new site based on the lesson06 folder as described in the "Getting Started" section at the beginning of this book.

Your Account page is also where you'll find any updates to the lessons or to the lesson files. Look in the Lesson & Update Files tab to access the most current content.



Dreamweaver's productivity tools and site-management capabilities are among its most useful features for a busy designer.

Creating a template from an existing layout

A template is a type of master page from which you can create related child pages. Templates are useful for setting up and maintaining the overall look and feel of a website while providing a means for quickly and easily producing site content. A template is different from a regular HTML page in Dreamweaver. In a normal webpage, Dreamweaver can edit the entire page. In a template, designated areas are locked and cannot be edited. Templates enable a workgroup environment in which page content can be created and edited by several team members, while the web designer controls the page design and the specific elements that must remainunchanged.

Note

Create a new site based on the lesson06 folder before beginning the lesson

---

Although you can create a template from a blank page, converting an existing page into a template is far more practical and also far more common. In this exercise, you'll create a template from an existing layout.

. Launch Dreamweaver CC (2019 release) or later.

Note

Don’t be concerned if various parts of the page do not render properly in Design view. New and advanced CSS properties are not supported within that view

---

. Open **mylayout.html** from the lesson06 folder. Switch to Design view.

The first step in converting an existing page to a template is to save the page as a template. Most of the work of creating a template must be completed in Design or Code view. The template options will not be accessible within Live view.

. Choose File > Save As Template.

The Save As Template dialog appears.

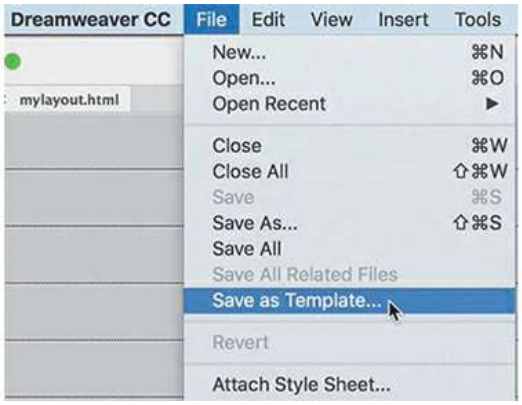
. If necessary, choose lesson06 from the Site pop-up menu.

Enter **GreenStart site template** in the Description field.

Type **mygreen_temp** in the Save As field. Click Save.

Tip

Adding the suffix “temp” to the filename is not a requirement, but it helps to visually distinguish this file from others in the site folder display.



An untitled dialog appears, asking whether you want to update links.

Templates are stored in their own folder, *Templates*, which Dreamweaver automatically creates at the site root level.

Note

A dialog may appear, asking about saving the file without defining editable regions; just click Yes to save anyway. You’ll create editable regions in the next exercise

---

- Click **Yes** to update the links.

Since the template is saved in a subfolder, updating the links in the code is necessary so that they will continue to work properly when you create child pages later. Dreamweaver automatically resolves and rewrites links as necessary when you save files anywhere in the site.

Although the page still looks exactly the same, you can identify that it's a template by the file extension displayed in the document tab: **.dwt**, which stands for Dreamweaver template.

A Dreamweaver template is *dynamic*, meaning that the program maintains a connection to all pages within the site that are derived from the template. Whenever you add or change content within the dynamic regions of the template and save it, Dreamweaver passes those changes to all the child pages automatically, keeping them up to date. But a template shouldn't be completely dynamic. Some sections of the page should contain areas where you can insert unique content. Dreamweaver allows you to designate these areas of the page as *editable regions*.

Inserting editable regions

When you create a template, Dreamweaver treats all the existing content as part of the master design. Child pages created from the template would be exact duplicates, and all the content would be locked and uneditable. This setup is great for repetitive features of the design, such as the navigation components, logos, copyright, contact information, and so on, but the downside is that it stops you from adding unique content to each child page. You get around this barrier by defining *editable regions* in the template. Dreamweaver creates two editable regions automatically, one for the `<title>` element and another for metadata or scripts that need to be loaded in the `<head>` section of the page; you have to create the rest.

First, give some thought to which areas of the page should be part of the template and which should be open for editing. At the moment, three sections of your current layout need to be editable: the main content area and the two `<aside>` elements.

- Open **mygreen_temp.dwt** from the lesson06 templates folder in Design view, if necessary. Maximize the program window to fill the entire screen.

The first step for creating each editable region is to update the placeholder text so that it's a bit more helpful.

- Select the text *Corporis expedita placeat* in the `<h1>` element.
- Type **Insert main heading here** to replace the text.

In Design view, you can enter and edit text directly without having to double-click it first.

- Select the text *Vero asperiores similique velit* in the `<h2>` element.
- Type **Insert article heading here** to replace the text.
- Select the five paragraphs of placeholder text in the `<article>` element.

Note

The template workflow currently works only in Design and Code views. You will not be able to perform any of these tasks in Live view

---

- Type **Insert content here.** to replace the text.

Note

Fonts may display differently between Design view and Live view and between Windows and macOS

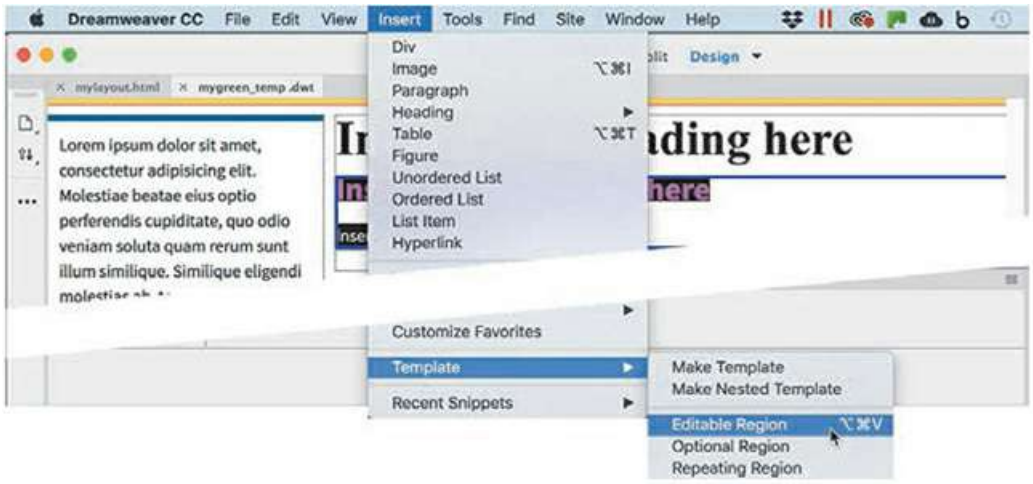
The main content is now ready to be converted. You'll want to add all three placeholders into your editable region, but there's a bug in Dreamweaver that will prevent you from applying the editable region to more than one selected element at a time. Instead, you'll convert the

<article> element first and then add the <h1> afterward.

- Click the article tag selector.

  Dreamweaver selects the entire <article> element. Although the selection includes a heading and paragraph text, you are selecting the single "parent" element, which will avoid the bug noted above.

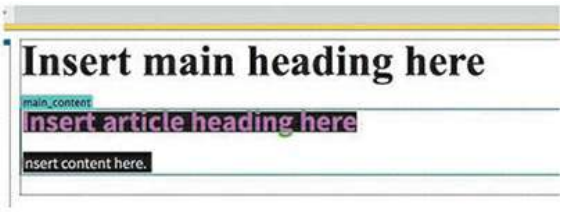
- Choose Insert > Template > Editable Region.



- In the New Editable Region dialog, enter **main_content** in the Name field.

  Each editable region must have a unique name, but no other special conventions apply. However, keeping the name short and descriptive is a good practice. The name is used solely within Dreamweaver and has no other bearing on the HTML code.

- Click OK.



  In Design view, you will see the new region name in a blue tab above the designated area, identifying it as an editable region. In Live view, the tabs appear orange in child pages. The editable region will encapsulate the <article> element. To include the <h1> you'll have to move it by hand.

- Switch to Split view.

Note that the HTML comment (around line 218) defining the editable region appears between the `<h1>` element and the `<article>`.

. Click the line number for the `<h1>` element.

The entire element is selected.

. Drag the selection below the HTML comment.

Note

You can also cut and paste to move the element

The `<h1>` is now inside the editable region.

. Save **mygreen_temp.dwt**.

You also need to add an editable region to the two aside elements, Sidebar 1 and Sidebar 2. Each of these sidebar regions needs a placeholder that you will then update on each child page. Let's update the placeholder text in Sidebar 1.

Building semantic content

Sidebar 1 will be used for environmentally themed quotations. Unlike with normal paragraph text, the value of a quotation is usually based on the perceived reputation of the author or source. HTML provides several elements designed specifically to identify this type of content.

- If necessary, open **mygreen_temp.dwt** in Design view.
- Insert the cursor in Sidebar 1.

  Examine the tag selectors showing the structure.

  The current structure is based on `<p>` and `<aside>` elements. Semantically, quotations should also include the `<blockquote>` element.

- Select the p tag selector.

  Press Ctrl+T/Cmd+T to edit the tag.

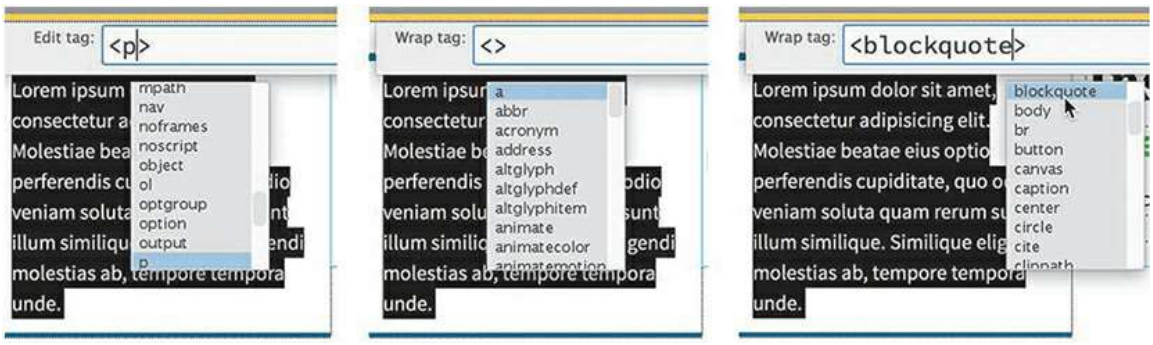
  The Quick Tag Editor appears, focused on the `<p>` tag. The editor has three modes: *Edit*, *Wrap*, and *Insert*. By default, the window opens in *Edit* mode. Using this mode, you could change the current tag to `blockquote` Instead, you'll want to add `blockquote` to the structure.

> **Tip**
>
> You can also access the Quick Tag Editor by right-clicking the element displayed in the Tag Selector interface

- Press Ctrl+T/Cmd+T again.

  The Quick Tag Editor switches to Wrap mode. You could press Ctrl+T/Cmd+T again to switch to Insert mode, if desired. Pressing the shortcut again would return to Edit mode.

- Type **blockquote** in the empty brackets.



As you type, the Code Hinting menu appears and will focus on the tag `<blockquote>`. Feel free to press Enter/Return to select and insert the tag using the menu. The new `<blockquote>` element has been created wrapping the existing `<p>` element.

. Press Enter/Return to close the Quick Tag Editor and complete the element.

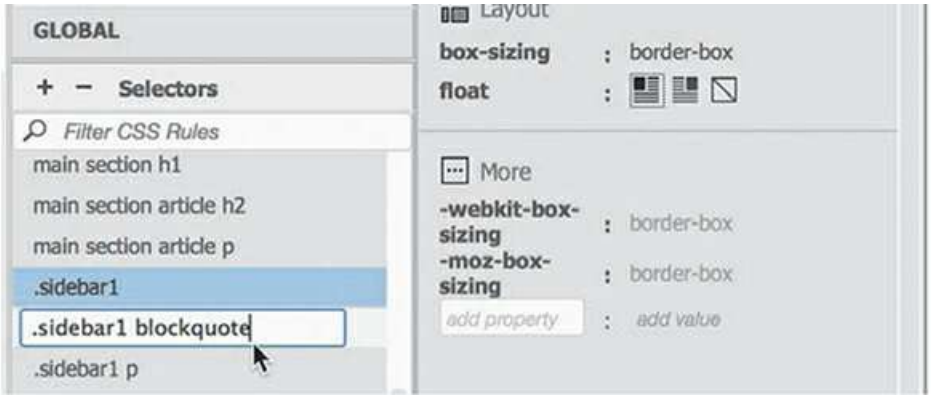
When the element is completed, the default styling of the `<blockquote>` element will apply indenting on the left and right. Such indentation is typical of material quoted within a term or research paper and may be desirable in the main content area, but it's totally unnecessary in the narrow `<aside>` elements. You'll need to create a new CSS rule to format these elements.

. In the CSS Designer, choose the selector, `sidebar1.`

Click the Add selector icon.

A new selector, `main .sidebar1 blockquote,` appears automatically.

. Press the up arrow on your keyboard to simplify the selector to **`.sidebar1 blockquote`** and press Enter/Return to complete the new selector.



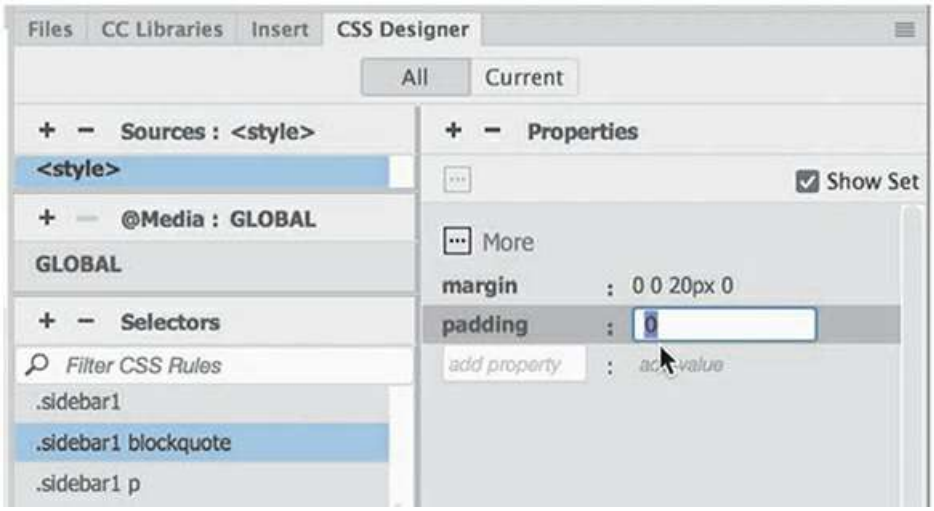
Tip

Whenever the dimension you need to enter is zero (0), you can enter the number without referencing a measurement system

---

. Create the following properties in the new rule:

```
margin: 0 0 20px 0
padding: 0
```



Typically, a blockquote element should contain the quoted text, either by itself or in one or more paragraphs, and an element providing the source or citation. Like `<blockquote>`, the `<cite>` element is designed specifically for this purpose.

- Select the placeholder text within the `<p>` element. Replace the text by typing **"Insert quotation here."**
- Press Enter/Return to create a new line.

  Type **Insert author here**

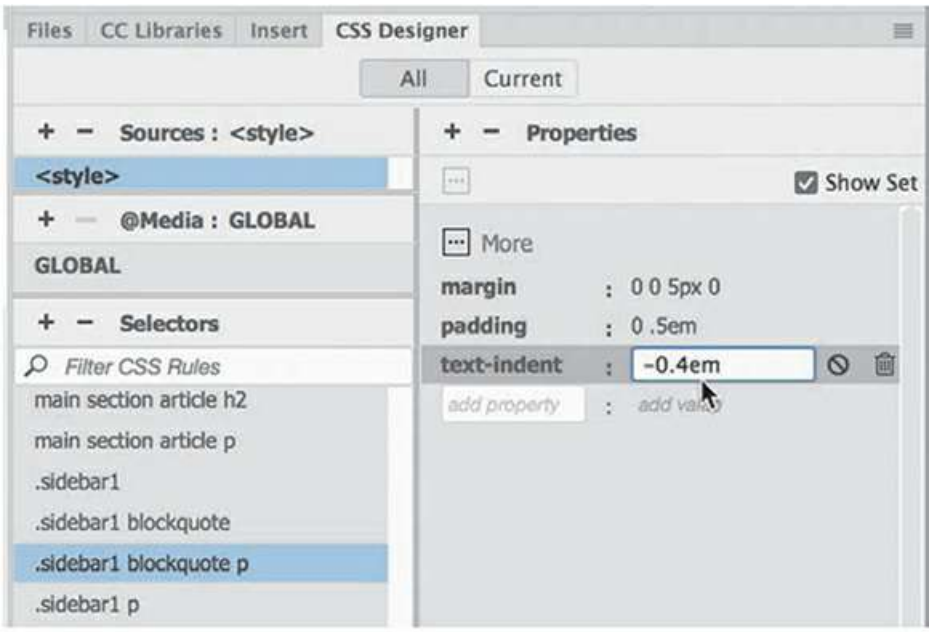
  Note that the quotation mark causes the first line of text to indent slightly, leaving it misaligned with the second line. Professional designers like to *outdent* such items to produce a *hanging* quotation mark.
- Choose .sidebar1 blockquote in CSS Designer.

  Create the following new selector:

  ```
  .sidebar1 blockquote p
  ```
- Create the following properties:

  Click here to view code image

  ```
  margin: 0 0 5px 0
  padding: 0 .5em
  text-indent: -0.4em
  ```



The effect can't be seen because both lines are `<p>` elements. Let's convert the second `<p>` element to create the `<cite>`.

. Insert the cursor in the text *Insert author here*. Select the `p` tag selector.

. Press Ctrl+T/Cmd+T to open the Quick Tag Editor.

Change the p tag to **`cite`** and press Enter/Return to complete the change.

Create a new rule to style the author name.

. Choose `.sidebar1 blockquote p` in CSS Designer.

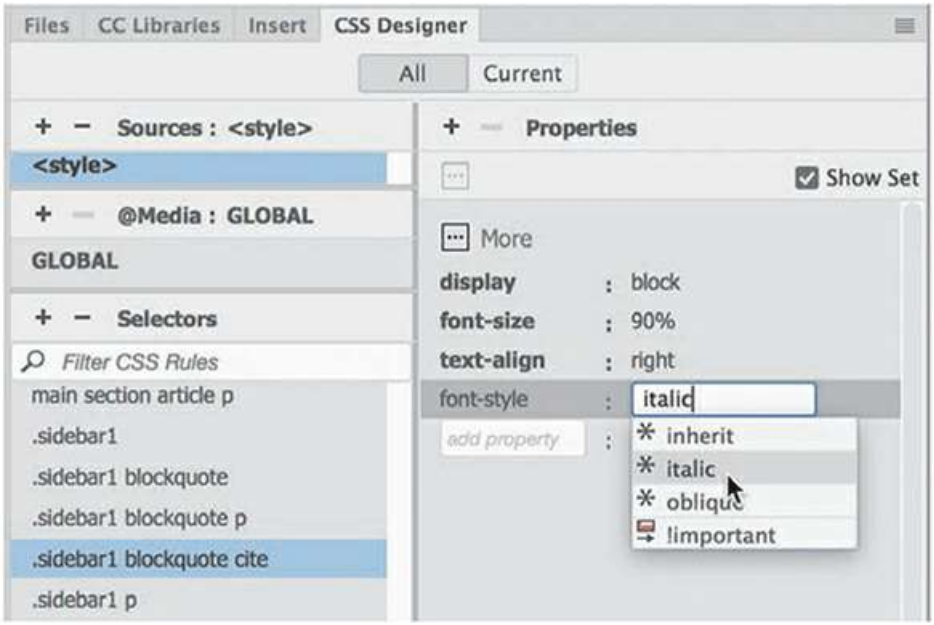
Create the following new selector:

```
.sidebar1 blockquote cite
```

. Create the following properties in the new rule:

Click here to view code image

```
display: block
font-size: 90%
text-align: right
font-style: italic
```



The `<blockquote>` structure is complete and contains a complete semantic structure. To remain semantically correct, each new quotation should be inserted into its own separate `<blockquote>` element.

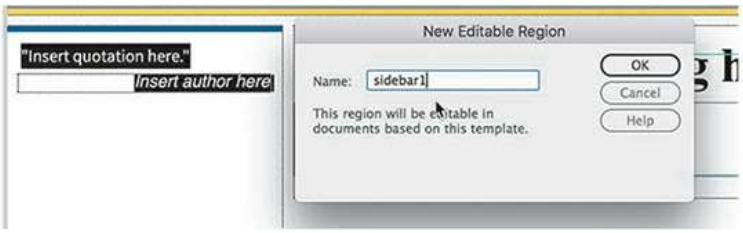
- Save the file.

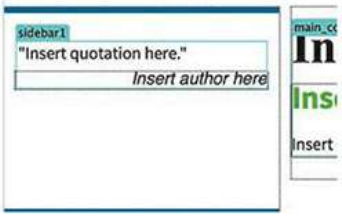
Now you are ready to add an editable region to Sidebar 1.

- Insert the cursor in Sidebar 1, if necessary.

Click the blockquote tag selector.

- Choose Insert > Template > Editable Region.
- In the New Editable Region dialog, type **sidebar1** in the Name field. Click OK.



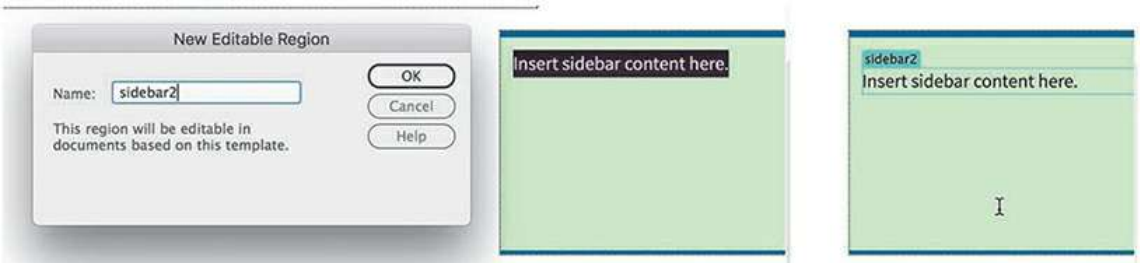


Let's complete Sidebar 2 next.

- Select the text in sidebar 2.

Type **Insert sidebar content here.** to replace it.

- Click the `aside.sidebar2` tag selector.
- Choose Insert > Template > Editable Region.
- In the New Editable Region dialog, enter **sidebar2** in the Name field. Click OK.



The Editable Regions are complete.

. Save the file.

The next thing you need to complete is the page footer. On most webpages, the footer element is usually the place where they put the copyright statement and other site or company information. Most designers like to use the actual copyright symbol in the statement as a professional touch, but there is no such symbol on the keyboard. So how can you create such a character if you can't type it?

Inserting HTML entities

As you learned in Lesson 2, "HTML Basics," every letter, punctuation mark, number, and special character is represented by an entity, even if you cannot type it from the keyboard. All entities can be added manually in Code view, but the most popular entities are supported directly by Dreamweaver. In this exercise, you will learn how to insert a copyright symbol into the footer using an HTML entity.

Note

Dreamweaver often uses *named* entities for special characters. Be aware that some web applications do not support named entities and often require the equivalent numbered entity instead. Check to make sure that the named entities you want to use are compatible with your workflow and site visitors

. If necessary, open **mygreen_temp.dwt** in Design view.

. Insert the cursor in the `<footer>` element. Select the placeholder text.

. Select Insert > HTML > Character > Copyright.

The copyright symbol (©) appears in the footer. Dreamweaver inserts the copyright character using the named entity `&copy;` in the code.

. Press the spacebar to insert a space.

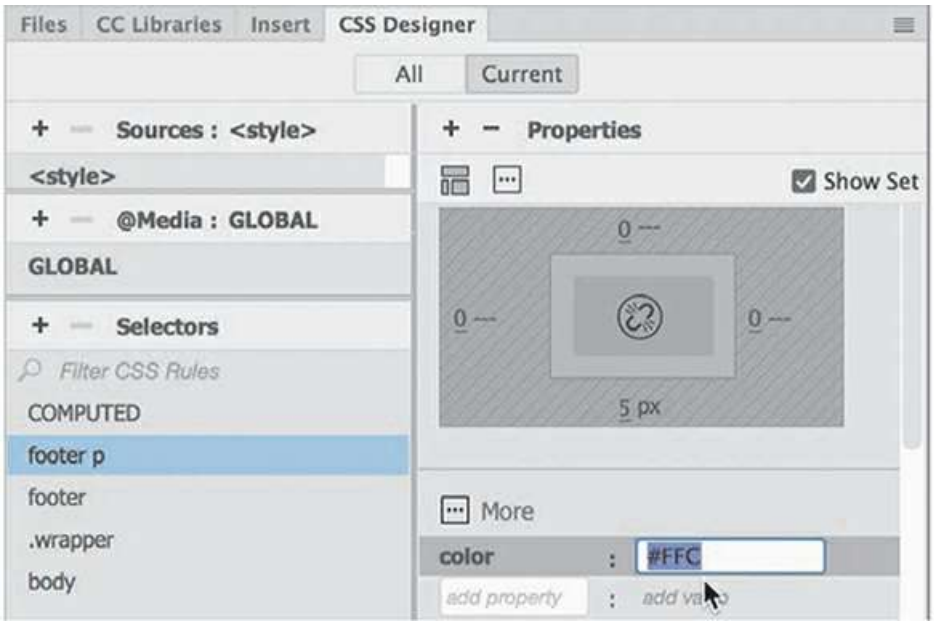
Type **2019 Meridien GreenStart. All rights reserved.**

Although you can't see the gradient background color in Design view, the black text will not be very legible on the dark green background. Let's match the color of the text in the top navigation menu.

Note

The numbered entity © is equivalent to the one inserted by Dreamweaver

- If necessary, select the rule `footer p`.
- Add the following property: **`color: #FFC`**



The pale yellow will look nice against the gradient background.

- Save the file.

The basic page layout for desktop media is complete. Once you have set up the visible components of the template, you should turn your attention to areas that are hidden from most visitors.

Inserting metadata

A well-designed webpage includes several important components that users may never see. One such item is the *metadata* that is often added to the `<head>` section of each page. Metadata is descriptive information about your webpage or its contents that is often used by other applications, such as a browser or a search engine.

Adding metadata—for instance, a piece of data such as the page *title*—is not only a good practice but also vital to your ranking and presence in the various search engines. Each title should reflect the specific content or purpose of the page. But many designers also append the name of the company or organization to help build more corporate or organizational awareness. By adding a title placeholder with the company name in the template, you will save time typing it in each child page later.

. If necessary, open **mygreen_temp.dwt in Design view.**

Tip

If the Property inspector is not visible, display it by choosing Window > Properties

---

. In the Document Title field of the Property inspector, select the placeholder text *Untitled Document*.

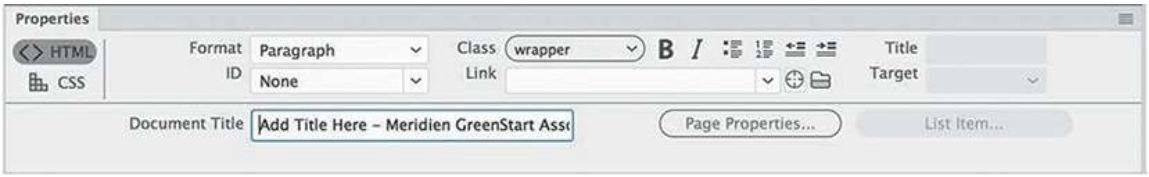
Many search engines use the page title in the listings of a search result. If you don't supply one, the search engine will pick one of its own. Let's replace the generic placeholder with one geared for this website.

Tip

The Document Title field is available in the Property inspector in all views

---

. Type **Add Title Here -Meridien GreenStart Association** to replace the text. Press Enter/Return to complete the title.



Along with the title, the other piece of metadata that usually appears in these search results is the page *description.* A description is a type of page summary that, in the past, succinctly described the contents in 160 characters or less. At the end of 2017, Google increased the size of the acceptable meta description to 320 characters.

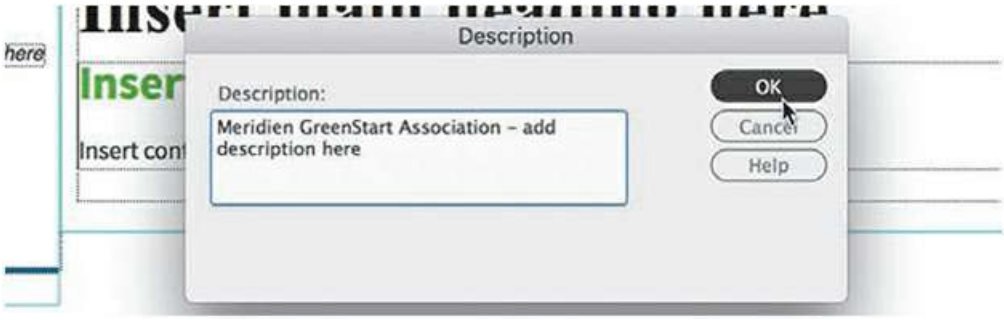
Over the years, web developers have tried to drive more traffic to their sites by writing misleading titles and descriptions or outright lies. But be forewarned—search engines have become wise to such tactics and will actually demote or even blacklist sites that use these tactics.

To achieve the highest ranking with the search engines, make the description of the page as accurate as possible. Try to avoid using terms and vocabulary that do not appear in the content. In many cases, the contents of the title and the description metadata will appear verbatim in the results page of a search.

- Choose Insert > HTML > Description.

  An empty Description dialog appears.

- Type **Meridien GreenStart Association - add description here**. Click OK.



Dreamweaver has added the two metadata elements to the page. Unfortunately, only one of them was implemented properly in the template.

- Switch to Code view. Locate and examine the `<title>` tag in the code and the surrounding markup.

```
3 ▼ <head>
4   <meta charset="UTF-8">
5   <!-- TemplateBeginEditable name="doctitle" -->
6   <title>Add Title Here - Meridien GreenStart Association</title>
7   <!-- TemplateEndEditable -->
8 ▼ <style type="text/css">
```

In most cases, the `<title>` will appear around line 6. Notice that the title appears between two comments that delineate an “editable” portion of the template named `"doctitle"`. This item was added correctly.

- Locate and examine the `<meta>` tag containing the `"description"` and the surrounding markup.

```
206  </style>
207  <!-- TemplateBeginEditable name="head" -->
208  <!-- TemplateEndEditable -->
209  <meta name="description" content="Meridien GreenStart Association - add
     description here">
210  </head>
```

You should find the description near the end of the `<head>` section, around line209. This element is not contained in an *editable section* of the template. This means that this metadata will be locked on all child pages, and you will not be able to customize it for that page.

Luckily, Dreamweaver comes to the rescue by providing an editable section designed for metadata just like this. In this case, it can't even get any more convenient—you'll find it just above the description delineated by the HTML comment markup `<!-- TemplateBeginEditable name="head"-->`. To make the description metadata editable, you just need to move it into this comment.

- Click the line number containing the entire description, or select the entire `<meta>` element using the cursor.

```
206  </style>
207  <!-- TemplateBeginEditable name="head" -->
208  <!-- TemplateEndEditable -->
209  <meta name="description" content="Meridien GreenStart Association - add
     description here">
210  </head>
```

The `<meta>` tag and its contents should occupy a single line of the markup.

- Drag or cut and paste the `<meta>` tag inside the HTML comment `<!-- TemplateBeginEditable name="head" -->`.

```
206  </style>
207  <!-- TemplateBeginEditable name="head" -->
208  <meta name="description" content="Meridien GreenStart Association - add
     description here">
209  <!-- TemplateEndEditable -->
210  </head>
```

The description is now contained within the editable template region named `"head"`.

- Choose File > Save.

You now have three editable regions—plus editable metadata for the title and description—that you can change as needed when you create new child pages using this template.

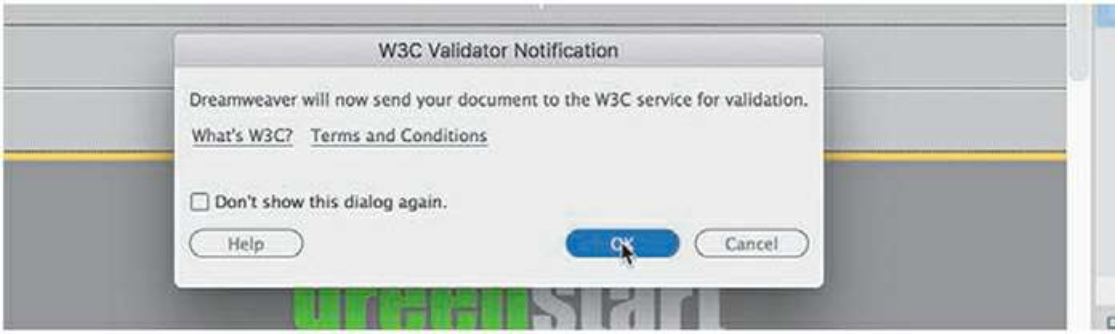
- Choose File > Close.

Before you use your template to create new pages, you should validate the quality of the code you created.

Validating HTML code

The goal whenever you create a webpage is to create code that will work flawlessly in all modern browsers. As you make major modifications in the sample layout, there's always a possibility that you may accidentally break an element or create invalid markup. These changes could have ramifications in the quality of the code or on whether it displays in the browser effectively. Before you use this page as your project template, you should check to make sure that the code is correctly structured and that it meets current web standards.

- If necessary, open **mygreen-temp.dwt** in Dreamweaver.
- Choose File > Validate > Current Document (W3C).



A W3C Validator Notification dialog appears, indicating that your file will be uploaded to an online validator service provided by the W3C. Before clicking OK, make sure you have a live Internet connection.

- Click OK to upload the file for validation.

After a few moments, you receive a report listing any errors in your layout. If you followed the instructions in the lessons correctly, there should be no errors.

- Close the file.

Congratulations! You created a workable basic page layout for your project template and learned how to insert additional components, placeholder text, and headings; modified existing CSS formatting and created new rules; and validated the HTML code successfully. Now it's time to learn how to use a Dreamweavertemplate.

Producing child pages

Child pages are the *raison d'être* for Dreamweaver templates. Once a child page has been created from a template, only the content within the editable regions can be modified in the child page. The rest of the page remains locked within Dreamweaver. It's important to remember that this behavior is supported only within Dreamweaver and a few other HTML editors. Be aware that if you open the page in a text editor, like Notepad or TextEdit, the code is fully editable.

Creating a new page

The decision to use Dreamweaver templates for a site should be made at the beginning of the design process so that all the pages in the site can be made as child pages of the template. In fact, that was the purpose of the layout you've built up to this point: to create the basic structure of your site template.

- Launch Dreamweaver CC (2019 release) or later, if necessary.

  The template workflow functions only in Design and Code views. You can also access site templates from the New Document dialog.

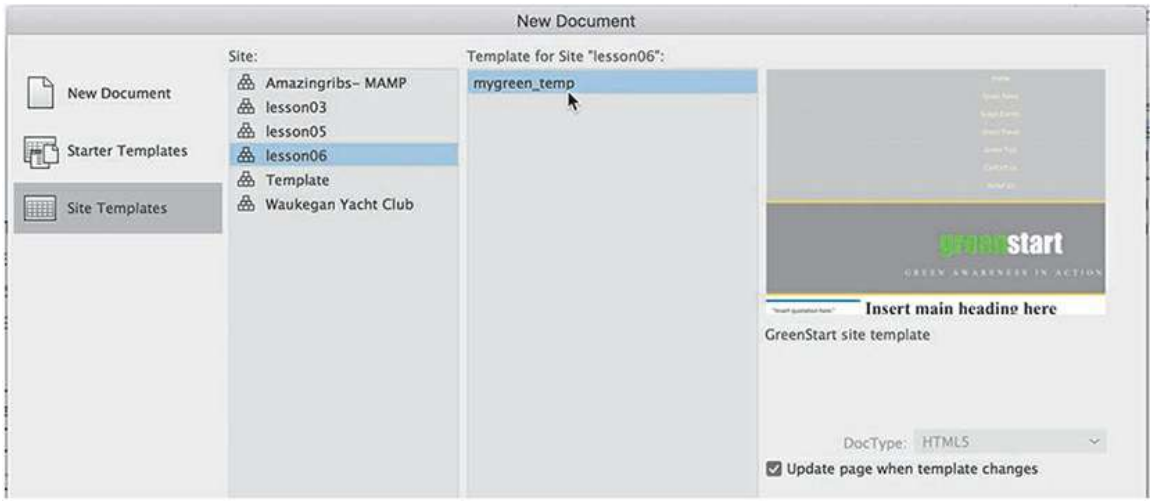
- Choose File > New, or press Ctrl+N/Cmd+N.

  The New Document dialog appears.

- In the New Document dialog, select the Site Templates option.

  Select lesson06 in the Site list, if necessary.

  Select **mygreen_temp** in the Template For Site "lesson06" list.

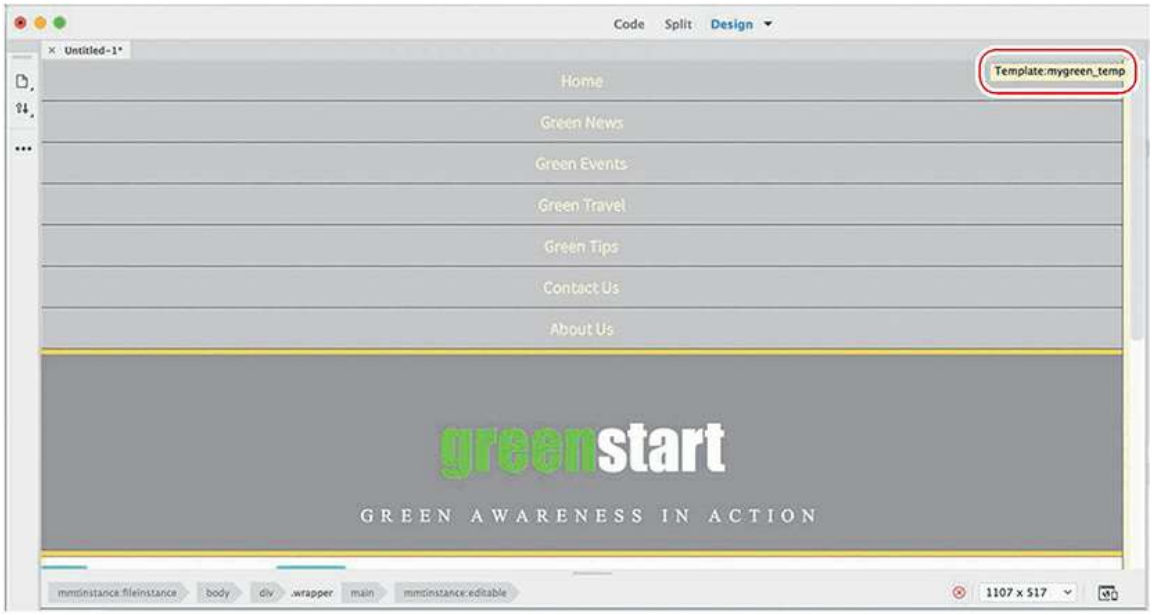


- Select the Update Page When Template Changes option, if necessary.

  Click Create.

  Dreamweaver creates a new page based on the template.

- If necessary, switch to Design view.



Typically, Dreamweaver defaults to the last document view (Code, Design, or Live) you were using for the new document. In Design view, you will see the name of the template file displayed in the upper-right corner of the document window. Before modifying the page, you should save it.

- Choose File > Save.

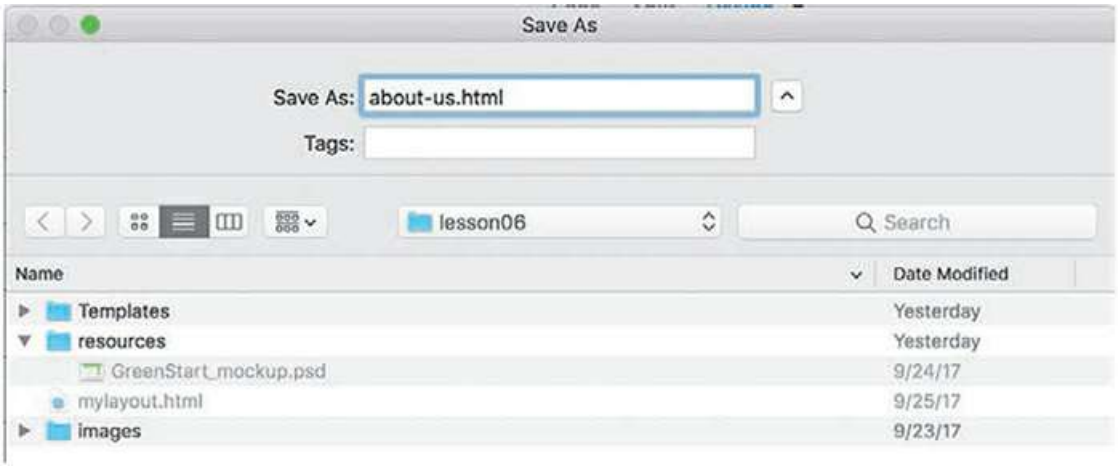
The Save As dialog appears.

Tip

The Save As dialog provides a handy button to take you to the site root with a single click. Feel free to use it in any exercise, as needed.

- In the Save As dialog, navigate to the site root folder.

Name the file **about-us.html** and click Save.



The child page has been created. When you save the document in the site root folder, Dreamweaver updates all links and references to external files. The template makes it easy to add new content.

Adding content to child pages

When you create a page from a template, only the editable regions can be modified.

- Open **about-us.html** in Design view, if necessary.

  You'll find that many of the features and functionality of templates work properly only in Design view, although you should be able to add or edit content in the editable regions from Live view.

- Position the cursor over each area of the page. Observe the cursor icon.

Warning

If you open a template in a text editor, all the code is editable, including the code for the noneditable regions of the page.

When the cursor moves over certain areas of the page, such as the horizontal menu, header, and footer, the Locked icon appears. These areas are uneditable regions that are locked and cannot be modified within the child page inside Dreamweaver. Other areas, such as `sidebar1` and the main content section, can be changed.

. Open the Properties panel, if necessary.

In the Title field, select the placeholder text *Add Title Here*.

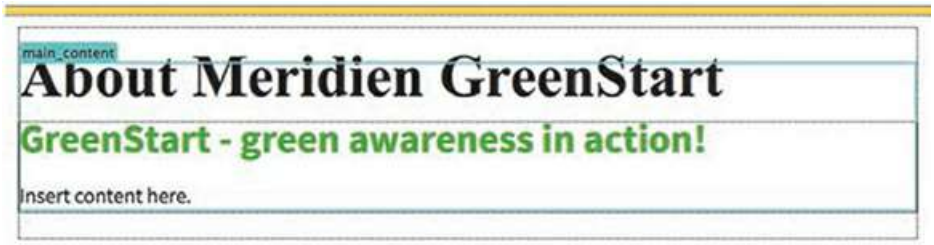
Type **About Meridien GreenStart** and press Enter/Return.

. Select the placeholder text *Insert main heading here*.

Type **About Meridien GreenStart** to replace the text.

. Select the placeholder text *Insert article heading here*.

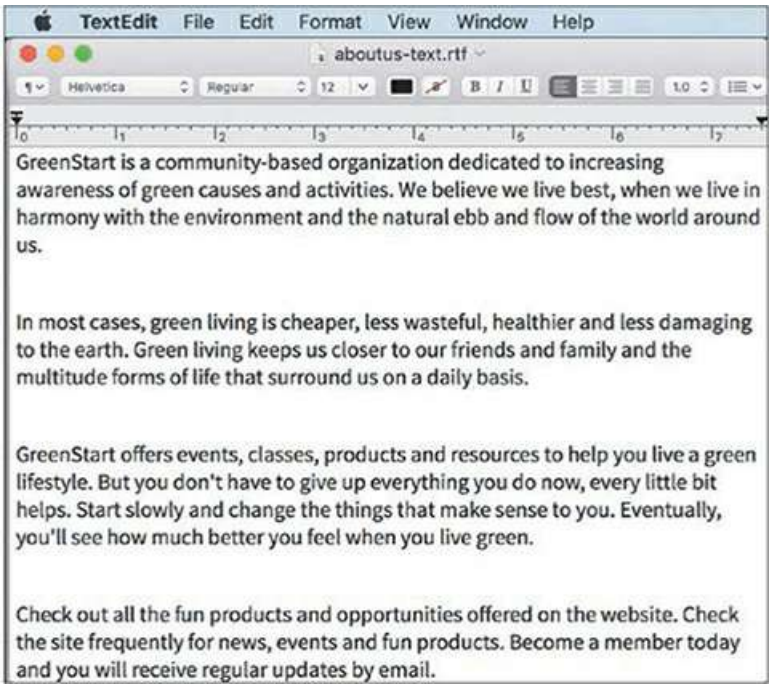
Type **GreenStart - green awareness in action!** to replace the text.



Tip

To add a little editorial flair, use the command Insert > HTML > Character > Em Dash to replace the hyphen in the heading with a long dash.

- In the Files panel, double-click **aboutus-text.rtf** in the lesson06 resources folder to open the file.



Dreamweaver opens only simple, text-based file formats, such as .html, .css, .txt, .xml, .xslt, and a few others. When Dreamweaver can't open the file, it passes the file to a compatible program, such as Word, Excel, WordPad, TextEdit, and so on. The file contains content for the main content section.

- Press Ctrl+A/Cmd+A to select all the text.

  Press Ctrl+C/Cmd+C to copy thetext.
- Switch back to Dreamweaver.
- Insert the cursor in the placeholder text *Insert content here*.

  Select the p tag selector.
- Press Ctrl+V/Cmd+V to paste the text.

The placeholder text is replaced by the new content. You can also add content to the sidebar elements.

. Open **sidebars06.html** in Design view from the site root folder.

The file contains content for each sidebar. The top half is composed of three environmentally themed quotations, and the bottom half is composed of environmental tips and news.

. Insert the cursor in the first paragraph and examine the tag selectors.

The tag selectors indicate a structure identical to what you created for the quotations sidebar in Lesson 5, "Creating a Page Layout," but unformatted by the CSS. Let's use this content to replace the existing sidebar placeholder.

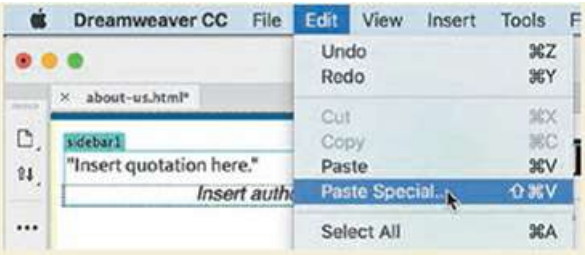
. Insert the cursor into the first quotation.

Click the `blockquote` tag selector.

The first entire quotation is selected. The tag selector interface shows the same HTML structure used in the template.

When paste won't work

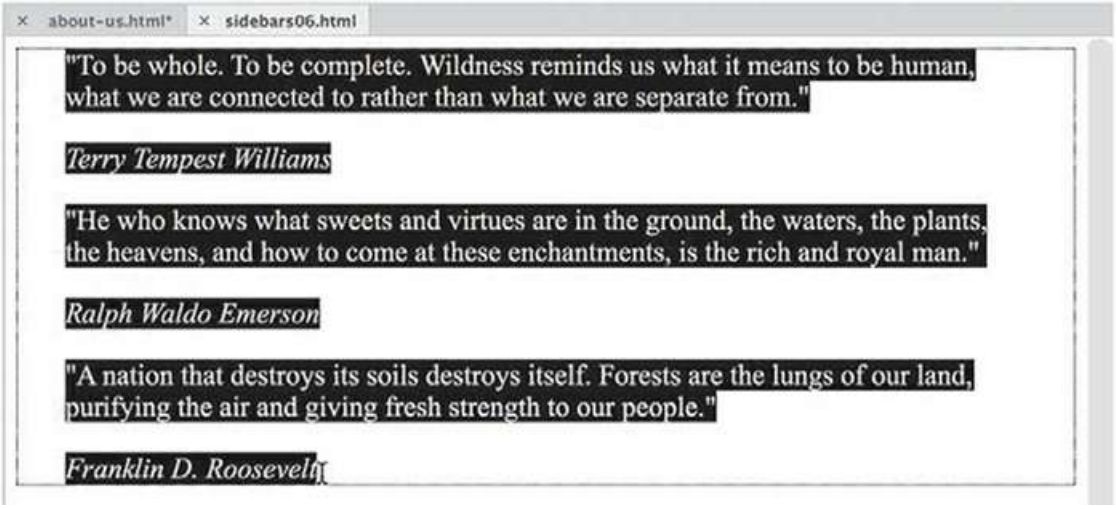
If Dreamweaver passes the file **about_us.rtf** to Word or to a similar word-processing program, you may find that using the paste command Ctrl+V/Cmd+V does not work. If nothing happens after you press the keyboard shortcut or use the menu option, you will have to use the Paste Special command.



Once you activate this command, a dialog will appear asking you how you want to paste the text.

In most cases, you will want to paste using the Text Only option and then apply formatting in Dreamweaver. Depending on the source program, and whether you are using Windows or macOS, text formatting may or may not come across the clipboard.

. Hold the Shift key and click at the end of the name *Franklin D. Roosevelt*.

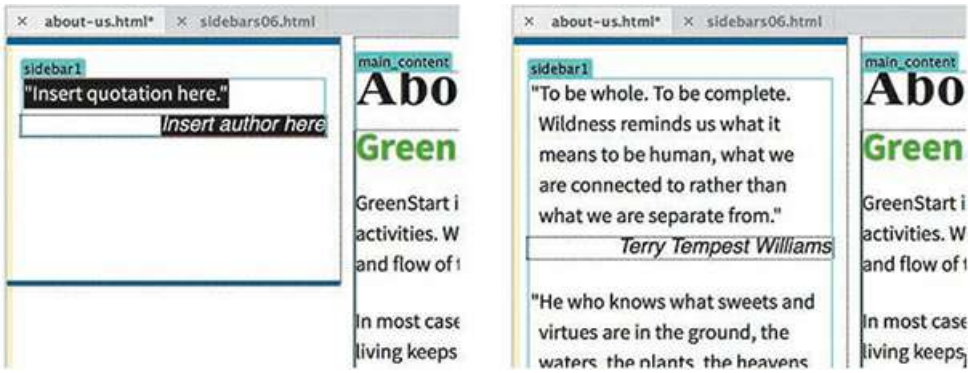


All three quotations are selected.

- Press Ctrl+X/Cmd+X to cut the quotations into memory.
- Select the **about-us.html** document tab.

  The child page appears in the document window again.
- Insert the cursor into Sidebar 1.

  Select the `blockquote` tag selector.
- Press Ctrl+V/Cmd+V to replace the sidebar placeholder.



  The replacement content appears.
- Select the **sidebars06.html** document tab.

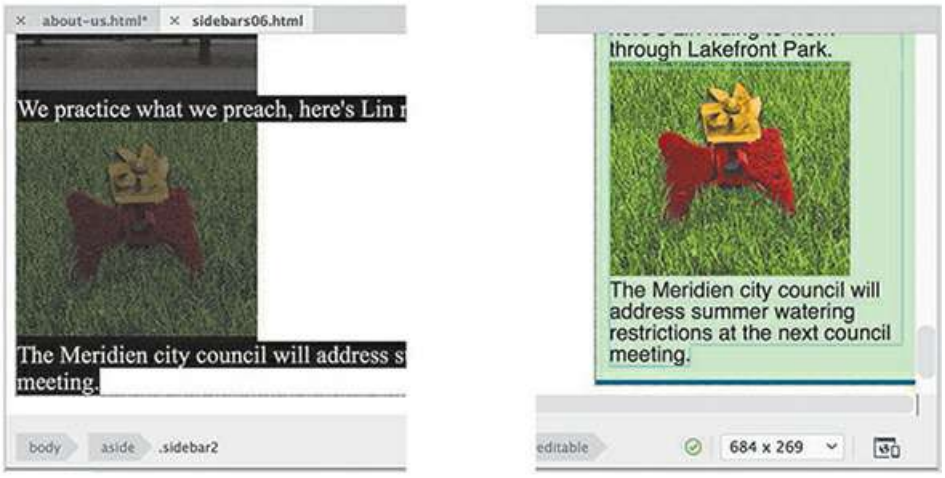
  The contents of **sidebars06.html** appear in the document window.
- Insert the cursor in the caption of the first image.

  Note the tag selectors.

  The caption is part of a `figure` element.
- Select the `figure` tag selector.

  The first image and caption are selected.
- Hold the Shift key and click at the end of the text "*…at the next council meeting.*" Cut the text into memory.

  All the figures have been selected.
- Switch back to **about-us.html**.
- Select p element in Sidebar 2 and paste the figures.



All three placeholders in the file are now filled with content.

- Save **about-us.html**.
- Close **sidebars06.html**. Do not save the changes.

  By not saving the changes, you preserve the content in the file if you want to repeat the exercise later.
- Switch to Live view to preview the page.

All the CSS styling kicks in again, and the page design and most of the page content now render properly. Although you can't see the template name in the upper-right corner anymore, the names of the editable regions now display in orange tabs above their corresponding element.

Notice how badly the text and images in Sidebar 2 are formatted.

- Select the caption element below the first image in Sidebar 2.

  Examine the tag selectors.

  The content in Sidebar 2 is based on the elements `<figure>` and `<figcaption>`, but the

original styling was based on the `p` element. This new structure is appropriate for an image with an associated caption. The default formatting for the figure element will need a little tweak to work properly here. In most formatting tasks, you should normally style the parent element first.

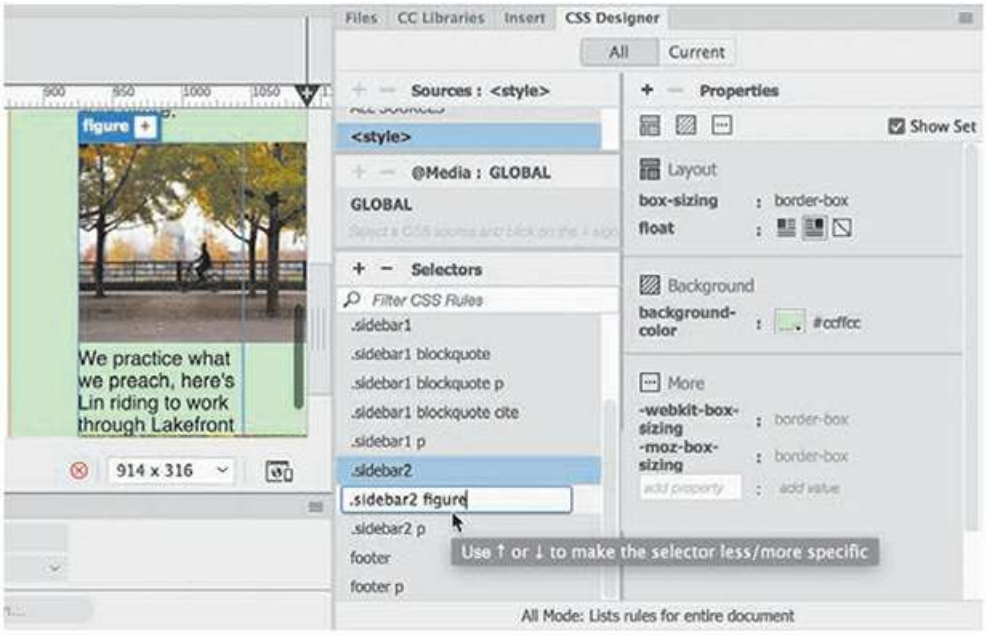
. Select the `figure` tag selector.

In the CSS Designer, click the Current button, if necessary.

The Selectors pane shows no rule targeting the current structure or element.

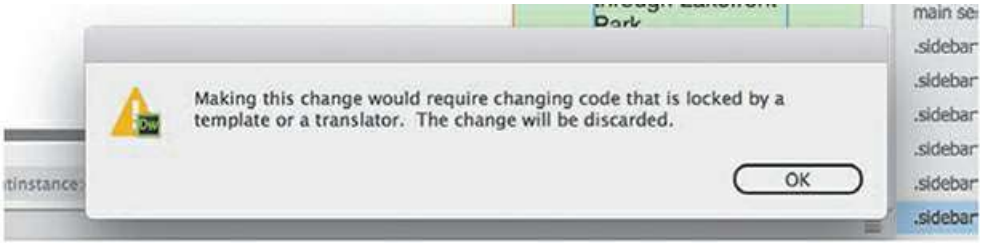
. Click the Add Selector icon.

A new selector field appears, with a selector targeting the selected element.



. If necessary, press the up arrow to create the selector `.sidebar2 figure`.

. Press Enter/Return to create the selector.



An error message appears indicating that creating the new selector would require changing code locked in the template. That's because this file uses an embedded style sheet that is controlled by the Dreamweaver template. If the style sheet is kept as part of the template, you will not be able to edit any of theCSS.

. Click OK to dismiss the warning.

You could move the embedded style sheet to an editable region of the template, but then each page of the site would have a different style sheet. Imagine having dozens of pages on your site with different style sheets. A change in one page would not be reflected in the other pages on the site. It would quickly become a maintenance nightmare.

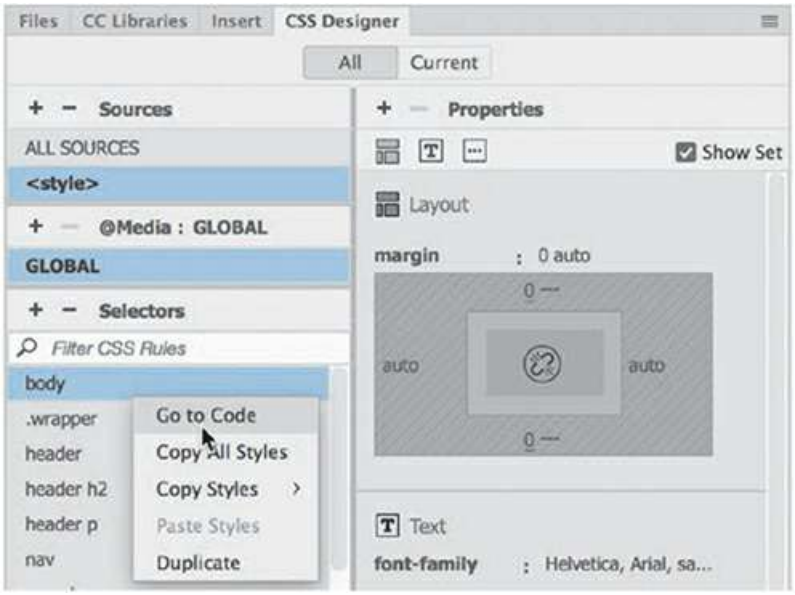
A better option, and the one used on most websites, is to move the style sheet to a separate file to which each page can be linked. To make this change, you have to open the template again.

Moving CSS styles to a linked file

Using an embedded style sheet is handy when developing the initial webpage design and site template. But embedded styles sheets are more difficult to update, and they can't provide the productivity advantages of an external, linked style sheet. In this exercise, you will move the embedded styles from the site template to a separate, external CSS file.

- In the Files panel, double-click **mygreen_temp.dwt** to open it.

  If necessary, switch to Design view.
- If necessary, select Window > CSS Designer to display the panel.
- In the Selectors panel, right-click the first rule in the list and select Go To Code in the context menu.

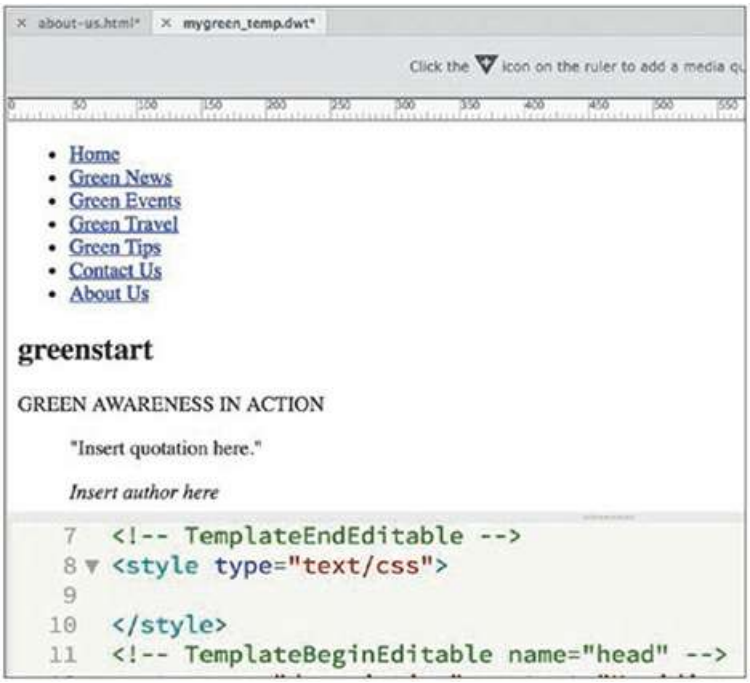


  The Code view window opens and is focused on the rule targeted. You want to move all the CSS rules but not the `<style>` and `</style>` tags themselves.
- Select all the CSS rules between the `<style>` and `</style>` tags in the `<head>` section of the page, but do not select the tags.

```
 7  <!-- TemplateEndEditable -->
 8  <style type="text/css">
 9  body {
10      margin: 0 auto;
11      font-family: Helvetica, Arial, sans
12  }
13
14  .wrapper {
15      overflow: auto;
16      width: 1100px;
17  }
18  header {
```

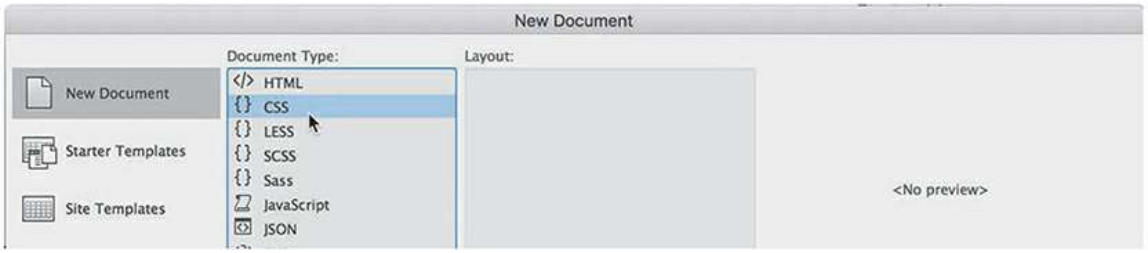
```
197      border-top: solid 3px #666;
198      background-image: -webkit-linear-gradien
         0%,rgba(3,185,36,1.00) 100%);
199      background-image: -moz-linear-gradient(9
         0%,rgba(3,185,36,1.00) 100%);
200      background-image: linear-gradient(0deg,
201  }
202  footer p {
203      margin: 0 0 5px 0;
204      color: #FFC;
205  }
206  </stvle>
```

. Press Ctrl+X/Cmd+X to cut the code.



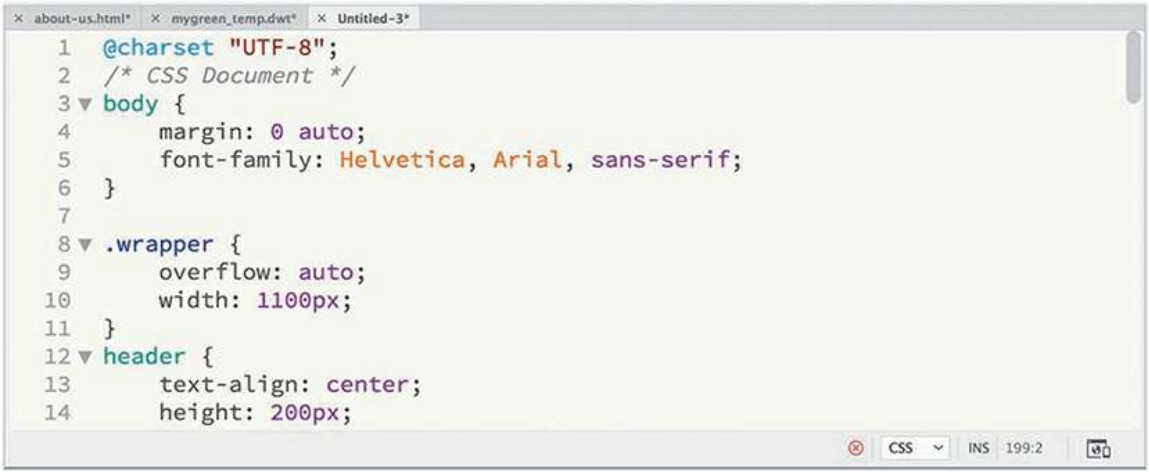
The formatting in the template disappears. Only the default HTML styling is visible.

. Select File > New. In the New Document dialog, select CSS in the Document Type column. ClickCreate.



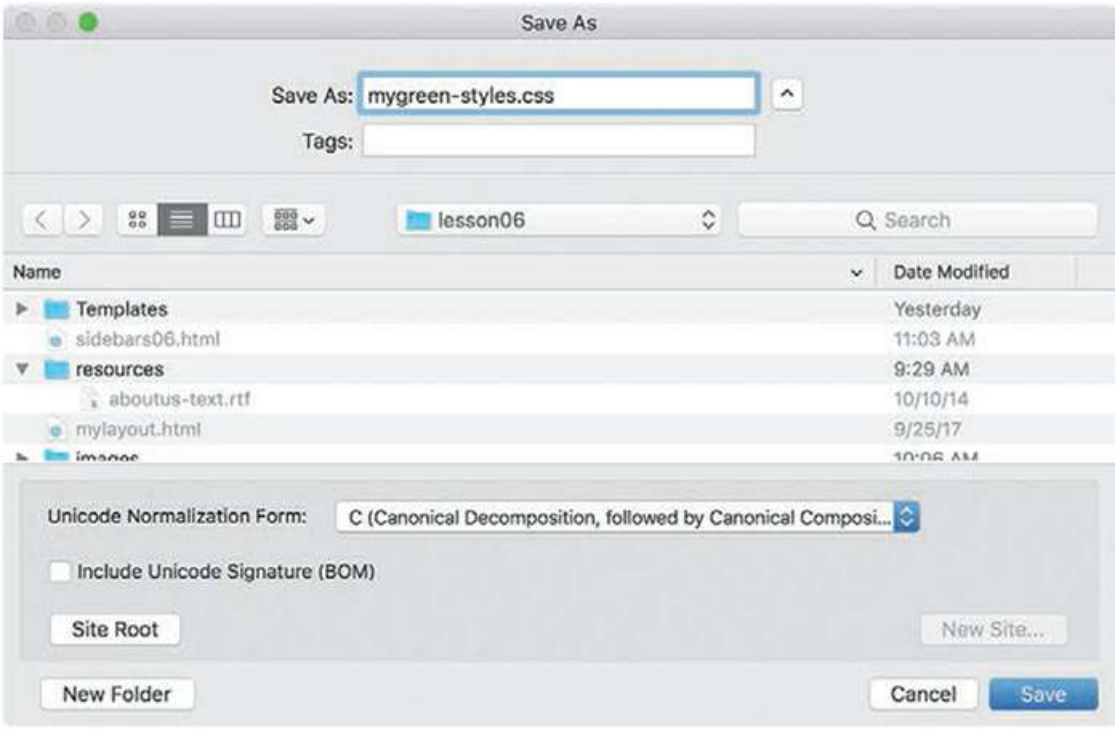
A new CSS file appears. The file contains the markup `@charset "UTF-8"; and /* CSS Document */` but is otherwise empty.

. Insert the cursor in line 3 and press Ctrl+V/Cmd+V.



The styles from the template appear in the sheet.

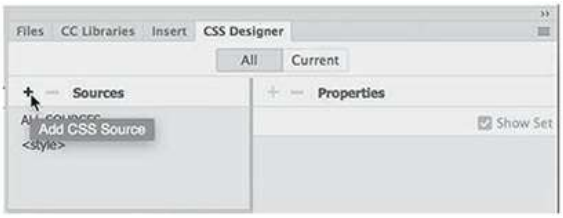
- Select File > Save. Name the file **mygreen-styles.css** and, if necessary, navigate to the site root folder.



- Click Save. Close the CSS file.

The styles have now been moved to an external CSS file. The last step is to link the style sheet to the template.

- In **mygreen_temp.dwt,** click the Add CSS Source + icon in the CSS Designer.
- Select Attach Existing CSS File in the drop-down menu.



The Attach Existing CSS File dialog appears.

. Click the Browse button.

Select **mygreen-styles.css** in the site root folder.

Click Open.

. Click OK.

The CSS file is now linked to the template. The template is styled again, but there is something wrong. The stripe and fern images are missing.

. In the CSS Designer, right-click the header rule and select Go To Code from the context menu.

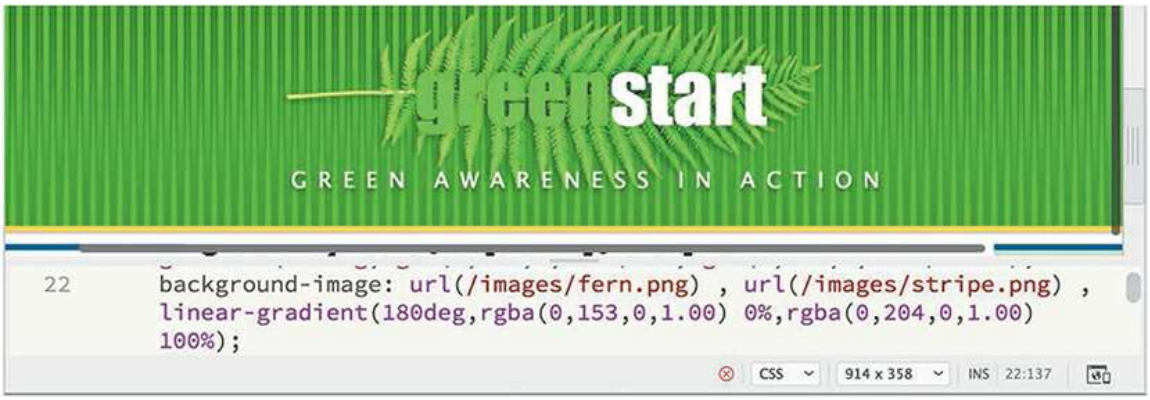
The Code view window focuses on the contents of the new external CSS file. Note the references to **stripe.png** and **fern.png**. You will see that Dreamweaver added two dots (..) before the pathname.

The template is located in the subfolder called *templates*. When creating pathnames from this location, you would add two dots to tell the browser to look for the image in the parent folder,

or one level up from the current location.

If the style sheet were saved in the templates folder too, this pathname would be correct. But the style sheet is located in the site root. The correct pathname would be **/images/stripe.png** and **/images/fern.png**.

- Remove the two dots (..) from the pathnames for **stripe.png** and **fern.png** in **mygreen-styles.css**.



The images may appear in the document window as soon as you delete the dots, but the change is not permanent yet.

- With the cursor in the Code view window, select File > Save to save **mygreen-styles.css**, or press Ctrl+S/Cmd+S.
- Close **mygreen-styles.css.**

The changes are permanent now. The embedded styles have been removed from the template and saved to an external CSS file. However, those changes have not been made to the child page **about-us.html**.

Updating a template

Templates can automatically update any child page made from that template. But only areas outside the editable regions will be updated. Let's make some other changes in the template to demonstrate how they work.

- Switch to Design view.

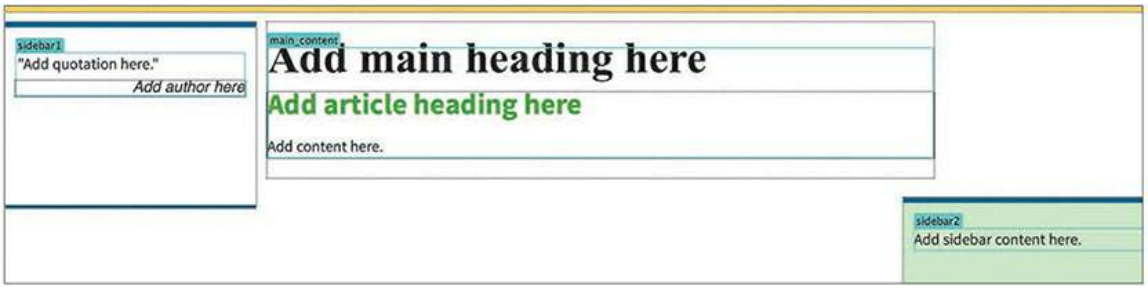
  In the navigation menu, select the text *Home*.

  Type **Green Home** to replace the text.

- In the horizontal menu, select the text *Green News*.

  Type **Headlines** to replace the text.

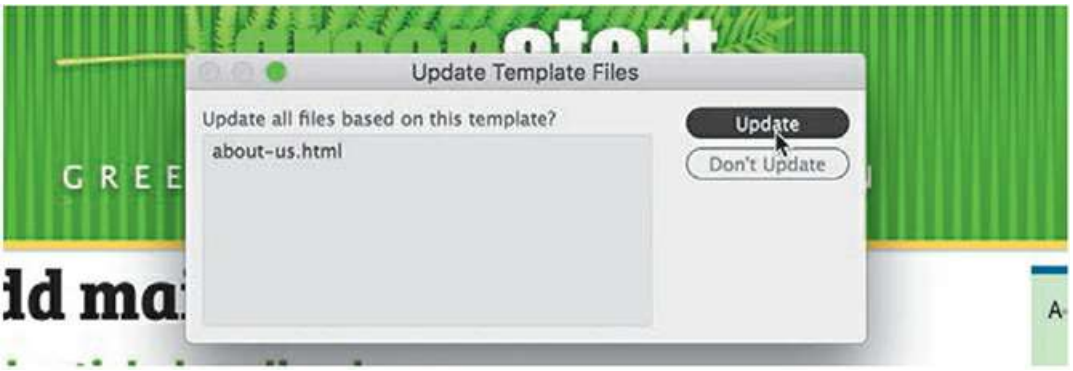
- Select and replace the text Insert with the word **Add** wherever it appears in the `main_content, sidebar1, and sidebar2` editable regions.



- Switch to Live view.

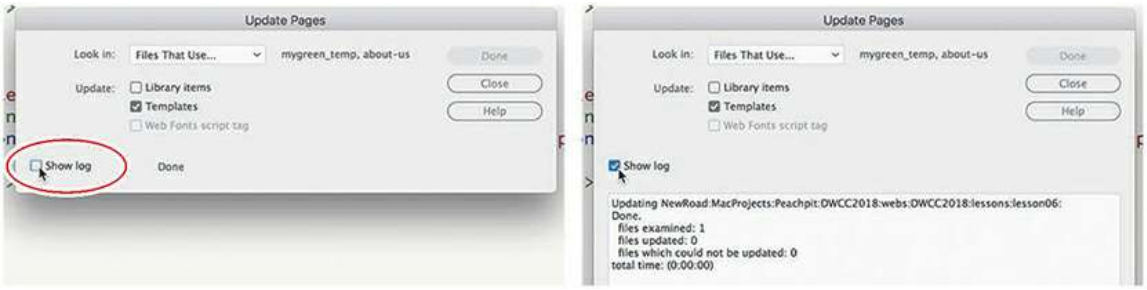
You can now clearly see the changes to the menu and content areas. In the template, the entire page is editable.

- Save the file.



The Update Template Files dialog appears. The filename **about-us.html** appears in the update list. This dialog will list all files based on the template.

- Click Update. The Update Pages dialog appears.
- If necessary, select the Show Log option.



Note

The Update Pages function can sometimes take a long time to complete. If your update freezes, the dialog provides a Stop button with which you can exit the process before it finishes

---

A window displays a report that lists which pages were successfully updated and which ones were not.

- Close the Update Pages dialog.
- Switch to **about-us.html** by clicking the document tab.

Observe the page and note any changes.

The changes made to the horizontal menu in the template are reflected in this file, but the changes to the sidebars and main content areas were ignored and the content you added to both areas earlier remains unaltered.

. In the CSS Designer click the All button, if necessary, and examine the Sourcespane.

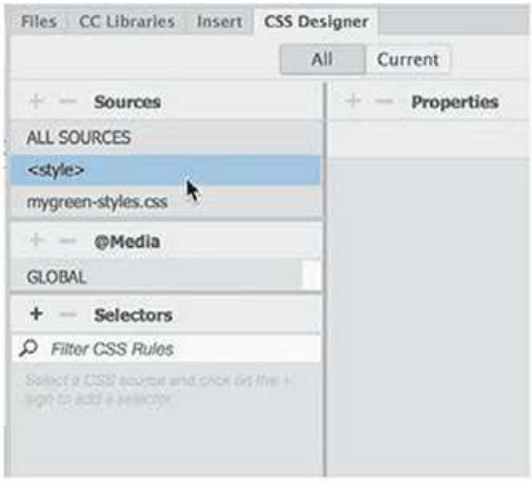
The Sources pane lists two CSS sources: `<style>` and **mygreen-styles.css**. The `style` item refers to the embedded style sheet in the `<head>` section of thepage.

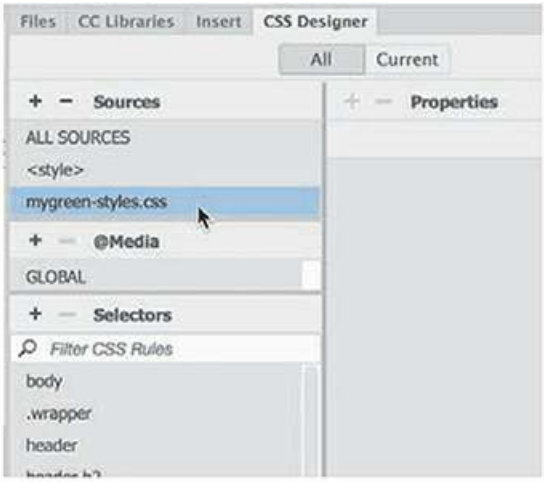
. Select `<style>` in the Sources pane.

Observe the Selectors pane.

The Selectors pane is empty, indicating that there are no rules defined in the embedded style sheet.

. Select **mygreen-styles.css** in the Sources pane. Observe the Selectors pane.





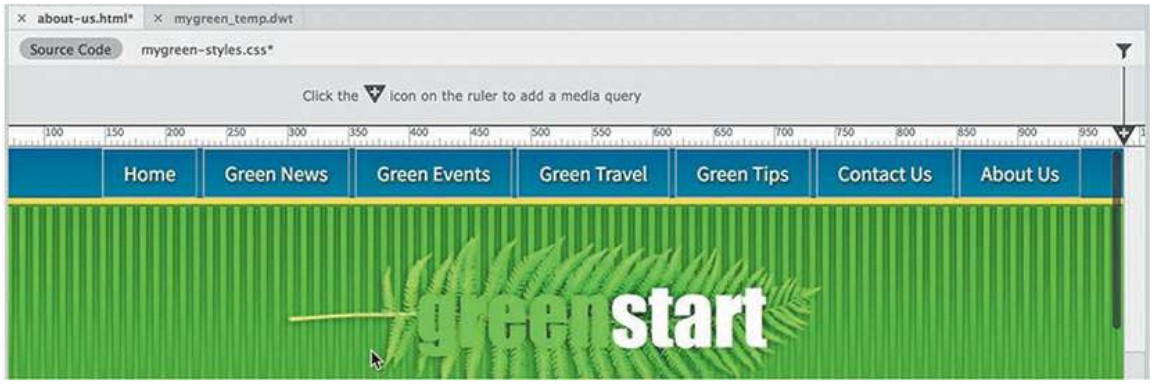
The Selectors pane displays all the CSS rules that previously were contained in the embedded style sheet. The changes made in the template were passed along to **about-us.html**.

As you can see, you can safely make changes and add content to the editable regions without worrying that the template will delete all your hard work. At the same time, the boilerplate elements of the header, footer, and horizontal menu all remain consistently formatted and up to date, based on the status of the template.

- Click the document tab for **mygreen_temp.dwt** to switch to the template file.
- Switch to Design view.
- Delete the word Green from the *Green Home* link in the navigation menu. Change the word *Headlines* back to **Green News**.

- Save the template and update the related files.
- Click the document tab for **about-us.html**.

Observe the page and note any changes.



The horizontal menu has been updated. Dreamweaver even updates linked documents that are open at the time. The only concern is that some changes have not been saved. Note that the document tab shows an asterisk, which means the file has been changed but not saved.

Tip

If an open page has been changed during the update, it will be updated but not

saved by Dreamweaver and show an asterisk by its name in the document tab.

If Dreamweaver or your computer were to crash at this moment, all the changes you made would be lost; you would have to update the page manually or wait until the next time you make changes to the template to take advantage of the automatic update feature.

Tip

Always use the Save All command whenever you have multiple files open that may have been updated by a template. In most cases, it's better to update when your files are all closed so that they are saved automatically

Note

Dreamweaver added a limited auto-backup feature in a previous version. If the program crashes, some or all of the changes you have made may be preserved.

- Select File > Save All.
- Close **mygreen_temp.dwt**.

Now that you moved the styles to the linked file, you should be able to create the rule to style the `figure` element in Sidebar 2.

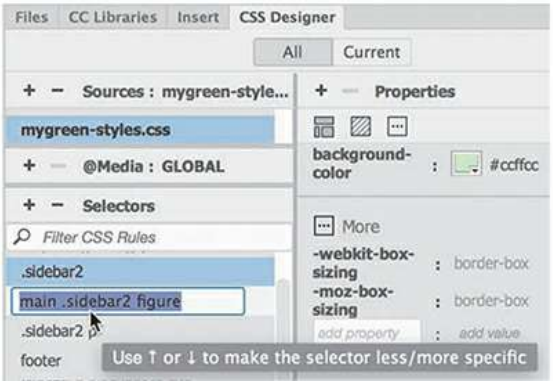
Formatting content in editable regions

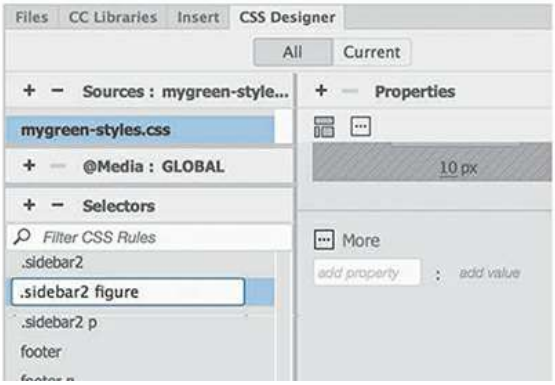
Often, content inserted into the editable regions will need custom styling to help it adapt to the layout. In this exercise, you will create rules to format pictures and text in Sidebar 2.

- Switch to Live view, if necessary.

  Select the first text caption in Sidebar 2.
- Select the `figure` tag selector.

  In the CSS Designer, click the Current button, if necessary.
- Click the Add Selector icon.
- If necessary, press the up arrow to simplify the selector name to `.sidebar2 figure`.





- Press Enter/Return to create the selector.

  The text and images in the sidebar appear over to the right side of the element. There were no existing rules formatting this content, so the styling must be the default HTML styling of the `figure` element. When the default styling of an element disrupts the normal flow or appearance of the page, you will often have to reset thestyling.

- Add the following property to `.sidebar2 figure`

  ```
  margin: 0 0 10px 0
  ```

  The `figure` element now extends to the left and right sides of the sidebar but still honors the 10 pixels of padding applied to it. Let's center the image in the `figure` element.

- Select the first image in the `figure` element.

  In CSS Designer, click the Currentbutton.

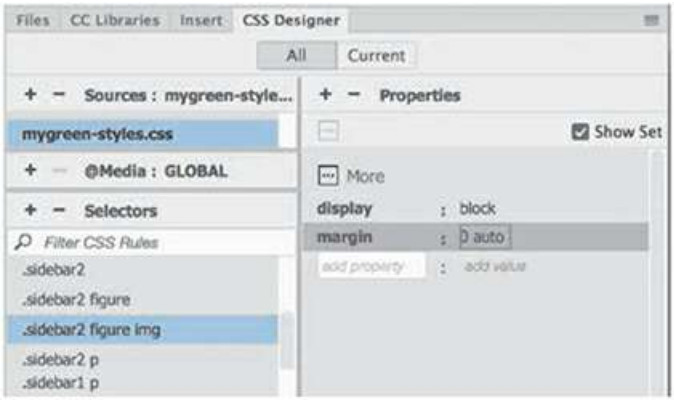
- Click the Add Selector icon.

  Create the following selector:

  ```
  .sidebar2 figure img
  ```

- Create the following properties in the new rule:

  ```
  display: block
  margin:0 auto
  ```



The images in Sidebar 2 align to the center of the column. Setting auto margins on the left and right forces elements toward the center. But this alone will not achieve the desired result. Images are considered inline elements, which ignore margin settings. Setting `display: block` allows images to honor the specifications.

. Select the first caption in Sidebar 2.

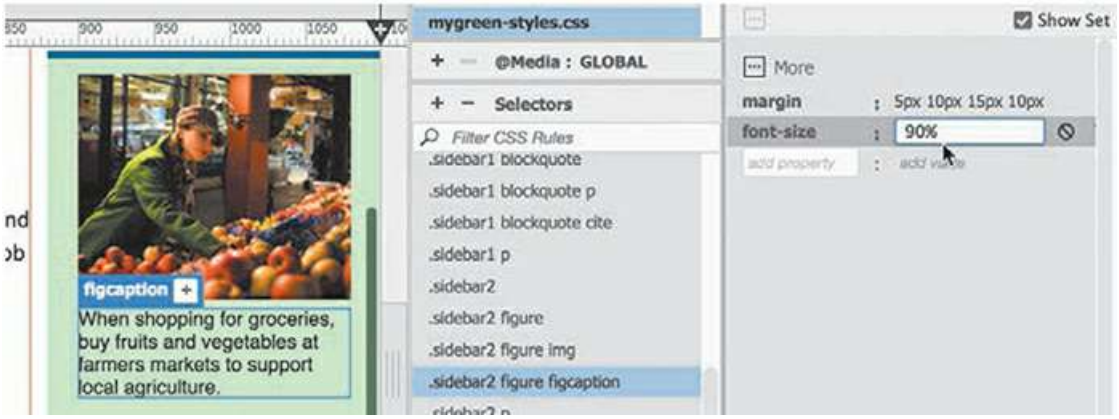
The element display appears focused on the `figcaption` element.

. Create the following rule:

```
.sidebar2 figure figcaption
```

. Add the following property to the rule:

```
margin: 5px 10px 15px 10px
font-size: 90%
```



The captions are now indented on the left and right and exhibit more space at the top and bottom. Other custom styling may be needed as you create more pages, but for now you're finished.

. Save all files.

. Choose File > Close All.

Dreamweaver's templates help you build and automatically update pages quickly and easily. In the upcoming lessons, you will use the newly completed template to create files for the project site. Although choosing to use templates is a decision you should make when first creating a new site, it's never too late to use them to speed up your workflow and make site maintenance faster and easier.

Review questions

1. How do you create a template from an existing page?
2. Why is a template “dynamic”?
3. What must you add to a template to make it useful in a workflow?
4. How do you create a child page from a template?
5. Can templates update pages that are open?

Review answers

1. To create a .dwt file, choose File > Save As Template and enter the name of the template in the dialog.
2. A template is dynamic because Dreamweaver maintains a connection to all pages created from it within a site. When the template is updated, it passes any changes to the locked areas of the child pages and leaves the editable regions unaltered.
3. You must add editable regions to the template; otherwise, unique content can’t be added to the child pages.
4. Choose File > New, and in the New Document dialog, select Site Templates. Locate the desired template, and click Create. Or right-click the template name in the Assets > Template category, and choose New From Template.
5. Yes. Open pages based on the template are updated along with files that are closed. The only difference is that files that are open are not automatically saved after being updated.

7 Working with Text, Lists, and Tables

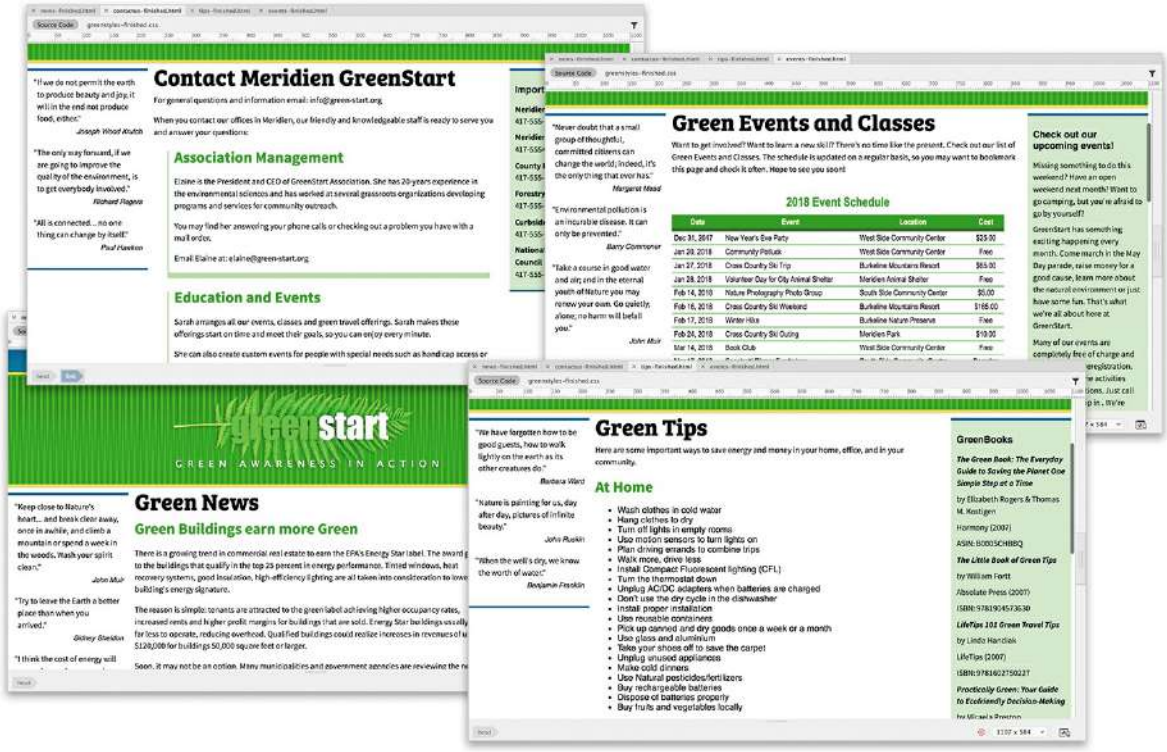
Lesson overview

In this lesson, you'll create several webpages from your new template and work with headings, paragraphs, and other text elements to do the following:

- Enter heading and paragraph text
- Insert text from another source
- Create bulleted lists
- Create indented text
- Insert and modify tables
- Spellcheck your website
- Search and replace text

This lesson will take about 3 hours to complete. If you have not already done so, please log in to your account on peachpit.com to download the project files for this lesson as described in the "Getting Started" section at the beginning of this book and follow the instructions under "Accessing the Lesson Files and Web Edition." Define a site based on the lesson07 folder.

Your Account page is also where you'll find any updates to the lessons or to the lesson files. Look on the Lesson & Update Files tab to access the most current content.



Dreamweaver provides numerous tools for creating, editing, and formatting web content, whether it's created within the program or imported from other applications.

Previewing the completed file

To get a sense of the files you will work on in this lesson, let's preview the completed pages in Dreamweaver.

. Launch Adobe Dreamweaver CC (2019 release) or later, if necessary. If Dreamweaver is already running, close any open files.

. Define a new site for the lesson07 folder, as described in the "Getting Started" section at the beginning of the book. Name the new site **lesson07.**

. If necessary, press F8 to open the Files panel.

Select lesson07 from the site drop-down list.

Dreamweaver allows you to open one or more files at the same time.

. Open the lesson07/finished-files folder.

. Select **contactus-finished.html**.

Hold Ctrl/Cmd, and then select **events-finished.html**, **news-finished.html**, and **tips-finished.html**.

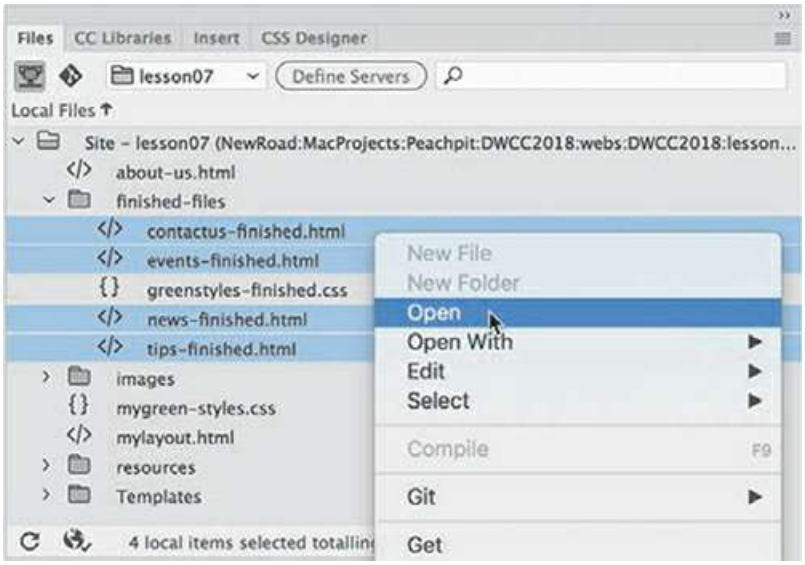
By holding Ctrl/Cmd before you click, you can select multiple non-consecutive files.

. Right-click any of the selected files.

Choose Open from the context menu.

Note

To open consecutive files, hold the Shift key before selecting. If the files are not listed consecutively, use the Ctrl/Cmd key to select the files.



All four files open. Tabs at the top of the document window identify each file.

Note

Be sure to use Live view to preview each of the pages.

. Click the **news-finished.html** tab to bring that file to the top, and switch to Live view if necessary.

Note the headings and text elements used.

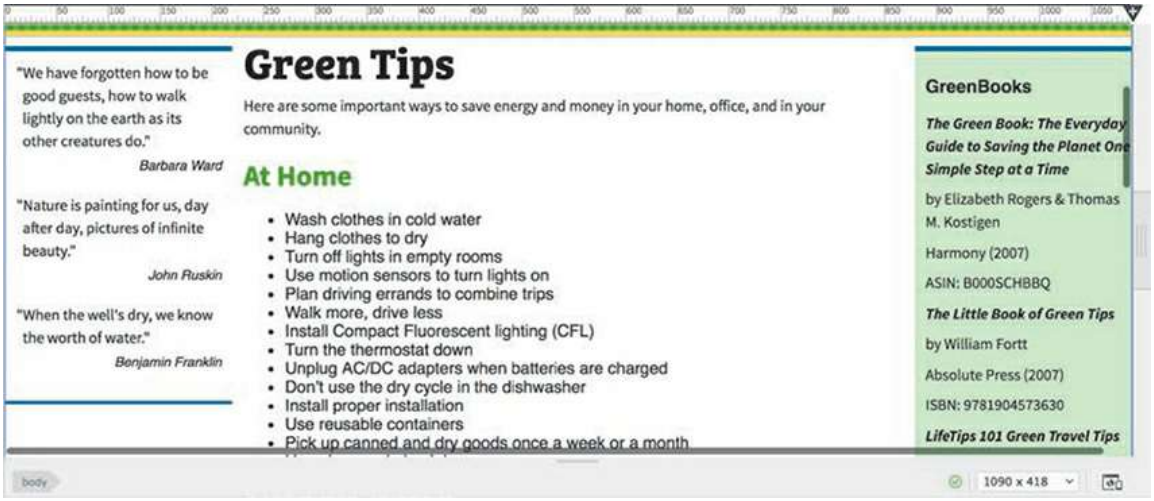
. Click the **events-finished.html** document tab to bring that file to the top.

Note the two HTML-based tables used.

. Click the **contactus-finished.html** tab to bring that file to the top.

Note that the text elements are indented and formatted.

. Click the **tips-finished.html** tab to bring that file to the top.



Note the bulleted list elements used.

. Choose File > Close All.

In each of the pages, a variety of elements are used, including headings, paragraphs, lists, bullets, indented text, and tables. In the following exercises, you will create these pages and learn how to format each of these elements.

Creating and styling text

Most websites are composed of large blocks of text with a few images sprinkled in for visual interest. Dreamweaver provides a variety of means for creating, importing, and styling text to meet any need.

Importing text

In this exercise, you'll create a new page from the site template and then insert heading and paragraph text from a text document.

Tip

The Assets panel may open as a separate, floating panel. To save screen space, feel free to dock the panel on the right side of the screen, as shown in Lesson 1, "Customizing Your Workspace."

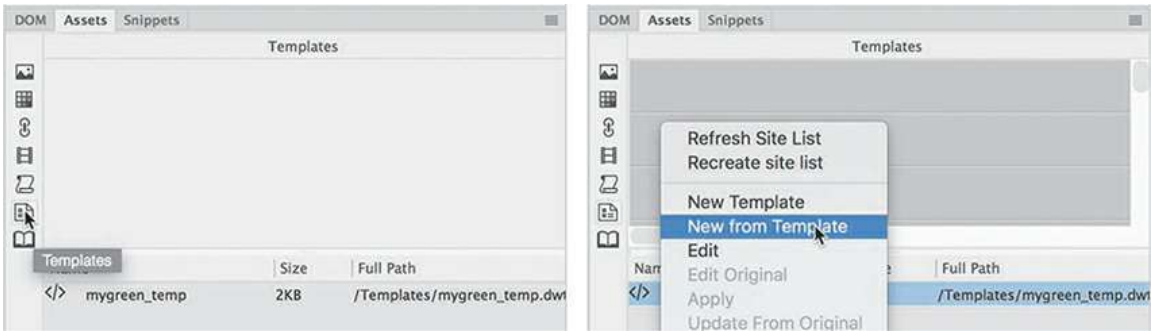
Note

The Templates tab of the Asset panel appears only in Design and Code views when documents are open. You will also be able to see it and select a template when no document is open.

. Choose Window > Assets to display the Assets panel.

Select the Templates category icon.

Right-click **mygreen_temp** and choose New From Template from the context menu.



A new page is created based on the site template.

. Save the file as **news.html** in the site root folder.

When you create a file, it's a good idea to immediately update or replace the various metadata placeholder text elements in the new page. These items are often overlooked or forgotten in all the hubbub around creating the text and images for the main content. First, you'll update the page title.

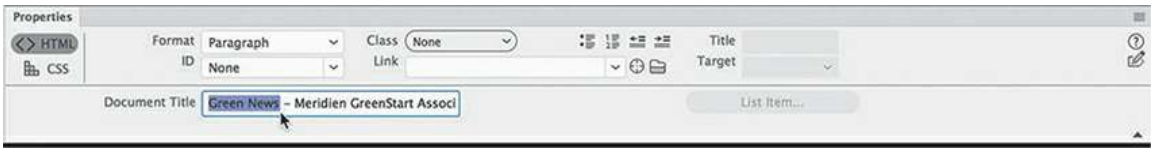
Tip

The Property inspector may not be visible in the default workspace. You can access it in the Window menu and dock it to the bottom of the document window

---

- If necessary, choose Window > Properties to display the Property inspector.
- In the Document Title field, select the placeholder text *Add Title Here*.

  Type **Green News** and press Enter/Return to complete the title.



Each page also has a meta description element, which provides valuable information about your page content to search engines. You'll have to edit it in Code view.

- Switch to Code view.

  The meta description should appear around line 13.

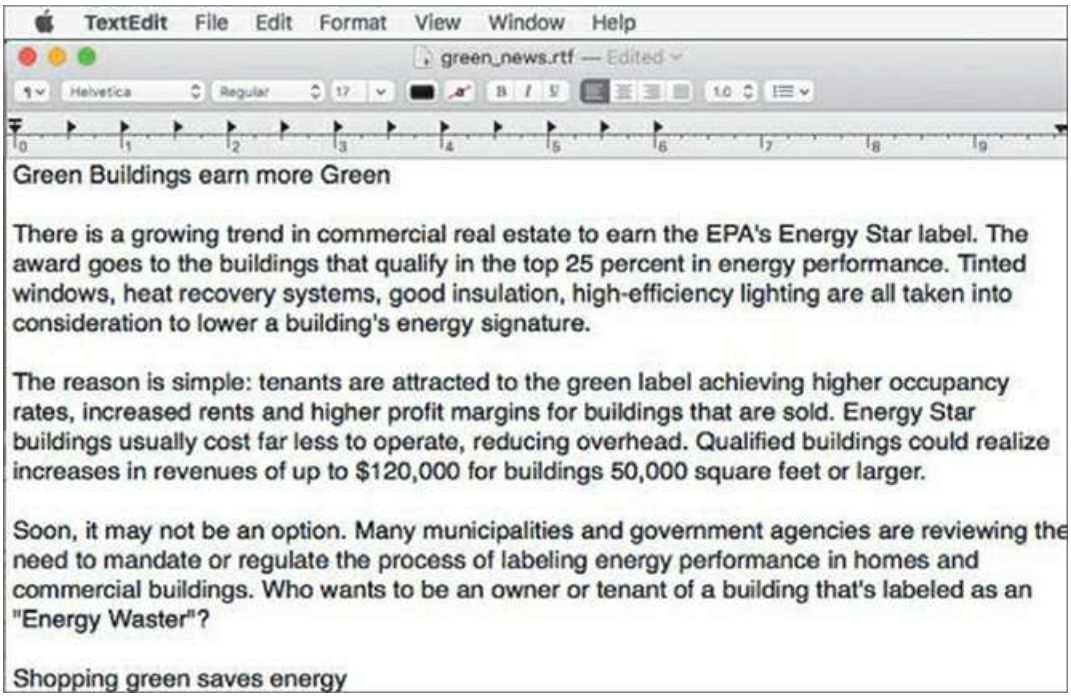
- Scroll to the editable region in the `<head>` section, around line 13.
- Select the text *add description here* and type the following:

  Read the latest eco-news and commentary for and about Meridien

```
/mygreen-styles.css" rel="stylesheet" type="text/css">
12  <!-- InstanceBeginEditable name="head" -->
13  <meta name="description" content="Meridien GreenStart Association - Read the
    latest eco-news and commentary for and about Meridien">
14  <!-- InstanceEndEditable -->
15  </head>
```

Once the metadata is updated, you can start working on the main content.

- In the Files panel, double-click **green-news.rtf** in the lesson07/resources folder.



Dreamweaver automatically launches a program compatible to the file type selected. The text is unformatted and features extra lines between each paragraph. These extra lines are intentional. For some reason, Dreamweaver swaps out single paragraph returns for `<br>` tags when you copy and paste them from another program. Adding a second paragraph return forces Dreamweaver to use paragraph tags instead of the break tag.

This file contains four news stories. After you move the stories to the webpage, you will create semantic structures, as you did for the quotation placeholders.

As explained earlier, semantic web design attempts to provide a context for your web content so that it will be easier for users and web applications to find the information and reuse it.

- In the text editor or word-processing program, position the cursor before the text *Green Buildings earn more Green*.

Tip

When you use the clipboard to bring text into Dreamweaver from other programs, you can then use Live or Design view if you want to honor the paragraph returns.

- Drag to select the next four paragraphs, ending with the text *"Energy Waster"?*

  You have selected the first "article" in the news content.

- Press Ctrl+X/Cmd+X to cut the text.
- Switch back to Dreamweaver.

- Switch to Live view, if necessary.

  The sample page has a page title and one `<article>` element placeholder.

  The content you cut from **green-news.rtf** will be inserted into the existing placeholder.
- Double-click to edit the text.

  Select *Add main heading here.*

  Type **Green News** to replace it.

Tip

Remember that you have to double-click an element in Live view to enter editing mode.

- Select and delete the heading element

  Add article heading here.
- *Select the* `<p>` element *Add content here.*

  The Element Display appears focused on the `<p>` element.
- Press Ctrl+V/Cmd+V to paste the text from the clipboard.

Tip

Remember this technique when you want to paste multiple paragraphs into Live view.

The text from **green-news.rtf** appears in the layout below the placeholder text, preserving the various paragraph elements. Notice that the placeholder element was not replaced. In Live view, the selected element is not replaced when pasting new content. You no longer need the text placeholder.

- Select and delete the entire line *Add content here.*
- Save the file.

Although human visitors may be able to distinguish where one story ends and another begins, adding a semantic structure to your content makes it easier for search engines and assistive devices to derive sense from the content. Adding semantic structures should be your goal whenever possible. This is encouraged not only to support accessibility standards but also to improve your SEO ranking.

Creating semantic text structures

In this exercise, you will insert the remaining content and create HTML5 `<section>` elements to help define the individual news stories.

- If necessary, open **news.html** in Live view and **green-news.rtf** in your text editor.

  The current document has one `<article>` containing a single news story. Often it's easier to add the content before creating the semantic structures.

- In **green-news.rtf** press Ctrl+A/Cmd+A to select the remaining text.

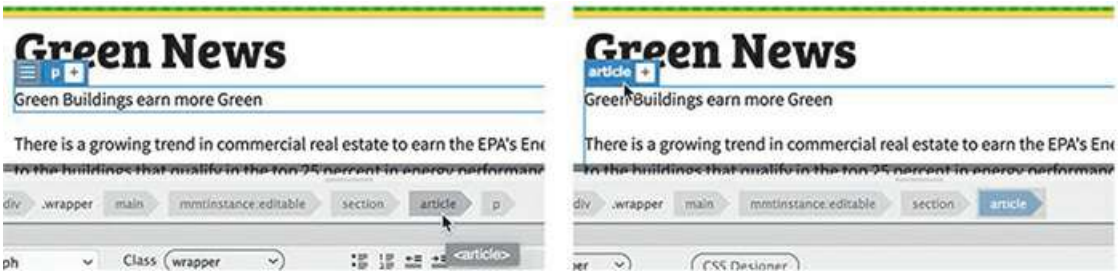
  All the text in the document is selected.

- Press Ctrl+X/Cmd+X to cut the text.
- Close **green-news.rtf**. Do not save any changes.
- In Dreamweaver, select the text element *Green Buildings earn more Green.*

  The Element Display appears focused on the `<p>` element. If you examine the tag selectors, you will see that `article` is the parent of the selected element. The new text should be pasted after this element. To change the focus of the Element Display, you can press the up or

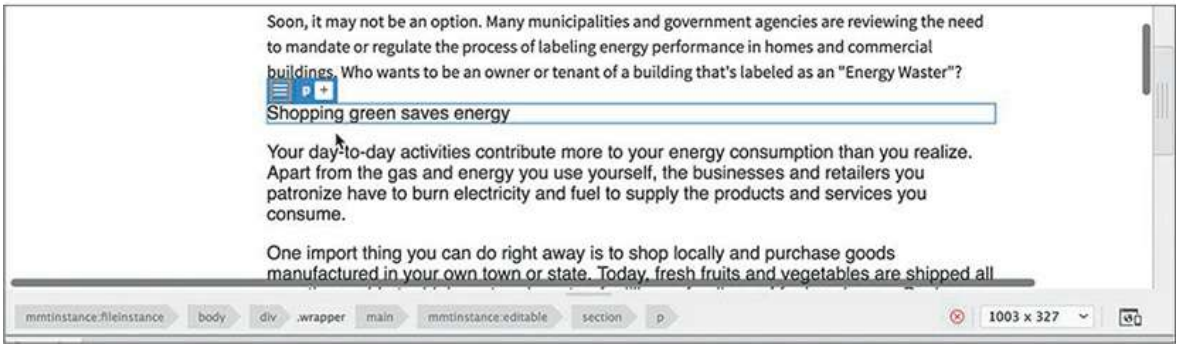
down arrows on your keyboard.

. Press the up arrow on your keyboard.



The Element Display is now focused on the `article` element.

. Press Ctrl+V/Cmd+V to paste the remaining news stories.



The news stories appear below and outside of the `article` element. You can see by the difference in styling that the CSS is not being applied properly to these new paragraphs. This will be rectified as soon as you add the proper HTML structure.

Using the Quick Tag Editor

There are several methods for adding semantic structure. In this exercise, you'll use the Quick Tag Editor.

Note

At the time of this writing, only the Quick Tag Editor allows you to wrap Live view selections.

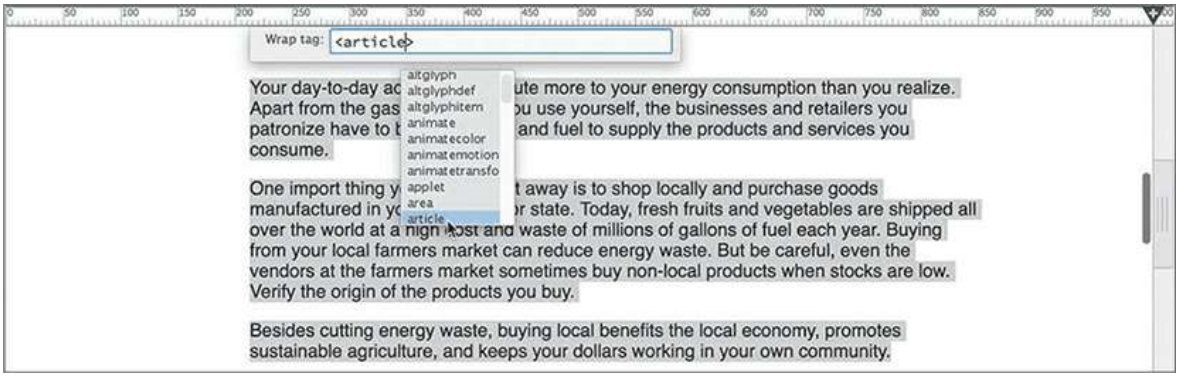
. Starting at the text *Shopping green saves energy*, drag to select the next four paragraphs, ending with the text *in your own community.*

The second news story is selected and highlighted in blue.

- Press Ctrl+T/Cmd+T to open the Quick Tag Editor.

  The Quick Tag Editor appears in Wrap mode. You can now type any tag name to wrap the selection.

- Type **article** and press Enter/Return twice to create the new element.



The new element appears in the tag selector interface wrapping the first news story. You can also use the DOM panel to create semantic structures.

Using the DOM panel

In this exercise, you'll use the DOM panel to add semantic structure.

- Choose Window > DOM to display the DOM panel, if necessary.
- In Live view, click the text *Recycling isn't always green*.

  The Element Display appears in Live view focused on the `p` element. In the DOM panel, the `p` element is also highlighted.

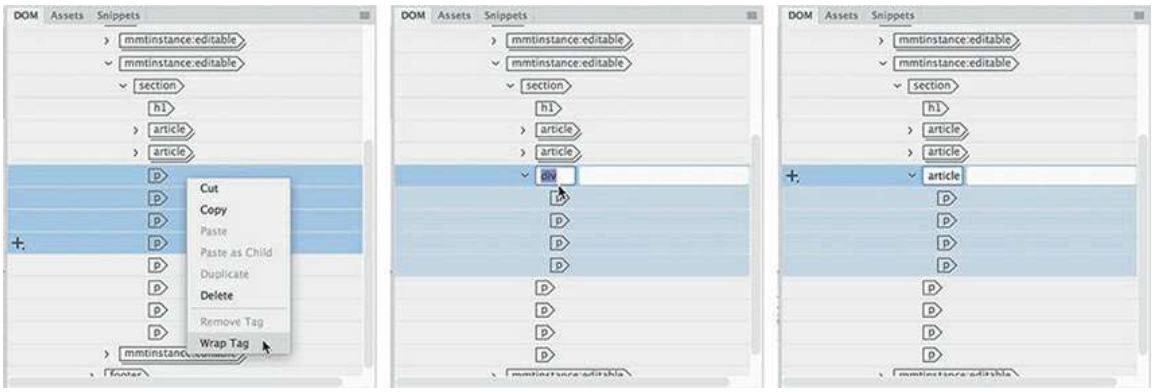
- Holding the Shift key, select the next three `p` elements in the DOM panel.

  All four paragraphs are selected.

- Right-click the selection and choose **Wrap Tag** from the context menu.

  Enter **article** in the element field.

  Press Enter/Return as necessary to complete the new element.



- Using the Quick Tag Editor or the DOM panel to wrap the remaining four paragraphs in a new `<article>` element.

When you're finished, you should have four `<article>` elements, one for each news story.

- Save **news.html**.

Each news story has its own heading, but they are currently formatted as paragraph elements. In the next exercise, you'll apply the proper tag to them.

Creating headings

In HTML, the tags `<h1>, <h2>, <h3>, <h4>, <h5>,` and `<h6>` create headings. Any browsing device, whether it is a computer, a Braille reader, or a cellphone, interprets text formatted with any of these tags as a heading. On the web, headings introduce distinct sections with helpful titles, just as they do in books, magazine articles, and even term papers.

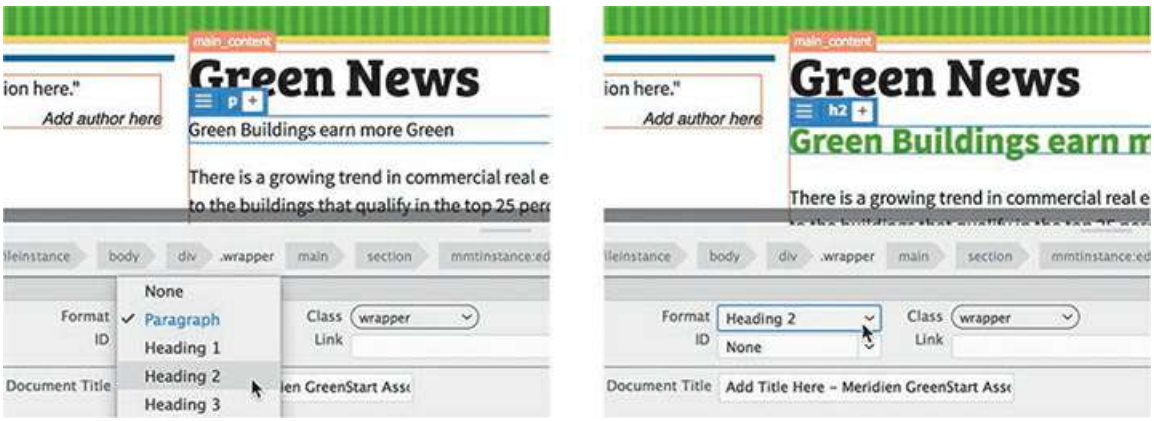
You are using one `<h1>` element per page as the primary page title. Typically, any other headings used on the page should descend in order from the `<h1>`. Since each news story has equal importance, they can all begin with a second-level heading, or `<h2>`. At the moment, all the pasted text is formatted as `<p>` elements. Let's format the story headings as `<h2>` elements.

Tip

If the Format menu is not visible, select the HTML mode of the Property inspector.

. In Live view, select the text *Green Buildings earn more Green*.

  Choose **Heading 2** from the Format menu in the Properties panel.

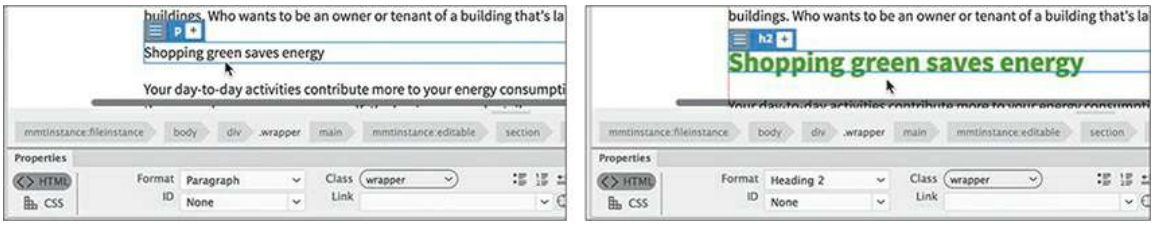


  The text is formatted as an `<h2>` element. Dreamweaver also provides a keyboard shortcut for this operation.

. Click the text *Shopping green saves energy*.

  The Element Display appears focused on the `p` element.

. Press Ctrl+2/Cmd+2 or select Heading 2 from the Format dropdown menu.



  The text is now formatted as an `<h2>` element.

. Format the paragraphs *Recycling isn't always Green* and *Fireplace: Fun or Folly?* as `h2` headings.

  All the news stories are properly structured and formatted.

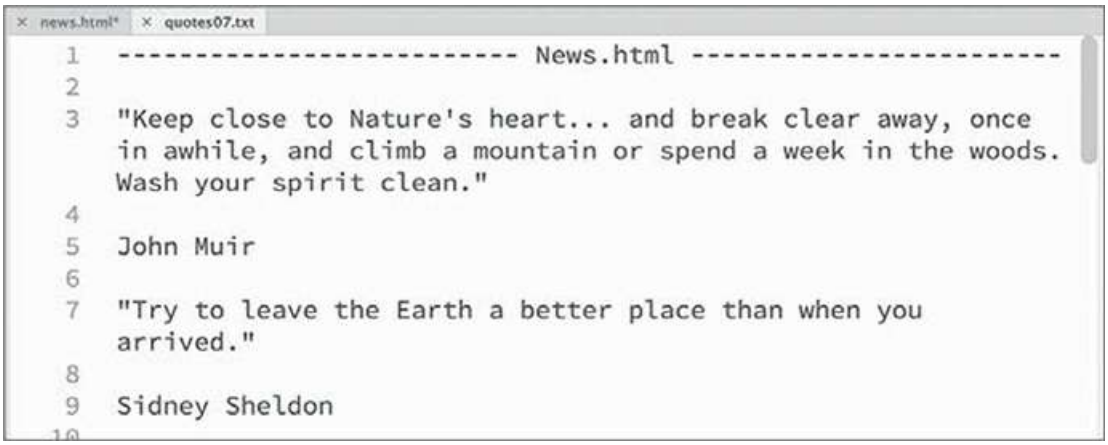
. Save all files.

Adding other HTML structures

Descendant selectors are often sufficient for styling most elements and structures in a webpage. But not all the structural elements are available from the Insert menu or panel. In this exercise,

you will learn how to build a custom HTML structure for a quotation and an attribution using the Quick Tag Editor.

- Open **news.html** in Live view, if necessary.
- In the Files panel, open **quotes07.txt** from the lesson07 resources folder.



Since this is a plain-text file, Dreamweaver can open it. The file contains quotations you will insert in the various pages that will be created in this lesson.

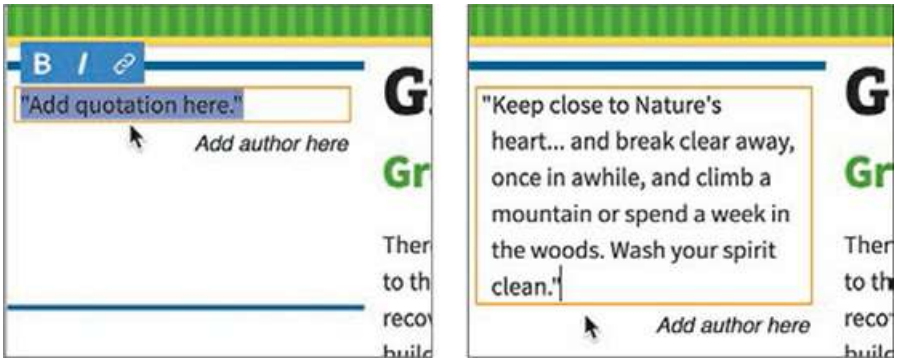
- Select the text of the first quotation, excluding the author name.

  Press Ctrl+X/Cmd+X to cut the text.
- Switch to **news.html**. Select the first quotation placeholder.

  Press Ctrl+V/Cmd+V.

Note

Remember to double-click the placeholder to open the orange editing box.



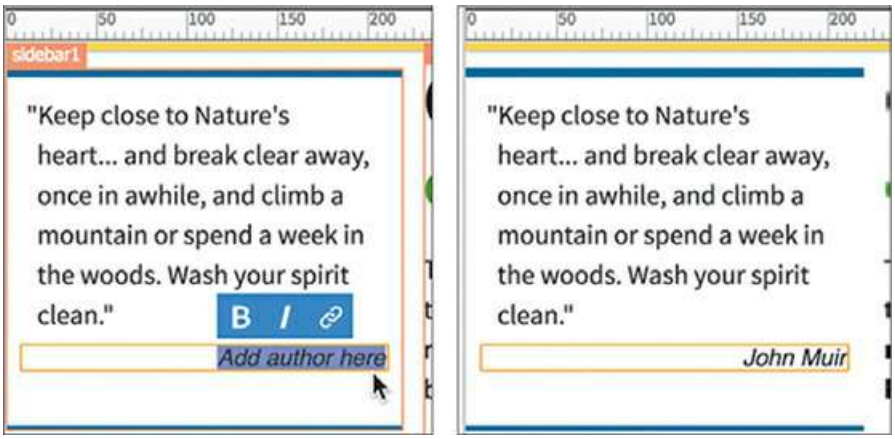
The quotation has replaced the placeholder.

- Switch to **quotes07.txt**.

Select and cut the author name, *John Muir*.

- Switch to **news.html**.

Select the *Add author here* placeholder text and paste the text.



John Muir replaces the author name placeholder.

Since there was only one quotation placeholder in the template, you'll have to create the other quotation structures from scratch.

- Switch to **quotes07.txt**.

Select and cut the next two quotations and authors.

- In **news.html**, click to select the first quotation.

Click the `blockquote` tag selector.

- Press Ctrl+V/Cmd+V to paste the new quotations and authors.

The text appears inserted in the `<aside>` element but after the `blockquote`. The new text is not styled properly.

- Using the cursor and the tag selector interface, compare the structure of the three quotations and note the differences.

The new quotations and author names appear in two separate p elements. Part of the styling problem is caused by the missing `blockquote` element. Dreamweaver has no menu option for adding this specific tag, but you can use the Quick Tag Editor in a pinch to build all types of custom structures.

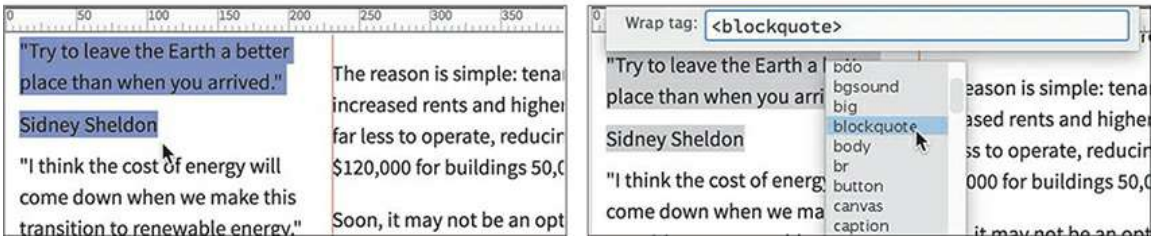
. Drag to select the text for the second quotation, including the author name, *Sidney Sheldon*. Press Ctrl+T/Cmd+T.

The Quick Tag Editor appears. Since you have more than one element selected, it should default to Wrap mode.

Tip

Press Ctrl+T/Cmd+T to toggle between modes in the Quick Tag Editor, if necessary.

. Type **`blockquote`** and press Enter/Return twice to add the element as a parent to the two paragraphs.



The quotation text is now formatted properly, but the author name needs one more tweak: You need to change the tag applied to it. As with `blockquote`, there's no menu option for the `<cite>` tag.

. Insert the cursor in the author name, *Sidney Sheldon*.

Select the tag selector for the `<p>` element.

Press Ctrl+T/Cmd+T.

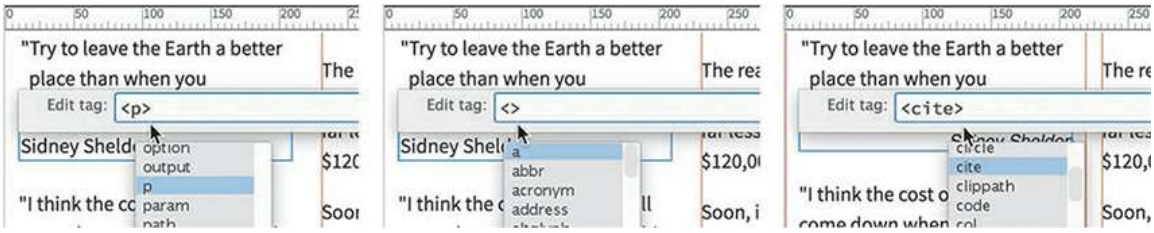
The Quick Tag Editor appears. Since you have only one element selected, it should default to Edit mode. If the correct mode is not visible, press Ctrl+T/Cmd+T until it is.

. Press the Backspace key to delete "`p`" from the selected tag.

Type **`cite`** and press Enter/Return twice to complete the change.

Note

In HTML5, the `cite` element is used to identify the attribution of a quotation.



The author name now appears in a `<cite>` element and is styled identically to the other author.

. Repeat steps 11 through 14 to create the `blockquote` and `cite` structure for the third quotation in **news.html**.

All three quotations are now structured properly in the first column.

. Save and close **news.html**.

. Close **quotes07.txt**. Do not save changes.

Closing the text file without saving the changes will preserve the original content in case you want to repeat this exercise later.

Creating lists

Formatting should add meaning, organization, and clarity to your content. One method of doing this is to use the HTML list elements. Lists are the workhorses of the web because they are easier to read than blocks of dense text; they also help users find information quickly.

In this exercise, you will learn how to make an HTML list.

Note

The Template category is not visible in Live view. To create, edit, or use Dreamweaver templates, you must switch to Design view or Code view or close all open HTML documents.

- Choose Window > Assets to bring the Assets panel to the front.

  In the Template category, right-click **mygreen_temp**.

  From the context menu, choose New From Template.

  A new page is created based on the template.
- Save the file as **tips.html** in the site root folder. Switch to Live view, if necessary.
- In the Property inspector, select the placeholder text *Add Title Here* in the Document Title field. Type **Green Tips** to replace the text and press Enter/Return.
- Switch to Code view. Locate the meta description element.

  Select the text *add description here.*
- Type **Learn the best eco-tips for your home, office, and your community** and save the file.

```
12  <!-- InstanceBeginEditable name="head" -->
13  <meta name="description" content="Meridien GreenStart
    Association - Learn the best eco-tips for your home, office,
    and your community">
14  <!-- InstanceEndEditable -->
```

  The new description replaces the placeholder.
- In the Files panel, double-click **green-tips.rtf** in the resources folder of lesson07.

  The file will open outside Dreamweaver. The content consists of three individual lists of tips on how to save energy and money at home, at work, and in the community. As in the news page, you will insert each list into its own `<section>` element.
- In **green-tips.rtf**, press Ctrl+A/Cmd+A.

  Press Ctrl+X/Cmd+X to cut the text.

  Close, but do not save changes to, **green-tips.rtf**.

  You have selected and cut all the text.
- Switch back to Dreamweaver. Switch to Live view.
- Select *Add main heading here.*

  Type **Green Tips** to replace it.
- Select and delete the entire `<h2>` element

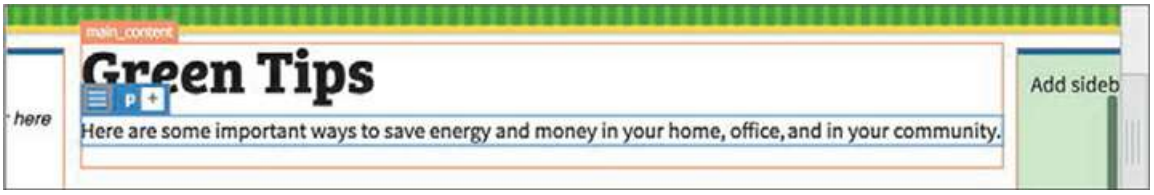
  Add article heading here.

Note

When removing the placeholder text, be sure to delete the HTML tags too. The best way to select and delete entire elements is by using the tag selectors.

. Double-click to edit the text *Add content here*.

Type **Here are some important ways to save energy and money in your home, office, and in your community.**



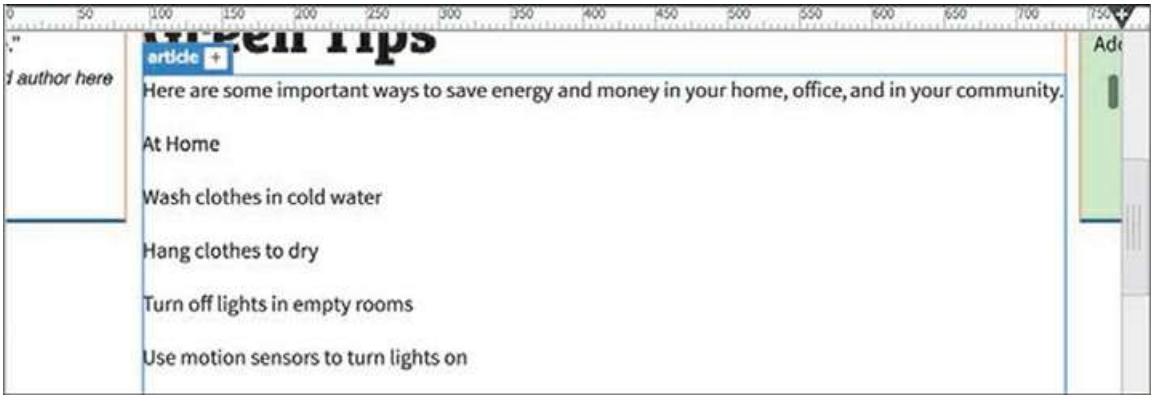
The new text replaces the placeholder.

. Click outside the orange editing box.

The orange box disappears.

. Click to select the new paragraph.

Press Ctrl+V/Cmd+V.

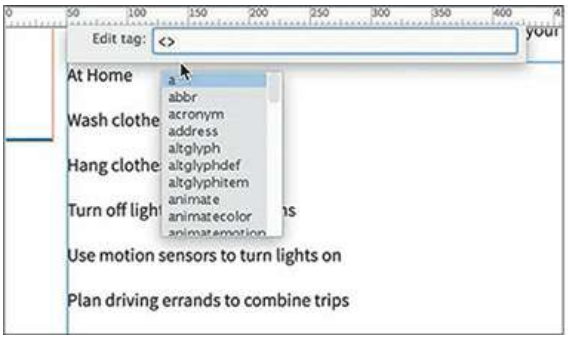


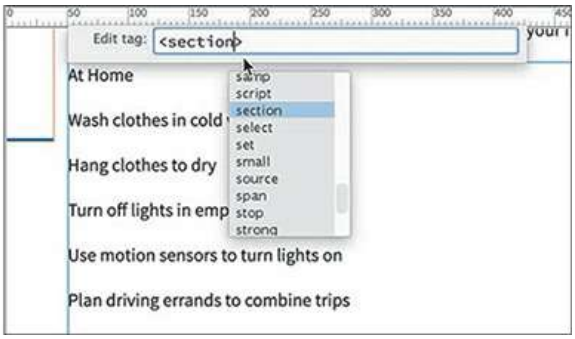
The text for all three lists appears.

. Drag to select the text starting at *At Home* and ending with *Buy fruits and vegetables locally*.

. Press Ctrl+T/Cmd+T.

Type **`section`** and press Enter/Return twice.





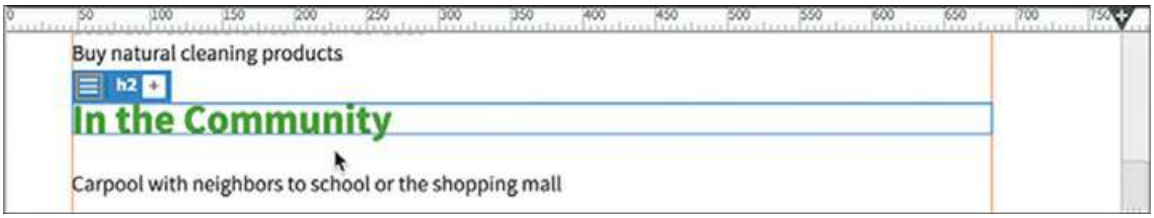
The `<section>` element appears, wrapping the first list.

- Select the text starting at *At Work* and ending with *Buy natural cleaning products.*
- Wrap the selection with a `section` element, as in step 15.
- In Dreamweaver, repeat steps 14 and 15 to create the third list and `section` structure with the remaining text.

  All three lists now appear in their own `<section>` elements.

  As you did with the titles of the news stories, apply HTML headings to introduce the list categories.

- Apply `<h2>` formatting to the text *At Home*, *At Work*, and *In the Community*.



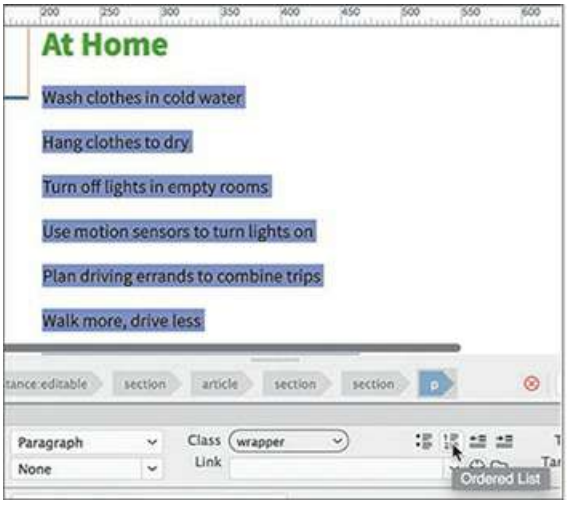
The remaining text is currently formatted entirely as HTML paragraphs. Dreamweaver makes it easy to convert this text into an HTML list. Lists come in two flavors: *ordered* and *unordered*.

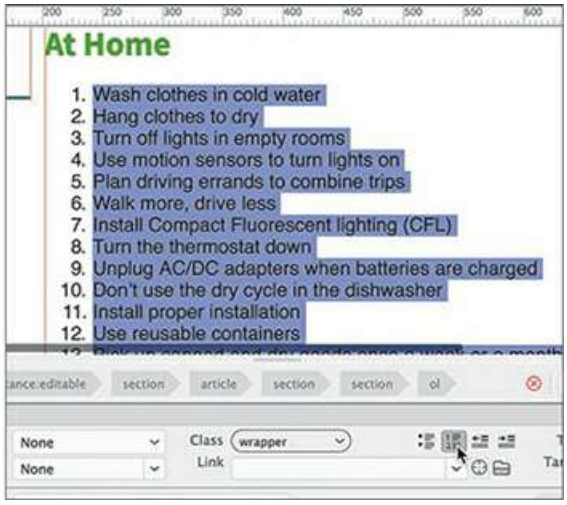
Creating an ordered list

In this exercise, you will convert the paragraph text into an HTML ordered list.

- Select all the `<p>` elements under the heading *At Home*.

  In the Property inspector, click the Ordered List icon.





Tip

The easiest way to select the entire list is to use the <ol> tag selector.

An ordered list adds numbers automatically to the entire selection. Semantically, it prioritizes each item, giving them intrinsic values relative to one another. However, this list doesn't seem to be in any particular order. Each item is more or less equal to the next one, so it's a good candidate for an unordered list—used when the items are in no particular order. Before you change the formatting, let's take a look at the markup.

. Switch to Split view.

Observe the list markup in the Code section of the document window.

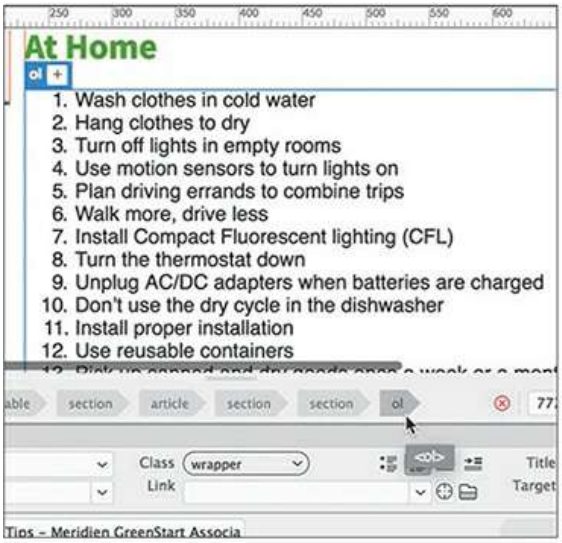
The markup consists of two elements:<ol>and <li>. Note that each line is formatted as an <li> (list item). The <ol> parent element begins and ends the list and designates it as an ordered list. Changing the formatting from numbers to bullets is simple and can be done in Code view or Design view.

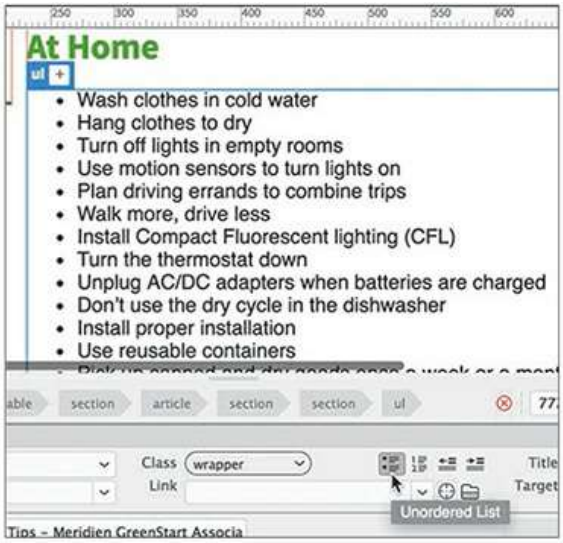
Before changing the format, ensure that the formatted list is still entirely selected. You can use the <ol> tag selector, if necessary.

Creating an unordered list

In this exercise, you will convert the ordered list into an unordered list.

. In the Properties panel, click the Unordered List icon.





All the items are now formatted as bullets.

If you observe the list markup, you'll notice that the only thing that has changed is the parent element. It now says `<ul>`, for *un*ordered list.

Tip

You could also change the formatting by editing the markup manually in the Code view window. But don't forget to change both the opening and closing tags.

- Select all the `<p>` formatted text under the heading *At Work*.
- In the Properties panel, click the Unordered List icon .
- Repeat steps 2 and 3 with all the text following the heading *In the Community*.

All three lists are now formatted with bullets.

- In Dreamweaver, save and close **tips.html**.

Creating indented text

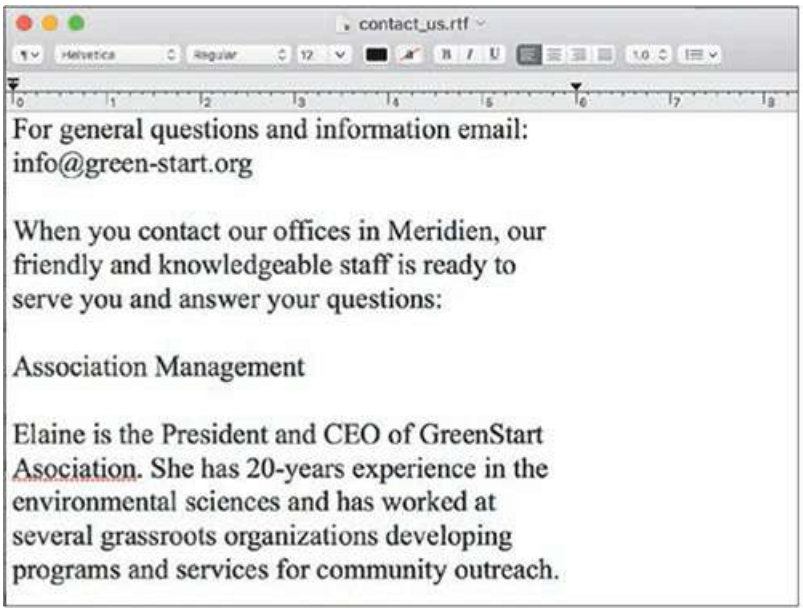
In Sidebar 1, you're using the `<blockquote>` element in the semantically correct way to identify sections of text quoted from other sources. But some designers still use the element as an easy way to indent headings and paragraph text.

Normally, text formatted this way will appear indented and set off from the regular paragraphs. If you want to comply with web standards, you should leave this element for its intended purpose and instead use custom CSS classes to indent text, as you will in this exercise.

- Create a new page from the site template.

Save the file as **contact-us.html** in the site root.

- Switch to Design view, if necessary. In the Property inspector, enter **Contact Meridien GreenStart** to replace the placeholder text *Add Title Here*.
- In Code view, select the meta description placeholder text *add description here* and type **Meet the amazing staff of Meridien GreenStart** to replace it.
- In the Files panel, open **contact-us.rtf** from the lesson07/resources folder.



The text consists of five department sections, including headings, descriptions, and email addresses for the managing staff of GreenStart. You will insert each department into its own `<section>` element.

- In **contact-us.rtf**, select all the text and cut it.

  Close the file and do not save the changes.
- In Dreamweaver, switch to Live view.

  Type **Contact Meridien GreenStart** to replace the placeholder heading *Add main heading here*.
- Select and delete the entire heading

  Add article heading here.
- Select the element *Add content here.*

  Press Ctrl+V/Cmd+V to paste the content.

  All the content cut from **contact-us.rtf** appears directly after the placeholder text. Selecting the placeholder before pasting allows you to insert the text into the existing `article` element.
- Select and delete the text *Add content here.*

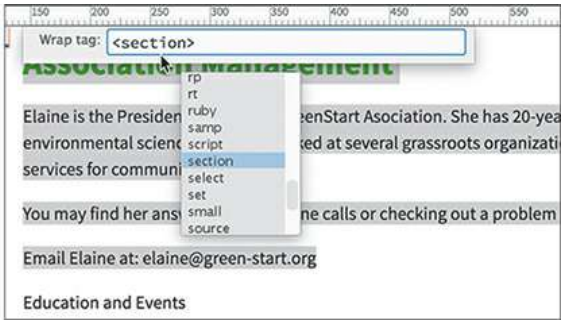
  We no longer need the placeholder text.

- Format the text *Association Management* as a Heading 2.

  As you did with the lists, you need to wrap each content group in its own `<section>` element.

- Drag to select the heading and the rest of the text for this section, including Elaine's email address.
- Press Ctrl+T/Cmd+T.

  Type **`section`** and press Enter/Return twice.



- Repeat steps 10 through 12 to structure and format the content for the *Education and Events*, *Transportation Analysis*, *Research and Development*, and *Information Systems* sections.

  With all the text in place, you're ready to create the indent styling. If you wanted to indent a single paragraph, you would probably create and apply a custom class to the individual `<p>` element. In this instance, you want to indent the entire `<section>` element to produce the desired visual effect.

  To make sure the styling is applied only to these employee profiles, you'll need to create a custom class that you can assign to them and style separately. First, let's create a custom CSS class and add it to the style sheet.

- Click in any element in *Association Management*

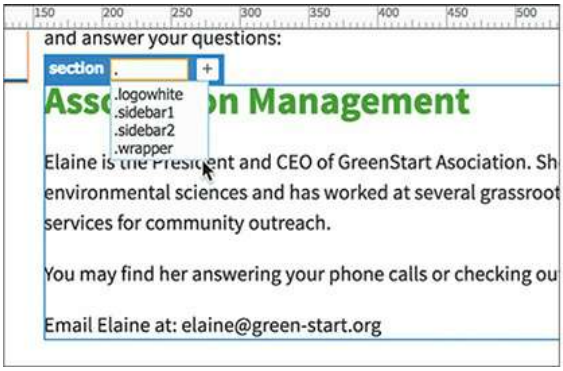
  Click the `<section>` tag selector.

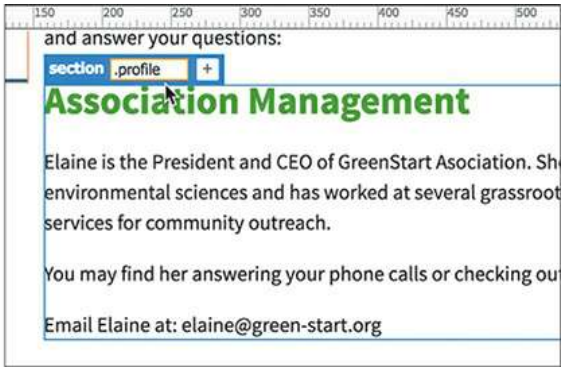
  The Element Display appears focused on the `<section>` tag.

- Click the Add Class/ID icon.
- Type **`.profile`** as the new class name.

Note

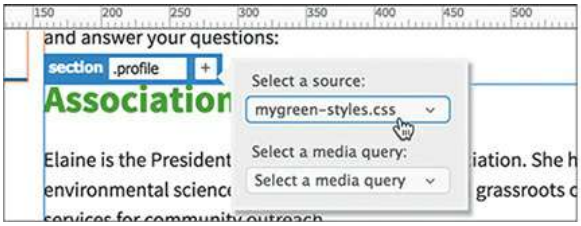
Don't forget to type the period at the beginning of the class name.

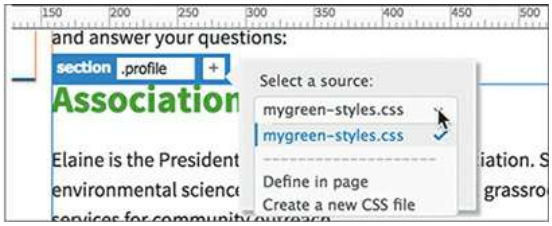




As you type, a hinting list appears and displays the names of existing rules, filtering the list to match the text you're typing. When using this feature, feel free to use the mouse or keyboard to select any name from the list. The class `.profile` doesn't exist yet, but the Element Display enables you to create it on the fly.

. Press Enter/Return once.





The CSS Source pop-up window appears.

Note

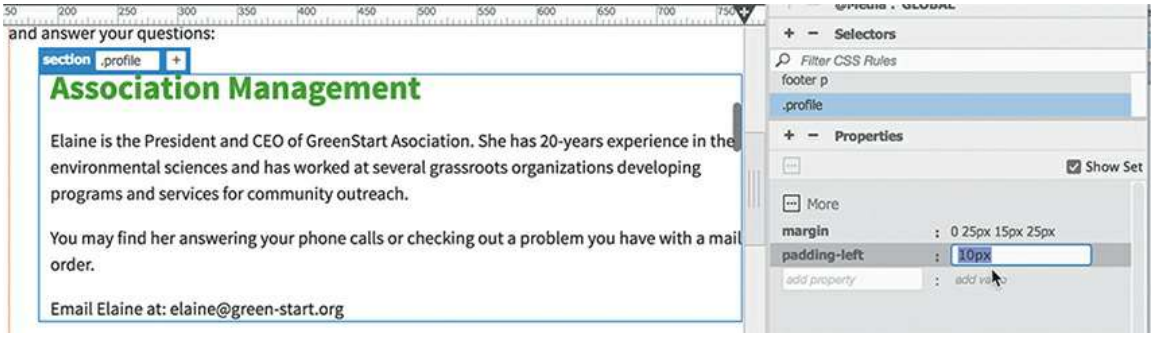
A new rule will normally be added to the end of the style sheet. This may cause conflicts if any media queries are defined. If you notice your new rule is not working, check to see that it appears before the media queries.

Whenever you enter a new class or id in the Element Display that does not exist in a linked or embedded style sheet, the CSS Source pop-up window will appear. This pop-up enables you to create a new matching selector in any style sheet embedded in the file or linked to it. You can even use it to start a new style sheet, if necessary. Since the site template is already linked to an external style sheet, **mygreen-styles.css** appears in the Select A Source drop-down menu.

. Press Enter/Return a second time.

Because you pressed Enter/Return again, the selector .profile is created in the default style sheet. If you do not want to create a selector for the class or id entered, press the Esc key instead. Once the selector is created, you can use it to style the content.

. Display the CSS Designer. Click the Current button.



The class `.profile` appears at the top of the list of selectors. If you look at the Properties pane, you can see that no styles are set.

Tip

When creating specifications manually, enter the property name in the field and press Tab. A value field will appear to the right. When Show Set is enabled, hinting may not appear in the values field.

. Enable the Show Set option, if necessary. Enter the following properties:

Click here to view code image

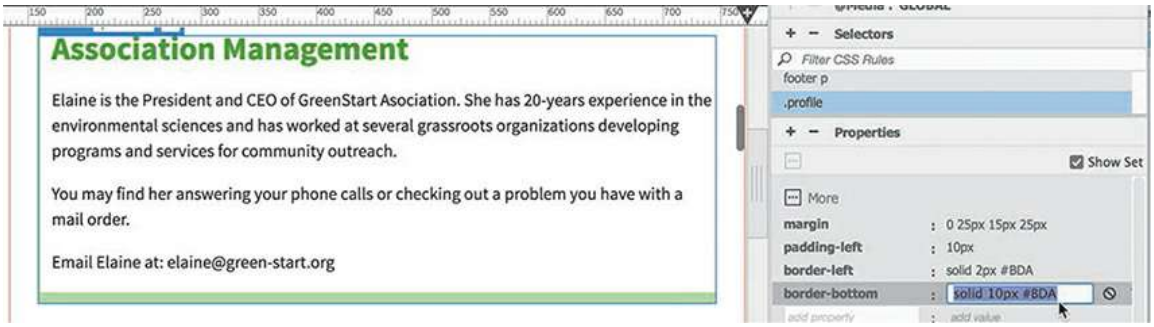
```
margin: 0 25px 15px 25px
padding-left: 10px
```

As with margins, border specifications can be entered using shorthand specifications.

. Enter the following specifications for the left and bottom borders:

Click here to view code image

```
border-left: solid 2px #BDA
border-bottom: solid 10px #BDA
```



The borders appear on the left and bottom of the section element. The borders help to visually group the indented text under its heading. Each department in the organization can now be

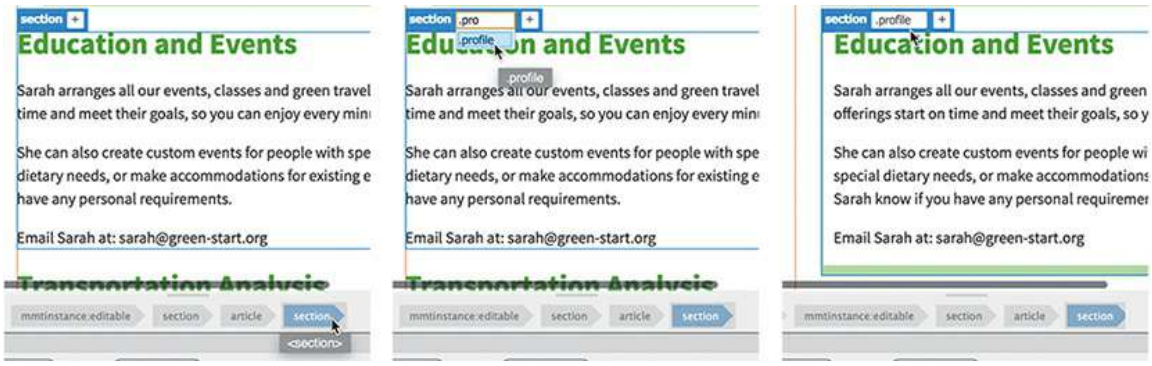
styled the same way.

. Insert the cursor anywhere in the *Education and Events* section.

Click the `<section>` tag selector.

The Element Display appears focused on the `<section>` element.

. In the Element Display, click the Add Class/ID icon and type **`.profile`** in the text field.



As you type, a hinting menu will show the matching class names. Feel free to select the name from the list. As soon as you add the class to the element, the formatting is applied to match the first `<section>`.

. Repeat steps 22 and 23 to apply the profile class to the remaining `<section>` elements.

Each `<section>` is indented and displays the custom border.

. Save and close all files.

Whenever you add new components or styling to a site, you need to make sure the elements and styling work well on all screen sizes and devices.

Creating and styling tables

Before the advent of CSS, HTML tables were often used to create page layouts. At that time it was the only way to create multicolumn layouts and maintain some control over the content elements. But tables proved to be inflexible and hard to adapt to the changing Internet, as well as just being a bad design choice. CSS styling provides so many more options for designing and laying out a webpage that tables were quickly dropped from the designer's toolkit.

That doesn't mean tables are no longer used on the web at all. Although tables are not good for page layout, they are good, and necessary, for displaying many types of data, such as product lists, personnel directories, and timetables, to name a few.

Dreamweaver enables you to create tables from scratch, to copy and paste them from other applications, and to create them instantly from data supplied from other sources, including database and spreadsheet programs such as Microsoft Access or Microsoft Excel.

Creating tables from scratch

In this exercise, you will learn how to create an HTML table.

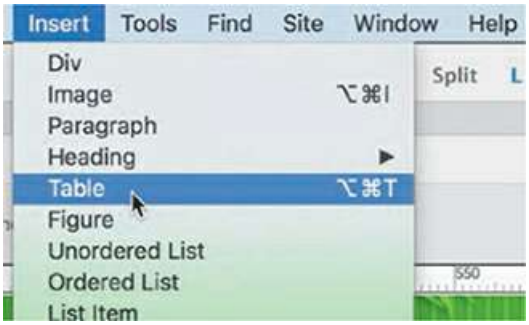
- Create a new page from **mygreen_temp**. Save the file as **events.html** in the site root folder.
- Enter **Green Events and Classes** to replace the *Title* placeholder text in the Property inspector.
- Select the meta description placeholder and type **Meridien GreenStart hosts and sponsors a variety of eco events and classes for anyone interested in learning more about the environment or their community** to replace it.

- Switch to Live view, select the *Add main heading here* placeholder heading, and type **Green Events and Classes** to replace it.
- Delete the *Add article heading here* placeholder.
- Select the text *Add content here*.
- Type the following text: **Want to get involved? Want to learn a new skill? There's no time like the present. Check out our list of Green Events and Classes. The schedule is updated on a regular basis, so you may want to bookmark this page and check it often. Hope to see you soon!**
- Click outside the orange editing box.

  The orange box closes, completing the paragraph.
- Click the edited paragraph to select it.

  Choose Insert > Table.

  The Position Assist dialog appears.



. Select After.

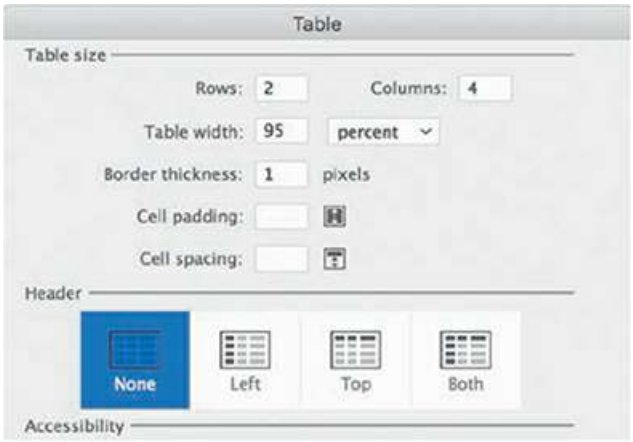
The Table dialog appears.

Although CSS has taken over most of the design tasks formerly done by HTML attributes, some aspects of the table may still be controlled and formatted by those attributes. The only advantage HTML has is that its attributes continue to be well supported by all popular browsers, both old and new. When you enter values in this dialog, Dreamweaver still applies them via HTML attributes. But whenever you have a choice, avoid using HTML to format tables.

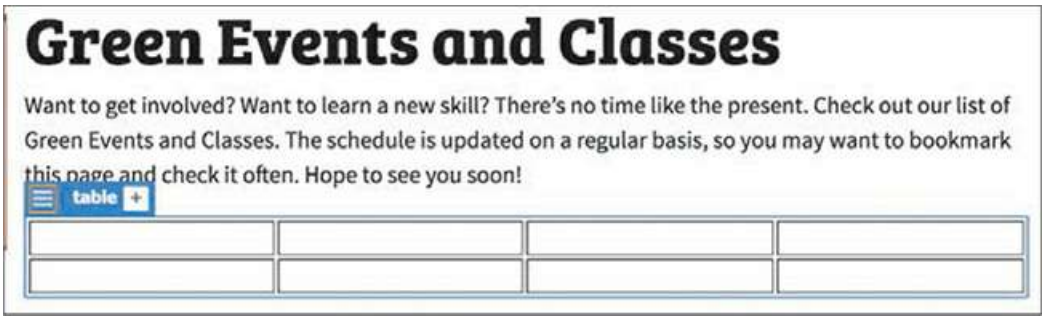
Note

Tables may not be visible in Live view when Border Thickness is set to 0 (zero). At the end of the exercise you will set the border to 0 (zero) pixels.

. Enter the following specifications for the table: Rows: **2** Columns: **4** Table Width: **95%** Border Thickness: **1.**



. Click OK to create the table.

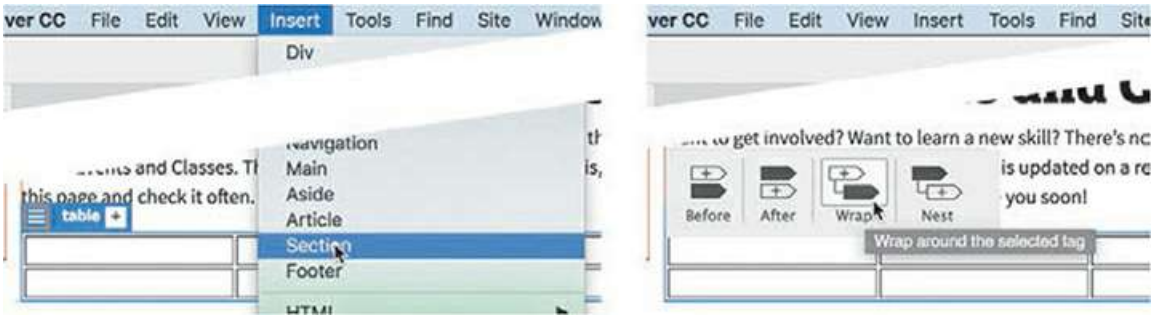


A four-column, two-row table appears below the main heading. Note that it fills the column from left to right. Let's wrap it in a `<section>` element.

. Select the `table` tag selector. Select Insert > Section.

The Position Assist dialog appears.

. Select Wrap.



The table is wrapped in a `<section>` element. The table is ready to accept input, but Live view is not optimized for data entry. If you have large amounts of data to enter, you're better off using Design view.

Adding data to a table

In this exercise, you'll learn how to add data to a table manually.

Tip

When your cursor is in a table cell in Design view, pressing the Tab key moves the cursor to the next cell on the right. Hold the Shift key before pressing the Tab key to move to the left, or backward, through the table.

. Switch to Design view.

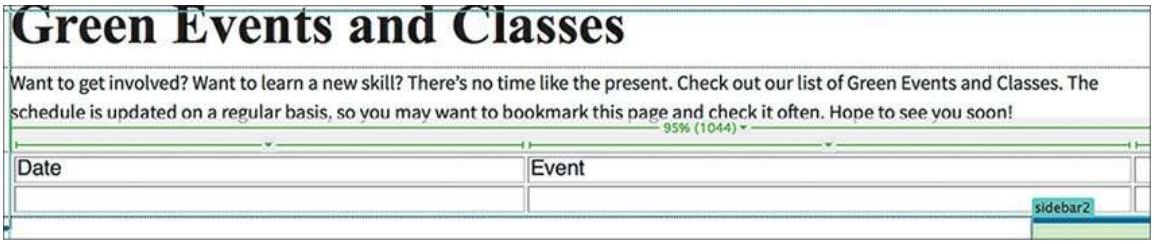
. Insert the cursor in the first cell of the table.

Type **Date** and press the Tab key.

The cursor moves into the next cell of the same row.

- In the second cell, type **Event** and press Tab.

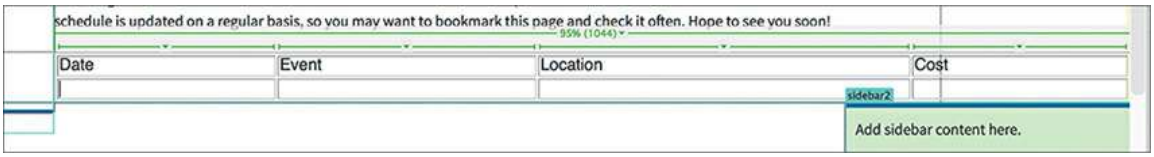
> **Note**
>
> Design view does not display complex CSS styling properly. The sidebars may overlap the tables. If the preview is too hard to work with, try adjusting the width of the document window.



The text appears and the cursor moves to the next cell, but you may find it hard to see it. In some cases, you may need to adjust the size of the document window.

- Type **Location** and press Tab.

  Type **Cost** and press Tab.



The cursor moves to the first cell of the second row.

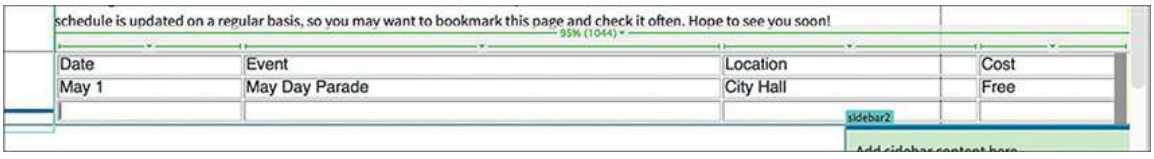
- In the second row, type **May 1** (in cell 1), **May Day Parade** (in cell 2), **City Hall** (in cell 3), and **Free** (in cell 4).

When the cursor is in the last cell, inserting additional rows in the table is easy.

Adding rows to an existing table

Dreamweaver provides several ways to add rows and columns to an existing table. In this exercise, you will learn how to add rows to a table.

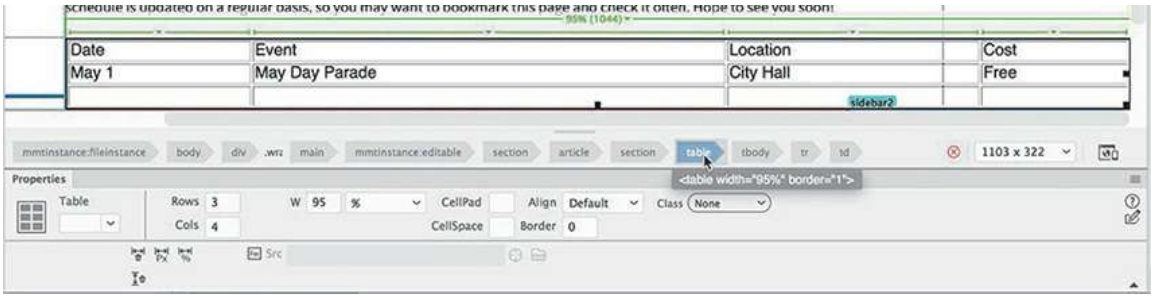
- Press Tab.



A new blank row appears at the bottom of the table. Dreamweaver also allows you to insert multiple new rows at once.

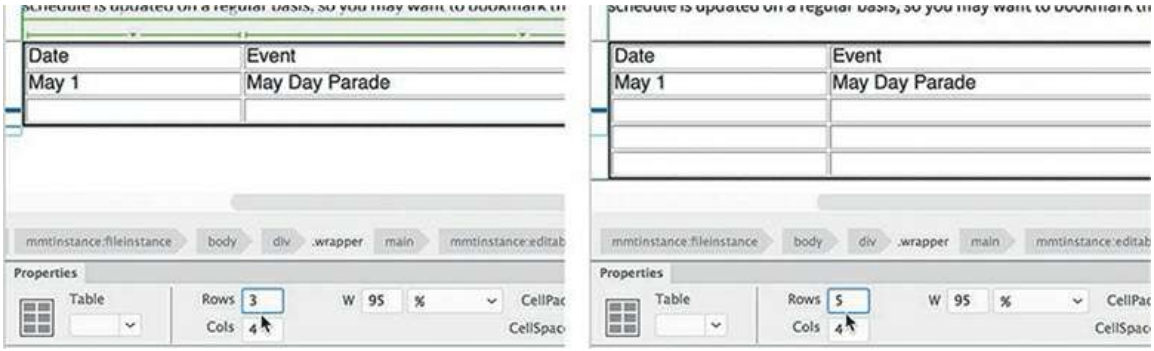
- Select the `<table>` tag selector at the bottom of the document window.

> **Tip**
>
> If the Properties panel is not visible, select Window > Properties. Dock the panel to the bottom of the document window.



The Properties panel fields create HTML attributes to control various aspects of the table, including table width, cell width and height, text alignment, and so on. It also displays the current number of rows and columns and even allows you to change the number.

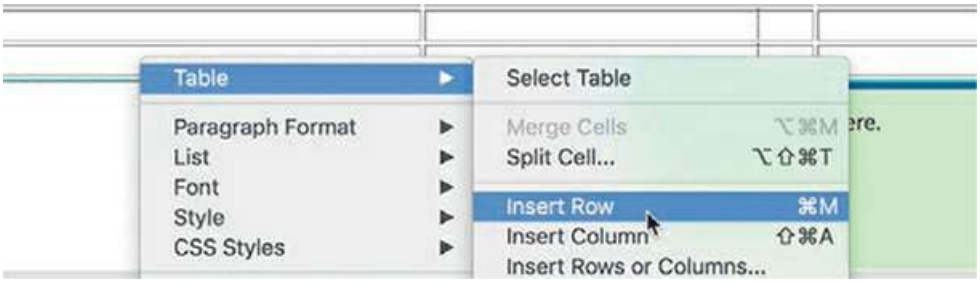
- Select the number 3 in the Rows field. Type **5** and press Enter/Return.



Dreamweaver adds two new rows to the table. You can also add rows and columns to the table interactively using the mouse.

- Right-click the last row of the table.

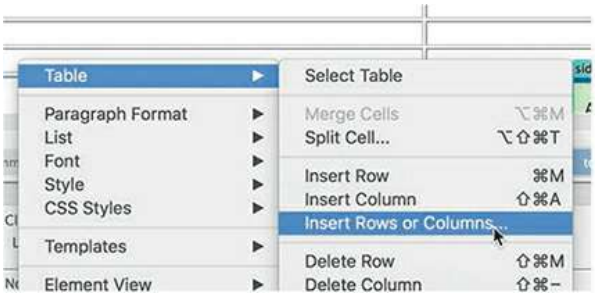
  Choose Table > Insert Row from the context menu.

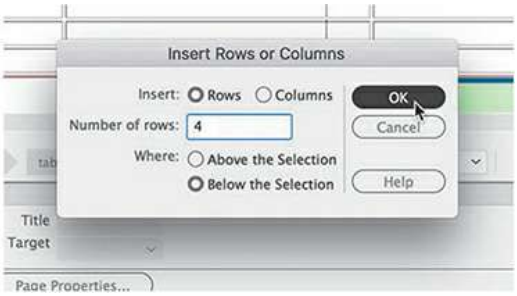


Another row is added to the table. The context menu can also insert multiple rows and/or columns at once.

. Right-click the last row of the table.

Choose Table > Insert Rows Or Columns from the context menu.





The Insert Rows Or Columns dialog appears.

. Insert four rows below the selection and click OK.

Four more rows are added to the table, for a total of 10 rows.

. Save all files.

Creating tables from scratch is a handy feature in Dreamweaver, but in many cases the data you need already exists in digital form—say, in a spreadsheet or even another webpage. Luckily, Dreamweaver provides support for moving such data from one page to another or even for creating tables directly from it.

Copying and pasting tables

Although Dreamweaver allows you to create tables manually inside the program, you can also move tables from other HTML files, or even from other programs, by using copy and paste.

Note

Dreamweaver allows you to copy and paste tables from some other programs, such as Microsoft Word. Unfortunately, copy and paste doesn't work with every program.

- Open the Files panel and double-click **calendar.html** in the lesson07/resources folder to open it.

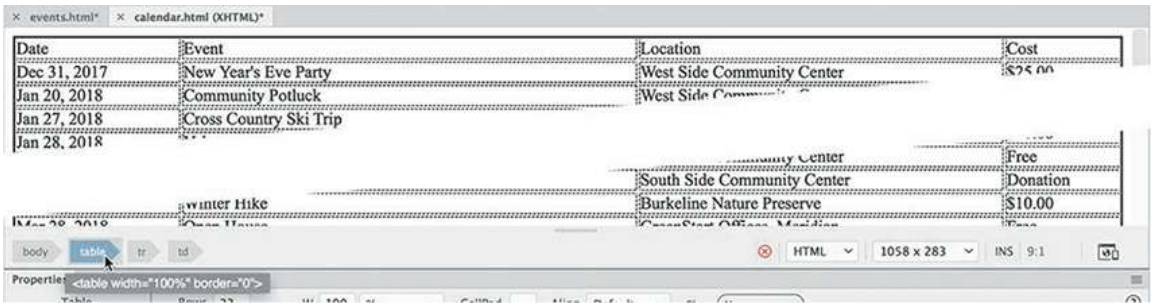
  This HTML file opens in its own tab in Dreamweaver. Note the table structure—it has four columns and numerous rows.

  When moving content from one file to another, it's important to match views in both documents. Since you were working in Design view in **events.html**, you should use Design view in this file too.

- Switch to Design view, if necessary.
- Insert the cursor in the table.

  Click the `<table>` tag selector.

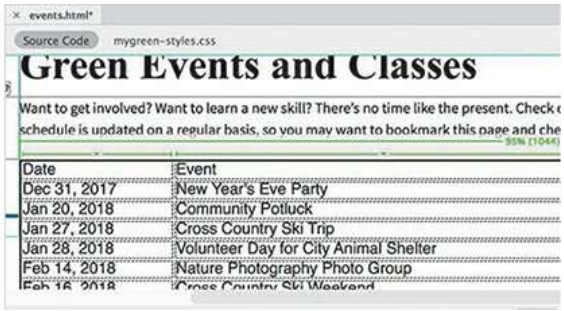
  Press Ctrl+C/Cmd+C to copy the table.



- Close **calendar.html**.
- In **events.html**, insert the cursor in the table.

  Select the `<table>` tag selector.

  Press Ctrl+V/Cmd+V to paste the table.



  The new table element completely replaces the existing table. This workflow will work in Design and Code views. But you must match views in both documents before you copy and paste.

- Save the file.

Styling tables with CSS

It’s clear that the formatting for the table is being supplied only by HTML settings. The CSS styling applied to the rest of the page doesn’t seem to be affecting the text in the table. You’ll have to create custom rules for the table text.

- Switch to Live view. Click the table.

  Select the table tag selector.

- In the CSS Designer, select **mygreen-styles.css**.

  Create a new selector: `section table`

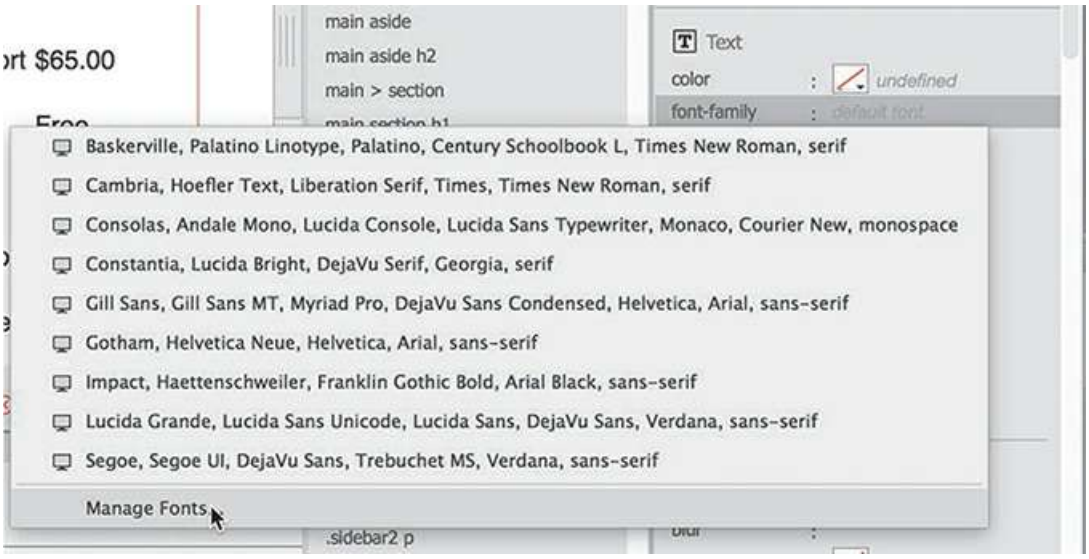
  The text in the table is larger than the other text on the page and seems crowded. In cases like this, you’ll want to pick a typeface and font that uses the space more economically.

Face vs. font: Know the difference?

People throw around the terms typeface and font all the time as if they were interchangeable. They are not. Do you know the difference? Typeface refers to the design of an entire family of letterforms. Font refers to one specific design. In other words, a typeface is usually composed of multiple fonts. Typically, a typeface will feature four basic designs: regular (or roman), italic, bold, and bold-italic.

When you choose a font in a CSS specification, you usually choose the regular format, or font, by default. When a CSS specification calls for italic or bold, the browser will normally load the italic or bold versions of the typeface automatically. However, you should be aware that many browsers can actually generate italic or bold effects when these fonts are not present or available. Purists resent this capability and go out of their way to define rules for italic and bold variations with specific calls to italic and bold versions of the typefaces they want to use. But, in the end, if the font is not installed, the browser cannot display it.

- In the CSS Designer Properties window, deselect the option Show Set, if necessary.
- In the Text category, click to open the `font-family` property.



A pop-up window appears showing Dreamweaver's nine predefined font groups, or *stacks*. You can select one of these or create one of your own. Are you wondering why you don't see the entire list of fonts installed on your computer?

The answer is a simple but ingenious solution to a problem that has nagged web designers from the beginning. Until recently, the fonts you see in your browser were not actually part of the webpage or the server; they were supplied by the computer browsing the site.

Although most computers have many fonts in common, they don't always have the same fonts, and users are free to add or remove fonts at will. So if you choose a specific font and it isn't installed on the visitor's computer, your carefully designed and formatted webpage could immediately, and tragically, appear in Courier or some other equally undesirable typeface.

For most people, the solution has been to specify fonts in groups, or *stacks*, giving the browser a second, third, and perhaps fourth (or more) choice to default to before it picks for itself (egads!). Some call this technique *degrading gracefully*. Dreamweaver CC (2019 release) offers nine predefined font stacks.

As you can see, the predefined font stacks are pretty limited. If you don't see a combination you like, you can click the Manage Fonts option at the bottom of the Set Font Family pop-up menu and create your own.

. Click Manage Fonts.

The Manage Fonts dialog gives you three options (tabs) for using web fonts: Adobe Edge Web Fonts, Local Web Fonts, and Custom Font Stacks. The first two tabs provide access to a new technique for using custom fonts on the web.

The Adobe Edge Web Fonts option supports the Edge Web Fonts service, through which you can access hundreds of fonts in multiple design categories right inside the program.

Local Web Fonts allows you to define the use of fonts that you can buy or find free on the Internet and that you can host on your own website.

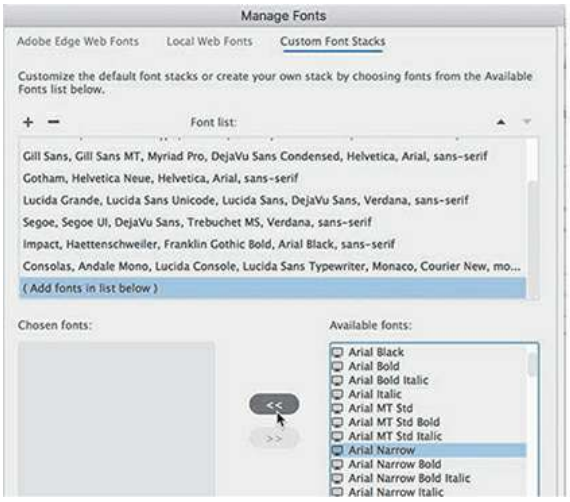
The option Custom Font Stacks enables you to build font groups using the new web-hosted fonts, web-safe fonts (fonts universally installed on most computers), or a combination of both. For the table text, it would be good to select a condensed or reduced-width font.

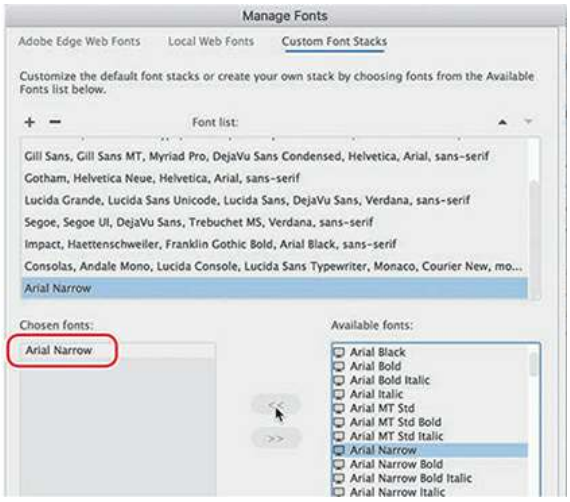
- In the Manage Fonts dialog, click the Custom Font Stacks tab.

  Arial Narrow is a condensed font and is considered a *web-safe* font.

- In the Available Fonts list, locate **Arial Narrow**.

  Click the << button to move the font to the Chosen Fonts list.





Note

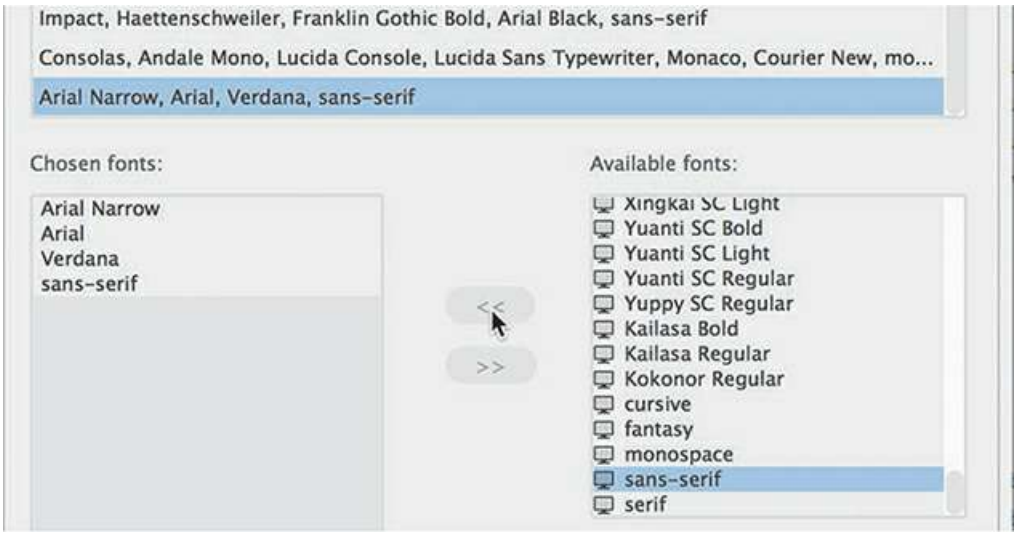
Whenever typing a font name manually, you must spell the font name correctly. Any typos will cause the font to fail to load.

Note

Not all font formats are universally supported across all computers and devices. Make sure your chosen font is supported within your desired audience.

If you cannot find a font in the list, you can type the name in the text field at the bottom of the dialog and press the << button.

. Repeat step 7 to add **Arial**, **Verdana**, and **sans-serif** to the Chosen Fonts list.

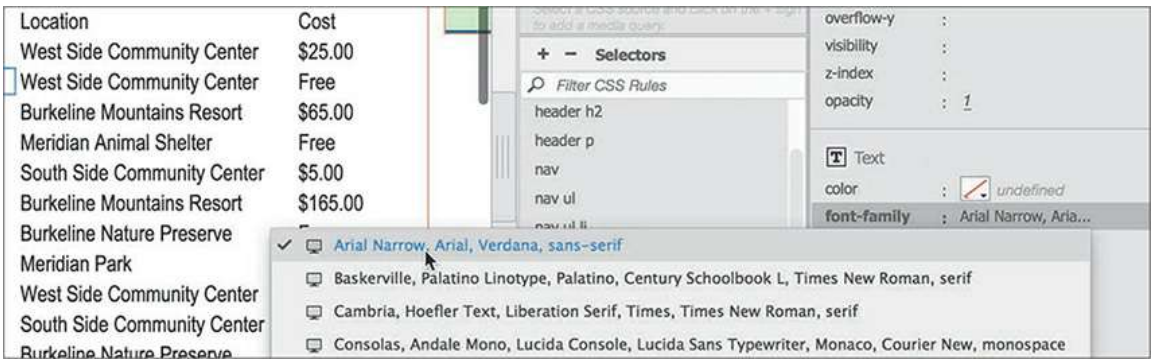


Feel free to add more web or web-safe fonts to your list as desired. If any fonts you want to use are not installed on your computer, type the names into the text field, and then add them to the stack using the <<button.

. Click Done.

The Manage Fonts dialog closes. The font stack was created but not applied.

. In the `font-family` property, select your new custom font stack.



The text in the table is now styled with Arial Narrow. It looks much better, but the table styling could still use some more tweaks.

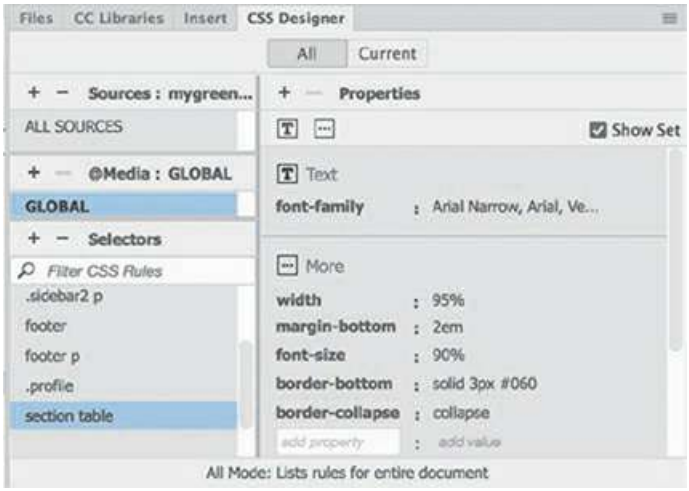
. Create the following specifications for `section` table:

Click here to view code image

```
width: 95%
margin-bottom: 2em
font-size: 90%
border-bottom: solid 3px #060
border-collapse: collapse
```

Note

Feel free to enable the Show Set option when creating these specifications, as shown.



The table displays a dark green border at the bottom, and the content has been reduced in size.

You have applied styling to one aspect of the table properties, but there are plenty of things you still need to address within this element.

. Save all files.

The rule you just created formats only the overall structure of the table, but it can't control or format the individual rows and columns. In the next exercise, you will turn your attention to a table's inner workings.

Styling table cells

Just as with tables, column styling can be applied by HTML attributes or CSS rules. Formatting for columns can be applied via two elements that create the individual cells:`<th>`for table header and `<td>` for table data.

Note

Remember that the order of the rules can affect the style cascade as well as how and what styling is inherited.

It's a good idea to create a generic rule to reset the default formats of the`<th>`and `<td>` elements. Later, you will create custom rules to apply more specific settings.

- Create a new selector in **mygreen-styles.css**:

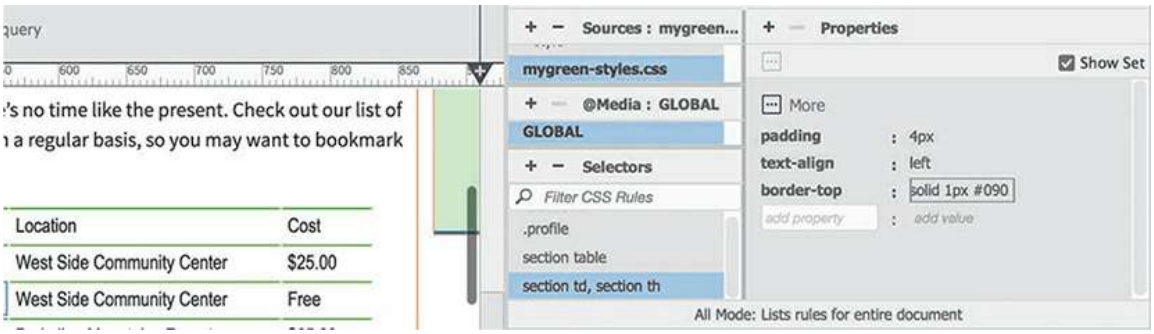
  `section td, section th`

  This simplified selector will work fine. Since<td>and <th> elements have to be in tables anyway, there's really no need to put >table> in the selector name.

- In the Properties window, select the Show Set option.
- Create the following properties for the new rule:

  Click here to view code image

```
padding: 4px
text-align: left
border-top: solid 1px #090
```



  Now that that you've added a border to the rows, the table border is no longer needed.

- Select the <table> tag selector.

  The Properties panel should display the table properties.

- Change the border value to 0 (zero).

  A thin green border appears above each row of the table, making the data easier to read. You may not be able to see the border properly unless you use Live view.

  Long columns and rows of undifferentiated data can be tedious to read and hard to decipher. Headers are often used to help the reader identify data. By default, the text in header cells is formatted in bold and centered to help it stand out from the normal cells, but some browsers do not honor this default styling. So don't count on it. You can make the headers stand out by giving them a touch of color of their own.

- Create a new rule: `section th`

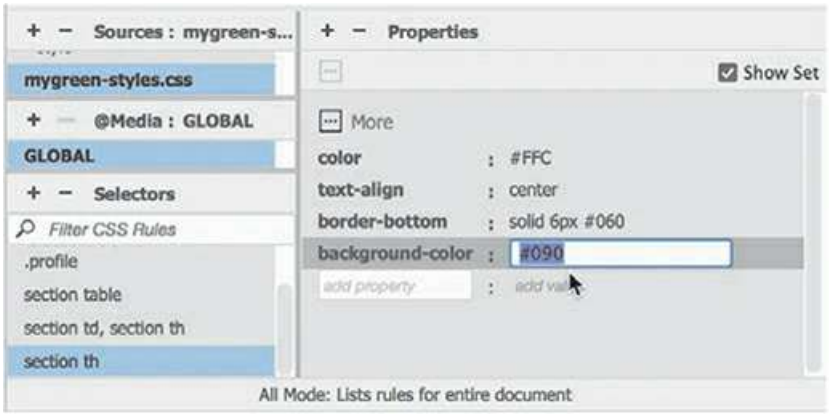
Note

The standalone th rule for the <th> element must appear after the rule styling th and td elements in the CSS or some of its formatting will be reset.

- Create the following properties in section th:

  Click here to view code image

```
color: #FFC
text-align: center
border-bottom: solid 6px #060
background-color: #090
```

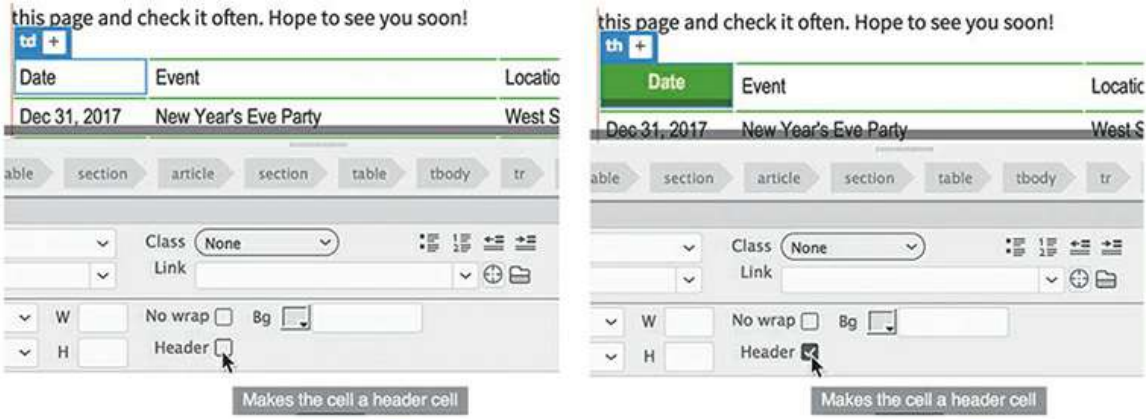


The rule is created, but it still needs to be applied. There are no headers in the table yet. Dreamweaver makes it easy to convert existing <td> elements into <th> elements.

- Click the first cell of the first row of the table.

  In the Property inspector, select the Header option.

  Note the tag selector and Element Display.



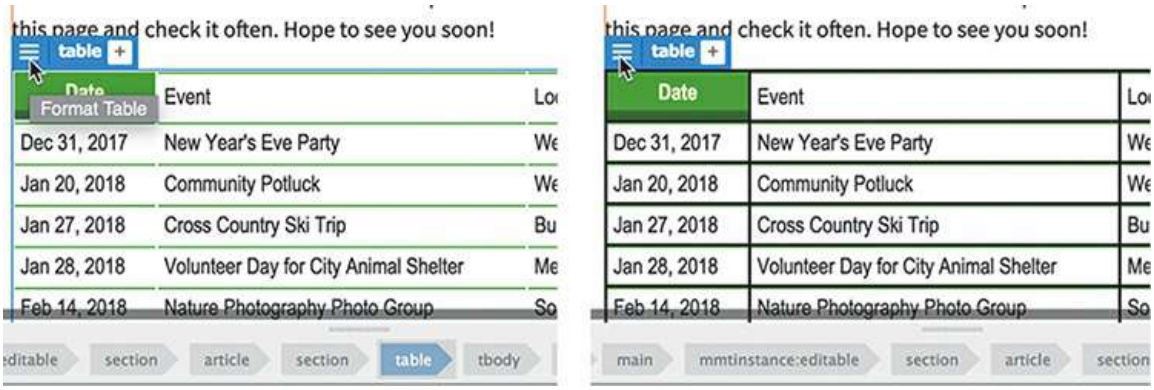
The cell background is filled with green. The Element Display changes from the td element to <th>.

When you click the Header checkbox, Dreamweaver automatically rewrites the markup, converting the existing<td>tags to <th> and thereby applying the CSS formatting. This functionality will save you lots of time over editing the code manually. In Live view, to select more than one cell, you have to use the enhanced table-editing function.

- Select the `<table>` tag selector.

  The Element Display appears focused on the `<table>` element. To enable the special editing mode for tables, you must first click the sandwich icon in the Element Display.

- Click the sandwich icon.



  When you click the icon, Dreamweaver enables an enhanced table-editing mode. Now you can select two or more cells, entire rows, or columns.

- Click the second cell of the first row and drag to select the remaining cells in the first row. Or you can select an entire row at once by positioning the cursor at the left edge of the table row and clicking when you see the black selection arrow appear to the left of the row.

- In the Property inspector, select the Header option to convert the table cells to header cells.

  The whole first row is filled with green as the table cells are converted to header cells.

- Save all files.

Controlling table display

Unless you specify otherwise, empty table columns will divide the available space between them equally. But once you start adding content to the cells, all bets are off. Tables seem to get a mind of their own and divvy up the space in a different way. In most cases, they'll award more space to columns that contain more data, but that's not guaranteed to happen.

To provide the highest level of control, you'll assign unique classes to the cells in each column.

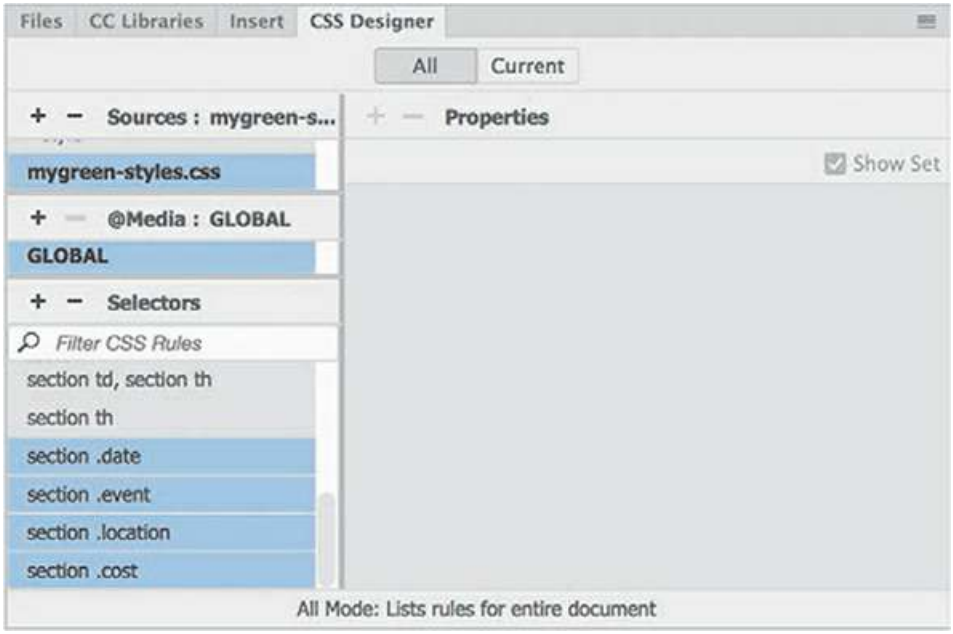
Creating them first makes it easier to assign them to the various elements later.

. Choose **mygreen-styles.css**> `GLOBAL`.

Create the following new selectors:

Click here to view code image

```
section .date
section .event
section.location
section .cost
```

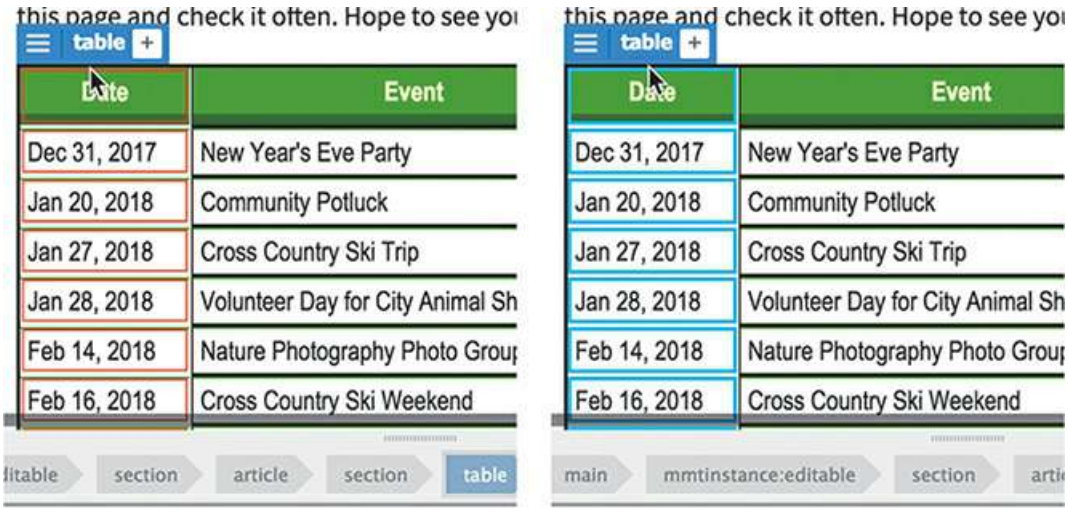


Four new rules appear in the Selectors window but contain no styling information. Even without styling, the classes can be assigned to each column. Dreamweaver makes it easy to apply classes to an entire column.

. Using the enhanced table-editing mode, position the cursor at the top of the first column of the table. Click to select the entire column.

Note

If you have difficulty working with tables in Live view, you can perform all these actions in Design view.



The column borders turn blue, indicating that the column is selected.

. Click to open the Class menu in the Property inspector.

A list of classes appears in alphabetical order.

. Choose *date* from the list.

The cells in the first column should now have the class `.date` applied to them. But after applying the class to the first column, you may notice that Dreamweaver has returned the table to normal mode again.

. Click the sandwich icon again.

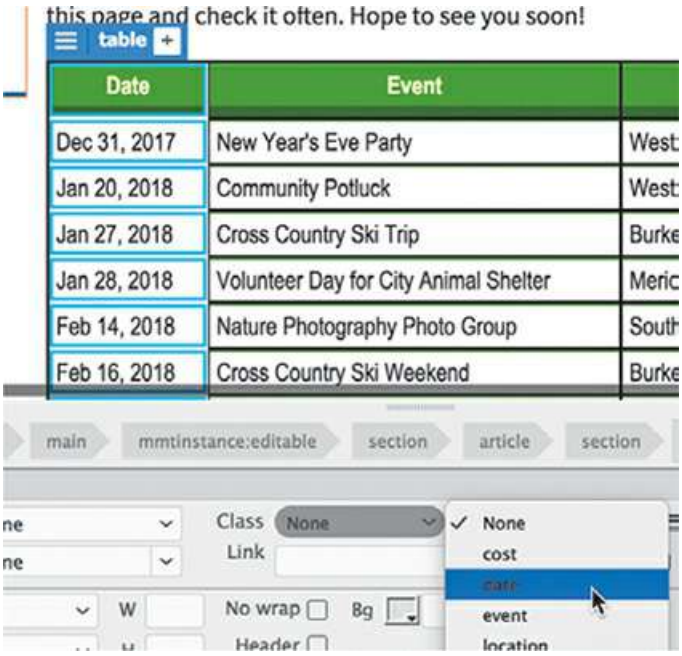
Apply the **`event`** class to the second column.

. Repeat step 5 to apply the appropriate classes to the remaining columns.

Controlling the width of a column is quite simple. Since the entire column must be the same width, you can apply a width specification to only one cell. If cells in a column have conflicting specifications, typically the largest width wins. Since you just applied a class to each column, any settings added to the class will affect every cell in that column.

Note

Even if you apply a width that's too narrow for the existing content, by default a cell can't be any smaller than the largest word or graphic element contained within it.



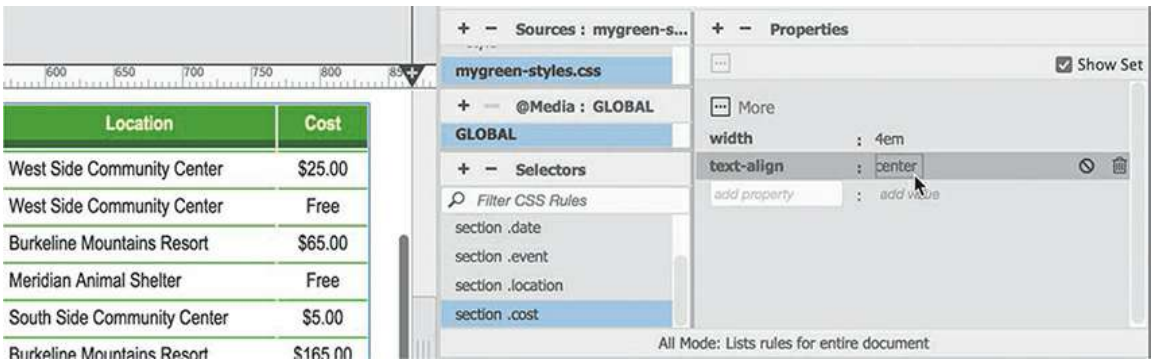
. Add this property to the rule `section` .date:

```
%
width: 6em
```

The Date column resizes. The remaining columns automatically divvy up the space left over. Column styling can also specify text alignment as well as width. Let's apply styling to the contents of the Cost column.

. Add these properties to the rule `section` .cost:

```
width: 4em
text-align: center
```



The Cost column resizes to a width of 4 ems, and the text aligns to the center.

. Save all files.

Now, if you want to control the styling of the columns individually you have the ability to do so. Note that the tag selectors and the Element Display show the class names for each cell, such as `th.cost or td.cost.`

Inserting tables from other sources

In addition to creating tables by hand, you can also create them from data exported from databases and spreadsheets. In this exercise, you will create a table from data that was exported from Microsoft Excel to a comma-separated values (CSV) file. The import feature does not work in Live view.

. Switch to Design view.

Insert the cursor in the existing Events table.

Select the `<section>` tag selector.

Be sure to select the `section` element that contains the Events table.

. Press the right arrow key.

In Design view, this technique moves the cursor after the closing `</section>` tag within the code.

. Choose File > Import > Tabular Data.

The Import Tabular Data dialog appears.

- Click the Browse button and select **classes.csv** from the lesson07/resources folder. Click Open. Comma should be automatically selected in the Delimiter menu.
- Select the following options in the Import Tabular Data dialog: Table Width: **95%** Border: **0**.

  Although you set the width in the dialog, as you did for the Events table, remember that the table width will actually be controlled by the table rule created earlier. HTML attributes will be honored in browsers or devices that do not support CSS. Because this is the case, make sure that the HTML attributes you use don't break the layout.
- Click OK.

  A new table—containing a class (course) schedule—appears below the first. To conform to the structure you created for the first table, you should insert the new one into its own `<section>` element.
- Select the `table` tag selector for the new table.
- Choose Insert > Section.

  Select **Wrap Around Selection** from the Insert menu.

  Click OK to insert the `<section>` element.

- Switch to Live view.

  The new table is inserted into the `<section>` element. Green lines appear between the rows, but the header cells are not styled the same as in the first table.
- Select the first row of the Class schedule.

  In the Property inspector, select the Header option.

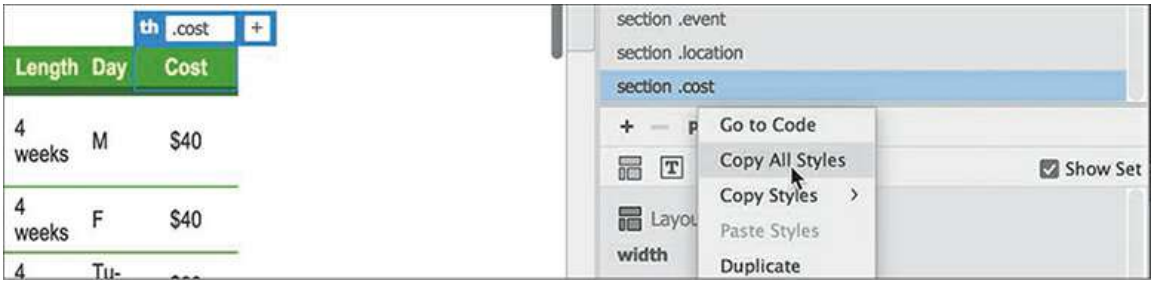
  The header cells now display in green with reversed text.

  The new table has one more column than the first one, and the text may be wrapping awkwardly in the last three columns. You will fix this display by using the `<.cost>` class created earlier and by creating additional custom classes.
- Using enhanced table-editing mode select the Cost column.

  In the Property inspector, choose `cost` from the Class menu.

  The Cost columns in both tables are now the same width.

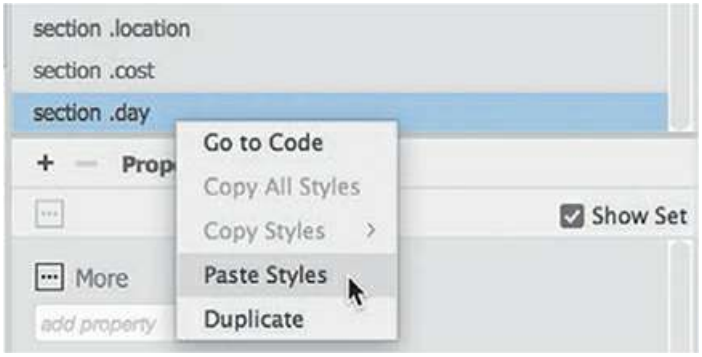
. In the CSS Designer, right-click the rule `section .cost`. Choose Copy All Styles from the context menu.



. Create a new selector: **`section .day`**

Right-click the new selector.

Select Paste Styles from the context menu.



The new rule now has the same styling as the `section .cost` rule.

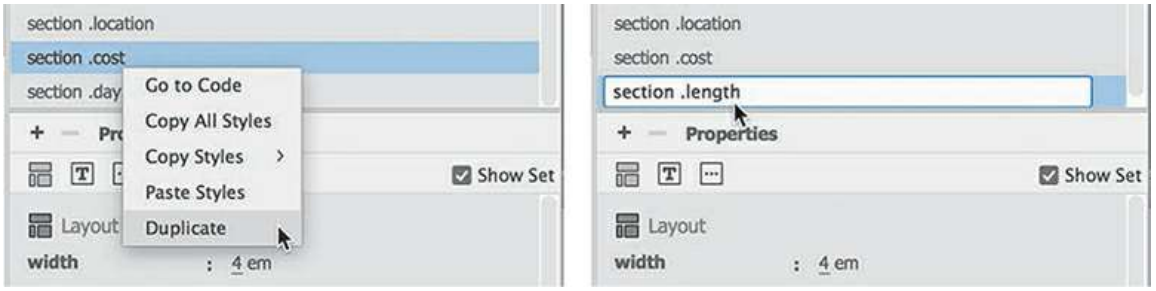
. Repeat step 11 to apply the day class to the Day column in the Classes table.

Dreamweaver also provides an option for duplicating rules.

. Right-click the rule section.cost.

Choose Duplicate from the context menu.

Enter **`section .length`** as the new selector.



. Apply the .length class to the Length column in the Classes table, as in step 10.

By creating and applying custom classes to each column, you have the means to modify each

column individually. You need to make two more rules: one to format the Class column and the other to format the Description column.

. Duplicate the rule `section .date`.

Enter **`section .class`** as the new rule name.

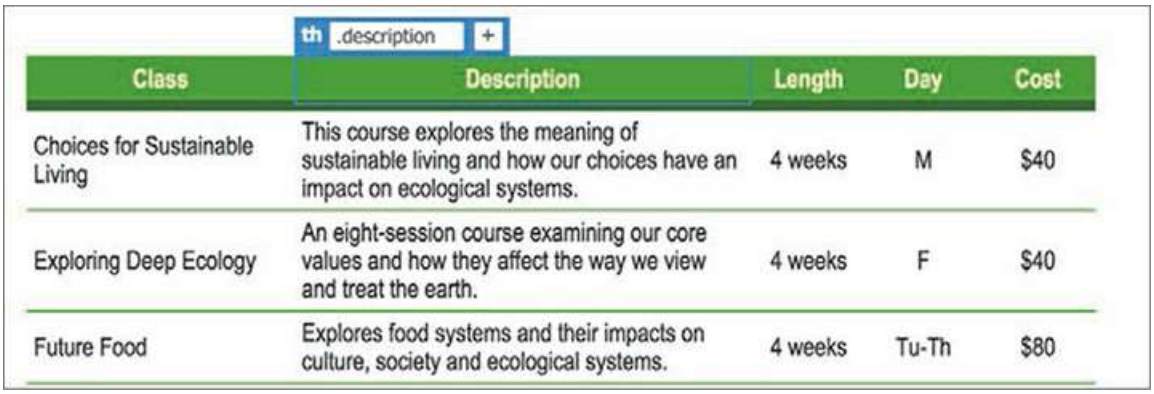
Change the width to **`10em`**.

. Duplicate the rule `section .event`.

Enter **`section .description`** as the new name.

. Apply the **`.class`** class to the Class column.

Apply the **`.description`** class to the Description column.



All columns in both tables now have custom CSS classes assigned to them.

. Save all files.

As with articles, tables should have descriptive titles that help visitors and search engines differentiate between them.

Adding and formatting caption elements

The two tables you inserted on the page contain different information but don't feature any labels or titles. Let's add a title to each. The `<caption>` element was designed to identify the content of HTML tables. This element is inserted as a child of the `<table>` element itself.

. Open **events.html** in Live view, if necessary.

. Insert the cursor in the first table.

Select the `table` tag selector. Switch to Code view.

By selecting the table first in Live view, Dreamweaver automatically highlights the code in Code view, making it easier to find.

. Locate the opening `<table>` tag.

Insert the cursor directly after this tag.

Press Return/Enter to insert a new line.

. Type **`<caption>`** or select it from the code-hinting menu when it appears.

. Type **`2019 Event Schedule`** and then type **`</`** to close the element, if necessary.

```
<section>
  <table width="95%" border="0">
    <caption>
      2019 Event Schedule
      </caption>
    <tr>
      <th class="date">Date</th>
      <th class="event">Event</th>
```

. Switch to Live view.

The caption is complete and inserted as a child element of the table.

. Repeat steps 2 through 4 for the Classes table.

Type **`2019 Class Schedule`** and then type **`</`** to close the element, if necessary.

```
<section>
  <table width="95%" border="0">
    <caption>
    2019 Class Schedule
    </caption>
    <tr>
      <th class="class">Class</th>
      <th class="description">Description</th>
```

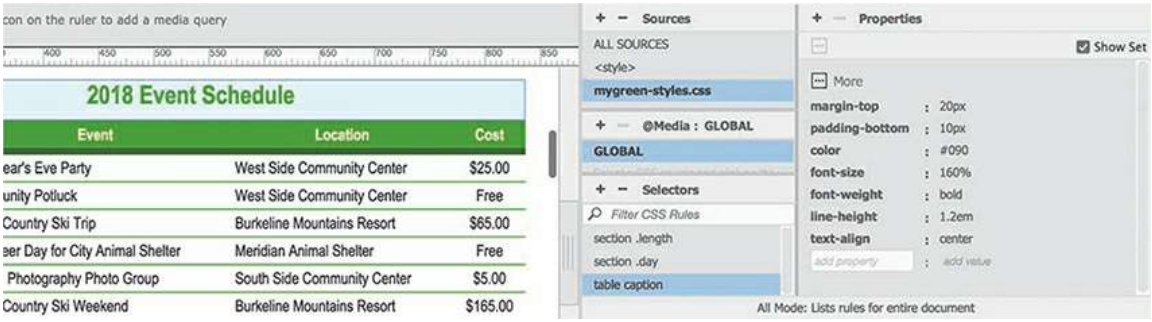
. Switch to Live view.

The default caption styling is relatively small and understated. The captions are lost against the color and formatting of the table. Let’s beef them up a bit with their own custom CSS rule.

- Create a new selector: `table caption`
- Create these properties for the rule `table caption`:

  Click here to view code image

```
margin-top: 20px
padding-bottom: 10px
color: #090
font-size: 160%
font-weight: bold
line-height: 1.2em
text-align: center
```



  The captions now appear sufficiently large and impressive above each table.

- Save all files.

  Formatting the tables and the captions with CSS has made them much easier to read and understand. Feel free to experiment with the size and placement of the captions and with the other specifications affecting the tables.

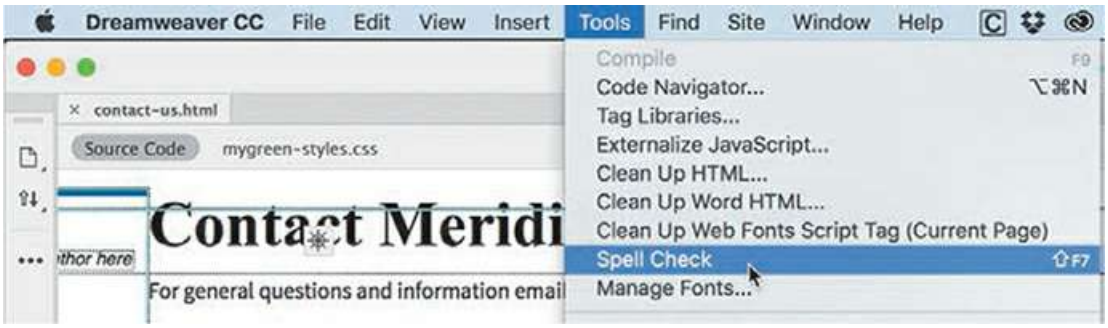
Spell-checking webpages

It's important to ensure that the content you post to the web is error-free. Dreamweaver includes a robust spell-checker capable of identifying commonly misspelled words and of creating a custom dictionary for nonstandard terms that you might use on a regular basis.

- Open **contact-us.html**, if necessary.
- Switch to Design view. Insert the cursor at the beginning of the heading *Contact Meridien GreenStart*. Choose Tools > Spell Check.

Note

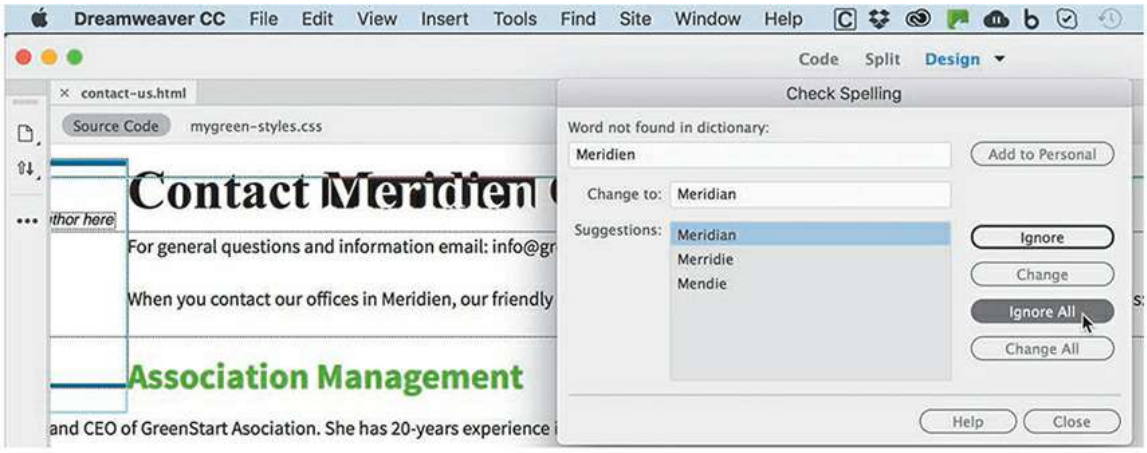
The spell-checker runs only in Design view. If you are in Code view or Live view, the command will be grayed out.



The spell-checker starts wherever the cursor has been inserted. If the cursor is located lower on the page, you will have to restart the spell-checker at least once to examine the entire page. It also does not check content locked in non-editable template regions.

The Check Spelling dialog highlights the word *Meridien*, which is the name of the fictional city where the GreenStart association is located. You could click the option Add To Personal to insert the word into your custom dictionary, but for now you will skip over other occurrences of the name during this check.

. Click Ignore All.

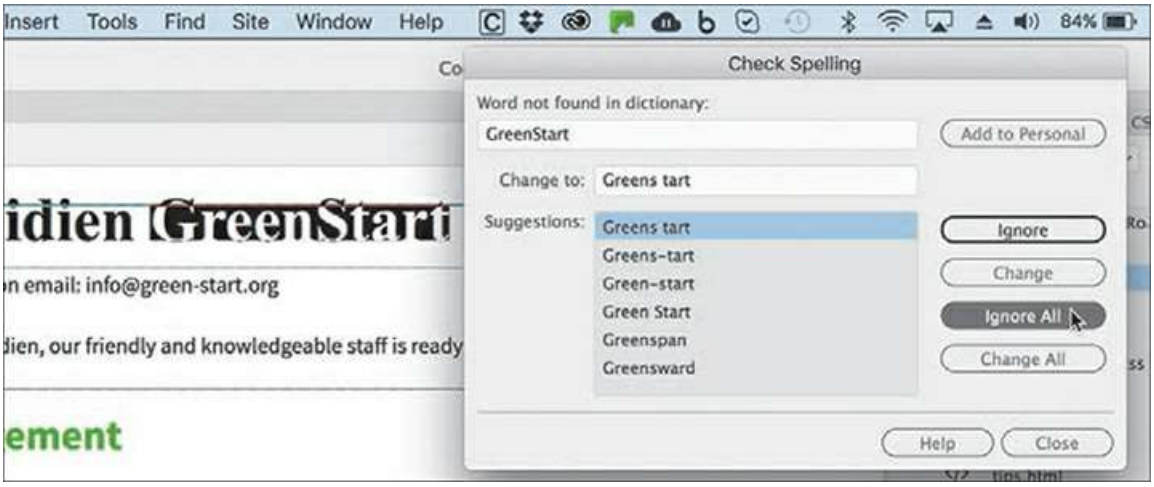


Dreamweaver's spell-checker highlights the word *GreenStart*, which is the name of the association. If GreenStart were the name of your own company, you'd want to add it to your custom dictionary. However, you don't want to add a fictional company name.

. Click Ignore All again.

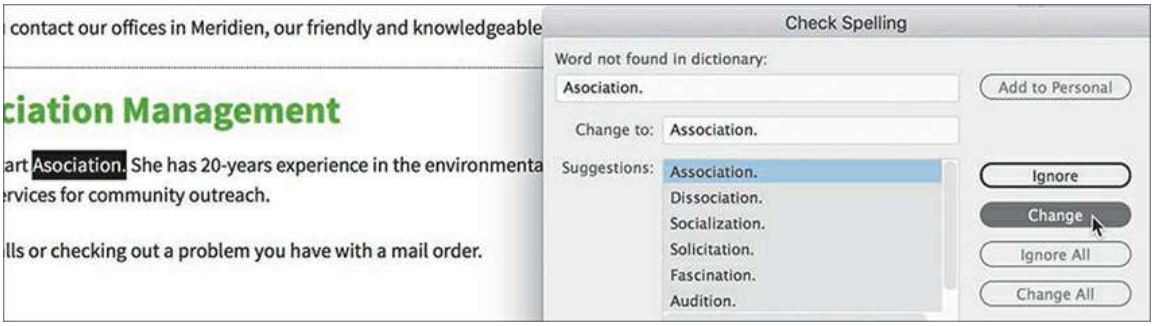
Dreamweaver highlights the domain for the email address info@greenstart.org.

. Click Ignore All.



Dreamweaver highlights the word *Asociation*, which is missing an "*s*."

- To correct the spelling, locate the correctly spelled word (*Association*) in the Suggestions list and click Change.



- Continue the spell-check to the end.

  Correct any misspelled words and ignore proper names, as necessary. If a dialog prompts you to start the check from the beginning, click Yes.

  Dreamweaver will start spell-checking from the top of the file to catch any words it may have missed.

- Click OK when the spell-check is complete. Save the file.

  It's important to point out that the spell-checker is designed to find only words that are *spelled* incorrectly. It will not find words that are *used* incorrectly. In those instances, nothing takes the place of a careful reading of the content.

Finding and replacing text

The ability to find and replace text is one of Dreamweaver's most powerful features. Unlike other programs, Dreamweaver can find almost anything, anywhere in your site, including text, code, and any type of whitespace that can be created in the program. You can search the entire markup, or you can limit the search to the rendered text or to the underlying tags. Advanced users can enlist powerful pattern-matching algorithms known as *regular expressions* to perform sophisticated find-and-replace operations. And then, Dreamweaver takes it one step further by allowing you to replace the targeted text or code with similar amounts of text, code, and whitespace. If you are a user of previous versions of Dreamweaver, you will see some significant changes in the Find And Replace function.

In this exercise, you'll learn some important techniques for using the Find And Replace feature.

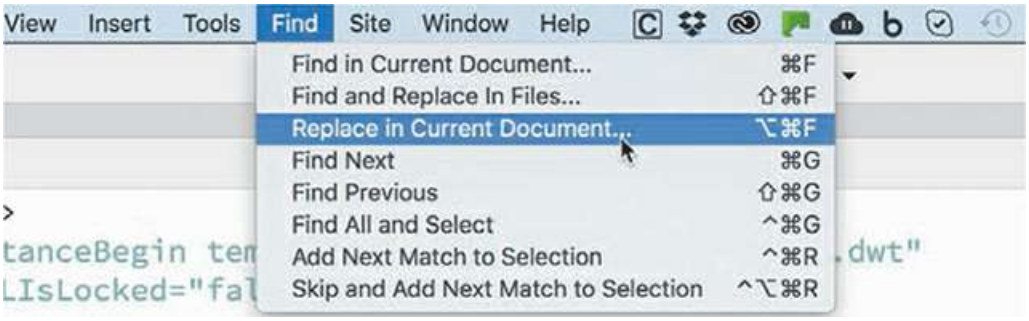
. Select the **events.html** document tab, if necessary, or open it from the site root folder.

  There are several ways to identify the text or code you want to find. One way is to simply type it in the Find field. In the Events table, the name *Meridien* was spelled incorrectly as *Meridian*. Since *Meridian* is an actual word, the spell-checker won't flag it as an error and give you the opportunity to correct it. So you'll use find and replace to make the change instead.

. Switch to Code view, if necessary.

  Click in the *Green Events and Classes* heading.

  Choose Find > Replace In Current Document.



  The Find And Replace panel appears at the bottom of the document window. If you have not used the feature before, the Find field should be empty.

. Type **Meridian** in the Find field.

  Dreamweaver finds the first occurrence of *Meridian* and indicates how many matches it has

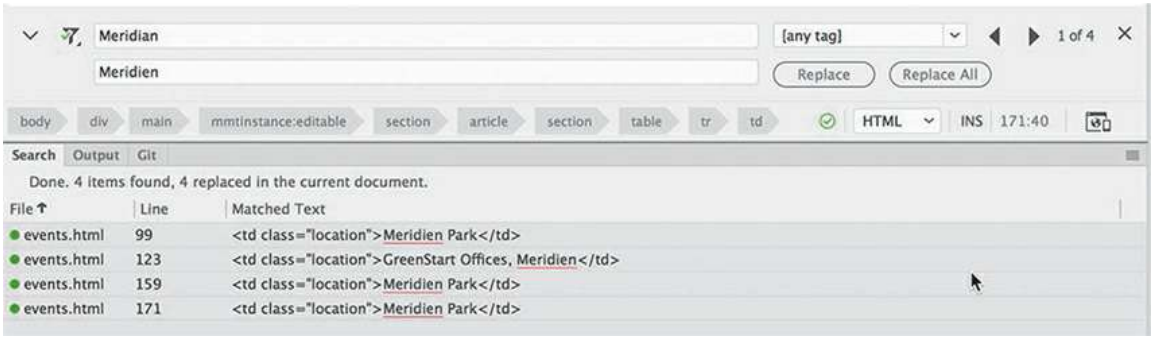
found in the document.

- Type **Meridien** in the Replace field.

- Click Replace.

  Dreamweaver replaces the first instance of *Meridian* and immediately searches for the next instance. You can continue to replace the words one at a time, or you can choose to replace all occurrences.

- Click Replace All.

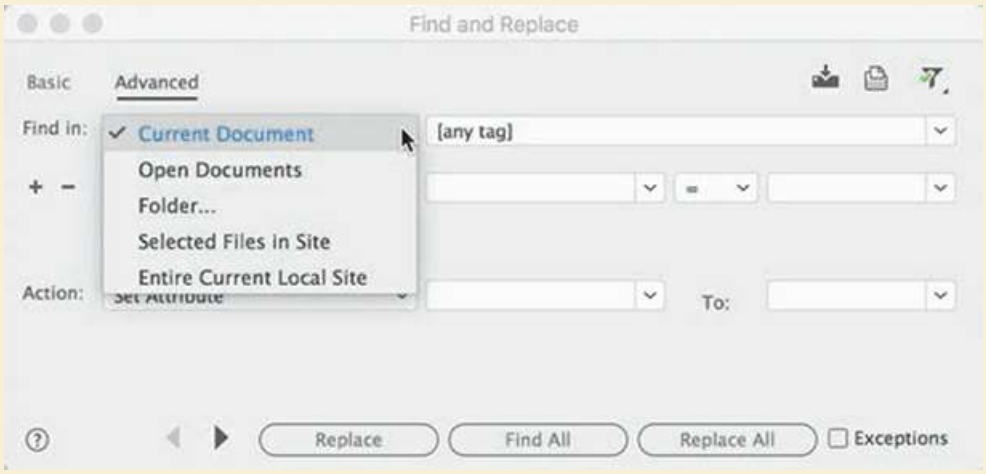


When you click Replace All, the Search Report panel expands to list all the changes made.

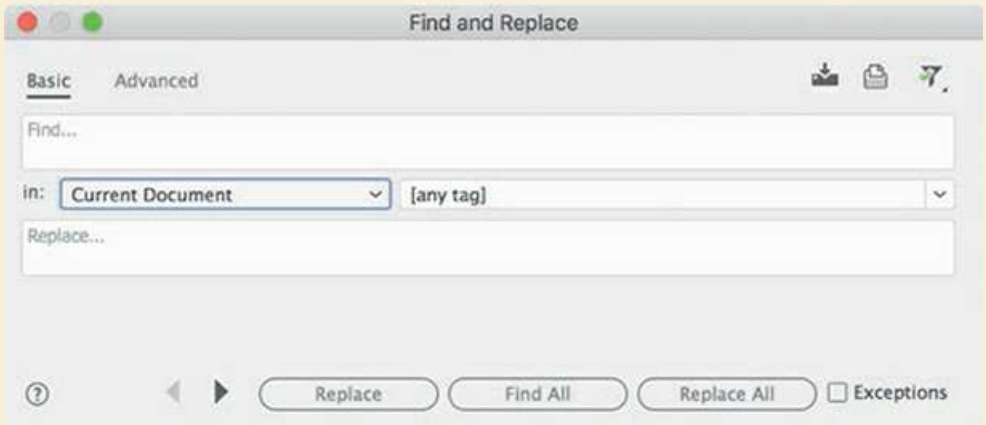
- Right-click the Search Report tab and select Close Tab Group from the context menu.

  Another method for targeting text and code is to select it *before* activating the command. This method can be used in either Design or Code view.

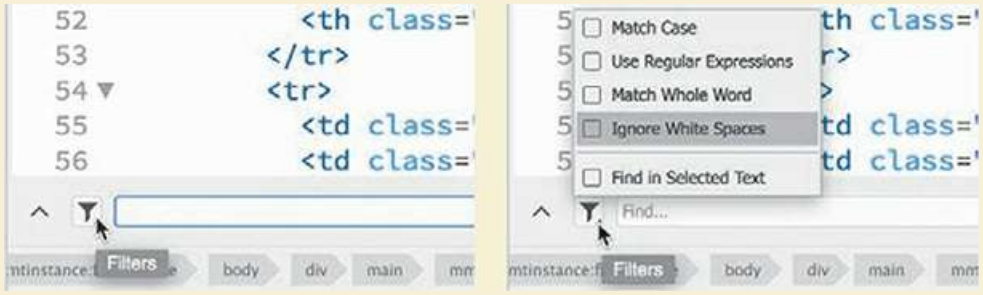
Superpowerfindelicious!



To access even more *findelicious* powers select Find > Find and Replace in Files. A standalone Find and Replace panel will appear. Note the Find and Filter options in the panel. The power and flexibility of Dreamweaver shine brightest here. Use the command Find And Replace in Files to search in selected text, in the current document, in all open documents, in a specific folder, in selected files of the site, or in the entire current local site.



But as if those options weren't enough, Dreamweaver also allows you to target, or limit, the search to the source code, to text only, based on case, and to whole words, and it gives you the ability to use *regular expressions* and to *ignore whitespace.*



. In Code view, locate and select the first occurrence of the text *Burkeline Nature Preserve* in the Location column of the Events table.

Choose Find > Find In Current Document.

Note

The Find and Replace panel typically appears with the Replace function hidden.

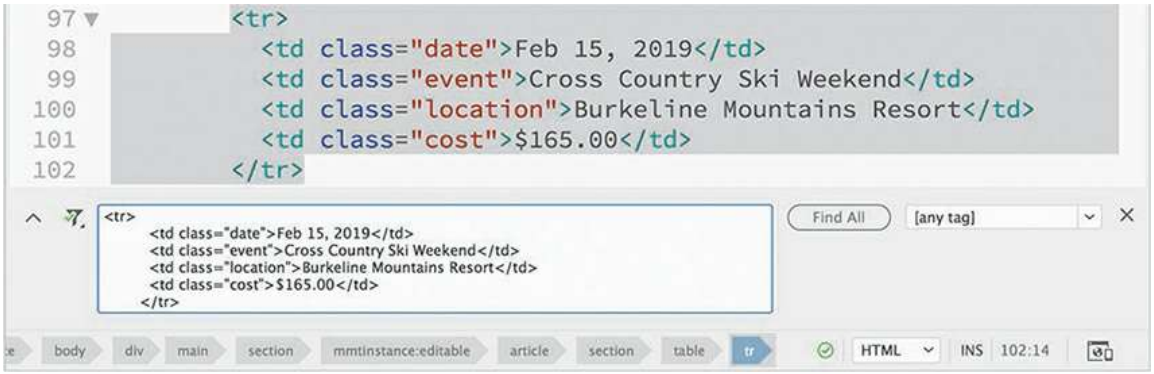
The Find And Replace panel appears. The selected text is automatically entered into the Find field by Dreamweaver. This technique will work with small snippets of text or code. With larger selections, you'll need to use copy and paste.

. With the cursor still inserted in the *Burkeline Nature Preserve* text, click the `<tr>` tag selector at the bottom of the document window.

. Press Ctrl+C/Cmd+C to copy the selection.

. If necessary, choose Find > Find In Current Document.

Insert the cursor into the Find field and press Ctrl+V/Cmd+V.



The selected code is entered into the Find field in its entirety, including the line breaks and other whitespace. The reason this is remarkable is that there's no way to enter this type of markup in the Find field manually.

. Select the code in the Find field.

Press Delete to remove it.

Type **<tr>** and press Enter/Return to insert a line break.

Observe what happens.

Pressing Enter/Return did not insert a line break; instead, it activated the Find command, which finds the next occurrence of the `<tr>` element. In fact, you can't manually insert any type of line break within the field.

You probably don't think this is much of a problem, since you've already seen that Dreamweaver inserts text or code when it's selected. Unfortunately, the method used in steps 9 through 11 doesn't work with large amounts of text or code. In those instances, you'll need to copy and paste.

. In Code view, click the `<table>` tag selector.

Copy the selection.

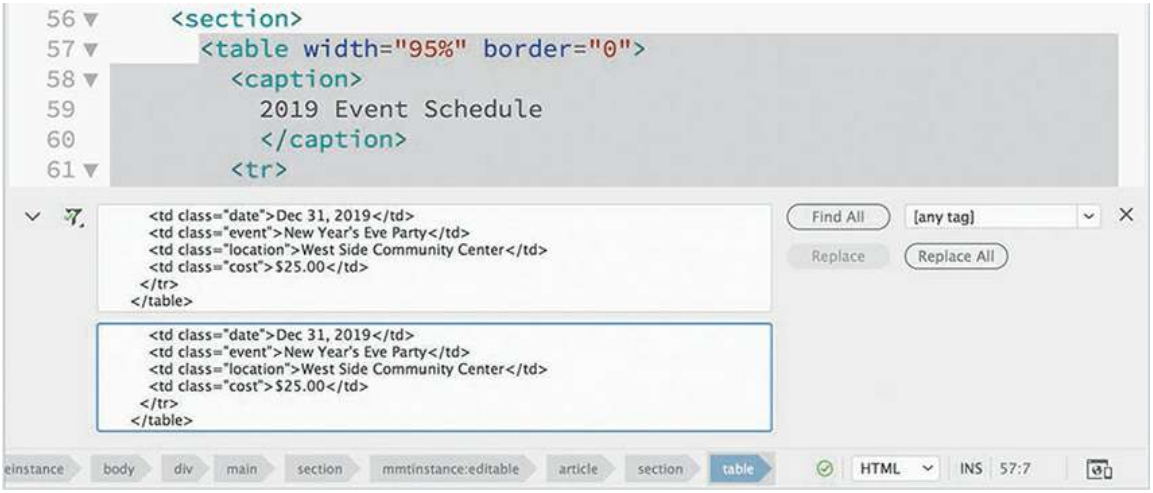
- Insert the cursor into the Find field and press Ctrl+V/Cmd+V.

> **Tip**
>
> If the Replace field does not appear, click the Show More ^ icon.

- Insert the cursor in the Replace field.

  Press Ctrl+A/Cmd+A to select the field contents.

  Press Ctrl+V/Cmd+V to replace any contents.



The entire table is pasted into the Find and Replace fields. Obviously, the two fields currently contain identical markup, but it illustrates how easy it would be to change or replace large amounts of code when needed.

- Close the Find And Replace panel. Save all files.

In this lesson, you created four new pages and learned how to import text from multiple sources. You formatted text as headings and lists, and then styled it using CSS. You inserted and formatted two tables and added captions to both. And you reviewed and corrected text using Dreamweaver's spell-checker and Find And Replace tools.

Optional self-paced exercise

At the end of the lesson, the four pages you created are only partially completed. Before

proceeding to the next lesson, go ahead and finish each page using the resources in the files **quotes07.txt** and **sidebar2-07.txt**, located in the resources folder. Don't forget to add meta titles and descriptions to each file too. If you have any questions about how the content should be created or formatted, check out the finished files with the same names within the finished-files folder for lesson07. Be sure to save all your changes when you are finished.

Review questions

1. How do you format text to be an HTML heading?
2. Explain how to turn paragraph text into an ordered or unordered list.
3. Describe two methods for inserting HTML tables into a webpage.
4. What element controls the width of a table column?
5. What items will not be found by Dreamweaver's spell-checker?
6. Describe three ways to insert content in the Find field.

Review answers

1. Use the Format menu in the Property inspector to apply HTML heading formatting, or press Ctrl+1/Cmd+1, Ctrl+2/Cmd+2, Ctrl+3/Cmd+3, and so on.
2. Highlight the text with the cursor and click the Ordered List button in the Property inspector. Then click the Unordered List button to change the numbered list to bullets.
3. You can copy and paste a table from another HTML file or from a compatible program. Or you can insert a table by importing the data from a delimited file.
4. The width of a table column is controlled by the widest`<th>`or `<td>` element that creates the individual table cell within the specific column.
5. The spell-checker finds only words that are *spelled* incorrectly, not those that are *used* incorrectly.
6. You can type text into the Find field, you can select text before you open the panel and then allow Dreamweaver to insert the selected text, or you can copy the text or code and then paste it into the field.

8 Working with Images

Lesson overview

In this lesson, you'll learn how to work with images and include them in your webpages in the following ways:

- Insert an image into a webpage
- Use Photoshop Smart Objects
- Copy and paste an image from Photoshop
- Make images responsive to different device and screen sizes
- Use tools in Dreamweaver to resize, crop, and resample web-compatible images

This lesson will take about 1 hour to complete. Please log in to your account on peachpit.com to download the project files for this lesson, as described in the "Getting Started" section at the beginning of this book. Follow the instructions under "Accessing the Lesson Files and Web Edition." Define a site based on the lesson08 folder.

Your Account page is also where you'll find any updates to the lessons or to the lesson files. Look on the Lesson & Update Files tab to access the most current content.

Dreamweaver provides many ways to insert and adjust graphics, both within the program and in tandem with other Creative Cloud tools, such as Adobe Fireworks and Adobe Photoshop.

Web image basics

The web is not so much a place as it is an experience. Essential to that experience are the images and graphics—both still and animated—that populate most websites. In the computer world, graphics fall into two main categories: vector and raster.

Vector graphic formats excel in line art, drawings, and logo art. Raster technology works better for storing photographic images.

Vector graphics

Vector graphics are created by math. They act as discrete objects, which you can reposition and resize as many times as you want without affecting or diminishing their output quality. The best application of vector art is wherever geometric shapes and text are used to create artistic effects. For example, most company logos are built from vector shapes.

Vector graphics are typically stored in the AI, EPS, PICT, or WMF file formats. Unfortunately, most web browsers don't support these formats. The vector format that is supported is SVG (Scalable Vector Graphic). The simplest way to get started with SVG is to create a graphic in your favorite vector-drawing program—such as Adobe Illustrator or CorelDRAW—and then export it to this format. If you are a good programmer, you may want to try creating SVG graphics using XML (Extensible Markup Language). Check out www.w3schools.com/html/html5_svg.asp to find out more about creating SVG graphics.

Raster graphics

Although SVG has definite advantages, web designers primarily use raster-based images in their webpages. Raster images are built from *pixels*, which stands for *picture elements*. Pixels have three basic characteristics:

- They are perfectly square in shape.
- They are all the same size.
- They display only one color at a time.

Raster-based images are composed of thousands, even millions, of pixels arranged in rows and columns, in patterns that create the illusion of an actual photo, painting, or drawing. It's an illusion, because there is no real photo on the screen, just a bunch of pixels that fool your eyes into seeing an image. And as the quality of the image increases, the illusion becomes more realistic. Raster image quality is based on three factors: resolution, size, and color.

The inset image shows an enlargement of the flowers, revealing the pixels that compose the image itself.

Resolution

Resolution is the best known of the factors affecting raster image quality. It is the expression of image quality measured in the number of pixels that fit in 1 inch (ppi). The more pixels you can fit in 1 inch, the more detail you can depict in the image. But better quality comes at a price. An unfortunate byproduct of higher resolution is larger file size. That's because each pixel must be stored as bytes of information within the image file—information that has real overhead in computer terms. More pixels means more information, which means larger files.

Resolution has a dramatic effect on image output. The web image on the left looks fine in the browser but doesn't have enough quality for printing.

Note

Printers and printing presses use round “dots” to create photographic images. Quality on a printer is measured in dots per inch, or dpi. The process of converting the square pixels used in your computer into the round dots used on the printer is called screening.

Luckily, web images have to appear and look their best only on computer screens, which are based mostly on a resolution of 72 ppi. This is low compared to other applications or output—such as professional four-color printing—where 300 dpi is considered the lowest acceptable quality. The lower resolution of the computer screen is an important factor in keeping most web image files at a reasonable size for downloading from the Internet.

Size

Size refers to the vertical and horizontal dimensions of the image. As image size increases, more pixels are required to create it, and therefore the file becomes larger. Since graphics take more time to download than HTML code, many designers in recent years have replaced graphical components with CSS formatting to speed up the web experience for their visitors. But if you need or want to use images, one method to ensure snappy downloads is to keep image size small. Even today, with the proliferation of high-speed Internet service, many websites still avoid using full-screen graphics, although that too is changing.

Although these two images share the identical resolution and color depth, you can see how image dimensions can affect file size.

Color

Color refers to the color space, or *palette*, that describes each image. Most computer screens display only a fraction of the colors that the human eye can see. And different computers and applications display varying levels of color, expressed by the term *bit depth*. Monochrome, or 1-bit color, is the smallest color space, displaying only black and white, with no shades of gray. Monochrome is used mostly for line-art illustrations, for blueprints, and to reproduce handwriting or signatures.

The 4-bit color space describes up to 16 colors. Additional colors can be simulated by a process known as *dithering*, where the available colors are interspersed and juxtaposed to create an illusion of more colors. This color space was created for the first color computer systems and game consoles. Because of its limitations, this palette is seldom used today.

The 8-bit palette offers up to 256 colors or 256 shades of gray. This is the basic color system of all computers, mobile phones, game systems, and handheld devices. This color space also includes what is known as the *web-safe* color palette. Web-safe refers to a subset of 8-bit colors that are supported on both Mac and Windows computers. Most computers, game consoles, handheld devices, and even phones now support higher color palettes, so 8-bit is not as important anymore. Unless you need to support non-computer devices, you can probably disregard the web-safe palette altogether.

Today, only a few older cellphones and handheld games support the 16-bit color space. This palette is named *high color* and sports a grand total of 65,000 colors. Although this sounds like a lot, 16-bit color is not considered good enough for most graphic design purposes or professional printing.

The highest color space is 24-bit color, which is named *true color*. This system generates up to 16.7 million colors. It is the gold standard for graphic design and professional printing. Several years ago, a new color space was added to the mix: 32-bit color. It doesn't offer any additional colors, but it provides an additional 8 bits of data for an attribute known as *alpha transparency*.

Alpha transparency enables you to designate parts of an image or graphic as fully or partially transparent. This trick allows you to create graphics that seem to have rounded corners or curves and can even eliminate the white bounding box typical of raster graphics.

24-bit color **8-bit color** **4-bit color**

Here you can see a dramatic comparison of three color spaces and what the total number of available colors means to image quality.

As with size and resolution, color depth can dramatically affect image file size. With all other aspects being equal, an 8-bit image is more than seven times larger than a monochrome image. And the 24-bit version is more than three times larger than the 8-bit image. The key to the effective use of images on a website is finding the balance of resolution, size, and color to achieve the desired optimal quality.

Optimizing your images is essential, even as more people get smartphones and tablets, because there are still millions of people all across the United States, and around the world, who don't have high-speed wired access to the Internet. In February of 2018, Pew research published a study reporting that only 65% of American households had access to broadband internet. Check out https://tinyurl.com/pew-broadband-report to see specific details. Using large images on your site is becoming more popular, but it could also cause problems for your target audience, depending on where they live.

Raster image file formats

Raster images can be stored in a multitude of file formats, but web designers have to be concerned with only three: GIF, JPEG, and PNG. These three formats are optimized for use on the Internet and compatible with virtually every browser. However, they are not equal in capability.

GIF

GIF (Graphics Interchange Format) was one of the first raster image file formats designed specifically for the web. It has changed only a little in the last 30 years. GIF supports a maximum of 256 colors (8-bit palette) and 72 ppi, so it's used mainly for web interfaces—buttons and graphical borders and such. But it does have two interesting features that keep it pertinent for today's web designers: index transparency and support for simple animation.

JPEG

JPEG, also written JPG, is named for the Joint Photographic Experts Group that created the image standard back in 1992 as a direct reaction to the limitations of the GIF file format. JPEG is a powerful format that supports unlimited resolution, image dimensions, and color depth. Because of this, most digital cameras use JPEG as their default file type for image storage. It's also the reason most designers use JPEG on their websites for images that must be displayed in high quality.

This may sound odd to you, since "high quality" (as described earlier) usually means large file size. Large files take longer to download to your browser. So why is this format so popular on the web? The JPEG format's claim to fame comes from its patented user-selectable image compression algorithm, which can reduce file size as much as 95 percent. JPEG images are compressed each time they are saved and then decompressed as they are opened and displayed.

Unfortunately, all this compression has a downside. Too much compression damages image quality. This type of compression is called *lossy*, because it loses quality. In fact, the loss in quality is great enough that it can potentially render an image totally useless. Each time designers

save a JPEG image, they face a trade-off between image quality and file size.

Here you see the effects of different amounts of compression on the file size and quality of an image.

PNG

PNG (Portable Network Graphics) was developed in 1995 because of a looming patent dispute involving the GIF format. At the time, it looked as if designers and developers would have to pay a royalty for using the .gif file extension. Although that issue blew over, PNG has found many adherents and a home on the Internet because of its capabilities.

PNG combines many of the features of GIF and JPEG and adds a few of its own. For example, it offers support for unlimited resolution, 32-bit color, and full alpha transparency. It also provides lossless compression, which means you can save an image in PNG format and not worry about losing any quality when you save the file.

The only downside to PNG is that its most important feature—alpha transparency—is not fully supported in older browsers. Luckily, these browsers are retired year after year, so this issue is becoming of little concern to most web designers.

But as with everything on the web, your own needs may vary from the general trends. Before using any specific technology, it's always a good idea to check your site analytics and confirm which browsers your visitors are actually using.

Previewing the completed files

To get a sense of the files you will work on in this lesson, let's preview the completed pages in a browser.

- Launch Adobe Dreamweaver CC (2019 release) or later.
- Define a new site for the lesson08 folder, as described in the "Getting Started" section at the beginning of the book. Name the new site lesson08.

Note

If you have not already downloaded the project files for this lesson to your computer from your Account page, make sure to do so now. See "Getting Started" at the beginning of the book.

. Open **contactus-finished.html** from the lesson08/finished-files folder.

The page includes several images, as well as a Photoshop Smart Object.

. Open **news-finished.html** from the lesson08/finished-files folder.

The news page contains images of varying sizes and composition.

- Close all sample files.

In the following exercises, you will insert these images into these pages using a variety of techniques and format them to work on any screen.

Inserting an image

Images are key components of any webpage, both for developing visual interest and for telling stories. Dreamweaver provides numerous ways to populate your pages with images, using built-in commands and even using copy and paste from other Adobe apps. Let's start with some of the tools built into Dreamweaver itself, such as the Assets panel.

Note

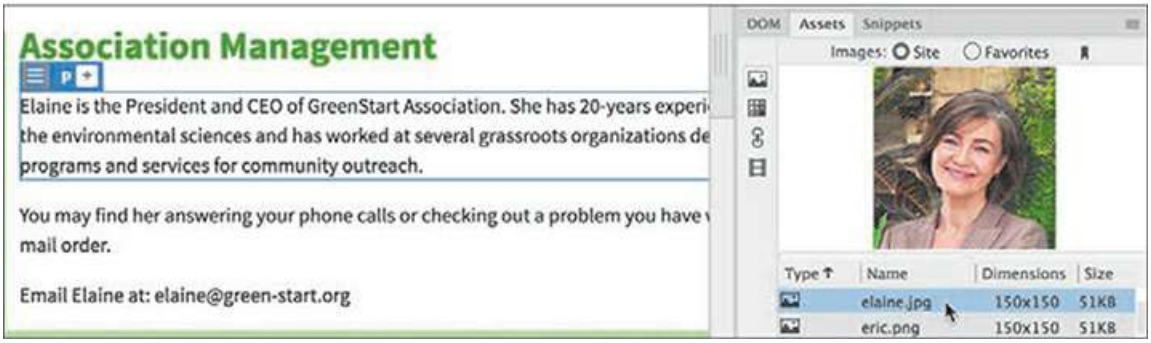
When working with images in Dreamweaver, you should be sure that your site's default images folder is set up according to the directions in the "Getting Started" section at the beginning of the book.

- In the Files panel, open **contact-us.html** in Live view.
- Click the first paragraph under the heading *Association Management*.

  The Element Display appears focused on the p element.
- Choose Window > Assets to display the Assets panel, if necessary. Click the Images category icon  to display a list of all images stored within the site.
- Locate and select **elaine.jpg** in the list.

Tip

The Assets panel should be populated as soon as you define a site and Dreamweaver creates the cache. If the panel is empty, click the Refresh Site List icon.



A preview of **elaine.jpg** appears in the Assets panel. The panel lists the image's name, dimensions in pixels, size in kilo- or megabytes, and file type, as well as its full directory path.

Note

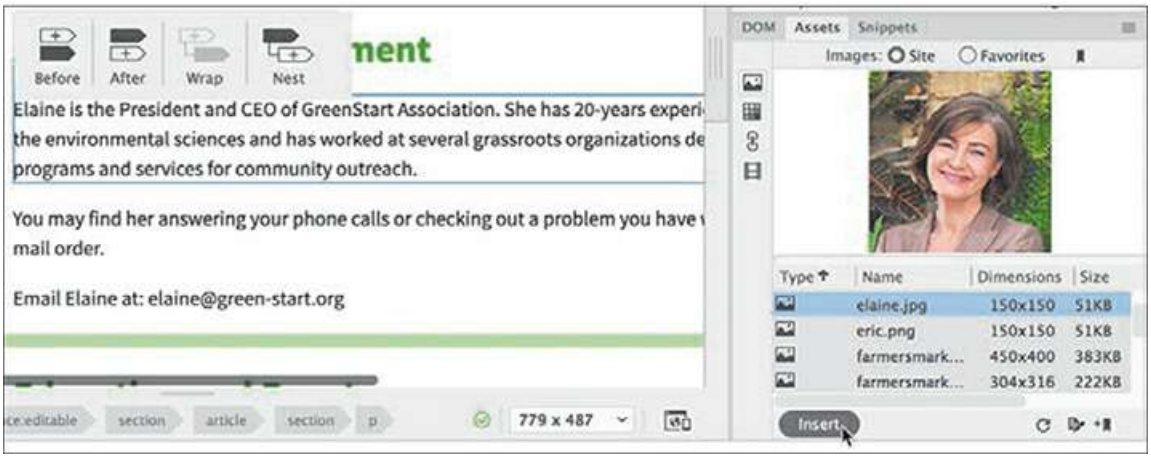
You may need to drag the edge of the panel to widen it to see all the asset information.

. Note the dimensions of the image: 150 pixels by 150 pixels.

. At the bottom of the panel, click the Insert button.

Note

The Images window shows all images stored anywhere in the defined site—even ones outside the site's default images folder—so you may see listings for images stored in the lesson subfolders too.



The Position Assist dialog appears.

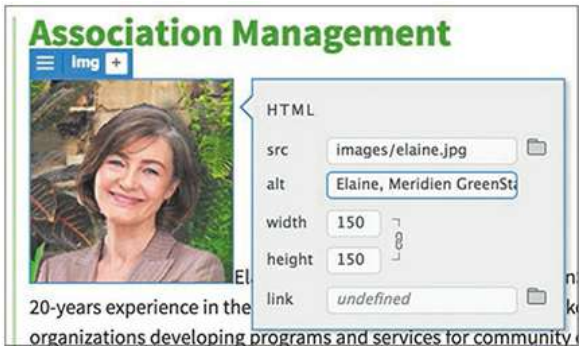
- Click Nest.

The image appears at the beginning of the paragraph. The Element Display now focuses on the img element. You can use the Quick Property inspector to add alt text to the image.

- Click the Edit HTML Attribute icon 

The Quick Property inspector's HTML Attribute dialog appears.

- In the Alt field in the Element Display, enter **Elaine, Meridien GreenStart President and CEO** as the alternate text.



- Choose File > Save.

Note

Alt text provides descriptive metadata about images; in some browsers, alt text may be seen if the image doesn't load properly, or it may be accessed by individuals with visual disabilities.

You inserted Elaine's picture in the text, but it doesn't look very nice at its current position. In the next exercise, you will adjust the image position using a CSS class.

Controlling image positions with CSS classes

The `<img>` element is an inline element by default. That's why you can insert images into paragraphs and other elements. When the image is taller than the font size, the image will increase the vertical space for the line in which it appears. In the past, you could adjust its position using either HTML attributes or CSS, but many of the HTML-based formatting attributes have been deprecated from the language as well as from Dreamweaver. Now you should rely completely on CSS-based techniques.

In this instance, the employee photos will alternate from right to left going down the page and the text will wrap around the image to use the space more effectively. To do this, you'll create a custom CSS class to provide options for left and right alignment. You can use the Element Display to create and apply the new class at the same time.

- If necessary, open **contact-us.html** in Live view.
- Click the image of Elaine in the first paragraph of the Association Management section.

  The Element Display appears focused on the `img` element.
- Click the Add Class/ID icon .
- Type `.flt-rgt` in the text field.

  The new class name is short for "float right," hinting at what CSS command you're going to use to style the images.
- Press Enter/Return.

  The CSS Source dialog appears.
- If necessary, select **mygreen-styles.css** from the Select A Source drop-down menu.

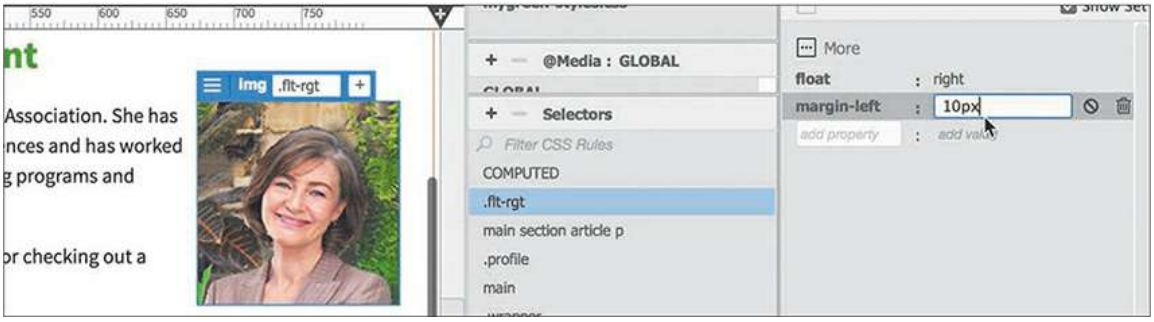
- Press Enter/Return to complete the class.

  The CSS Source dialog disappears, and a new class is created in the style sheet. Let's take a look.
- If necessary, select Elaine's picture.

  In the CSS Designer, click the Current button.

  The new selector appears at the top of the Properties pane.
- Create the following properties:

```
float: right
margin-left: 10px
```



The image moves to the right side of the section element; the text wraps around on the left. As you learned in Lesson 3's online bonus content, "CSS Basics Bonus," applying a float property removes an element from the normal flow of the HTML structure, although it still maintains its width and height.

The margin setting keeps the text from touching the edge of the image. You will create a similar rule to align images to the left in the next exercise.

Working with the Insert panel

The Insert panel duplicates key menu commands and makes inserting images and other code elements both quick and easy. You can even dock it to the top of the document window to have it available all the time. In this exercise, you will use the Insert panel to add an image to the layout.

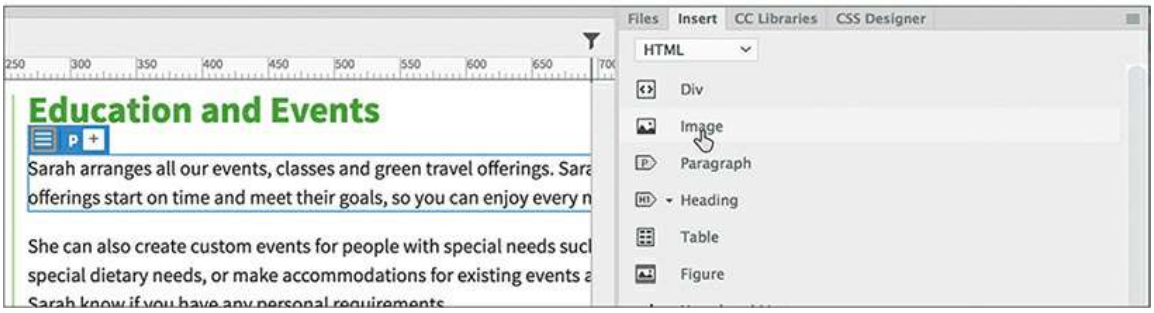
. In Live view, click the first paragraph under the heading *Education and Events*.

The Element Display appears focused on the p tag.

. Choose Window > Insert to display the Insert panel, if necessary.

. In the Insert panel, choose the HTML category.

. Click Image.



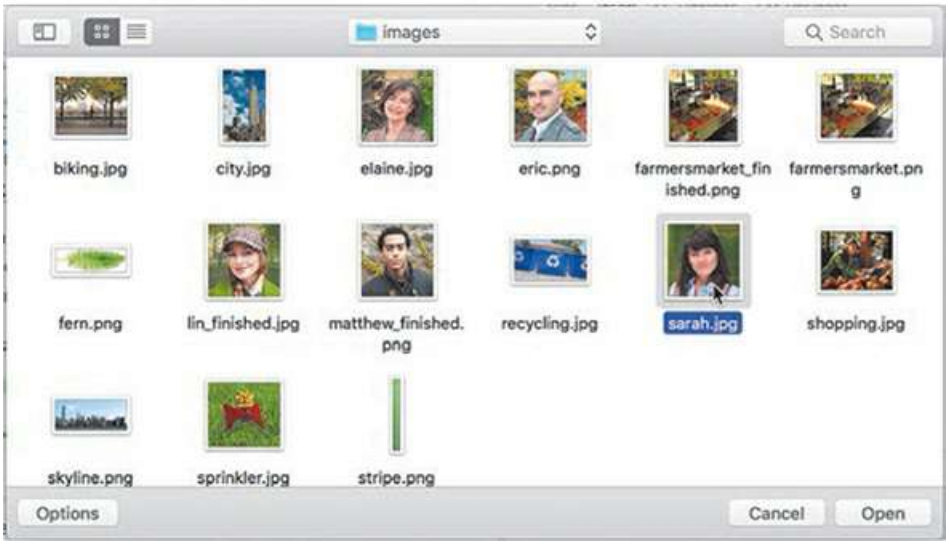
The Position Assist dialog appears.

. Click Nest.

The Select Image Source dialog appears.

- Select **sarah.jpg** from the site images folder.

  Click OK/Open.



- In the Property inspector, enter **Sarah, GreenStart Events Coordinator** in the Alt field.

Now you'll create a new rule for images aligned to the left. In the last exercise, you created the class in the Element Display first. You can also create classes in CSS Designer.

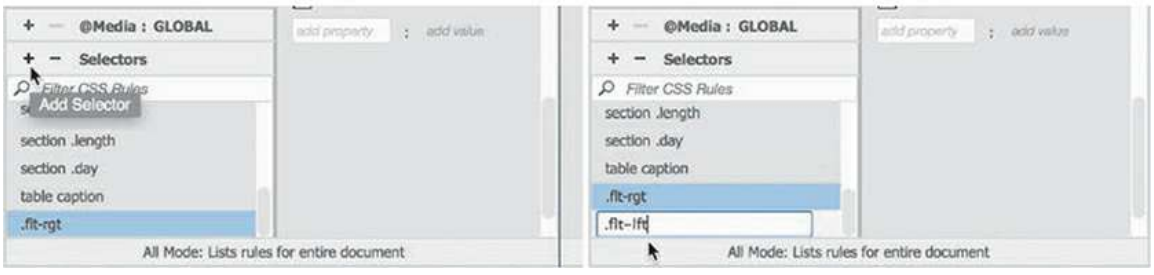
- In CSS Designer, click the All button, if necessary.

  Select the class `.flt-rgt` class.

  If you select a class before creating a new selector, Dreamweaver inserts the new selector directly after the selected rule in the style sheet.

- Click the Add Selector icon ✚.

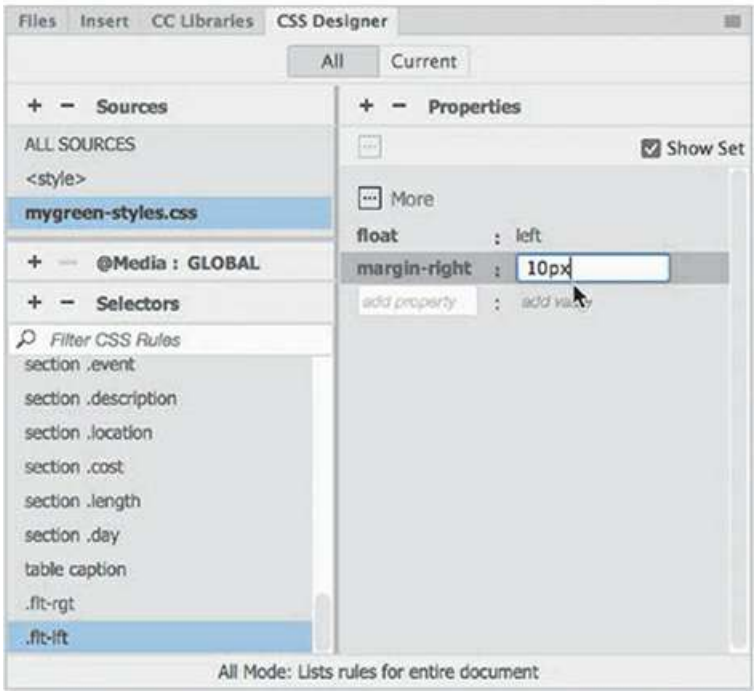
  Type **`.flt-lft`** and press Enter/Return.



The name is short for “float left.”

. Create the following properties in the new rule:

```
float: left
margin-right: 10px
```



. Select Sarah’s image.

. Click the Add Class/ID icon ✚ on Sarah’s image.

Type `.flt-lft` in the field and press Enter/Return.

As you type, the new class name will appear in the hinting menu. Feel free to select the name when you see it in the list. After you apply the class, the image drops down into the paragraph on the left side, with the text wrapping to its right.

- Save the file.

Another way to insert images in your webpage is by using the Insert menu.

Using the Insert menu

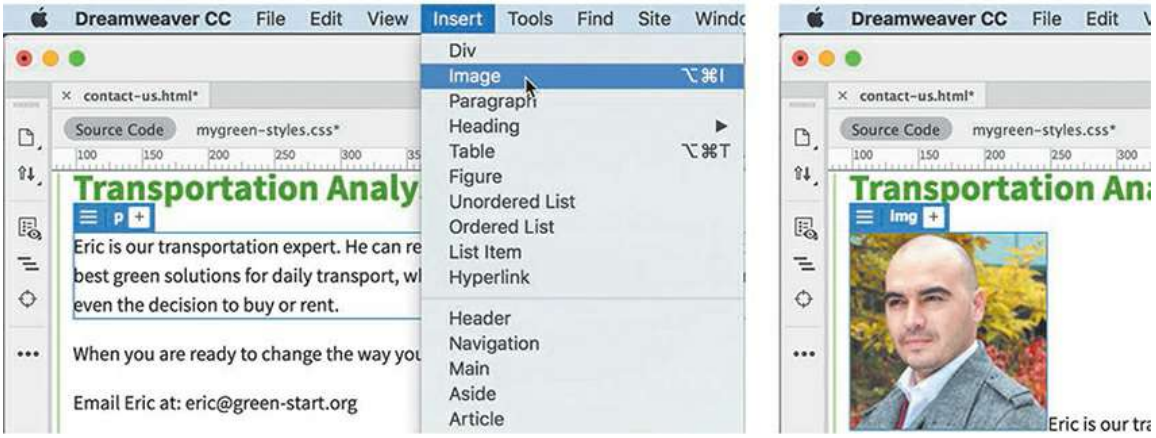
The Insert menu duplicates all the commands you'll find in the Insert panel. Some users find the menu faster and easier to use. Others prefer the ready nature of the panel, which allows you to focus on one element and quickly insert multiple copies of it at once. Feel free to alternate between the two methods as desired or even use the keyboard shortcut. In this exercise, you will use the Insert menu to add images.

- Click the first paragraph under the heading *Transportation Analysis*.
- Choose Insert > Image or press Ctrl+Alt+I/Cmd+Option+I.

  The Position Assist dialog appears.
- Click Nest.

  The Select Image Source dialog appears.
- Navigate to the images folder in lesson08.

  Select the file **eric.png** and click Open.



The **eric.png** image appears in the Dreamweaver layout. Once the classes have been created and defined, you simply have to add the appropriate class using the Element Display.

- Click the Add Class/ID icon , and type the following:

  `.flt-rgt`

  As you type, the class will appear in the hinting menu. You can click the name or use the arrow keys to highlight it, and you can press Enter/Return to select it. As soon as the class is selected, the image floats to the right side of the paragraph.
- In the Property inspector, type **Eric, Transportation Research Coordinator** in the Alt field.

- Save all files.

So far, you have inserted only web-compatible image formats. But Dreamweaver is not limited to the file types GIF, JPEG, and PNG; it can work with other file types too. In the next exercise, you will learn how to insert a Photoshop document (PSD) into a webpage.

Inserting non-web file types

Although most browsers will display only the web-compliant image formats described earlier, Dreamweaver also allows you to use other formats; the program will then automatically convert the file to a compatible format on the fly.

- Click the first paragraph under the heading *Research and Development.*
- Choose Insert > Image.

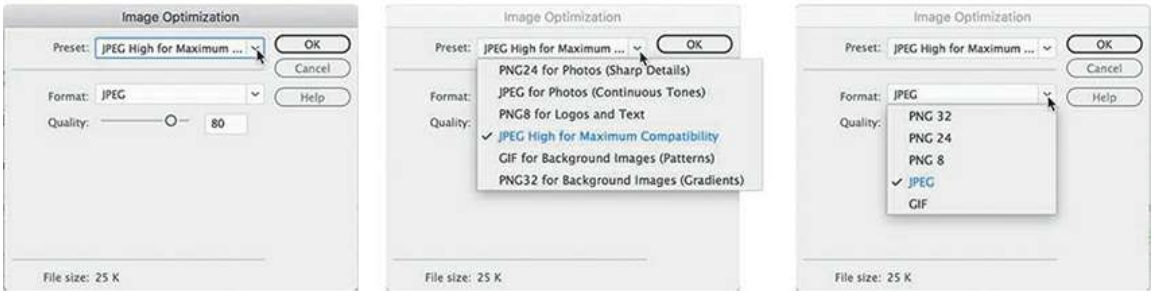
  Nest the image in the first paragraph.

  Navigate to the lesson08/resources folder.

  Select **lin.psd**.
- Click OK/Open to insert the image.

  The Image Optimization dialog appears; it acts as an intermediary that allows you to specify how and to what format the image will be converted.
- Observe the options in the Preset and Format menus.

  The Preset menu allows you to select from six predetermined options that have a proven track record for web-based images. The Format menu allows you to specify your own custom settings from among five options: GIF, JPEG, PNG 8, PNG 24, and PNG 32.
- Choose JPEG High For Maximum Compatibility from the Presets menu. Note the Quality setting.



This Quality setting produces a high-quality image with a moderate amount of compression. If you lower the Quality setting, you automatically increase the compression level and reduce the file size; increase the Quality setting for the opposite effect. The secret to effective design is to select a good balance between quality and compression. The default setting for the JPEG High preset is 80, which is sufficient for your purposes.

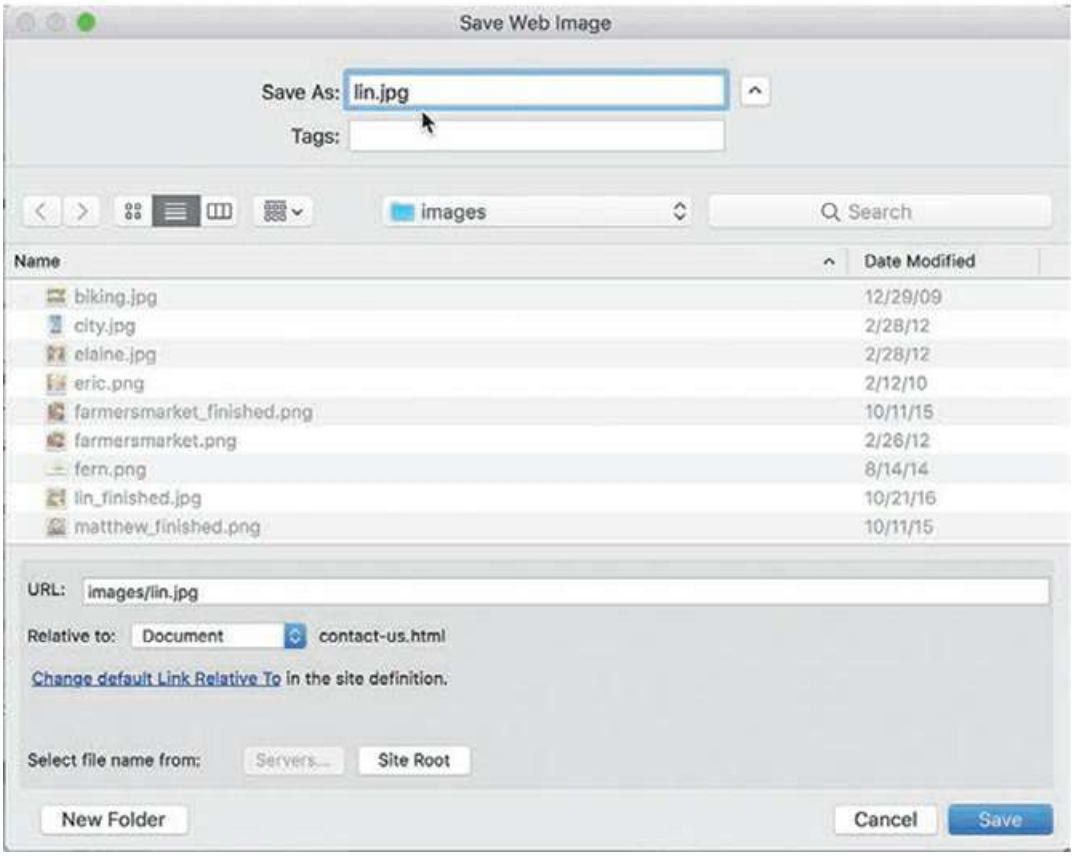
Note

The Image Optimization dialog displays the final file size of the image at the bottom of the dialog.

Note

When an image has to be converted this way, Dreamweaver usually saves the converted image into the site's default images folder. This is not the case when the images inserted are web-compatible. So before you insert an image, you should be aware of its current location in the site and move it to the proper folder first, if necessary.

. Click OK to convert the image.



The Save Web Image dialog appears with the name *lin* entered in the Save As field. Dreamweaver adds the .jpg extension to the file automatically. Be sure to save the file to the default site images folder. If Dreamweaver does not automatically point to this folder, navigate to it before saving the file.

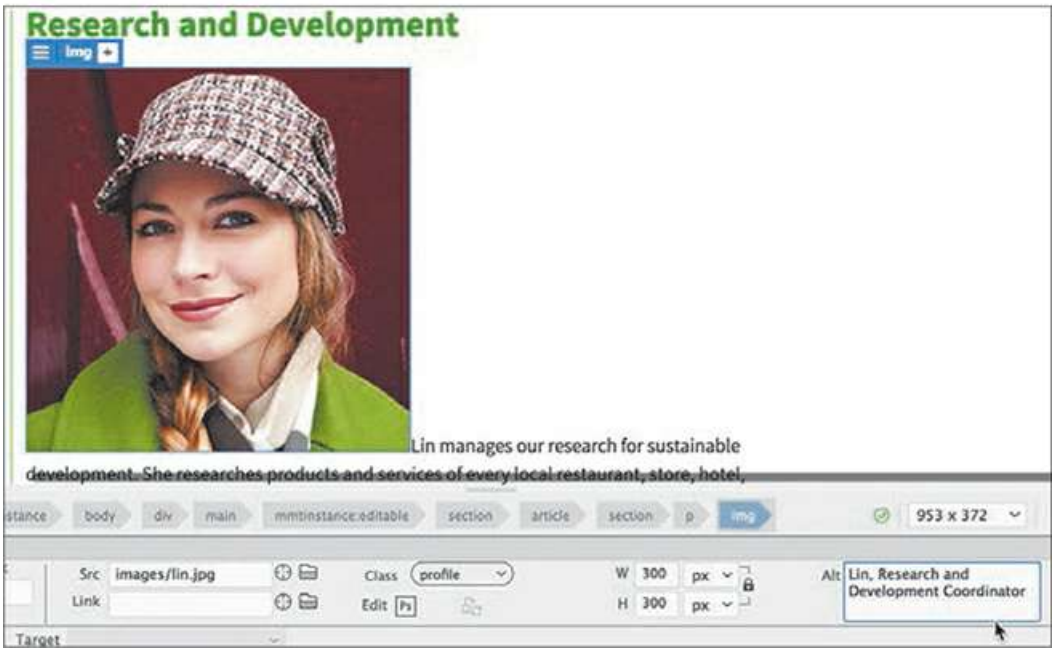
. Click Save.

The Save Web Image dialog closes. The image appears in the layout and is now linked to the JPEG file saved in the default images folder.

Tip

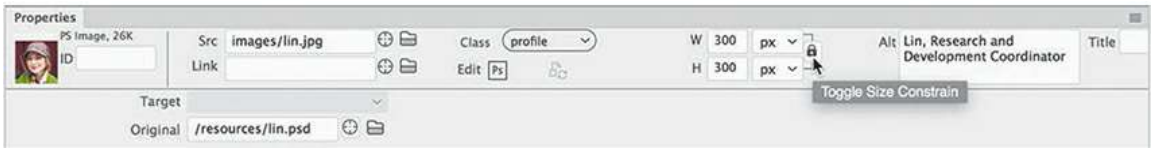
The Element Display and the Property inspector can be used interchangeably to enter alt text.

. Enter **Lin, Research and Development Coordinator** in the Alt field.



The image appears in Dreamweaver at the cursor position. The image has been resampled to 72 ppi but still appears at its original dimensions, so it's larger than the other images in the layout. You can resize the image in the Property inspector.

- If necessary, click the Toggle Size Constrain icon to display the closed lock. Change the Width value to **150px** and press Enter/Return.



Note

Whenever you change HTML or CSS properties, you may need to press Enter/Return to complete the modification.

When the lock icon appears closed, the relationship between width and height is constrained, and the two change proportionally to each other—change one and they both change. The change to the image size is only temporary at the moment, as indicated by the Reset and Commit icons. In other words, the HTML attributes specify the size of the image as 150 pixels by 150 pixels, but the JPEG file holds an image that's still 300 pixels by 300 pixels—four times as many pixels as it needs to have.

- Click the Commit icon

The image is now resized to 150 by 150 pixels permanently.

. Apply the `flt-lft` class to this image using the Element Display. Save all files.

In Live view, the image now appears like the others in the layout; however, this image has a difference. But you can't see it in Live view.

. Switch to Design view.

In Design view, you can now see an icon in the upper-left corner of the image that identifies this image as a Photoshop Smart Object.

. Save all files.

Right size, wrong size

Until the latest mobile devices appeared on the scene, deciding what size and resolution to use for web images was pretty simple. You picked a specific width and height and saved the image at 72 pixels per inch. That's all you needed to do.

But today, web designers want their sites to work well for all visitors, no matter what type or size of device they want to use. So, the days of picking one size and one resolution may be gone forever. But what's the answer? At the moment, there isn't one perfect solution.

One trend simply inserts an image that is larger or has higher resolution and resizes it using CSS. This allows the image to display more clearly on high-resolution screens, like Apple's Retina display. The downside is that lower-resolution devices are stuck downloading an image that's larger than they need. This not only slows the loading of the page for no reason, but it can incur higher data charges for smartphone users.

Another idea is to provide multiple images optimized for different devices and resolutions and use JavaScript to load the proper image as needed. But many users object to using scripts for such basic resources as images. Others want a standardized solution.

So, W3C is working on a technique that uses a new element named, which will not require JavaScript at all. Using this new element, you would select several images and declare how they should be used, and then the browser would load the appropriate image. Unfortunately, this element is so new that Dreamweaver doesn't support it yet, and few browsers even know what it is.

Implementing a responsive workflow for images is outside the scope of this course. In Lesson 14, "Working with a Web Framework," you will learn how to adapt standard web images to a responsive template using CSS and media queries.

Working with Photoshop Smart Objects (optional)

Unlike other images, Smart Objects maintain a connection to their original Photoshop (PSD) file. If the PSD file is altered in any manner and then saved, Dreamweaver identifies those changes and provides the means to update the web image used in the layout. The following exercise can be completed only if you have Photoshop installed on your computer along with Dreamweaver.

. If necessary, open **contact-us.html** in Design view.

Scroll down to the **lin.jpg** image in the *Research and Development* section. Observe the icon in the upper-left corner of the image.

The  icon indicates that the image is a Smart Object. The icon appears only within Dreamweaver itself; visitors see the normal image in the browser, as you saw originally in Live view. If you want to edit or optimize the image, you can simply right-click the image and choose the appropriate option from the context menu.

To make substantive changes to the image, you will have to open it in Photoshop. (If you don't have Photoshop installed, copy lesson08/resources/smartobject/**lin.psd** into the

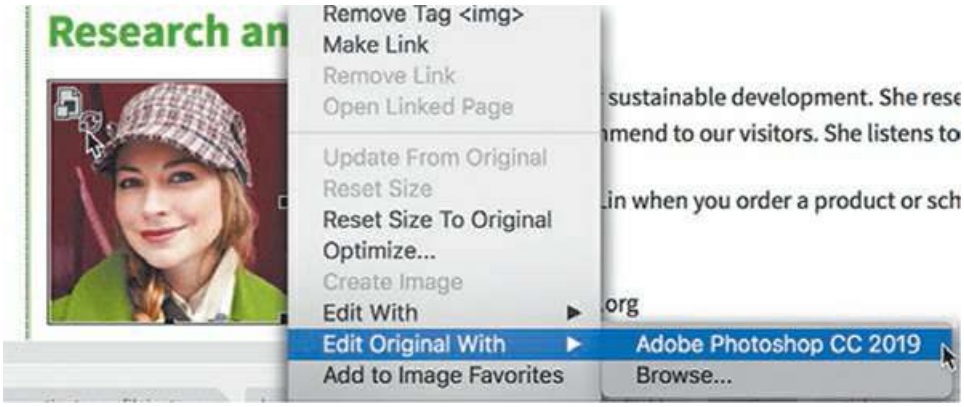
lesson08/resources folder to replace the original image, and then skip to step 6.) In this exercise, you will edit the image background using Photoshop.

Note

The exact name of the apps appearing in the menu may differ depending on your operating system and what version of Photoshop you have installed. If no version of Photoshop is installed at all, you may not see any program listed.

- Right-click the **lin.jpg image.**

  Choose Edit Original With > Adobe Photoshop CC 2019 from the context menu.

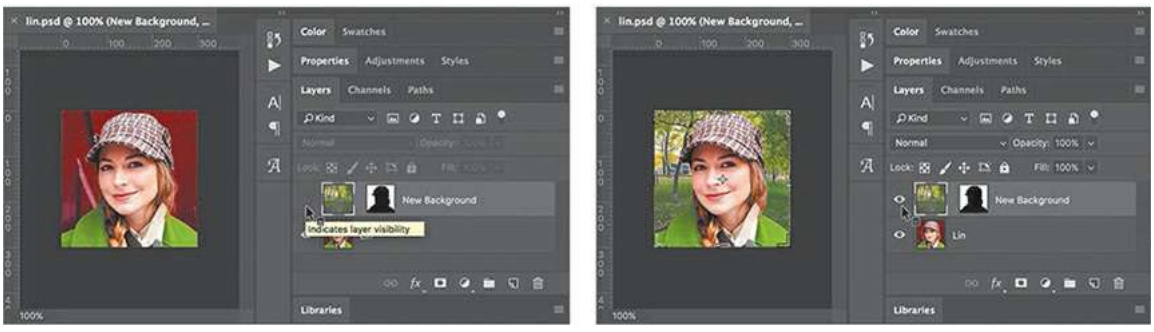


  Photoshop launches—if it is installed on your computer—and loads the file.

- In Photoshop, choose Window > Layers to display the Layers panel, if necessary. Observe the names and states of any existing layers.

  The image has two layers: *Lin* and *New Background*. *New Background* is turned off.

- Click the eye icon for the New Background layer to display its contents.



  The background of the image changes to show a scene from a park.

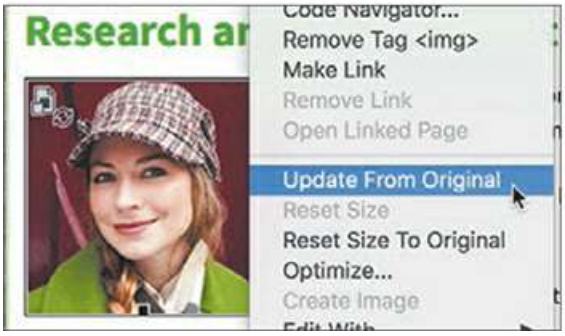
- Save the Photoshop file.

- Switch back to Dreamweaver.

  Position the cursor over the Smart Object icon .

  A tool tip appears indicating that the original image has been modified. You don't have to update the image at this time, and you can leave the out-of-date image in the layout for as long as you want. Dreamweaver will continue to monitor its status as long as it's in the layout. But for this exercise, let's update the image.

- Right-click the image and choose Update From Original from the context menu.



  This Smart Object, and any other instances of it, changes to reflect the new background. You can check the status of the Smart Object by positioning the pointer over the image. A tool tip should appear showing that the image is synced. You can also insert the same original PSD image multiple times in the site using different dimensions and image settings under different filenames. All the Smart Objects will stay connected to the PSD and will allow you to update them as the PSD changes.

- Save the file.

As you can see, Smart Objects have several advantages over a typical image workflow. For frequently changed or updated images, using a Smart Object can simplify updates to the website in the future.

Copying and pasting images from Photoshop (optional)

As you build your website, you will need to edit and optimize many images before you use them in your site. Adobe Photoshop is an excellent program for performing these tasks. A common workflow is to make the needed changes to the images and then manually export the optimized GIF, JPEG, or PNG files to the default images folder in your website. But sometimes simply copying images and pasting them directly into your layout is faster and easier.

Note

You should be able to use any version of Photoshop for this exercise. But Creative Cloud subscribers can download and install the latest version at any time.

- Launch Adobe Photoshop, if necessary.

  Open **matthew.tif** from the lesson08/resources folder.

  Observe the Layers panel.

  The image has only one layer. In Photoshop, by default you can copy only one layer at a time to paste into Dreamweaver. To copy multiple layers, you have to merge or flatten the image first, or you have to use the command Edit > Copy Merged to copy images with multiple active layers.

- Choose Select > All, or press Ctrl+A/Cmd+A, to select the entire image.
- Choose Edit > Copy, or press Ctrl+C/Cmd+C, to copy the image.
- Switch to Dreamweaver. Scroll down to the Information Systems section in **contact-us.html**. Insert the cursor at the beginning of the first paragraph in this section and before the name *Matthew*.
- Press Ctrl+V/Cmd+V to paste the image from the clipboard.

Tip

When inserting images that are outside the default site images folder, Dreamweaver may try to save the image in its original location, which may be outside the site folder. When in doubt, use the Site Root button in the Save As dialog to focus the dialog on the site folder. Then select the images folder from there.

The image appears in the layout with the Image Optimization dialog.

- Choose the preset PNG24 For Photos (Sharp Details), and choose PNG24 from the Format menu. Click OK.

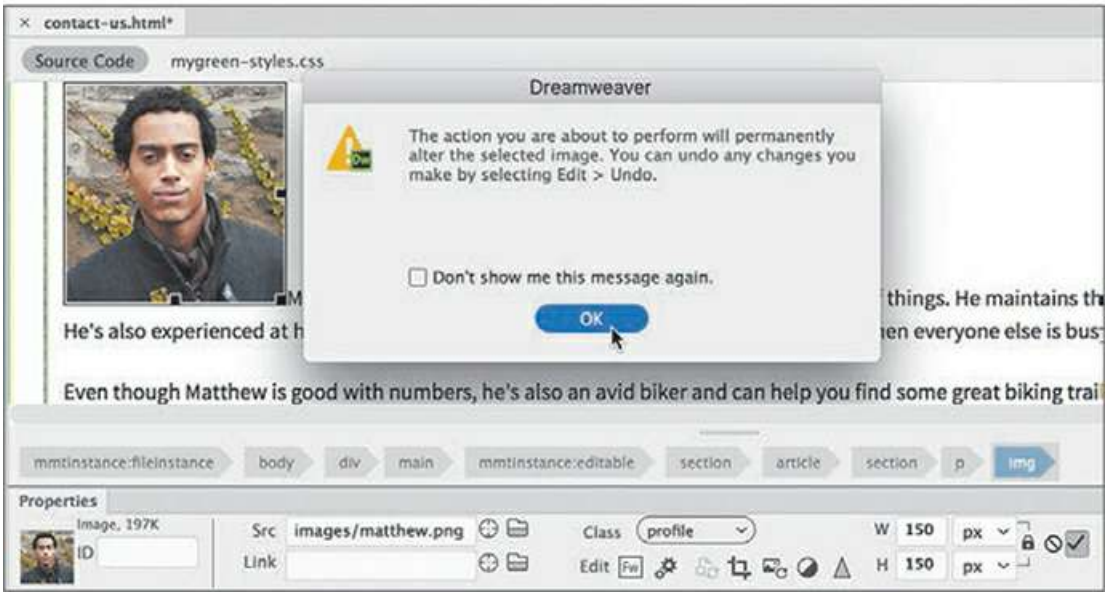
The Save Image dialog appears.

- If necessary, navigate to the default site images folder. Name the image **matthew.png** and select the default site images folder, if necessary. Click Save.

You have now saved the image as a web-compatible PNG file in the site images folder. Just like the image of Lin, Matthew's image is larger than the others.

- Click on the image to select it. In the Properties inspector, change the image dimensions to

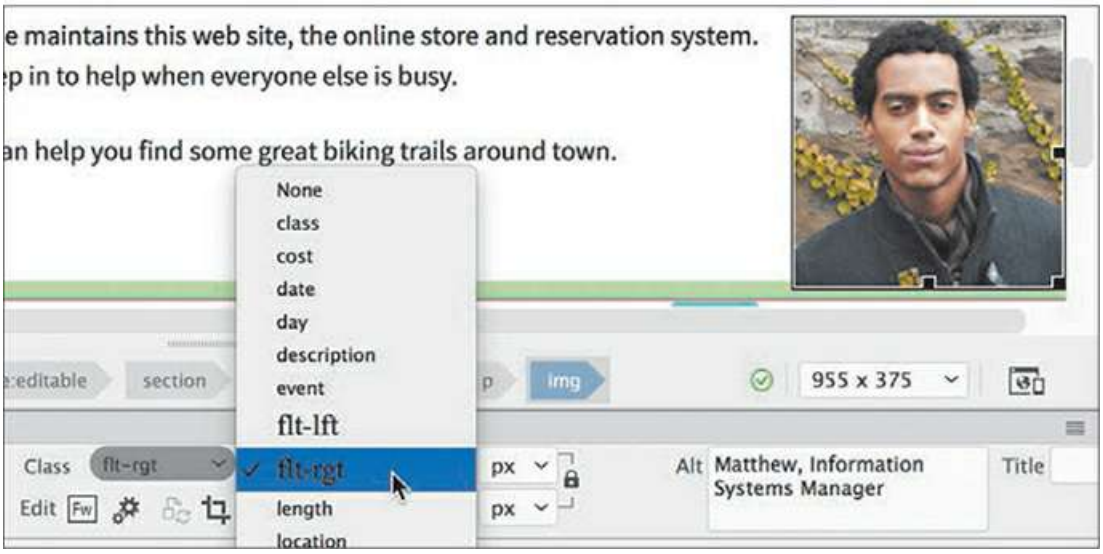
150px by **150px**. Click the Commit icon ✔ to apply the change. Click OK in the dialog that appears, acknowledging that the change is permanent.



Note

Raster images can be scaled down in size without losing quality, but the opposite is not true. Unless a graphic has a resolution higher than 72 ppi, scaling it larger without noticeable degradation may be impossible.

- If necessary, select the image for Matthew and enter **Matthew, Information Systems Manager** in the Alt field in the Property inspector.
- Apply the `flt-rgt` class to **matthew.png** using the Class menu in the Property inspector.



The image appears in the layout at the same size as the other images and aligned to the right. Although this image came from Photoshop, it's not "smart" like a Photoshop Smart Object and can't be updated automatically. It does, however, give you an easy way to load the image into Photoshop or another image editor to perform any modifications.

Note

The exact name displayed in the menu may differ depending on the program version or operating system installed.

. In the layout, right-click **matthew.png**.

Choose Edit With > Photoshop CC 2019 from the context menu.

If Photoshop CC 2019 is not installed, select the program that is displayed.

Tip

If no image-editor program is displayed, you may need to browse for a compatible editor. The executable program file is usually stored in the Program Files folder in Windows and in the Applications folder on a Mac.

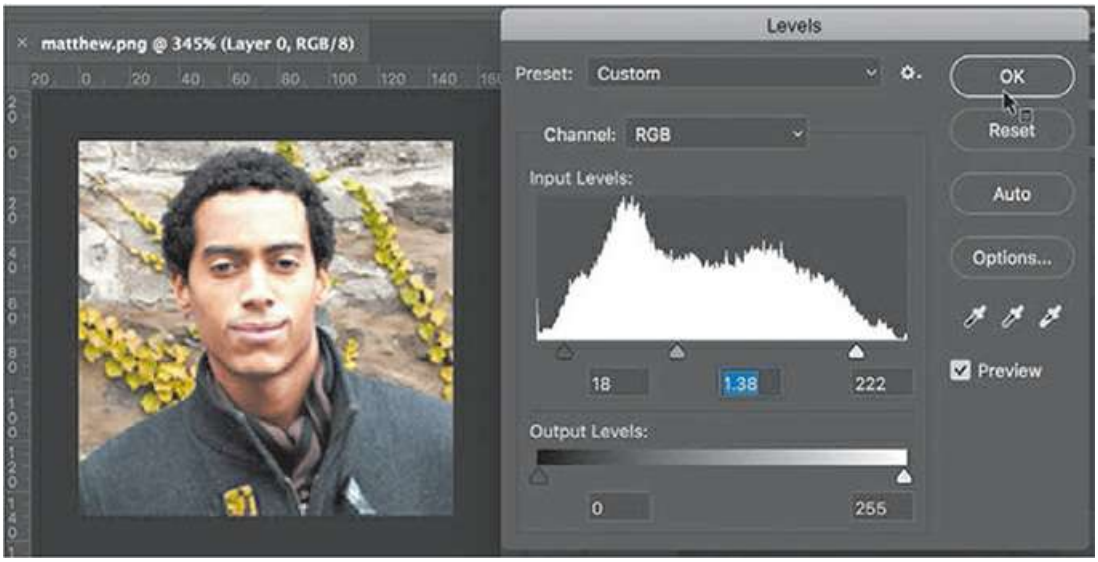
The program launches and displays the PNG file from the site images folder. If you make changes to this image, you merely have to save the file to update the image in Dreamweaver.

. In Photoshop, press Ctrl+L/Cmd+L to open the Levels dialog.

Adjust the brightness and contrast. Save and close the image.

Note

This exercise is geared specifically to Photoshop, but the changes can be made in most image editors.



. Switch back to Dreamweaver.

Scroll down to view the **matthew.png** image in the Information Systems section.

The image should be updated in the layout automatically. Since you saved the changes under the original filename, no other action is necessary. This method saves you several steps and avoids any potential typing errors.

Note

Although Dreamweaver automatically reloads any modified file, most browsers won't. You will have to refresh the browser display before you see any changes.

- Save all files.

In the next exercise, you will insert an image using drag and drop.

Inserting images by drag and drop

Most of the programs in Creative Cloud offer drag-and-drop capabilities. Dreamweaver is no exception.

- Open **news.html** from the site root folder in Live view.
- Choose Window > Assets to display the Assets panel, if necessary.

  The Assets panel may not be opened by default in the Dreamweaver workspace. You can leave it as a floating dialog or dock it to keep it out of the way.
- If necessary, drag the Assets panel to dock it beside the Files or DOM tab.
- In the Assets panel, click the Images category icon .

Tip

Tip: If you don't see specific image files listed in the Assets panel, click the Refresh icon to reload site images.

- Drag the **skyline.png** icon from the panel and position the cursor between the first paragraph and the heading *Green Buildings earn more Green*.

If you position the cursor correctly, you will see a green line between the heading and the paragraph, indicating where the image will be inserted once you release the mouse.

Unlike the images used in the previous exercises, **skyline.png** was inserted between the `<h2>` and `<p>` elements. It is not part of any of the paragraphs, so no float command is needed.

- Enter **Green buildings are top earners** in the Property inspector's Alt field.
- Save all files.

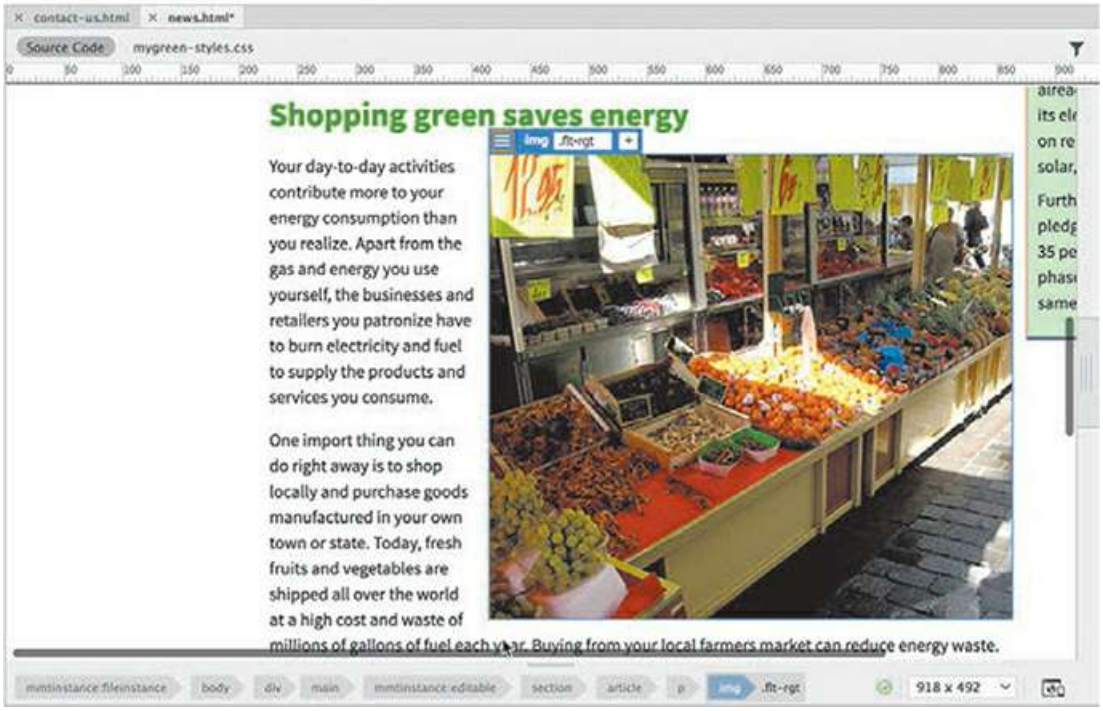
For users who do not have Photoshop or another image editor, Dreamweaver provides tools for basic image processing.

Optimizing images with the Property inspector

Optimized web images try to balance image dimensions and quality against file size. Sometimes you may need to optimize graphics that have already been placed on the page. Dreamweaver has built-in features that can help you achieve the smallest possible file size while preserving image quality. In this exercise, you'll use tools in Dreamweaver to scale, optimize, and crop an image for the web.

- If necessary, open **news.html** in Live view or switch to it.
- Click to select the first paragraph below the *Shopping green saves energy* heading.
- Choose Insert > Image. Click Nest in the Position Assist dialog.

  Select **farmersmarket.png** from the site images folder. Click Open.
- Enter **Buy local to save energy** in the Alt field.
- Apply the `.flt-rgt` class to the image.



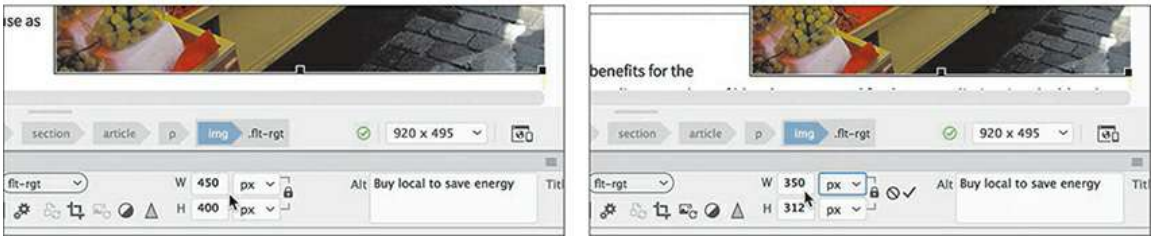
The image is too large, and there's barely any room for it in the column. It could really use some resizing and cropping. Dreamweaver's built-in tools work only in Design view.

- Switch to Design view and observe the Property inspector.

Whenever an image is selected, image-editing tools appear below the Class menu in the Property inspector. The icons allow you to edit the image in Photoshop or Adobe Fireworks or to adjust several aspects in place. See the sidebar "Dreamweaver's graphic tools" at the end of the lesson for an explanation of each tool.

There are two ways to reduce the dimensions of an image in Dreamweaver. The first method changes the size of the image temporarily by imposing user-defined dimensions.

- Select **farmersmarket.png**. If necessary, click the Toggle Size Constrain icon in the Property inspector to lock the image proportions. Change the image width to **350 pixels** and press the Tab key.



When the size constraint is locked, the height automatically conforms proportionally to the new width. Note that Dreamweaver indicates that the new size is not permanent by displaying the current specifications in bold and by displaying the Reset and Commit icons.

- Click the Commit icon .

A dialog appears that indicates the change will be permanent.

- Click OK.

Dreamweaver can also crop images.

- With the image still selected, click the Crop icon in the Property inspector.

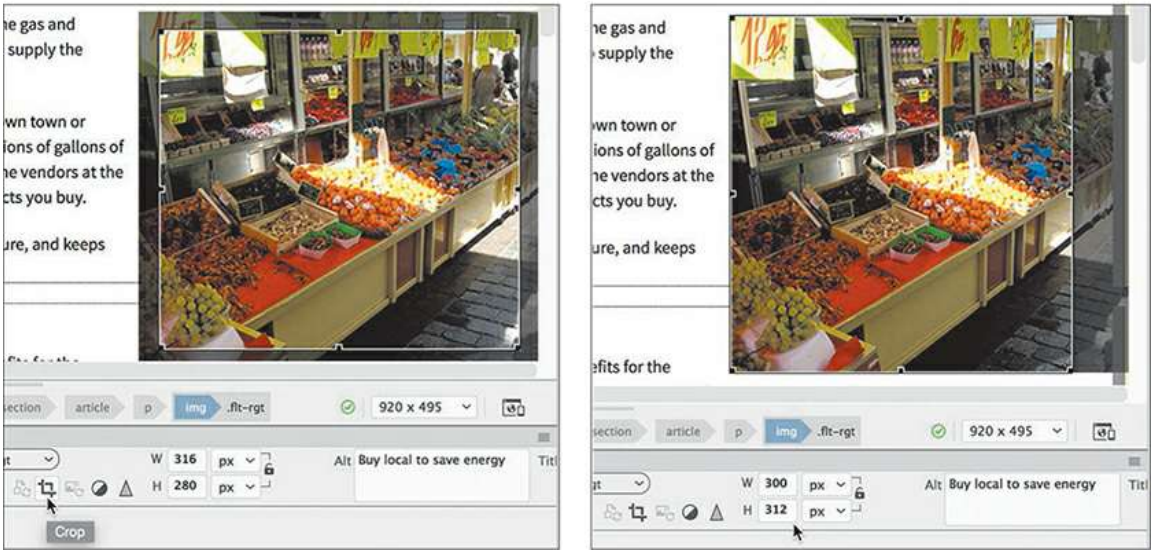
A dialog appears indicating that the action will permanently change the image.

- Click OK.

Crop handles appear slightly inset from the edges of the image. You want to crop the width but not the height.

> **Tip**
>
> Dimensions may also be entered manually if you know the final proportions.

. Drag the crop handles to set the image to a width of 300 pixels and a height of 312 pixels.



. Press Enter/Return to apply the change.

. Save all files.

Most designers edit and resize images prior to bringing them into Dreamweaver, but it's nice to know that these tools are available for any last-minute changes or fast turnarounds.

In this lesson, you learned how to insert images and Photoshop Smart Objects into a Dreamweaver page, copy and paste from Photoshop, and use the Property inspector to edit and resample images.

There are numerous ways to create and edit images for the web. The methods examined in this lesson show but a few of them and are not meant to recommend or endorse one method over another. Feel free to use whatever methods and workflow you desire based on your own situation and expertise.

Dreamweaver's graphic tools

All Dreamweaver's graphic tools appear in the Property inspector when an image is selected in Design view. Here are the seven tools:

Edit—Opens the selected image in the defined external graphics editor if you have one installed. You can assign a graphics-editing program to any given file type in the File Types/Editors category of the Preferences dialog. The button's image changes according to the program chosen. For example, if Fireworks is the designated editor for the image type, a Fireworks icon is shown; if Photoshop is the editor, you'll see a Photoshop icon. If neither app is installed, you will see a generic edit icon.

Edit Image Settings—Opens the Image Optimization dialog, allowing you to apply user-defined optimization specifications to the selected image.

Update From Original—Updates any placed Smart Object to match any changes to the original source file.

Crop—Permanently removes unwanted portions of an image. When the Crop tool is active, a bounding box with a series of control handles appears within the selected image. You can adjust the bounding box size by dragging the handles or by entering the final dimensions. When the box outlines the desired portion of the image, press Enter/Return or doubleclick the graphic to apply the cropping.

Resample—Permanently resizes an image. The Resample tool is active only when an image has been resized.

Brightness And Contrast—Offers user-selectable adjustments to an image's brightness and contrast; a dialog presents sliders for each value that can be adjusted independently. A live preview is available so that you can evaluate adjustments before committing to them.

Sharpen—Affects the enhancement of image details by raising or lowering the contrast of pixels on a scale from 0 to 10. Like the Brightness And Contrast tool, Sharpen offers a real-time preview.

You can undo most graphics operations by choosing Edit > Undo until the containing document is closed or you quit Dreamweaver.

Review questions

1. What are the three factors that determine raster image quality?
2. What file formats are specifically designed for use on the web?
3. Describe at least two methods for inserting an image into a webpage using Dreamweaver.
4. True or false: All graphics have to be optimized outside of Dreamweaver.

5. What is the advantage of using a Photoshop Smart Object over copying and pasting an image from Photoshop?

Review answers

1. Raster image quality is determined by resolution, image dimensions, and color depth.
2. The compatible image formats for the web are GIF, JPEG, PNG, and SVG.
3. One method to insert an image into a webpage using Dreamweaver is to use the Insert panel. Another method is to drag the graphic file into the layout from the Assets panel. Images can also be copied and pasted from Photoshop and Fireworks.
4. False. Images can be optimized even after they are inserted into Dreamweaver by using the Property inspector. Optimization can include rescaling, changing format, or fine-tuning format settings.
5. A Smart Object can be used multiple times in different places on a site, and each instance of the Smart Object can be assigned individual settings. All copies remain connected to the original image. If the original is updated, all the connected images are immediately updated as well. When you copy and paste all or part of a Photoshop file, however, you get a single image that can have only one set of values applied to it.

9 Working with Navigation

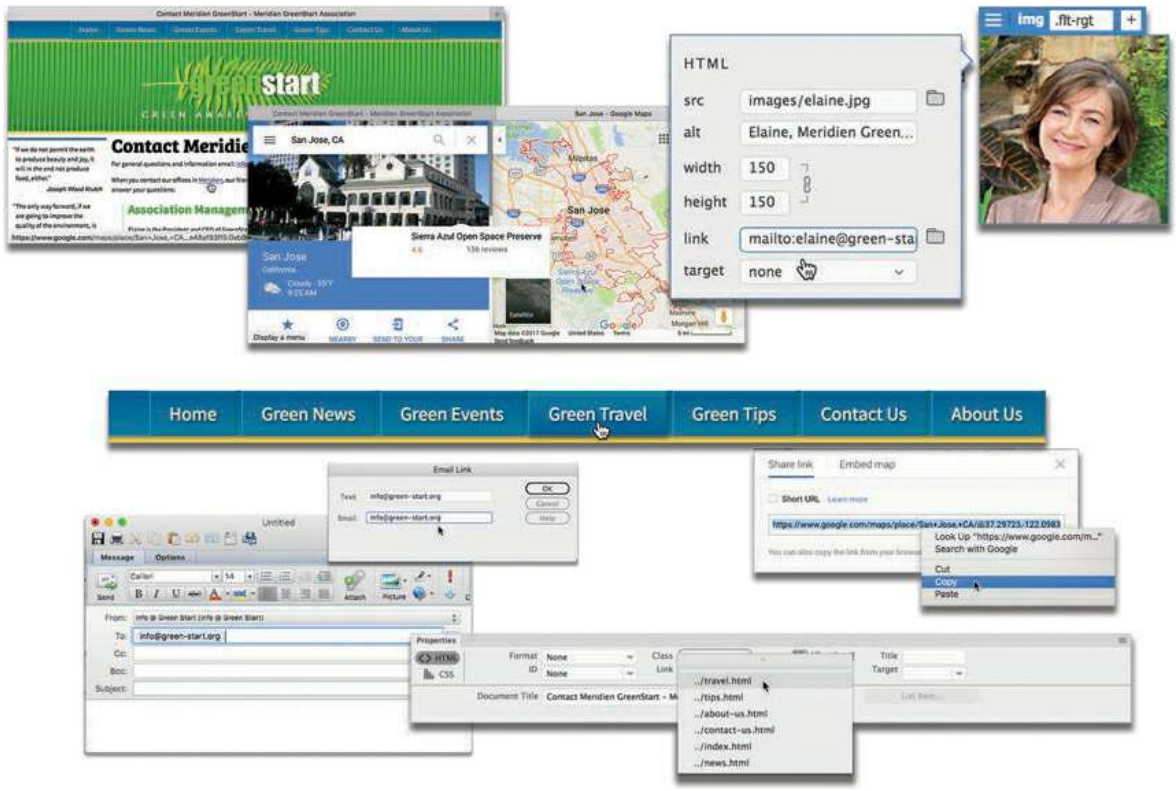
Lesson overview

In this lesson, you'll learn how to do the following:

- Create a text link to a page within the same site
- Create a link to a page on another website
- Create an email link
- Create an image-based link
- Create a link to a location within a page

This lesson will take 1 hour and 15 minutes to complete. Please log in to your account on peachpit.com to download the project files for this lesson, as described in the "Getting Started" section at the beginning of this book. Follow the instructions under "Accessing the Lesson Files and Web Edition." Define a site based on the lesson09 folder.

Your Account page is also where you'll find any updates to the lesson files. Look on the Lesson&Update Files tab to access the most current content.



Dreamweaver can create and edit many types of links—from text-based links to image-based links— and does so with ease and flexibility.

Hyperlink basics

The World Wide Web, and the Internet in general, would be a far different place without the hyperlink. Without hyperlinks, HTML (HyperText Markup Language) would simply be ML (Markup Language). The *hypertext* in the name refers to the functionality of the hyperlink. So what is a hyperlink?

A hyperlink, or *link*, is an HTML-based reference to a resource available on the Internet or within the computer hosting a web document. The resource can be anything that can be stored on and displayed by a computer, such as a webpage, an image, a movie, a sound file, a PDF—in fact, almost any type of computer file. A hyperlink creates an interactive behavior specified by HTML and CSS, or by the programming language you're using, and is enabled by a browser or other application.

An HTML hyperlink consists of the hey have one thing in common: they are enabled in HTML by the `<a>` *anchor* element. This element designates the address of the destination of the hyperlink and can then specify how it functions using several attributes. You'll learn how to create and modify the `<a>` element in the exercises that follow.

Internal and external hyperlinks

The simplest hyperlink—an internal hyperlink—takes the user to another part of the same document or to another document stored in the same folder or hard drive on the web server that hosts the site. An external hyperlink is designed to take the user to a document or resource outside your hard drive, website, or web host.

Internal and external hyperlinks may work differently, but they have one thing in common: they are enabled in HTML by the `<a>` *anchor* element. This element designates the address of the destination of the hyperlink and can then specify how it functions using several attributes. You'll learn how to create and modify the `<a>` element in the exercises that follow.

Relative vs. absolute hyperlinks

A hyperlink address can be written in two ways. When you refer to a target according to where it is stored in relation to the current document, it is known as a *relative* link. This is like telling a friend that you live next door to the blue house. If she were driving down your street and saw the blue house, she would know where you live. But those directions don't really tell her how to get to your house or even to your neighborhood. A relative link frequently will consist of the resource name and per-haps the folder it is stored within, such as news.`html or content/news.html.` Sometimes you need to spell out precisely where a resource is located. In those instances, you need an *absolute* hyperlink. This is like telling someone you live at 123 Main Street in Meridien. This is typically how you refer to resources outside your website. An absolute link includes the entire uniform resource locator, or URL, of the target and may even include a filename—such as http://www.adobe.com/products/dreamweaver.html—or just a folder within the site.

Both types of links have advantages and disadvantages. Relative hyperlinks are faster and easier to write, but they may not work if the document containing them is saved in a different folder or location in the website. Absolute links always work no matter where the containing document is saved, but they can fail if the targets are moved or renamed. A simple rule that most web

designers follow is to use rela-tive links for resources within a site and absolute links for resources outside the site. Of course, whether you follow this rule or not, it's important to test all links before deploying the page or site.

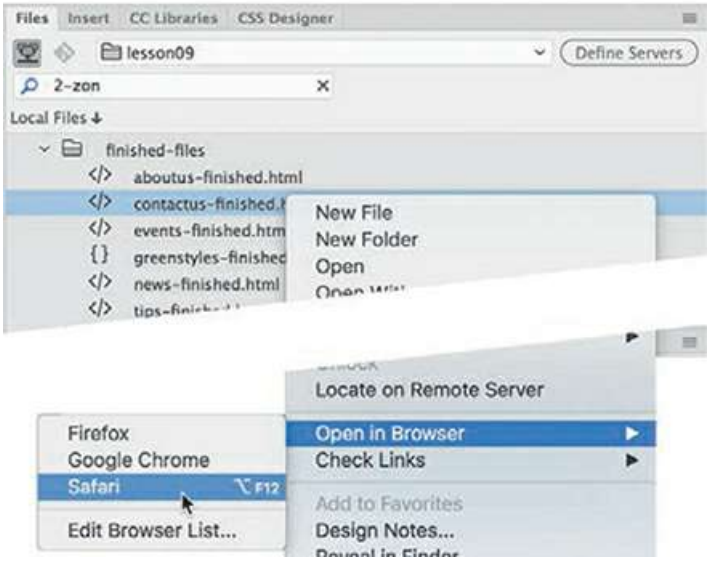
Previewing the completed file

To see the final version of the file you will work on in this lesson, let's preview the completed page in the browser.

Note

Before beginning this exercise, download the project files and define a new site based on the lesson09 folder using the instructions in the "Getting Started" section at the beginning of the book.

- Launch Adobe Dreamweaver CC (2019 release) or later.
- If necessary, press F8 to open the Files panel. Select lesson09 from the site list.
- In the Files panel, expand the lesson09 folder.
- In the Files panel, navigate to lesson09/finished-files folder and right-click **aboutus-finished.html**. Choose Open In Browser from the context menu, and select your favorite browser.



The **aboutus-finished.html** file appears in your default browser. This page features only internal links in the horizontal menu.

- Position the cursor over the horizontal navigation menu. Hover over each button and examine

the behavior of the menu.

The menu is the same one created and formatted in Lesson 5, "Creating a Page Layout," with a few changes.

- Click the *Green News* link. The browser loads the finished *Green News* page.
- Position the cursor over the *Contact Us* link.

  Observe the browser to see whether it's displaying the link's destination anywhere on the screen.

Typically, the browser shows the link destination in the status bar

---

Tip

Most browsers display the destination of a hyperlink in the status bar at the bottom of the browser window. In some browsers, this status bar may be turned off by default.

- Click the *Contact Us* link.

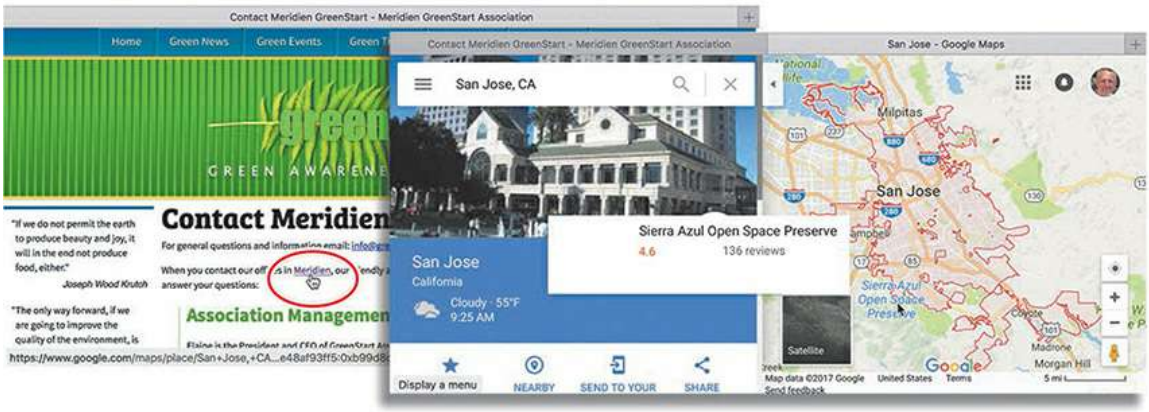
  The browser loads the finished *Contact Us* page, replacing the *Green News* page. The new page includes internal, external, and email links.
- Position the cursor over the *Meridien* link in the second paragraph of the main content area. Observe the status bar.

  The status bar displays an http://google.com/maps link.

Note

The display in Google Maps may differ from the one pictured.

- Click the *Meridien* link.



A new browser window appears and loads Google Maps. The link is intended to show the visitor where the Meridien GreenStart Association offices are located. If desired, you can even include address details or the company name in this link so that Google can load the exact map and directions.

Note that the browser opens a separate window or document tab when you click the link. This is a good behavior to use when directing visitors to resources outside your site. Since the link opens in a separate window, your own site is still open and ready to use. This practice is especially helpful if your visitors are unfamiliar with your site and may not know how to get back to it once they click away.

- Close the Google Maps window.

The *Contact Us* page is still open in the browser. Note that each employee has a link applied to their email address.

- Click an email link for one of the employees.

The default mail application launches on your computer. If you have not set up this application to send and receive mail, the program will usually start a wizard to help you set up this functionality. If the email program is set up, a new message window appears with the email address of the employee automatically entered in the To field.

- Close the new message window, if necessary, and exit the email program.
- Scroll down to the *Education and Events* section. Note that the menu sticks to the top of the page as you scroll down.
- Click the *events* link.

The browser loads the *Green Events and Classes* page. The browser focuses on the table containing the list of upcoming events near the top of the page. Notice that the horizontal menu is still visible at the top of the browser.

- Click the *Classes* link in the first paragraph.

The browser jumps down to the list of upcoming classes at the bottom of the page.

- Click the *Return to Top* link that appears above the class schedule. You may need to scroll up or down the page to see it.

The browser jumps back to the top of the page.

. Close the browser and switch to Dreamweaver, if necessary.

You have tested a variety of different types of hyperlinks: internal, external, relative, and absolute. In the following exercises, you will learn how to build each type.

Note

Many web visitors don't use email programs installed on their computers. They use web-based services such as AOL, Gmail, Hotmail, and so on. For these visitors, email links like the one you tested won't work. The best option is to create a web-hosted form on your site that sends the email to you via your own server.

Creating internal hyperlinks

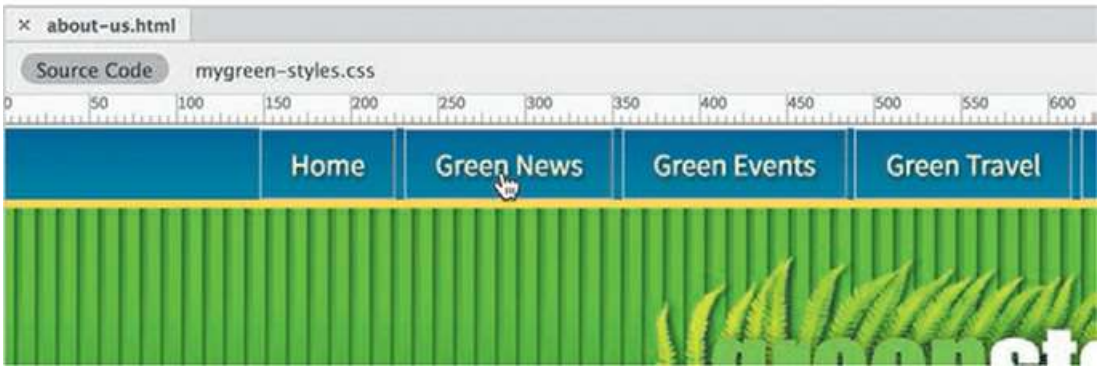
Creating hyperlinks of all types is easy with Dreamweaver. In this exercise, you'll create relative text-based links to pages in the same site, using a variety of methods. You can create links in Design view, Live view, and Code view.

Creating relative links

Dreamweaver provides several methods for creating and editing links. Links can be created in all three program views.

. Open **about-us.html** from the site root folder in Live view.

. In the horizontal menu, position the cursor over any of the horizontal menu items. Observe the type of cursor that appears.



The pointer indicates that the menu item is structured as a hyperlink. The links in the horizontal menu are not editable in the normal way, but this is something you can actually see only in Design view.

- Switch to Design view.

  Position the cursor over any item of the horizontal menu again.

  The “no” symbol appears, indicating that this section of the page is uneditable. The horizontal menu was not added to any of the editable regions you created in Lesson 6, “Working with Templates.” That means it’s considered part of the template and is locked within Dreamweaver. To add hyperlinks to this menu, you’ll have to open the template.

- Choose Window > Assets. Click the Templates icon in the Assets panel. Right-click **mygreen-temp** and choose Edit from the context menu.
- Switch to Design view, if necessary.

  In the horizontal menu, insert the cursor into the *Green News* link.

  The horizontal menu is editable in the template.

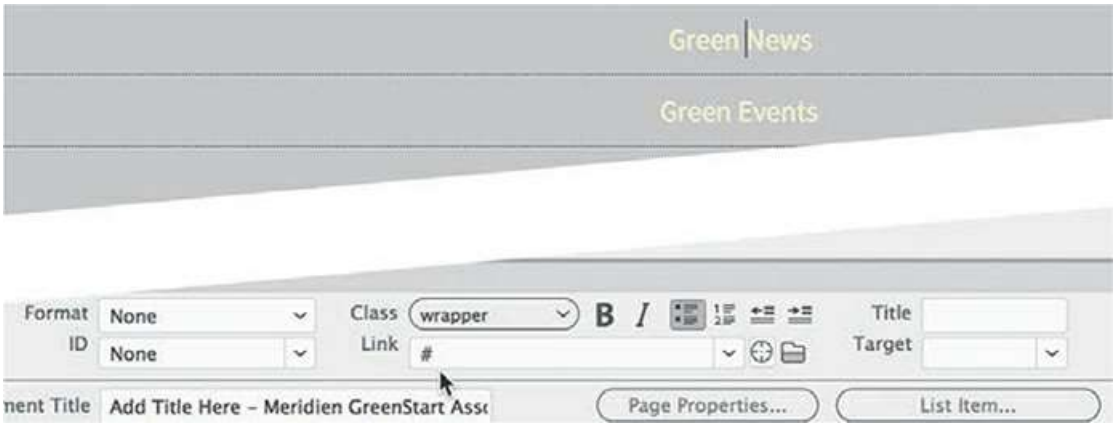
Note

The Template category is not visible in Live view. You will see it only in Design view and Code view or when no document is open.

- If necessary, choose Window > Properties to open the Property inspector. Examine the contents of the Link field in the Property inspector.

Tip

When editing or removing an existing hyperlink, you don’t need to select the entire link; you can just insert the cursor anywhere in the link text. Dream-weaver assumes you want to change the entire link by default.

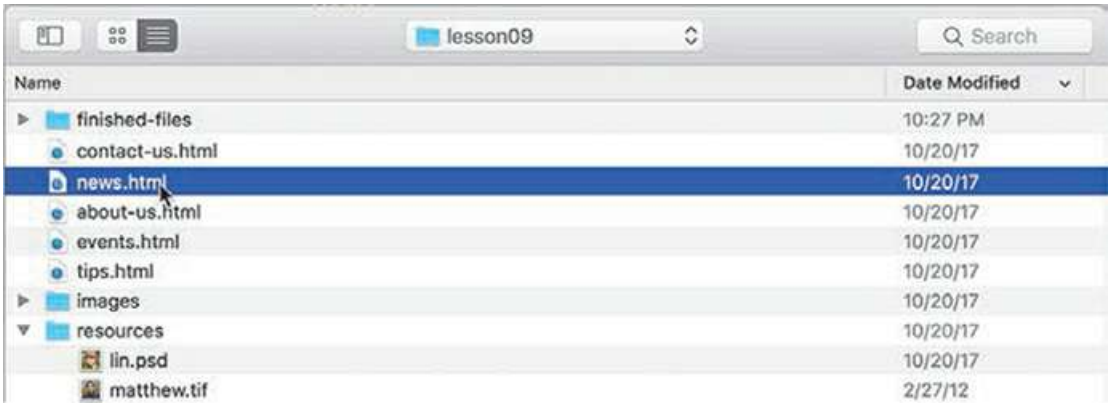


To create links, the HTML tab must be selected in the Property inspector. The Link field shows a hyperlink placeholder (#).

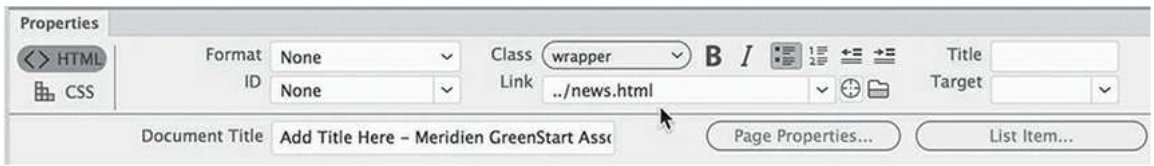
. In the Link field, click the Browse For File icon

A file selection dialog appears.

. Navigate to the site root folder, if necessary. Select **news.html** from the site root folder.



. Click OK/Open.



Note

The link won't have the typical hyperlink appearance—a blue underscore—because of the special formatting you applied to this menu in Lesson 5.

The link `../news.html` appears in the Link field in the Property inspector. You've created your first text-based hyperlink.

Since the template is saved in a subfolder, Dreamweaver adds the path element notation (../) to the filename. This notation tells the browser or operating system to look in the parent directory of the current folder.

This is necessary if the child page is saved in a subfolder, but it is unnecessary if the page is saved in the root of the site. Luckily, when you create a page from the template, Dreamweaver rewrites the link, adding or removing path information as needed.

If desired, Dreamweaver enables you to type links in the field manually.

. Insert the cursor in the *Home* link.

The home page does not exist yet. But that doesn't stop you from entering the link text by hand.

. In the Property inspector Link field, select the hash (#) symbol, type `../index.html` to replace the placeholder, and press Enter/Return.

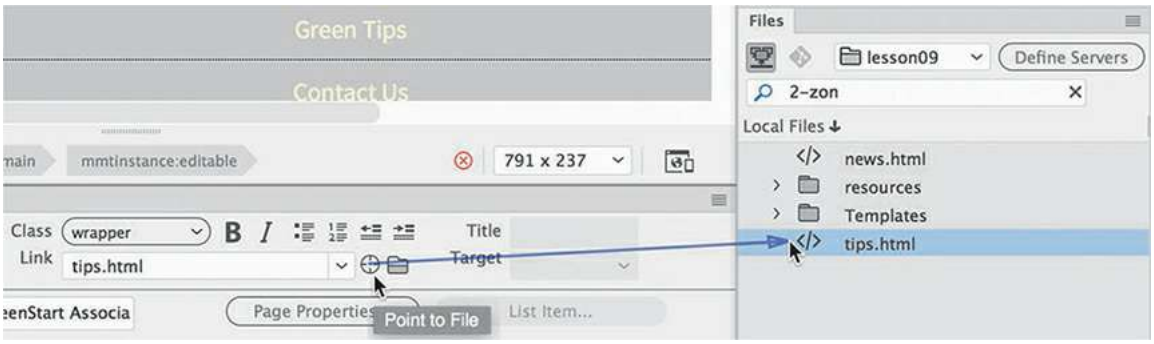
At any time, you may insert a link by typing it manually just this way. But entering links by hand can introduce a variety of errors that can break the very link you are trying to create. If you want to link to a file that already exists, Dream-weaver offers other interactive ways to create links.

. Insert the cursor in the *Green Tips* link.

. Click the Files tab to bring the panel to the top, or choose Window > Files.

You need to make sure you can see the Property inspector and the target file in the Files panel.

. In the Property inspector, drag the Point To File icon ⊕—next to the Link field—to **tips.html** in the site root folder displayed in the Files panel.



Dreamweaver enters the filename and any necessary path information into the Link field.

Tip

If a folder in the Files panel contains a page you want to link to but the folder is not open, drag the Point To File icon over the folder and hold it in place to expand that folder so that you can point to the desired file.

- Create the rest of the links as shown below using any of the methods you've learned:

  Green Travel: **../travel.html**

  Contact Us: **../contact-us.html**

  About Us: **../about-us.html**

Note

The **travel. html** file has not been created yet. You'll have to create the link manually, as you did with **index.html**. You will create the *Green Events* link later.

For files that have not been created, you will always have to enter the link manually. Remember that all the links added to the template pointing to files in the site root folder must include the ../ notation so that the link resolves properly. Remember also that Dreamweaver will modify the link as needed once the template is applied to the child page.

Creating a home link

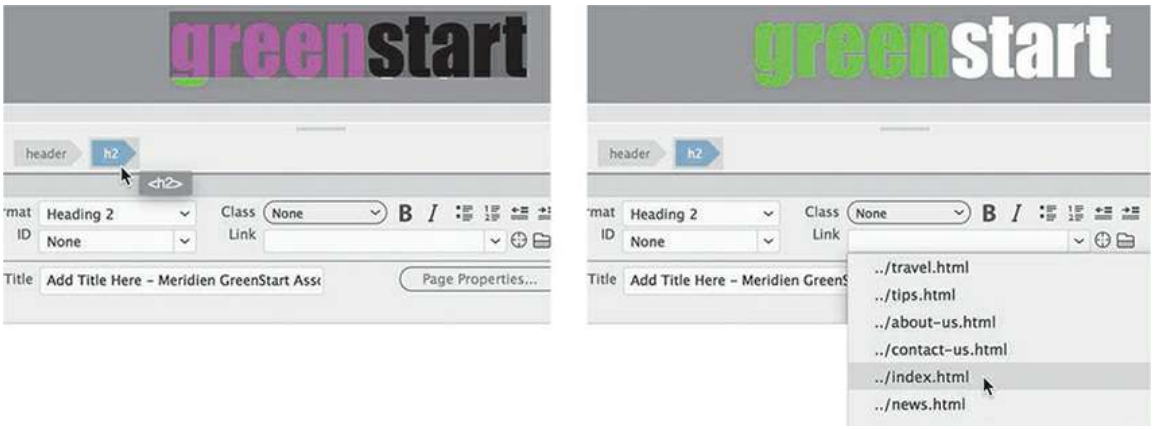
Most websites display a logo or company name, and this site is no different. The GreenStart logo appears in the header element—a product of two background graphics, a gradient, and some text. Frequently, such logos are used to create a link back to the site home page. In fact, this practice has become a virtual standard on the web. Since the template is still open, it's easy to add such a link to the Green-Start logo.

- Open **mygreen_temp.dwt** in Design view, if necessary. Insert the cursor in the *GreenStart* text in the element.

  Dreamweaver keeps track of links you create in each editing session until you close the program. You can access these previously created links from the Property inspector.

> **Note**
>
> You can select any range of text to create a link—from one character to an entire paragraph or more; Dreamweaver will add the necessary markup to the selection.

- Click the `h2` tag selector. In the Property inspector Link field, choose `../index.html` from the drop-down menu.



  This selection will create a link to the home page that you will build later. the `<a>` tag now appears in the tag selector interface, and the logo has changed color to match the default styling of hyperlinks. Although you may want normal hyperlinks to be styled this way, the logo is not supposed to be blue. It's a simple fix with CSS.

- In the CSS Designer, click mygreen-styles.css. Create the following selector: **`header h2 a:link, header h2 a:visited`**

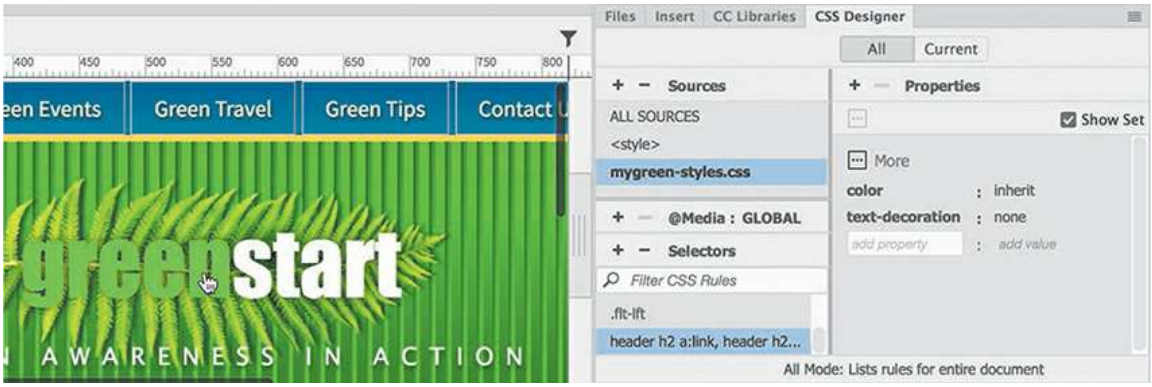
  This selector will target the "default" and "visited" states of the link within the logo.

- Add the following properties to the rule:

  `color: inherit`

  `text-decoration: none`

- Switch to Live view.



Tip

You may need to select the All button to see **mygreen-styles.css.**

Note

Design view will not render all the styling properly, but it will appear correctly in Live view and in a browser.

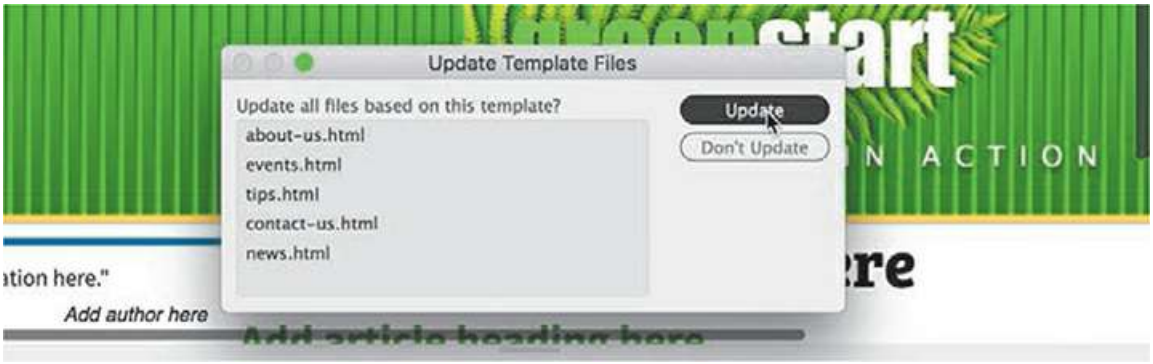
These properties will cancel the hyperlink styling and return the text to its original appearance. By using `inherit` for the color value, the color applied by the `header h2` rule will be passed automatically to the text. That way, anytime the color in the `header h2` rule changes, the hyperlink will be styled in turn without any additional work or redundant code.

So far, all the links you've created and the changes you've made are only on the template. The whole purpose of using the template is to make it easy to update pages in your site.

Updating links in child pages

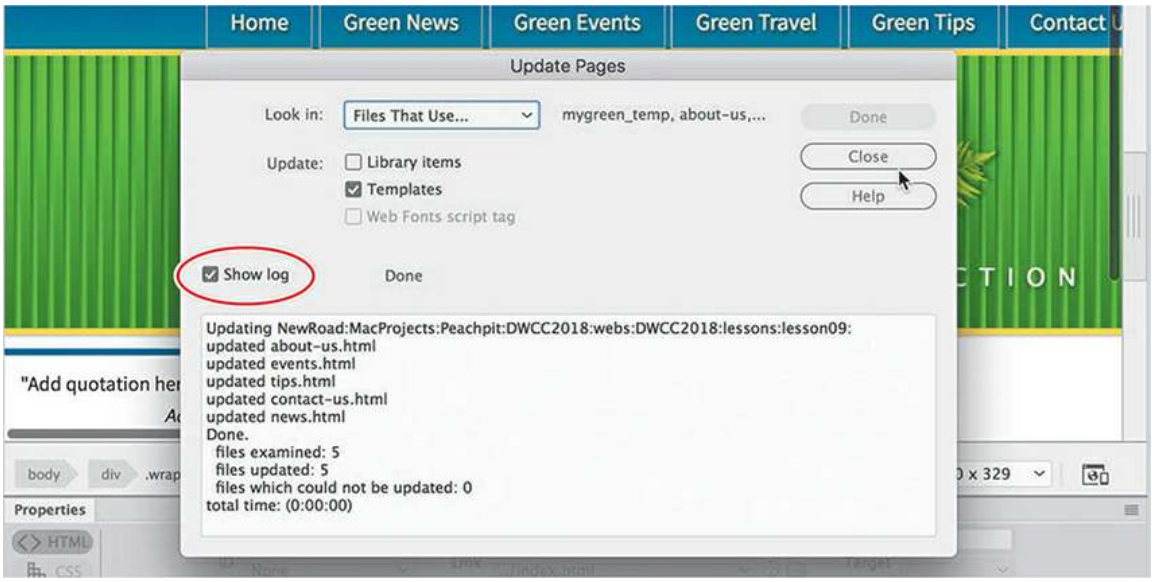
To apply the links you've created to all the existing pages based on this template, all you have to do is save it.

. Choose File > Save.



The Update Template Files dialog appears. You can choose to update pages now or wait until later. You can even update the template files manually, if desired.

. Click Update.



Dreamweaver updates all pages created by this template. the Update Pages dialog appears and displays a report listing the updated pages. If you don't see the list of updated pages, click the Show Log option in the dialog.

. Close the Update Pages dialog.

Close **mygreen_temp.dwt.**

Dreamweaver prompts you to save **mygreen-styles.css**.

Note

When you close templates or webpages, Dreamweaver may ask you to save changes to **mygreen—styles.css.** Whenever you see these warnings, always save the changes; otherwise, you could lose all your newly created CSS rules and properties.

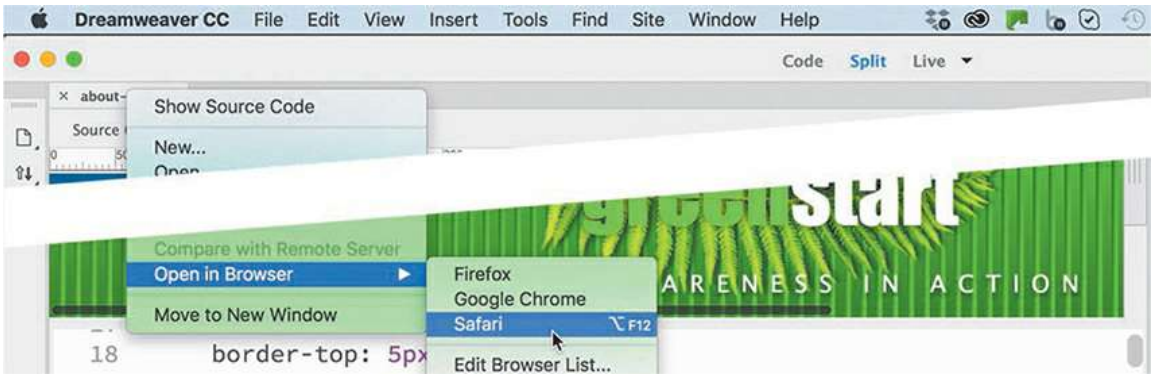
- Click Save.

  The file **about-us.html** is still open. Note the asterisk in the document tab; this indicates that the page has been changed but not saved.

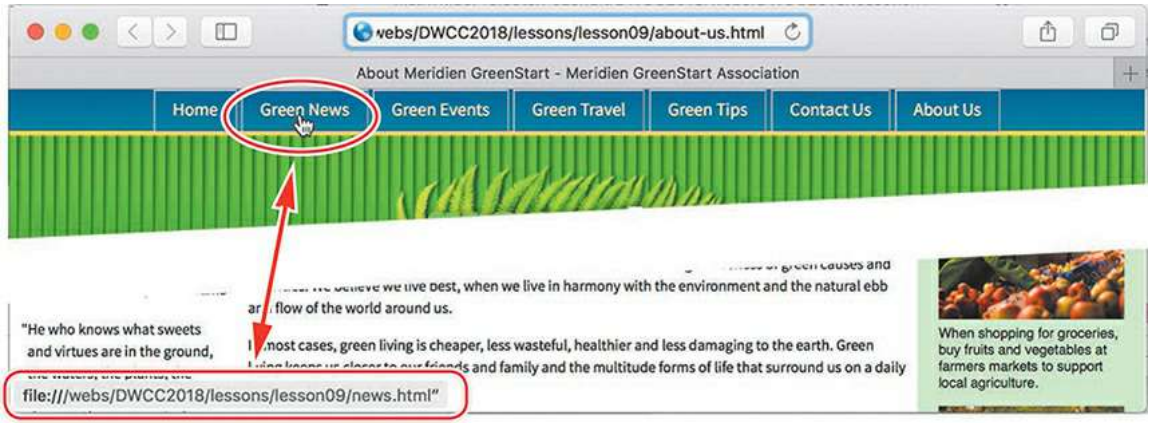
- Save **about-us.html.**

  Although Live view provides an excellent method to preview your HTML content and styling, the best way to preview links by far is in a web browser. Dreamweaver provides an easy way to preview webpages in your favorite browser.

- Right-click the document tab for **about-us.html**. Select Open In Browser. Choose your preferred browser from the context menu.



- Position the cursor over the *Home* and *Green News* links.



If you display the status bar in your browser, you can see the links applied to each item. When the template was saved, it updated the locked regions of the page, adding the hyperlinks to the horizontal menu. Child pages that are closed at the time of updating are automatically saved. Open pages must be saved manually or you will lose changes applied by the template.

. Click the *Contact Us* link.

The *Contact Us* page loads to replace the *About Us* page in the browser.

Tip

Thoroughly test every link you create on every page.

. Click the *About Us* link.

The *About Us* page loads to replace the *Contact Us* page. The links were added even to pages that weren't open at the time.

. Close the browser.

You learned three methods for creating hyperlinks with the Property inspector: typing the link manually, using the Browse For File function, and using the Point To File tool.

Creating an external link

The pages you linked to in the previous exercise were stored within the current site. You can also link to any page—or other resource—stored on the web if you know the URL.

Creating an absolute link in Live view

In the previous exercise, you used Design view to build all your links. As you build pages and format content, you'll use Live view frequently to preview the styling and appearance of your elements. Although some aspects of content creation and editing are limited in Live view, you

can still create and edit hyperlinks. In this exercise, you'll apply an external link to some text using Live view.

Tip

For this exercise, you can use any search engine or web-based mapping application.

Note

In some browsers, you can type the search phrase directly in the URL field.

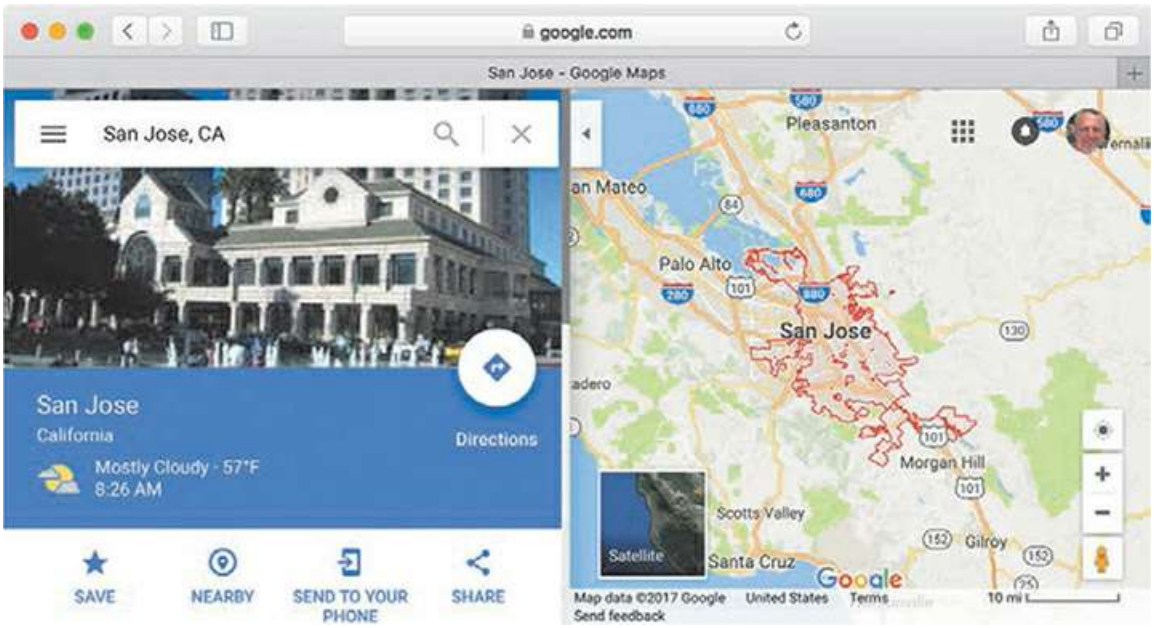
Note

We're using the Adobe headquarters in place of the fictional city of Meridien. Feel free to use your own location or another search term in its place.

- Open **contact-us.html** from the site root folder in Live view.
- In the second <p> element in the main_content region, note the word *Meridien*.

  You'll link this text to the Google Maps site.
- Launch your favorite browser. In the URL field, type **google.com/maps** and press Enter/Return.

  Google Maps appears in the browser window.
- Type **San Jose, CA** into the search field and press Enter/Return.

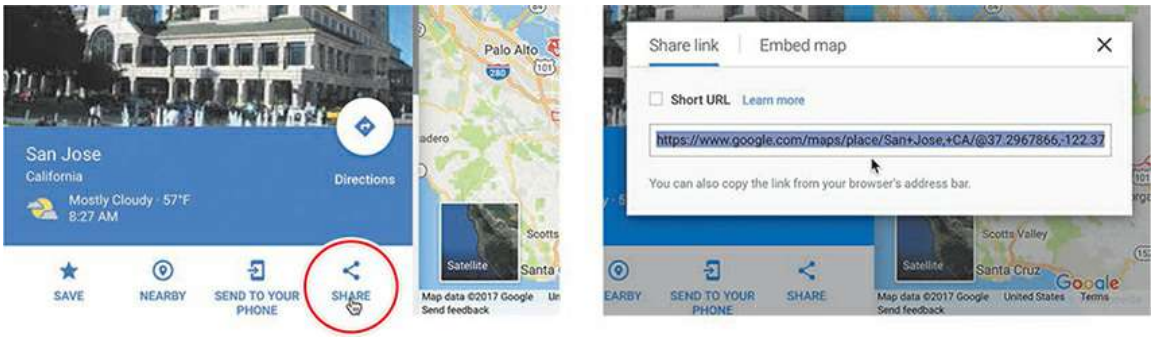


San Jose appears on a map in the browser. In Google Maps, somewhere on the screen you should see a settings or share icon.

. Open the sharing or settings interface as appropriate for your chosen mapping application.

Note

The technique for sharing map links is implemented differently in various browsers and search engines and may change over time.



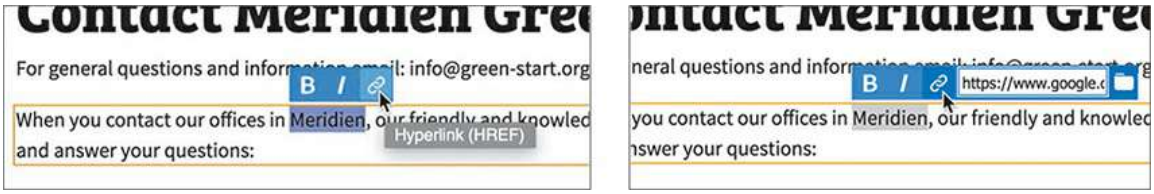
Search engines and browsers may display their link-sharing and embedding interface slightly differently than the one pictured. Google Maps, MapQuest, and Bing usually offer at least two separate code snippets: one for use within a hyperlink and the other to generate an actual map that you can embed in your site.

Note that the link contains the entire URL of the map, making it an absolute link. The advantage of using absolute links is that you can copy and paste them anywhere in the site without worrying about whether the link will resolve properly.

- Select and copy the link.
- Switch to Live view in Dreamweaver.

  Select the word Meridien.

  In Live view, you can select an entire element or insert the cursor within the element to edit or add text or apply hyperlinks, as desired. When an element or section of text is selected, Text Display will appear. The Text Display interface allows you to apply `<strong>` or `<em>` tags to the selection or (as in this case) to apply hyperlinks.
- Click the Hyperlink icon in the Text Display. Press Ctrl+V/Cmd+V to paste the link in the Link field. Press Enter/Return to complete the link.



  The selected text displays the default formatting for a hyperlink.

> **Tip**
>
> Double-click to select text in Live view.

- Save the file and preview it in the default browser. Test the link.

  When you click the link, the browser takes you to the opening page of Google Maps, assuming you have a connection to the Internet. But there is a problem: clicking the link replaced the *Contact Us* page in the browser; it didn't open a new window, as it did when you previewed the page at the beginning of the lesson. To make the browser open a new window, you need to add a simple HTML attribute to the link.
- Switch to Dreamweaver.

  Click the *Meridien* link in Live view.

  The Element Display appears focused on the `<a>` element. The Property inspector displays the value of the existing link.

> **Tip**
>
> You can access the Target attribute in Note the other options in the drop-down menu. the Property inspector

- Choose `_blank` from the Target menu in the Property inspector.
- Save the file and preview the page in the default browser again. Test the link.

  This time when you click the link, the browser opens a new window or document tab.
- Close the browser windows and switch back to Dreamweaver.

As you can see, Dreamweaver makes it easy to create links to both internal and external resources.

Where are you going?

The Target menu has six options. The *target* attribute specifies where to open the designated page or resource.

Default—This option does not create a target attribute in the markup. The default behavior of hyperlinks is to load the page or resource in the same window or tab.

`_blank`—Loads the page or resource in a new window or tab.

new—`HTML5` value that loads the page or resource in a new window or tab.

`_parent`—Loads the linked document in the parent frame or parent window of the frame that contains the link. If the frame containing the link is not nested, then the linked document loads in the full browser window.

`_self`—Loads the linked document in the same frame or window as the link. This target is the default, so you usually don't have to specify it.

`_top`—Loads the linked document in the full browser window, thereby removing all frames.

Many of the target options were designed decades ago for sites using framesets, which are now outmoded. As a result, the only option you need to consider today is whether the new page or resource replaces the existing window content or loads in a new window.

Setting up email links

Another type of link takes the visitor not to another page but to the visitor's email program. Email links can create automatic, pre-addressed email messages from your visitors for customer feedback, product orders, or other important communications. The code for an email link is slightly different from the normal hyperlink, and—as you probably guessed already—Dreamweaver can create the proper code for you automatically.

- If necessary, open **contact-us.html** in Design view.
- Select the email address (info@green-start.org) in the first paragraph underneath the heading and press Ctrl+C/Cmd+C to copy the text.

Tip

The Email Link menu cannot be accessed in Live view. But you can use the menu in Design view or Code view, or you can just create the links by hand in any view.

. Choose Insert > HTML > Email Link.

The Email Link dialog appears. The text selected in the document window in step 2 is automatically entered into the Text field.

. Insert the cursor in the Email field and press Ctrl+V/Cmd+V to paste the email address, if necessary.

. Click OK.

Examine the Link field in the Property inspector.

Tip

If you select the text before you access the dialog, Dreamweaver enters the text in the field for you automatically.

Dreamweaver inserts the email address into the Link field and also enters the `mailto:` notation, which tells the browser to automatically launch the visitor's default email program.

. Save the file and open it in the default browser.

Test the email link.

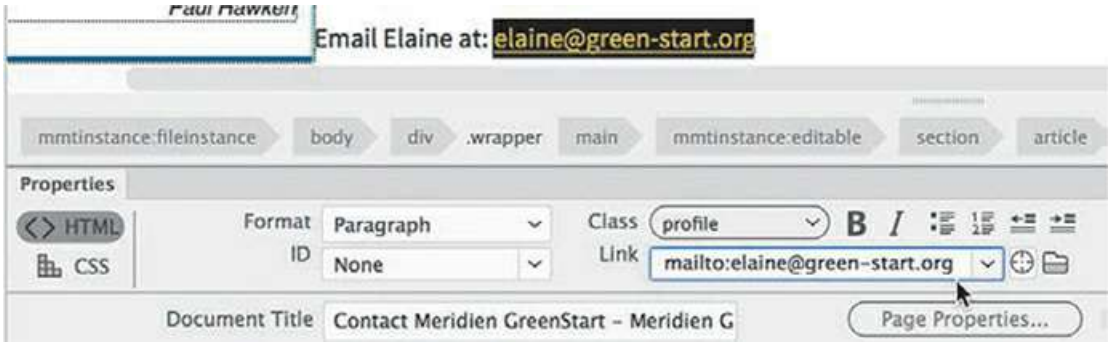
If your computer has a default email program installed, it will launch and create a new email message using the email address provided in the link. If there is no default email program, your computer's operating system may ask you to identify or install one.

. Close any open email program, related dialogs, or wizards. Switch to Dreamweaver.

You can also create email links manually.

. Select and copy the email address for Elaine.

. Type **`mailto:`** in the Property inspector Link field. Paste Elaine's email address directly after the colon. Press Enter/Return to complete the link.



Note

Be sure that there are no spaces between the colon and the link text.

The text `mailto:elaine@green-start.org` appears in the Text Display link field in Live view.

. Save the file.

You have learned a few techniques to add links to text content. You can add links to images too.

Creating an image-based link

Image-based links work like any other hyperlink and can direct users to internal or external resources. You can use the Insert menu in Design view or Code view or apply links and other attributes using the Element Display interface in Live view.

Creating image-based links using the Element Display

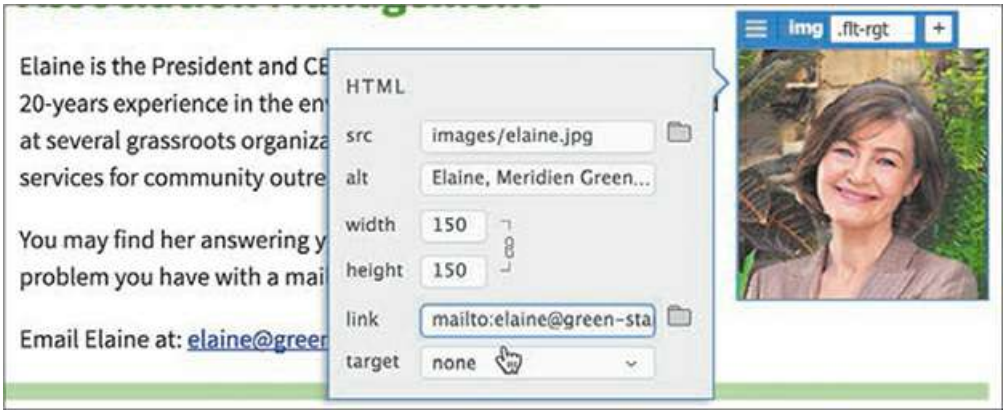
In this exercise, you will create and format an image-based link using the email addresses of each GreenStart employee via the Element Display.

- If necessary, open **contact-us.html** in Live view from the site root folder.
- Select the image of Elaine in the *Association Management* section.

  To access the hyperlink option, you must open the Quick Property inspector.
- In the Element Display, click the Edit HTML Attributes icon .

The Quick Property inspector opens and displays options for the image attributes `src`, `alt`, `link`, `width`, and `height`.

- Click in the link field. If the email address is still in memory from the previous exercise, simply enter **`mailto:`** and paste the address in the Link field. Otherwise, enter **`mailto:elaine@green-start.org`** in the Link field after the colon and press Enter/Return to complete the link. Press the Esc key to close the Quick Property inspector.



Note

In the past, images that featured a hyperlink were automatically styled with a blue border. That styling was deprecated in HTML5.

The hyperlink that is applied to the image will launch the default email program in the same fashion as it did with the text-based link earlier.

. Select and copy the email address for Sarah.

Repeat steps 2 through 4 to create an email link for Sarah's image.

. Create image links for the remaining employees using the appropriate email address for each.

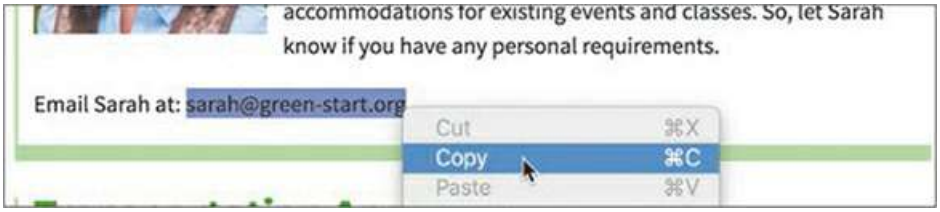
All the image-based links on the page are complete. You can create text-based links using the Text Display too.

Creating text links using the Text Display

In this exercise, you will create text-based email links as needed for the remaining employees.

. If necessary, open **contact-us.html** in Live view.

. Select and copy the email address for Sarah.



You can now select and copy text without having to activate text-editing mode in Live view.

. Double-click to edit the paragraph containing Sarah's email address. Select her email address.

The Text Display appears around the selected text.

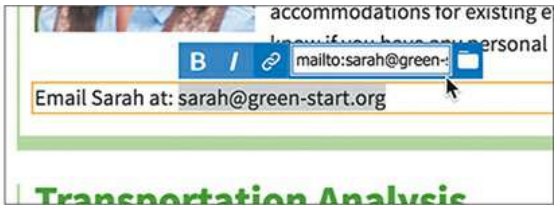
. Click the Link icon.

A link field appears. A folder icon displays on the right side of the link field. If you were linking to a file on the website, you could click the folder to target the file. In this case, we're creating an email link.

. Insert the cursor in the link field, if necessary.

Enter `mailto:` and paste Sarah's email address.

Press Enter/Return.



- Using the Text Display, create email links for the remaining email addresses displayed on the page.
- Save and close all files.

Attack of the killer robots

Although on the surface it sounds like a good idea to add email links to make it easier for your customers and visitors to communicate with you and your staff, emaillinks are a double-edged sword. The Internet is awash in bad actors and unethical companies that use intelligent programs, or robots, to constantly search for live email addresses (and other personal information) that they can flood with unsolicited email and spam. Putting a plain email address on your site as shown in these exercises is like putting a sign on your back that says “kick me.”

In place of active email links, many sites use a variety of methods for limiting the amount of spam they receive. One technique uses images to display the email addresses, since robots can’t read data stored in pixels (yet). Another leaves off the hyperlink attribute and types the address with extra spaces, like this:

```
elaine @ green-start .org
```

However, both of these techniques have drawbacks; if visitors try to use copy and paste, it forces them to go out of their way to remove the extra spaces or try to type your email address from memory. Either way, the chances of you receiving any communication decreases with each step the user has to accomplish without additional help.

At this time, there is no foolproof way to prevent someone from using an email address for nefarious purposes. Coupled with the fact that fewer users actually have a mail program installed on their computers anymore, the best method for enabling communication for your visitors is to provide a means built into the site itself. Many sites create web-hosted forms that collect the visitor’s information and message and then pass them along using server-based email functionality.

Targeting page elements

As you add more content, the pages get longer and navigating to that content gets more difficult. Typically, when you click a link to a page, the browser window loads the page and displays it starting at the top. But it can be helpful when you provide convenient methods for users to link to

a specific point on a page.

HTML 4.01 provided two methods to target specific content or page structures: a *named anchor* and an *id* attribute. In HTML5, the named anchor method has been deprecated in favor of ids. If you have used named anchors in the past, don't worry—they won't suddenly cease to function. But from this point on, you should start using ids exclusively.

Creating internal targeted links

In this exercise, you'll work with id attributes to create the target of an internal link. You can add ids in Live, Design, or Code view.

- Open **events.html** in Live view.
- Scroll down to the table containing the class schedule.

  When users move down this far on the page, the navigation menus are out of sight and unusable. The farther down the page they read, the farther they are from the primary navigation. Before users can navigate to another page, they have to use the browser scroll bars or the mouse scroll wheel to get back to the top of the page.

  Older websites dealt with this situation by adding a link to take visitors back to the top, vastly improving their experience on the site. Let's call this type of link an *internal targeted* link. Modern websites simply freeze the navigation menu at the top of the screen. That way, the menu is always visible and accessible to the user. You will learn how to do both techniques. First, let's create an internal targeted link.

  Internal targeted links have two parts: the link itself and the target, or destination. Which one you create first doesn't matter.
- Click the *2019 Class Schedule* table.

  Select the table's parent section tag selector.

  The Element Display appears focused on the section element.
- Open the Insert panel.

  Select the HTML category.

  Click the Paragraph item.

  The position assist dialog appears.
- Click Before.

A new paragraph element appears in the layout, with the placeholder text *This is the content for Layout P Tag*.

. Select the placeholder text.

Type **Return to Top** to replace it.

The text is inserted between the two tables, formatted as a <p> element.

The text would look better centered.

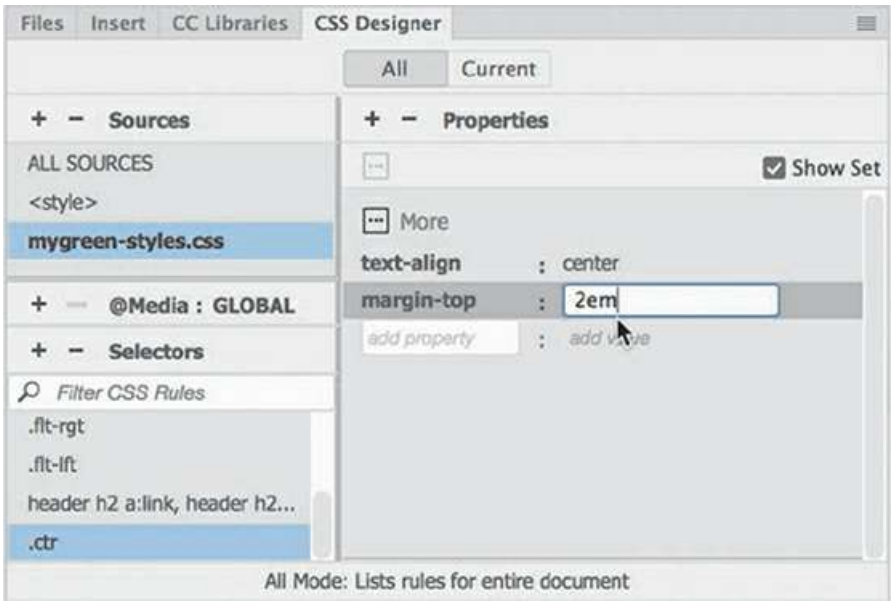
. Choose **mygreen-styles.css** in the CSS Designer.

Create a new selector: **.ctr**

. Create the following properties for .ctr:
%

text-align: center

margin-top: 2em



. Click the Add Class/ID icon for the selected <p>element.

. Type **.ctr** in the text field and press Enter/Return, or choose .ctr from the hinting menu.

The *Return to Top* text is aligned to the center. The tag selector now displays `p.ctr`.

- Select the element *Return to Top*. Click the Edit HTML Attributes icon and type **`#top`** in the Link field. Press Enter/Return to complete the link.

By using #top, you have created a link to the top of the current page. This target is now a default function in HTML5. If you use the plain hash (#) symbol or #top as the link target, the browser automatically assumes you want to jump to the top of the page. No additional code is needed.

- Save all files.
- Open **events.html** in a browser.
- Scroll down to the Class table.

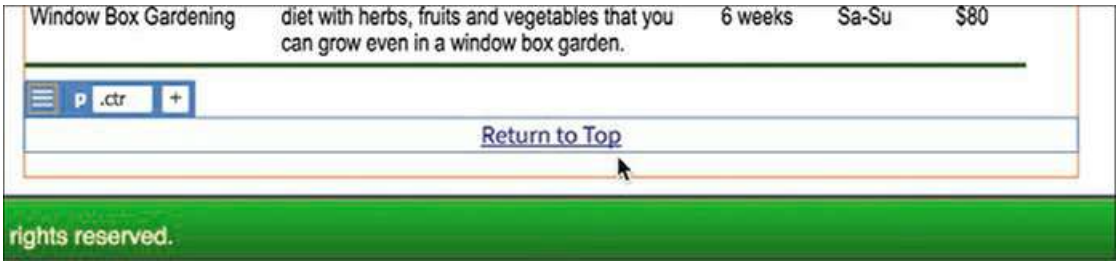
  Click the *Return to Top* link.

  The browser jumps back to the top of the page.

  You can copy the *Return to Top* link and paste it anywhere in the site you want to add this functionality.
- Select and copy the `<p>`element containing the text *Return to Top* and its link.
- Insert the cursor in the Class table.

  Using the tag selector, select the `<Section>` element.

  Press Ctrl+V/Cmd+V to paste.



A new p element and link appear at the bottom of the page.

. Save the file and preview it in the browser.

Test both *Return to Top* links.

Both links can be used to jump back to the top of the document. In the next exercise, you'll learn how to create link targets using element attributes.

Creating a destination link in the Element Display

In the past, destinations were often created by inserting a standalone element known as a *named anchor* within the code. In most cases, there's no need to add any extra elements to create hyperlink destinations, since you can simply add an id attribute to a handy element nearby. In this exercise, you will use the Element Display to add an id.

. Open **events.html** in Live view.

Click the *2019 Events Schedule* table.

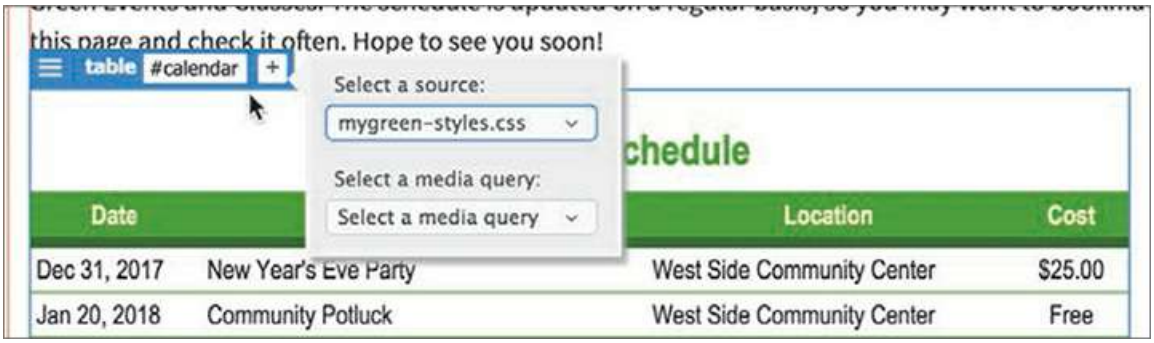
Select the table tag selector.

The Element Display and the Property inspector display the attributes currently applied to the Events table. You can add an id using either tool.

. Click the Add Class/ID icon. Type **#**

If any ids were defined in the style sheet but unused on the page, a list would appear. Since nothing appears, it means that there are no unused ids. Creating a new one is easy.

. Type **calendar** and press Enter/Return.



The CSS Source dialog appears. You do not need to add the id to any style sheet.

. Press Esc to close the dialog.

The tag selector now displays `#calendar` and no entry was made in the style sheet. Since ids are unique identifiers, they are perfect for targeting specific content on a page for hyperlinks.

You also need to create an id for the Class table.

Note

When creating ids, remember that they need to have names that are used only once per page. They are case sensitive, so look out for typos.

. Repeat steps 2 through 4 to create the id `#classes` on the Class table.

The tag selector now displays `#classes`.

Note

If you add the id to the wrong element, simply delete it and start over.

. Save all files.

You'll learn how to link to these ids in the next exercise.

Targeting id-based link destinations

By adding unique ids to both tables, you have provided ideal targets for internal hyperlinks to navigate to a specific section of your webpage. In this exercise, you will create a link to each table.

. If necessary, open **contact-us.html** in Live view.

Scroll down to the *Education and Events* section.

. Select the word *events* in the first paragraph of the section.

> **Tip**
>
> You can select single words by double-clicking them.

. Using the Text Display, create a link to the file **events.html**.

This link will open the file, but you're not finished. You now have to direct the browser to navigate down to the Events table.

> **Note**
>
> Hyperlinks cannot contain spaces; make sure the id reference follows the filename immediately.

. Type `#calendar` at the end of the filename to complete the link, and press Enter/Return.

The word *events* is now a link targeting the Events table in the **events.html** file.

. Select the word *classes*.

Create a link to the **events.html** file.

Type `#classes` to complete the link and press Enter/Return.

. Save the file and preview the page in a browser.

Test the links to the *Events* and *Class* tables.

The links open the *Events* page and navigate to the appropriate tables. You've learned how to

create a variety of internal and external links. The last things you need to do are learn how to freeze the horizontal navigation menu at the top of the screen and adjust a styling issue with the menu items.

Locking an element on the screen

Most elements you encounter on a webpage will move with the page as you scroll down through the content. This is the default behavior in HTML. For specific purposes you may want to freeze an element so that it stays on the screen. This has become very popular, especially with navigation menus. Keeping the menu visible at all times provides handy navigation options whenever desired.

Although the navigation menu is not editable directly in the child pages created from the template, the change you need to make can be done entirely in the CSS Designer. However, it helps to have one of the pages open

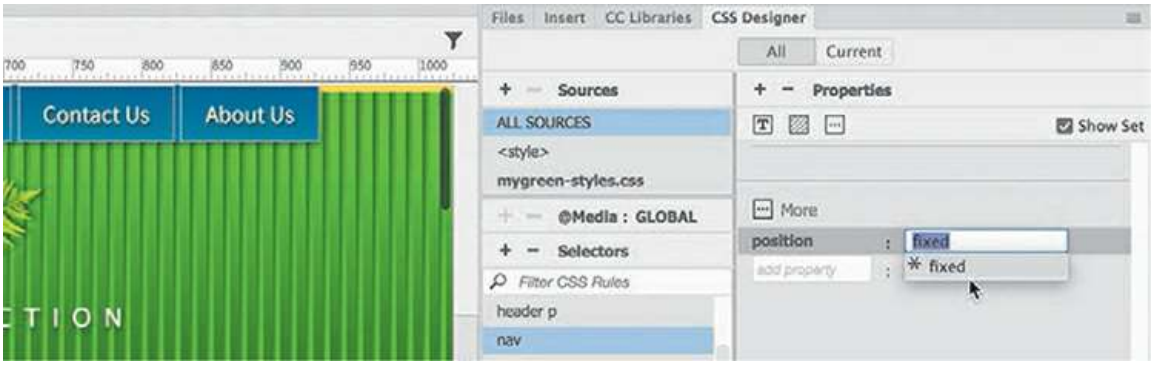
. If necessary, open **contact-us.html** in Live view.

  The navigation menu appears at the top of the page but scrolls with the rest of the content.

. Select the rule nav in the CSS Designer.

. Add the following property to the rule:

```
position: fixed
```



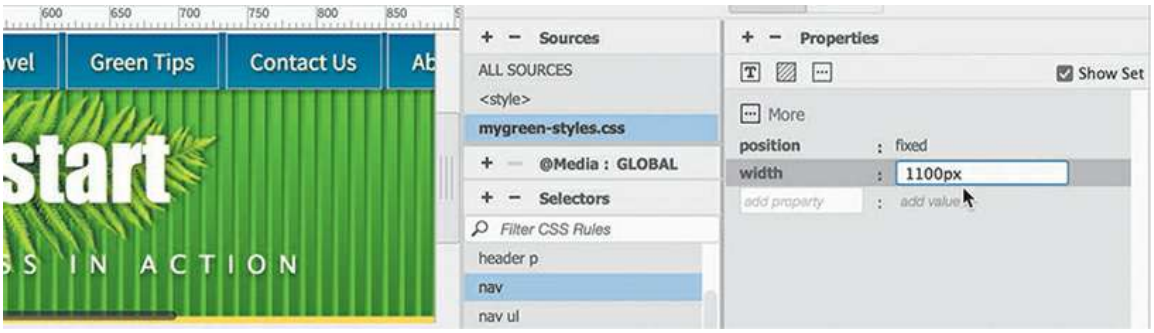
As soon as you create the property, the header element shifts under the horizontal menu. If you scroll down the page, you will see that in Live view the menu stays at the top of the window. It should work the same way in a browser.

The effect is close to what we wanted but not complete. Although the menu is fixed to the top of the screen, it no longer matches the width of the rest of the page and obscures part of the header. Applying the position property has basically taken the menu out of the document. It now exists in a separate world from the rest of the content. By default, it floats above the other elements.

To make the menu reorient properly to the original page design, you'll have to apply a width to it and add spacing above the header to move everything back into place.

. Add the following property to the nav rule:

```
width: 1100px
```

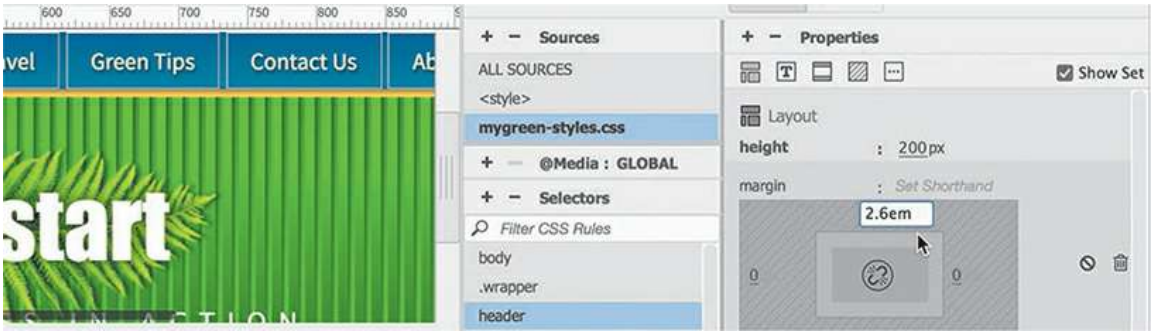


The horizontal menu now has the same width as `div.wrapper`. To move the header down below the menu, you need to add some space to the controlling rule.

. Add the following property to the header rule:

```
margin-top: 2.6em
```

> **Tip**
>
> Using em measurements for menus and other controls ensures that the structure will adapt better if visitors use larger font sizes, since ems are based on the font size.



The `header` element shifts back down to its original position.

. Save all files.

The menu is almost complete. You may have noticed that the borders on the menu items still sport gray borders. Let's adjust the colors to better match the site color scheme.

Styling a navigation menu

The gray borders on the menu items are a holdover from the original styling of the layout. When

you picked up the styling from the mockup in Lesson 5, it didn't include border colors for these items. Using the CSS Designer it'll be easy to identify the source of this styling. Although the menu is locked in the child pages, it's still possible to identify the pertinent CSS rules as long as you know how the component is constructed.

The horizontal menu is composed of nav, `ul, li`, and a elements. Knowing the structure allows you to troubleshoot any CSS styling scheme. It helps to have one of the child pages open.

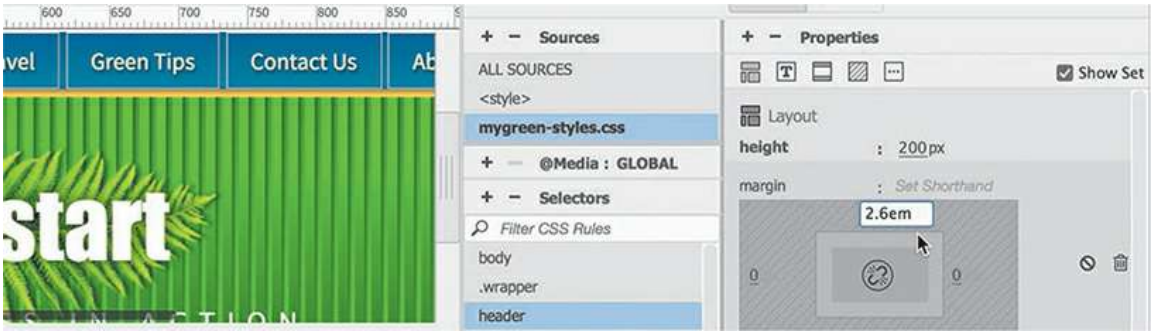
- If necessary, open **contact-us.html** in Live view.
- In the CSS Designer, locate any nav rules in the Selectors pane.

  There is one nav rule in the list.
- Select the `nav` rule.

  When you select the rule, the nav element highlights in the layout in Live view.
- Examine the properties of the rule.

  The entire menu is selected, but the gray borders are only on the menu items themselves. As you search rules styling the borders, the best method is to work down through the structure of the component.
- Select the rule `nav ul`.

  This time the highlighting focuses on the seven menu items as a group.
- Select the rule `nav ul li`.



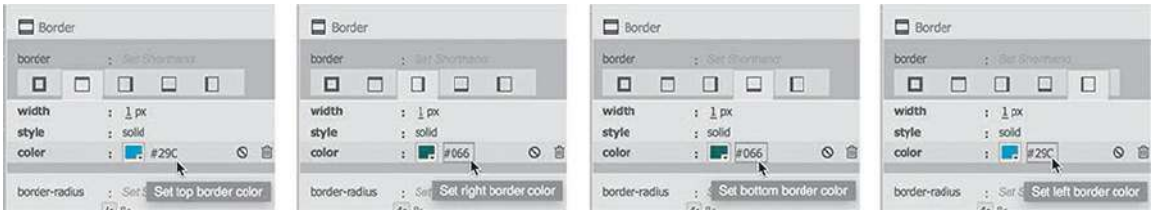
  The highlighting now focuses on each item.
- Examine the properties for the rule `nav ul li`.

  The rule does not contain any styling for borders. There's one rule left.
- Select the rule `nav ul li a:link, nav ul li a:visited`.

  This rule supplies the border colors. You may need to select one of the border tabs to see the current assigned color.
- Make the following changes to the rule

  Click here to view code image

```
nav ul li a:link,nav ul li a:visited:
border-top-color: #29C
border-right-color: #066
border-bottom-color: #066
border-left-color: #29C
```



The new colors give a slight 3D effect to the menu items. Besides styling the menu, it's always nice to add interactive behaviors to such components too. This is a popular technique on the web. Many designers take advantage of the default behaviors of the anchor element to do this. Read the sidebar "Hyperlink pseudo-classes" to learn more about these behaviors.

By providing a separate CSS rule for the :hover pseudo-class, you will change the styling of the menu item whenever a visitor positions the cursor over it.

As described in the sidebar "Hyperlink pseudo-classes," the `:hover` rule must appear after the `:link and :visited` selectors in the style sheet. There's a simple technique you can use in the CSS Designer to ensure that a specific rule appears after another.

Hyperlink pseudo-classes

The `<a>` element (hyperlink) provides five states, or distinct behaviors, that can be modified by CSS using what are known as pseudo—classes. A pseudo—class is a CSSfeature that can add special effects or functionality to certain selectors, such as the `<a>` anchor tag.

- The `a:link` pseudo-class creates the default display and behavior of the hyperlink and in many cases is interchangeable with the a selector in CSS rules. But a:linkis *more* specific and will override specifications assigned to a less specific selector if both are used in the style sheet.
- The `a:visited` pseudo-class formats the link after it has been visited by the browser. This resets to default styling whenever the browser cache, or history, is deleted.
- The `a:hover` pseudo-class formats the link when the cursor passes over it.
- The `a:active` pseudo-class formats the link when the mouse clicks it.
- The `a:focus` pseudo-class formats the link when accessed via keyboard as opposed to mouse interaction.

When used, the pseudo-classes must be declared in the order listed here to be effective.

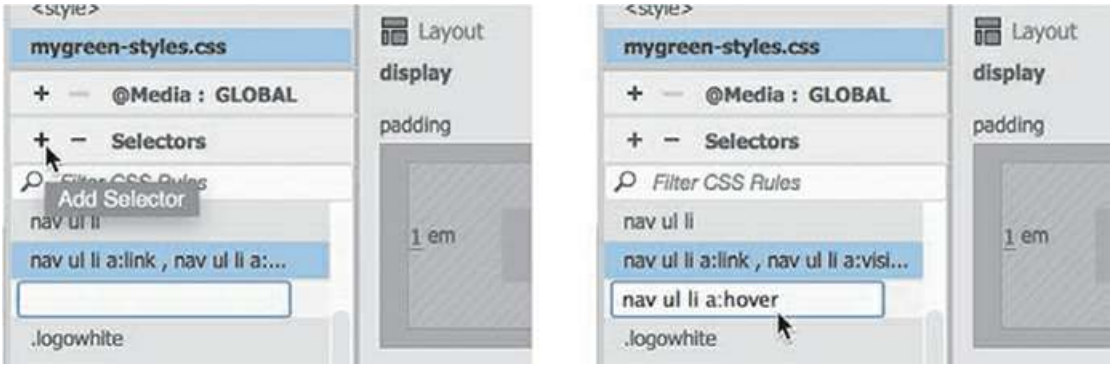
Remember that, whether declared in the style sheet or not, each state has a set of default formats and behaviors.

. Select the rule `nav ul li a:link, nav ul li a:visited`.

Click the Add Selector icon.

The new selector field appears following the selected rule. In turn, the new selector and declarations will be inserted after the `nav ul li a:link, nav ul li a:visited` rule in the style sheet.

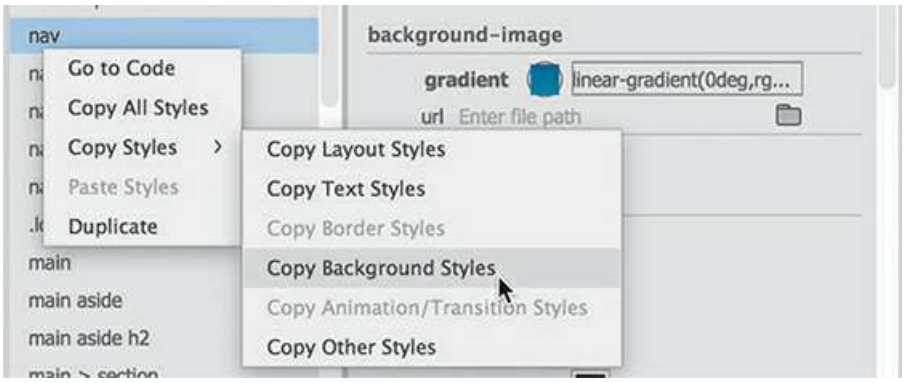
. Type **`nav ul li a:hover`** and press Enter/Return to create the selector.



The `:hover` selector kicks in when the cursor is positioned over the element. The simplest way to format the effect is to use the existing style of the menu and then modify it.

. Right-click the rule nav.

Select Copy Styles > Copy Background Styles.



. Right-click the rule `nav ul li a:hover`.

Select Paste Styles from the context menu.

The `:hover` rule now has styling that is identical to that of the nav rule. A simple change to the background gradient will provide a dramatic interactive appearance.

. Click the Gradient color picker in `nav ul li a:hover`.

Change the angle to `180`.

- Create the following new property in the rule: `color: #FFF`
- Position the cursor over any of the menu items.

The gradient background reverses direction and the text displays in white.

The effect provides a nice interactive behavior to the menu items.

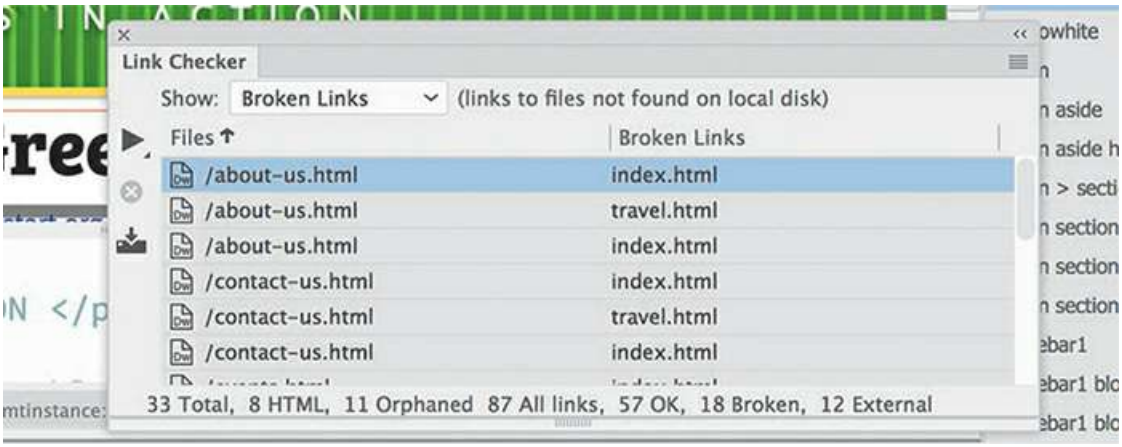
- Save all files.

You have learned how to create links in a variety of ways and even learned how to format an interactive behavior on the horizontal menu as well as freeze it to the top of the screen. Once you know how to create hyperlinks, you next need to learn how to test them.

Checking your page

Dreamweaver can check your page, as well as the entire site, for valid HTML, accessibility, and broken links. In this exercise, you'll learn how to check your links sitewide.

- If necessary, open **contact-us.html** in Design view.
- Choose Site > Site Options > Check Links Sitewide.



The Link Checker panel appears. The panel reports broken links to the files **index.html** and **travel.html**. These are links to nonexistent pages. You'll create these pages in an upcoming lesson, so you don't need to worry about fixing these broken links now. The Link Checker will also find broken links to external sites, should you have any.

Note

The total number and types of missing and broken links may vary from that pictured.

. Close the Link Checker panel, or, if it's docked, right-click the Link Checker tab and choose Close Tab Group from the context menu.

You've made big changes to the pages in this lesson by creating the main navigation menu with links to specific positions on a page, to email, and to an external site. You also applied links to images and learned how to check your site for broken links.

Adding destination links (optional)

Using the skills you have just learned, open **events.html** and create destination links for the words *Events* and *Classes* in the first paragraph.

Remember that each word should link to the appropriate tables on that page. Can you figure out how to construct these links properly? If you have any trouble, check out the **events-finished.html** file for the answer.

Review questions

1. Describe two ways to insert a link into a page.
2. What information is required to create a link to an external webpage?
3. What's the difference between standard page links and email links?
4. What attribute is used to create destination links?
5. What limits the usefulness of email links?
6. Can links be applied to images?
7. How can you check to see whether your links will work properly?

Review answers

1. Select text or a graphic, and then, in the Property inspector, click the Browse for File icon next to the Link field and navigate to the desired page. A second method is to drag the Point To File icon to a file within the Files panel.
2. To link to an external page, you must type or copy and paste the full web address (a fully formed URL, including http:// or other protocol) in the Link field of the Property inspector or the Text Display.
3. A standard page link opens a new page or moves the view to a position somewhere on the page. An email link opens a blank email message window if the visitor has an email application installed on their system.
4. You apply unique id attributes to any element to create a link destination, which can

appear only once in each page.

5. Email links may not be very useful because many users do not use built-in email programs, and the links will not automatically connect with Internet-based email services.
6. Yes, links can be applied to images and used in the same way text-based links are.
7. Run the Link Checker report to test links on each page individually or sitewide. You should also test every link in a browser.

10 Adding Interactivity

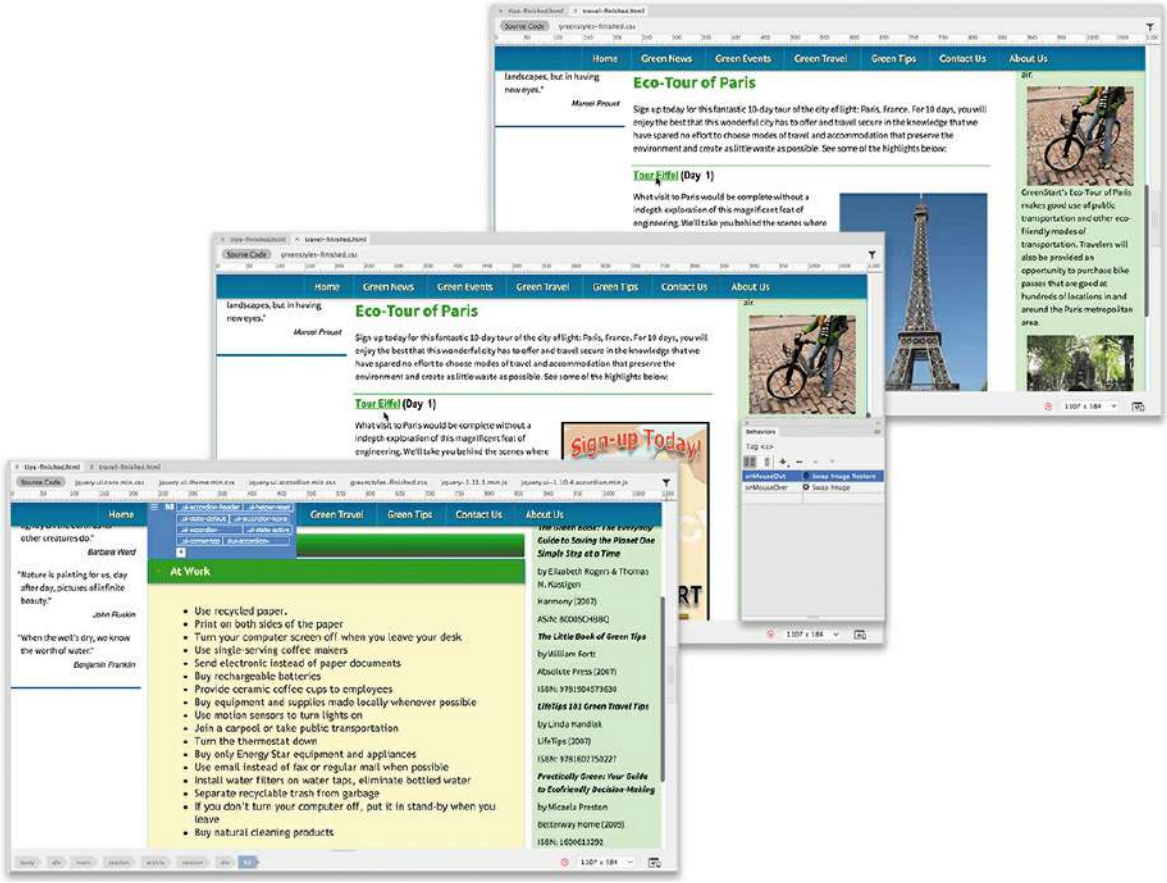
Lesson overview

In this lesson, you'll add Web 2.0 functionality to your webpages by doing the following:

- Using Dreamweaver behaviors to create an image rollover effect
- Inserting a jQuery accordion widget

This lesson will take about 90 minutes to complete. Please log in to your account on peachpit.com to download the project files for this lesson, as described in the "Getting Started" section at the beginning of this book. Follow the instructions under "Accessing the Lesson Files and Web Edition." Define a site based on the lesson10 folder.

Your Account page is also where you'll find any updates to the lessons or the lesson files. Look on the Lesson & Update Files tab to access the most current content.



Dreamweaver can create sophisticated interactive effects with behaviors and accordion panels using Adobe's Bootstrap and jQuery frameworks.

Learning about Dreamweaver behaviors

The term *Web 2.0* was coined to describe a major change in the user experience on the Internet—from mostly static pages, featuring text, graphics, and simple links, to a new paradigm of dynamic webpages filled with video, animation, and interactive content. Dreamweaver has always led the industry in providing a variety of tools to drive this movement, from its tried-and-true collection of JavaScript behaviors to jQuery, jQuery Mobile, and Bootstrap widgets. This lesson explores two of these capabilities: Dreamweaver behaviors and jQuery widgets.

Note

If you have not already downloaded the project files for this lesson to your computer from your Account page, make sure to do so now. See "Getting Started" at the beginning of the book.

A Dreamweaver *behavior* is predefined JavaScript code that performs an action—such as

opening a browser window or showing or hiding a page element—when it is triggered by an event, such as a mouse click. Applying a behavior is a three-step process.

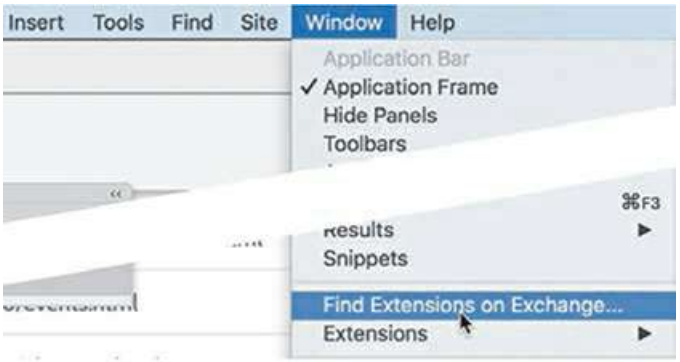
Note

To access Dreamweaver behaviors, you must have a file open.

- Create or select the page element that you want to trigger the behavior.
- Choose the behavior to apply.
- Specify the settings or parameters of the behavior.

The triggering element often involves a hyperlink applied to a range of text or to an image. In some cases, the behavior is not intended to load a new page, so it employs a dummy link enabled by the hash sign (#), similar to ones you used in Lesson 5, "Creating a Page Layout." The Swap Image behavior you will use in this lesson does not require a link to function, but keep this in mind when you work with other behaviors.

Dreamweaver offers more than 16 built-in behaviors, all accessed from the Behaviors panel (Window > Behaviors). You can download hundreds of other useful behaviors from the Internet for free or a small fee. Some are available from the Adobe Add-ons website, which you can add to the program by clicking the Add Behavior icon in the Behaviors panel and choosing Get More Behaviors from the pop-up menu. You can obtain other tools or features from third-party developers and install them in Dreamweaver as extensions. You can also access the Adobe Add-ons website by choosing Window > Find Extensions On Exchange.



When the Adobe Add-ons page loads in the browser, click the link to download the plug-in, extension, or other add-on. Often you can simply double-click the add-on to install it.

The following are some examples of the functionality available to you using the built-in Dreamweaver behaviors:

Opening a browser window

Swapping one image for another to create what is known as a *rollover effect*

Fading images or page areas in and out

Growing or shrinking graphics

Displaying pop-up messages

Changing the text or other HTML content within a given area

Showing or hiding sections of the page

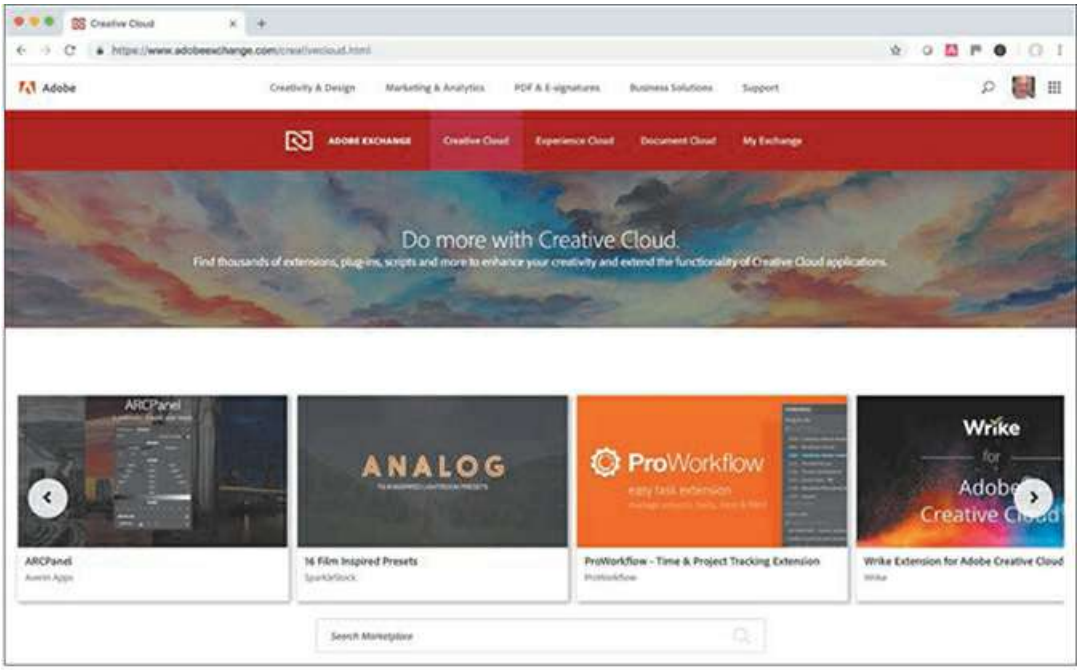
Calling a custom-defined JavaScript function

Not all behaviors are available all the time. Certain behaviors become available only in the presence and selection of certain page elements, such as images or hyperlinks. For example, the Swap Image behavior must target an image.

Each behavior invokes a unique dialog that provides relevant options and specifications. For instance, the dialog for the Open Browser Window behavior enables you to open a new browser window; set its width, height, and other attributes; and set the URL of the displayed resource. After the behavior is defined, it is listed in the Behaviors panel with its chosen triggering action. As with other behaviors, you can modify these specifications at any time.

Behaviors are extremely flexible, and you can apply multiple behaviors to the same trigger. For example, you could swap one image for another and change the text of the accompanying image caption—and do it all with one click. Although some effects may appear to happen simultaneously, behaviors are actually triggered in sequence. When multiple behaviors are applied, you can choose the order in which the behaviors are processed.

To learn more about Adobe Add-ons, check out **www.adobeexchange.com/creativecloud.html**. Select Dreamweaver in the "view by product" list to see add-ons developed specifically for that app.



The Adobe Add-ons website offers tons of resources for many of the applications in

Creative Cloud, including both free and paid add-ons.

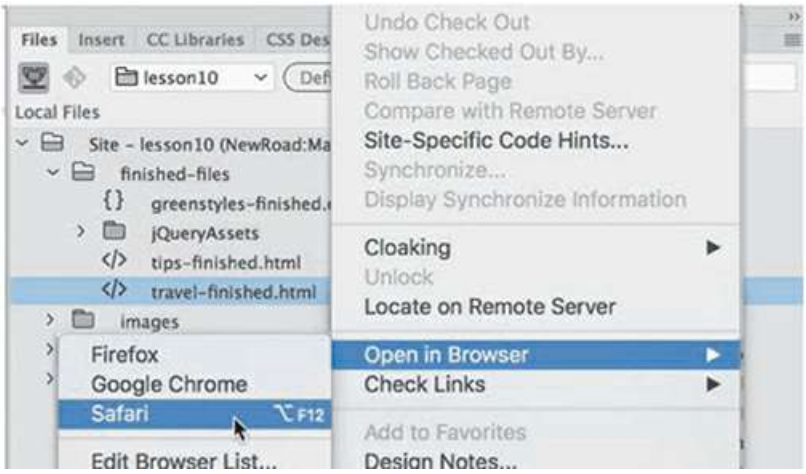
Previewing the completed file

In the first part of this lesson, you'll create a new page for GreenStart's travel services. Let's preview the completed page in a browser.

- Launch Adobe Dreamweaver CC (2019 release) or later.

  Define a site based on the lesson10 folder. Name the site **lesson10**.

- Open **travel_finished.html** directly in your favorite browser.



The page includes Dreamweaver behaviors. In the middle of the page is a two-column table. Some of the interactivity may not preview properly in Dreamweaver.

- If Microsoft Internet Explorer is your default browser, a message may appear in the browser window indicating that it has prevented scripts and ActiveX controls from running. If so, click Allow Blocked Content.

  This message appears only when the file is previewed from your hard drive. It doesn't appear when the file is actually hosted on the Internet.

- Position the cursor over the *Tour Eiffel* heading. Observe the image to the right of the text.

The existing image is swapped for one of the Eiffel Tower.

- Move the pointer to the *Paris Underground* heading.

  Observe the image to the right of the text.

  As the pointer moves off the *Tour Eiffel* heading, the image reverts to the Eco-Tour ad. Then, as the pointer moves over the heading *Paris Underground*, the ad image is swapped for one of underground Paris.
- Pass the pointer over each `<h3>` heading, and observe the image behavior.

  The image alternates between the Eco-Tour ad and images of each of the tours. This effect is the Swap Image behavior.
- When you're finished, close the browser window and return to Dreamweaver.
- Close **travel-finished.html**.

In the next exercise, you'll learn how to work with Dreamweaver behaviors.

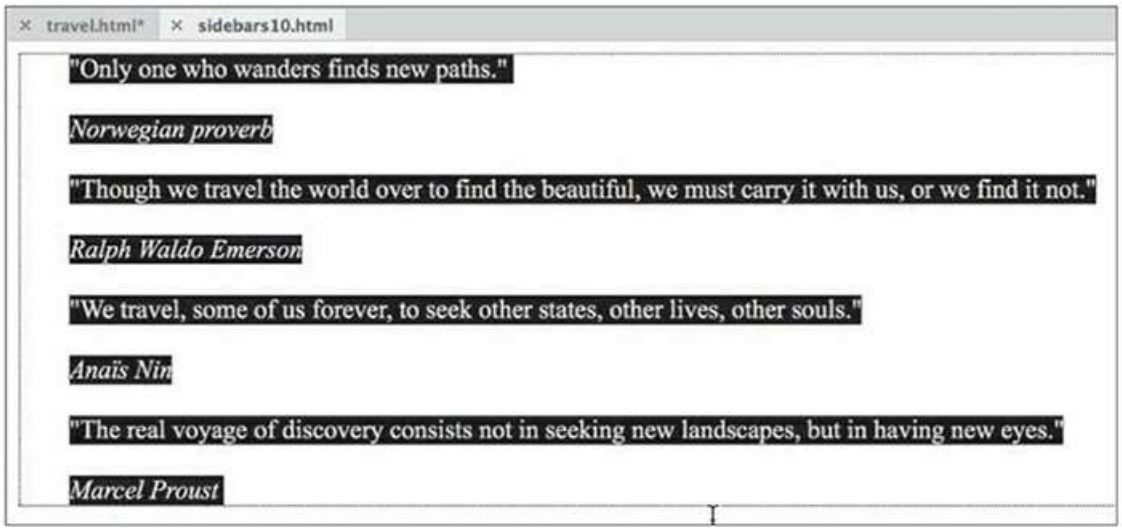
Working with Dreamweaver behaviors

Adding Dreamweaver behaviors to your layout is a simple point-and-click operation. But before you can add the behaviors, you have to create the travel page from the site template.

- Create a new page from **mygreen_temp.dwt**.
- Save the file as **travel.html** in the site root folder.

  Switch to Design view, if necessary.
- Open **sidebars10.html** in Design view from the lesson10/ resources folder. Insert the cursor into the first paragraph. Examine the tag selectors.

  The paragraph is a child of a `<blockquote>` within an `aside` element. The classes and structure in the new file are identical to Sidebar 1 in the site template.
- In **sidebars10.html**, drag to select the four quotes and citations.



. Copy the selected text in **sidebars10.html**.

. Switch to **travel.html**.

Insert the cursor into the quotation placeholder.

Select the `blockquote` tag selector.

. Paste the content from step 5.

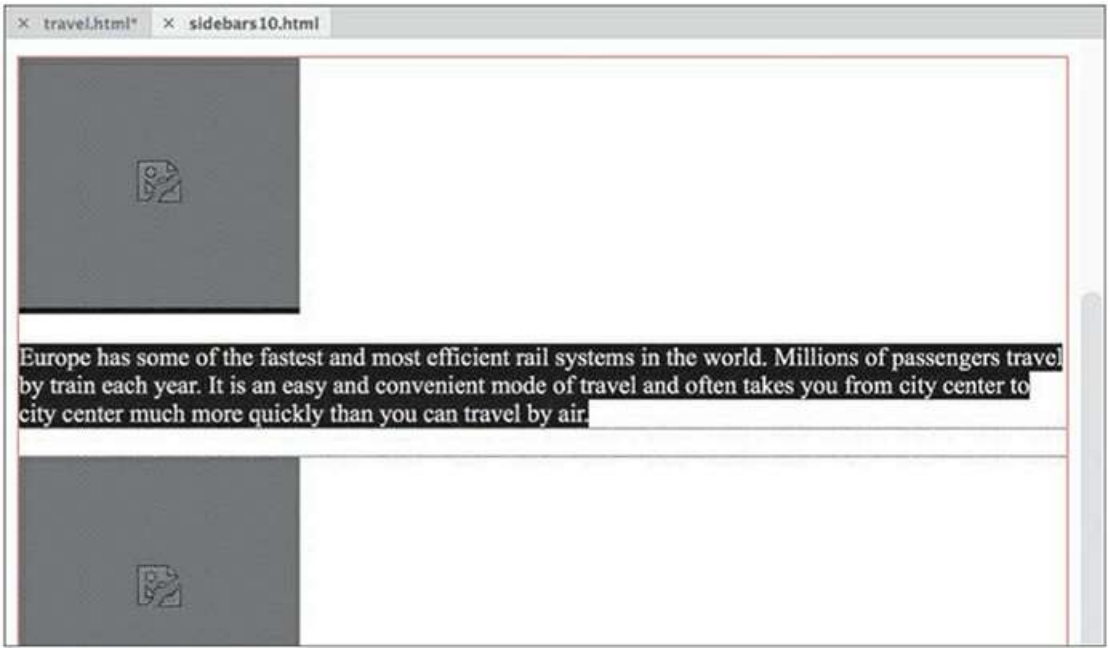
Note

It's vital that you use the same document view when copying and pasting content from one document to another in Dreamweaver.

The new content replaces the placeholder.

. In **sidebars10.html**, scroll down to the next content element.

Insert the cursor into the first paragraph. Examine the tag selectors.



The content is composed of figure and figcaption elements.

- Select the `figure` tag selector.

  Hold the Shift key and click to select all the content ending at "…welcome friendly visitors."

- Copy the selected content.

  Note that the text in Sidebar 2 in **sidebars10.html** contains images that are missing. If you check the source attributes, you'll see that the image paths are designed to work from the site root. Since **sidebars10.html** is saved in the *resources* subfolder, the images will not appear until the code is moved into the travel page and saved.

- Close **sidebars10.html**.
- In **travel.html**, insert the cursor in the Sidebar 2 placeholder.
- Select the `p` tag selector. Paste the text from step 10.

The text replaces the placeholder text in Sidebar 2. The images should now be visible.

- Open **travel-text.html** in Design view from the lesson10/resources folder.

  The **travel-text.html** file contains content in paragraphs and a table meant for the travel page. Note that the text and table are unformatted.

- Press Ctrl+A/Cmd+A to select all the text.

  Press Ctrl+C/Cmd+C to copy the contents.

  Close **travel-text.html**.

- In **travel.html**, select the text *Add main heading here*.

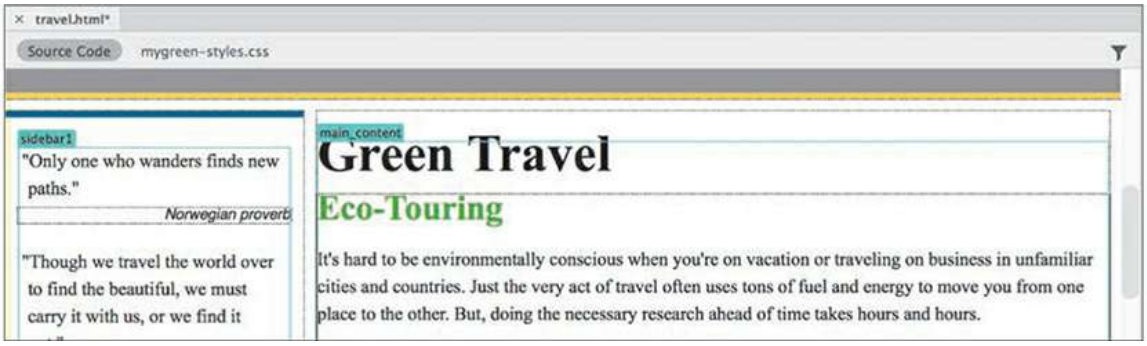
  Type **Green Travel** to replace the text.

- Select the placeholder *Add article heading here.*

  Type **Eco-Touring** to replace it.

- Select the `p` tag selector for the text *Add content here*.

  Press Ctrl+V/Cmd+V to paste.



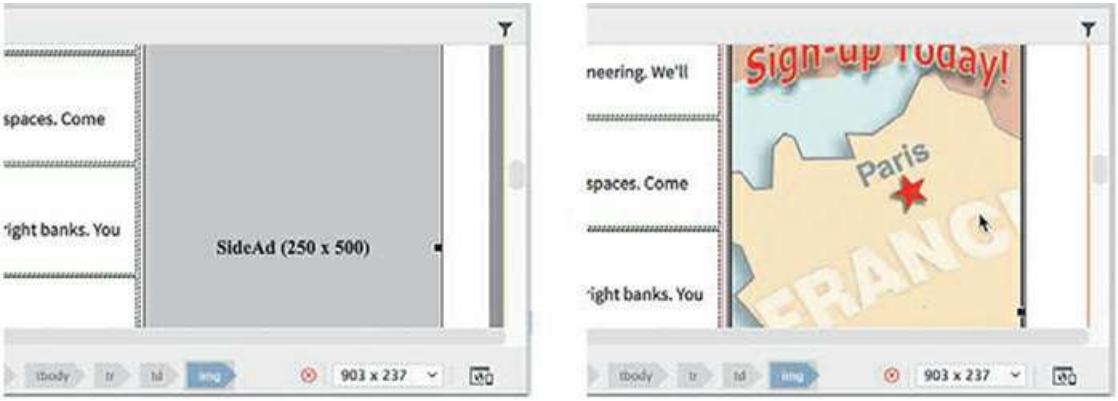
The content from **travel-text.html** appears, replacing the placeholder text. The new text will assume the default formatting for text and tables applied by the style sheet you created in Lesson 7, "Working with Text, Lists, and Tables."

Next, let’s insert the Eco-Tour ad, which will be the base image for the Swap Image behavior.

. In the table, double-click the *SideAd* placeholder.

Select **ecotour.png** from the images folder.

Click OK/Open.



The placeholder is replaced by the Eco-Tour ad. But before you can apply the Swap Image behavior, you have to identify the image you want to swap. You do this by giving the image an id.

Tip

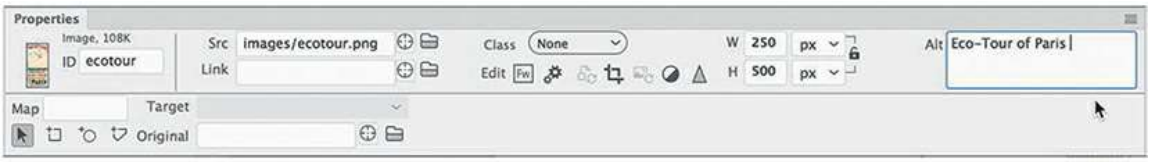
Although it takes more time, giving all your images unique ids is a good practice.

. Select **ecotour.png** in the layout.

In the Property inspector, select the existing id SideAd.

Type `ecotour` and press Enter/Return.

Enter `Eco-Tour of Paris` in the Alt field.



. Save the file.

Next, you’ll create a Swap Image behavior for the new image.

Applying a behavior

As described earlier, many behaviors are context sensitive, based on the elements or structure

present. A Swap Image behavior can be triggered by any document element, but it affects only images displayed within the page.

Note

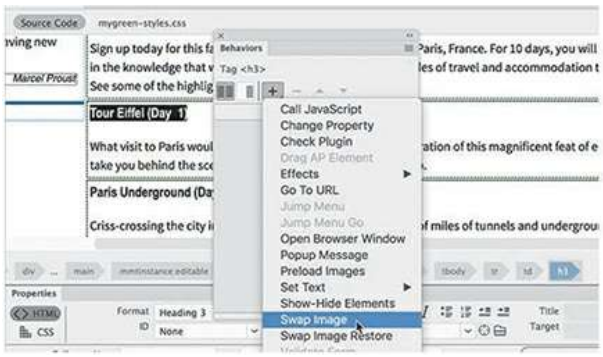
You will be able to access the Behaviors panel only when you are in Design view or Code view.

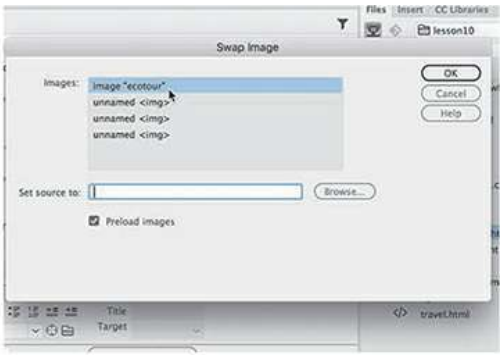
- Choose Window > Behaviors to open the Behaviors panel.
- Insert the cursor in the *Tour Eiffel* text and select the `<h3>` tag selector.
- Click the Add Behavior icon .

  Choose Swap Image from the behavior menu.

Note

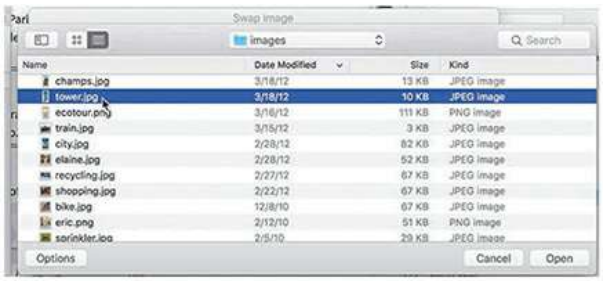
Feel free to dock the Behaviors panel with the other panels in the interface.

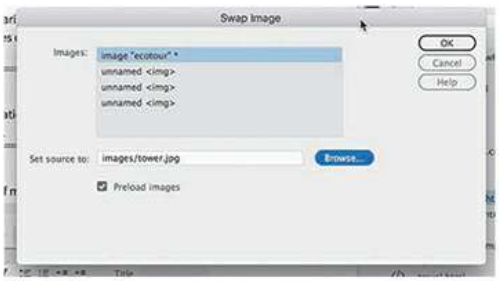




The Swap Image dialog lists any images on the page that are available for this behavior. This behavior can replace one or more of these images at a time.

- Select the `image "ecotour"` item and click Browse.
- In the Select Image Source dialog, select **tower.jpg** from the site images folder. Click OK/Open.





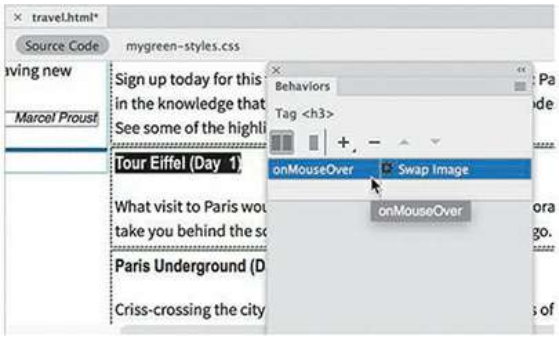
. In the Swap Image dialog, select the Preload Images option, if necessary, and click OK.

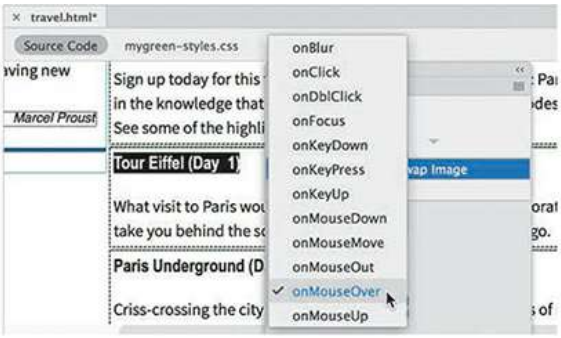
Note

The Preload Images option forces the browser to download all images necessary for the behavior when the page loads. That way, when the user interacts with the trigger, the image swap occurs without any lags or glitches.

A Swap Image behavior is added to the Behaviors panel with an attribute of `onMouseOver`. Attributes can be changed, if desired, using the Behaviors panel.

. Click the `onMouseOver` attribute to open the pop-up menu and examine the other available options.

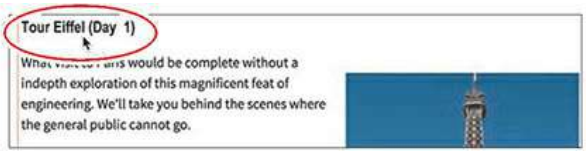




The menu provides a list of trigger events, most of which are self-explanatory. For now, however, leave the attribute as `onMouseOver`.

. Save the file. Switch to Live view to test the behavior.

Position the cursor over the *Tour Eiffel* text.



When the cursor passes over the text, the Eco-Tour ad is replaced by the image of the Eiffel Tower. But there is a small problem. When the cursor moves away from the text, the original

image doesn't return. The reason is simple: You didn't tell it to return. To bring back the original image, you have to add another command—Swap Image Restore—to the same element.

Applying a Swap Image Restore behavior

In some instances, a specific action requires more than one behavior. To bring back the Eco-Tour ad once the mouse moves off the trigger, you have to add a restore function.

- Switch to Design view.

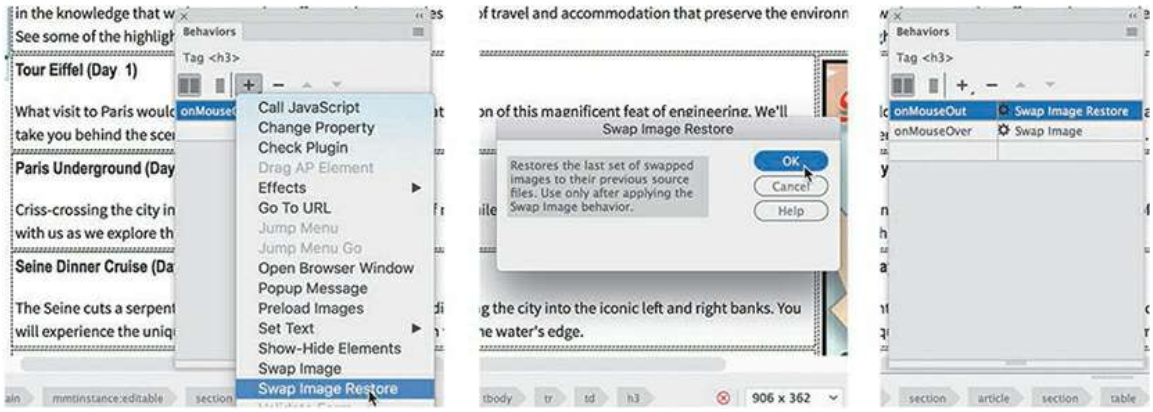
  Insert the cursor in the *Tour Eiffel* heading and examine the Behaviors panel.

  The inspector displays the currently assigned behavior. You don't need to select the element completely; Dreamweaver assumes you want to modify the entire trigger.

- Click the Add Behavior icon .

  Choose Swap Image Restore from the dropdown menu.

  Click OK in the Swap Image Restore dialog to complete the command.



  The Swap Image Restore behavior appears in the Behaviors panel with an attribute of `onMouseOut`.

- Switch to Code view and examine the markup for the *Tour Eiffel* text.

```
90          <td scope="col"><h3
            onMouseOver="MM_swapImage('ecotour','','images/tower.jpg',1)"
            onMouseOut="MM_swapImgRestore()">Tour Eiffel (Day 1)</h3>
91            <p>What visit to Paris would be complete without a indepth
```

  The trigger events—`onMouseOver` and `onMouseOut`—were added as attributes to the `<h3>` element. The rest of the JavaScript code was inserted in the document's `<head>` section.

- Save the file and switch to Live view to test the behavior.

  Test the text trigger *Tour Eiffel*.

When the pointer passes over the text, the Eco-Tour image is replaced by the one of the Eiffel Tower and then reappears when the pointer is withdrawn. The behavior functions as desired, but nothing is visibly "different" about the text. In other words, there is nothing here to prompt a user to roll their pointer over the heading. The result will be that many users will miss the swap image effect altogether.

Users sometimes need to be encouraged or directed to these types of effects. Many designers use hyperlinks for this purpose, since users are already familiar with how they function. Let's replace the current effect with one based on a hyperlink.

Removing applied behaviors

Before you can apply a behavior to a hyperlink, you need to remove the current Swap Image and Swap Image Restore behaviors.

- Switch to Design view. Open the Behaviors panel, if necessary.

  Insert the cursor in the *Tour Eiffel* text.

  The Behaviors panel displays the two applied events. Which one you delete first doesn't matter.

- Select the Swap Image event in the Behaviors panel.

  Click the Remove Event icon ..

  The Swap Image event is removed.

- Select the Swap Image Restore event.

  In the Behaviors panel, click the Remove Event icon ..

  Both events are now removed. Dreamweaver also removes any unneeded JavaScript code.

- Save the file and check the text in Live view again.

The text no longer triggers the Swap Image behavior. To reapply the behavior, you need to add a link or link placeholder to the heading.

Adding behaviors to hyperlinks

Behaviors can be added to hyperlinks even if the link doesn't load a new document. For this exercise, you'll add a link placeholder (#) to the heading to support the desired behavior.

- Switch to Design view.

  Select only the text *Tour Eiffel* in the `<h3>` element.

  Type **#** in the Property inspector Link field.

  Press Enter/Return to create the link placeholder.

  The text displays with the default hyperlink styling. The tag selector for the a tag appears.

- Click the Add Behavior icon .

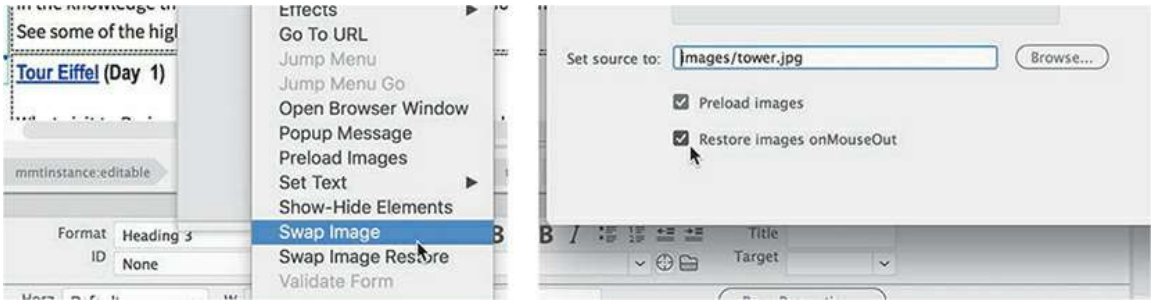
  Choose Swap Image from the pop-up menu.

  As long as the cursor is still inserted anywhere in the link, the behavior will be applied to the entire link markup.

- In the Swap Image dialog, select the item `image "ecotour".`

  Browse and select **tower.jpg** from the images folder. Click OK/Open.

- In the Swap Image dialog, select the **Preload Images** option and the **Restore Images onMouseOut** option, if necessary, and click OK.



  The Swap Image event appears in the Behaviors panel along with a Swap Image Restore event. Since the behavior was applied all at once, Dreamweaver provides the restore functionality as a productivity enhancement.

- Add a link placeholder (#) to the text *Paris Underground*. Apply the Swap Image behavior to the link. Use **underground.jpg** from the images folder.

- Repeat step 5 for the *Seine Dinner Cruise* text.

  Select the image **cruise.jpg**.

- Repeat step 5 for the *Champs Élysées* text.

  Select the image **champs.jpg**.

  The Swap Image behaviors are now complete, but the styling of the text and link don't match the site's color scheme. Let's create custom CSS rules to format them accordingly. You will need to create two rules: one for the heading element and another for the link itself.

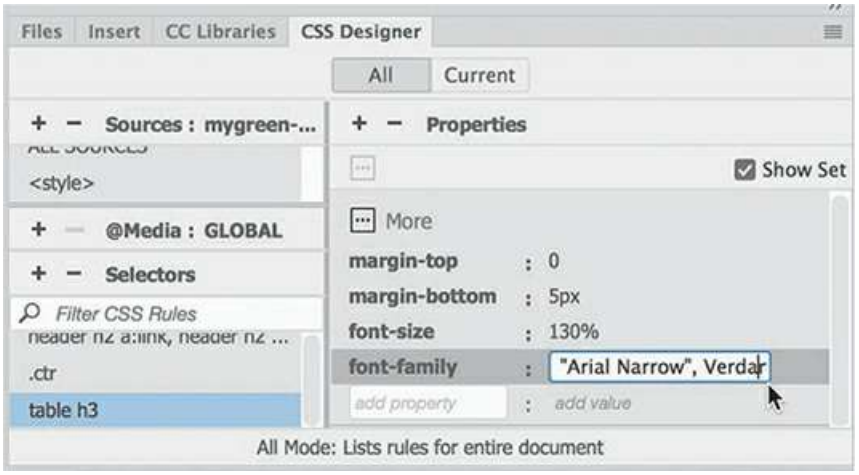
- In the CSS Designer, create a new selector in **mygreen-styles.css**:

  `table h3`

- Create the following properties in the new rule:

Click here to view code image

```
margin-top: 0px
margin-bottom: 5px
font-size: 130%
font-family: "Arial Narrow", Verdana, "Trebuchet MS",
sans-serif
```

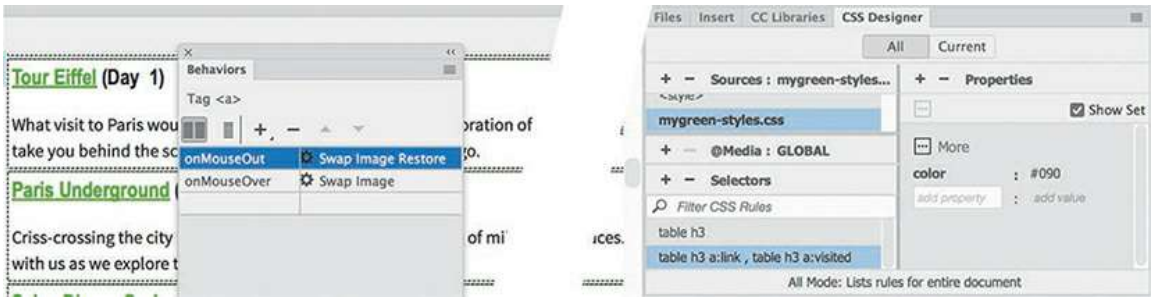


. Create a new selector:

```
table h3 a:link, table h3 a:visited
```

. Create the following properties in the new rules:

```
color: #090
```



The headings are now more prominent and styled to match the site theme.

. Save all files. Test the behaviors in Live view.

The Swap Image behavior should work successfully on all links, but only on the link itself. If one or more of the links do not function, check to make sure the behavior was assigned to the link successfully.

. Close all files.

In addition to eye-catching effects, such as the dynamic behaviors you've just been learning about, Dreamweaver also provides structural components—such as jQuery and Bootstrap widgets—that conserve space and add more interactive flair to your website.

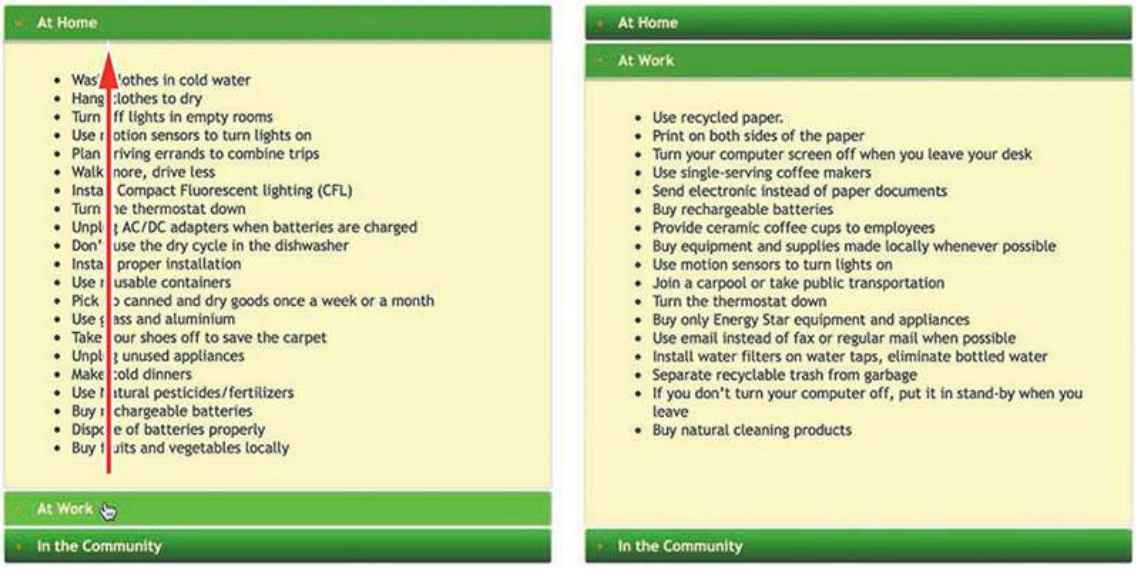
Working with jQuery accordion widgets

The jQuery accordion widget allows you to organize a lot of content into a compact space. In the accordion widget, the tabs are stacked, and when opened, they expand vertically rather than side by side. Let's preview the completed layout.

. In the Files panel, select **tips-finished.html** from the finished folder in lesson10 and open it directly in your favorite browser.

  The page content is divided among three panels using a jQuery accordion widget.

. Click each panel in turn to open and close each.



  When you click a tab, the panel slides open with a smooth action. The panels are set to a specific height; if the content is taller than the default panel size, the panel adjusts its height automatically. When the panels open and close, the bulleted lists of green tips are revealed. The accordion panel allows you to display more content in a smaller, more efficient footprint.

. Close your browser and return to Dreamweaver. Close **tips-finished.html**.

In the next exercise, you'll learn how to create and format a jQuery accordion widget.

Inserting a jQuery accordion widget

In this exercise, you'll incorporate a jQuery accordion widget into one of your existing layouts.

. Open **tips.html** in Live view.

  The page consists of three bulleted lists separated by `<h2>` headings. These lists take up a lot of vertical space on the page, requiring the user to scroll down two or more screens to read

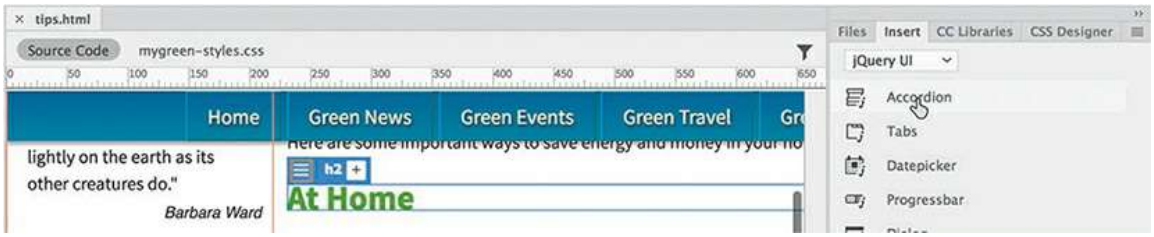
them. Keeping content on one screen as much as possible will make it easier to access and read.

One technique to maximize screen real estate is using tabbed or accordion panels. Dreamweaver (2019 Release) offers these types of components in both jQuery and Bootstrap frameworks. Since you're not using a Bootstrap layout here, let's use a jQuery accordion widget.

- Insert the cursor in the *At Home* heading and select the `<h2>` tag selector.
- Open the Insert panel.

  Select jQuery UI from the drop-down menu.

  Click the Accordion item.



The position assist dialog appears.

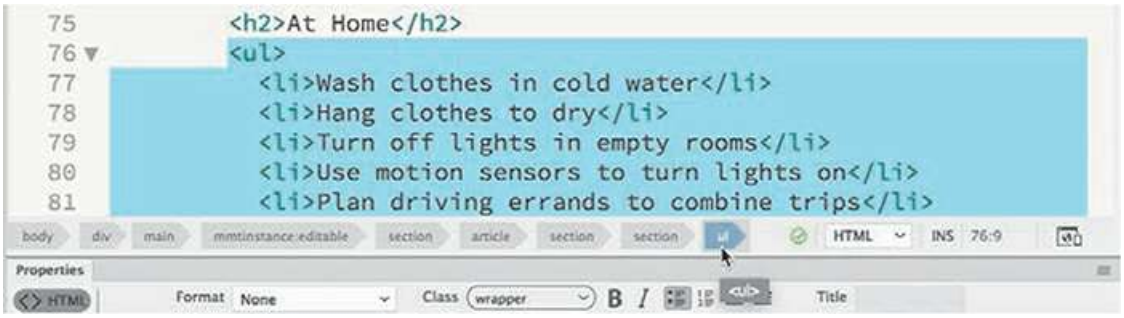
- Click Before.

Dreamweaver inserts the jQuery accordion widget element above the heading but inside the `<section>` element. The default element is a three-panel accordion widget that appears with the top panel open. The Element Display appears above the new object, focused on a div element with an id of `#Accordion1`.

The next step is to move the existing lists into the panels. Since two of the panels are hidden by default, the easiest way to work with the content will be in Code view.

- Switch to Code view.
- Scroll down and insert the cursor in the first bullet: `<li>Wash clothes in cold water.</li>` (around line 77).
- Select the ul tag selector.

  Press Ctrl+X/Cmd+X to cut the whole list.



- Delete the code `<h2>At Home</h2>`.
- Scroll up and select the heading *Section 1* (around line 62). Edit the heading to say **`At Home`**

  The new heading structure is based on an `<h3>` element.

Tip

You may not be able to see the tag selectors when you edit code elements. Remember to click the Refresh button as necessary.

- Insert the cursor into the text placeholder *Content 1* (approximately line 64). Select the p tag selector.

  The text appears in the `<div>` without any other structure. Make sure you do not delete the `<div>`.
- Press Ctrl+V/Cmd+V to paste the list.

The list markup appears in the `<div>`. The first accordion panel group is complete. You have to repeat this process for the other two lists.

- Scroll down to the *At Work* tip list (around line 101).
- Select the ul tag selector, as in step 7. Cut the list.
- Click the `<section>` tag selector. Press Delete.

  The `<section>` and the heading *At Work* are deleted.
- Select the heading *Section 2* and type **`At Work`** to replace it (approximately line 88).
- Delete the placeholder text *Content 2* and paste the list you cut in step 13 (around line 90).

- Repeat steps 12-16 to move the content for *In the Community* into the third panel.

```
109             <h3><a href="#">In the Community</a></h3>
110 ▼           <div>
111 ▼         <ul>
112           <li>Carpool with neighbors to school or the shopping mall</li>
113           <li>Put the leaf blowers away and get out the rakes</li>
114           <li>Water early in the morning or after the sun sets</li>
```

When you're finished, all three lists are now contained within Accordion 1, and all the empty `<section>` elements have been deleted.

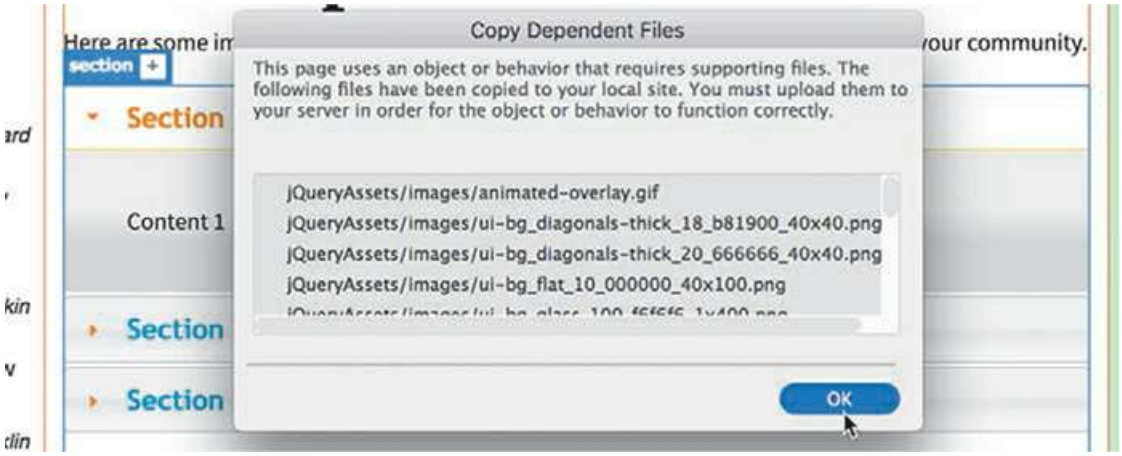
- Switch to Live view.

You inserted a jQuery accordion widget and added content to it.

- Test the panels by clicking each heading.

When clicked, the panel should open, revealing the list contained within. When you click a different heading, the new panel opens, closing the old one.

- Save all files.



When you save **tips.html**, a dialog should appear informing you that several new dependent files will be copied into the site folder. These include images, CSS, and JavaScript files that enable the interactivity of the accordion panel. These files will be stored in a folder named jQueryAssets. Whenever you upload this page to your web server, remember to include this entire folder.

- Click OK.

In the next exercise, you'll learn how to apply the site color scheme to the accordion widget.

By any other name

I keep throwing the terms jQuery and JavaScript around and you may be wondering,"What's the difference? Do I need to learn two different scripting languages?"

The good news is that jQuery is JavaScript. Created by John Resig, an American software engineer, a little over a decade ago, jQuery is a library of prebuilt JavaScript functions. Resig realized that he was constantly writing the same code over and over to perform specific tasks in JavaScript. By supplying a library of these functions, jQuery enables you to create a set of complex behaviors with a minimum amount of coding.

The Insert panel of Dreamweaver hosts 11 jQuery widgets, including tabbed panels, dialog boxes, date pickers, and other handy webpage components. Like the accordion, each item provides powerful interactive functionality to your site that can be added using a simple user interface that requires no programming knowledge orcoding experience.

Styling a jQuery accordion

As with the basic layout and the other jQuery components created by Dreamweaver, the accordion is formatted by the jQuery CSS and JavaScript files. You should avoid editing these files directly unless you know what you are doing. Instead, you'll apply the site design theme to Accordion 1 using your own custom style sheet. Let's start with the tabs.

Tip

As you work in Code view, you may need to click the Refresh button from time to time in the Property inspector to see the tag selectors.

. Click the *At Home* tab. Examine the tag selectors.

The tab is composed of two elements: <h3> and <a>. But that's only on the surface. Behind the scenes, the jQuery and CSS functions are manipulating the HTML and CSS to produce the various behaviors controlling the accordion. As you move your mouse over the tabs and click them, class attributes are being added and changed on the fly to produce the hover effects and animated panels. In most cases, you won't even see it happen.

As you learned earlier, hyperlinks exhibit four basic behaviors: `link`, `visited`, `hover`, and `active`. The jQuery library takes advantage of these default states to apply the various effects you see when interacting with Accordion 1 for the *At Home* list.

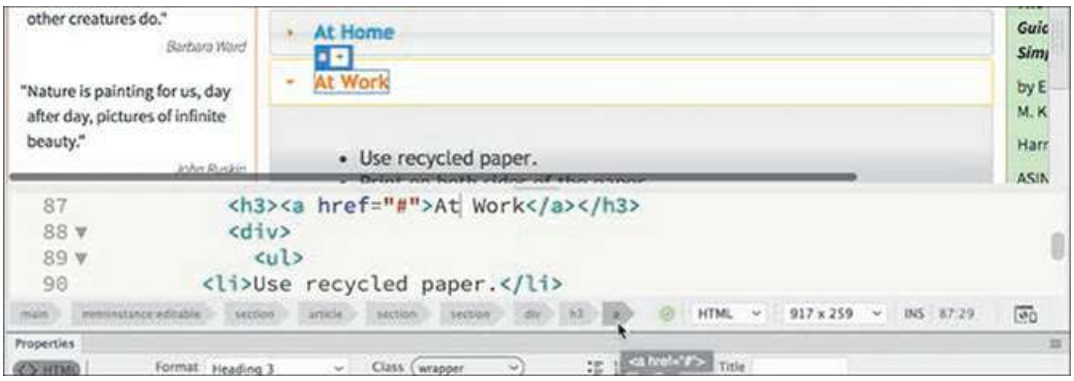
Your job will be to create several new rules that will override the default styling and apply the GreenStart theme instead. The first step is to format the default state of the tabs. Since only one tab can be open at a time, the closed state is considered the default state.

When the accordion is inserted, the first tab is typically open by default. Normally, you can troubleshoot the CSS by selecting an element in Live view. But since the tabs are designed to open when you click them, selecting a closed tab in Live view poses a problem. Luckily, you can use Code view.

. Select Split view, if necessary.

In Code view, insert the cursor in the text *At Work*.

Select the a tag selector.



Closed tabs are currently styled a light gray with a slight gradient background. You need to identify any rules that format the background color of the accordion tab. Be aware that there may be more than one rule affecting these properties and, as you learned in Lesson 9, "Working with Navigation," some of the styling is applied dynamically. So you have to distinguish rules that apply by default and ones that work only by user interaction.

. In CSS Designer, click the Current button.

The Selectors panel displays all the rules that are currently styling the selected element.

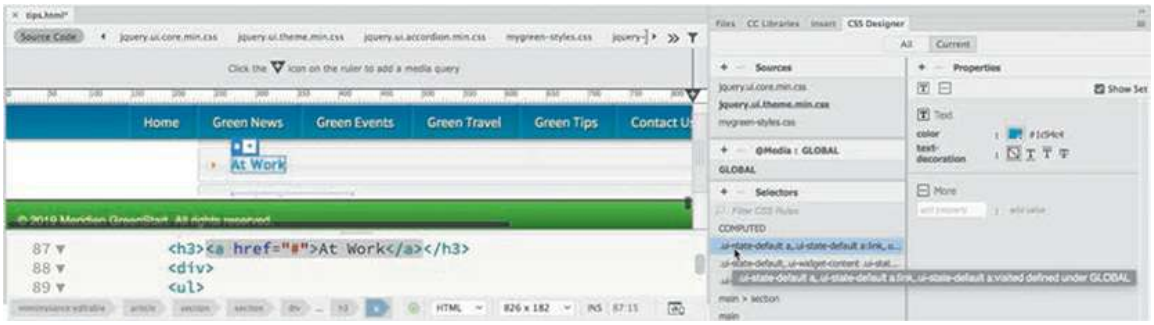
Note

Be aware that the selector display in CSS Designer changes based on whether the tab is open or closed.

. If necessary, click the first rule in the list:

Click here to view code image

```
.ui-state-default a,.ui-state-default a:link,.ui-state-default
a:visited
```



Note that the Element Display focuses only on the <a> element. You can see that the properties format the text color and turn off the underscore. It's clear that this rule does not format the entire tab, but we can use it later to style the text color.

One class, two class, three class, more!

You may have noticed that many of the rules that came in with the jQuery accordion feature multiple classes. You may be wondering, "What's up with that?"

As you learned in Lesson 3, "CSS Basics," classes are used to apply styling in special circumstances, such as an open or closed accordion tab. But why multiple classes in one selector?

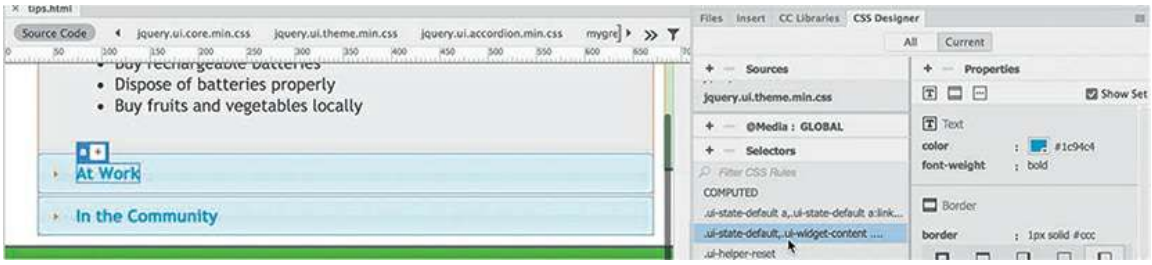
CSS rules often apply several properties to the targeted element. But if you need to provide different styles for different states of the object (open or closed, and so on), you may want to split the styles into several classes and apply several to one element. That way, one rule will supply the basic default styling and the others will only need to provide styling for special events.

You can sometimes figure out the intention of the class by seeing how it was named. In the jQuery style sheets, you will see several words used multiple times, like *default*, *active*, *hover*, and so on. Some of the events are supported directly by HTML, such as a:link and a:hover. Others, like *default* and *active*, will be applied and swapped out by JavaScript.

. Click the rule

Click here to view code image

```
.ui-state-default,.ui-widget-content .ui-state-default,.ui-
widget-header .ui-state-default
```



Notice that all three tabs are highlighted in the document window, although the Element Display continues to focus on the <a> element. You can also see that this rule includes styling for the tab background, which means you can use it to reformat the default styling.

By looking at the Sources pane, you can see that the highlighted rule is located in the file **jquery.ui.theme.min.css**. The goal is to duplicate this rule in **mygreen-styles.css** so that you can override the default styling on the tab.

Unfortunately, Dreamweaver doesn't have an easy, built-in feature that does everything that we need all at once. But you can still use the CSS Designer to get the same result.

. Note the exact name of the targeted rule.

Click the All button in the CSS Designer.

The Selectors window now shows all the rules in **jquery.ui.theme.min.css**. The rule you need to duplicate should still be visible, but it may no longer be highlighted.

. Double-click the selector

Click here to view code image

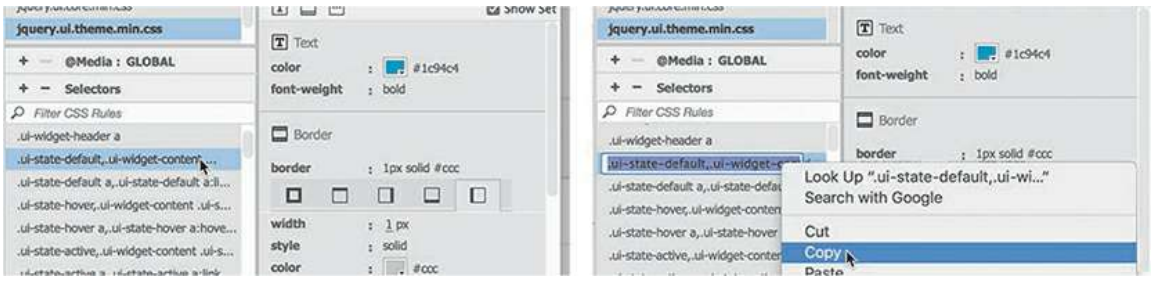
```
.ui-state-default,.ui-widget-content .ui-state-default, .ui-
widget-header .ui-state-default
```

The name should now be editable

Note

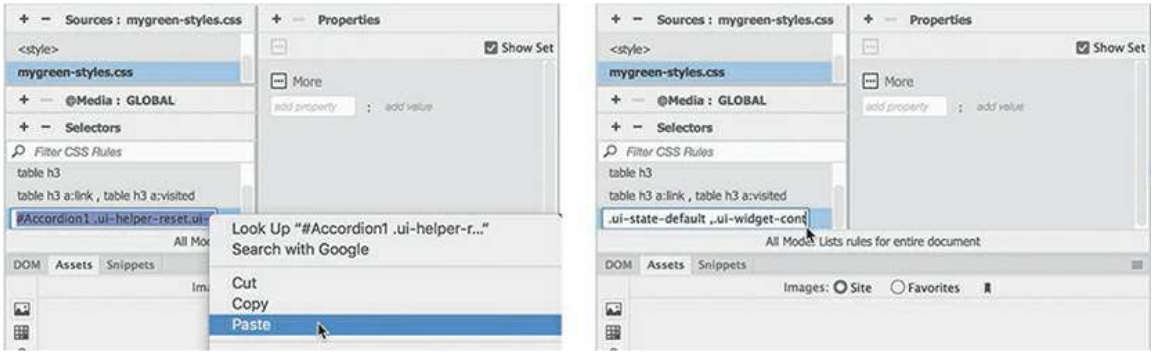
Selectors cannot be edited while the Current button is selected.

. Copy the selector.



- In the Sources pane, select **mygreen-styles.css**.
- Click the Add Selector icon.

  A new selector name appears.
- Paste the selector name copied in step 8 and press Enter/Return as necessary to create the new selector.



- In **jquery.ui.theme.min.css**, select the rule

  Click here to view code image

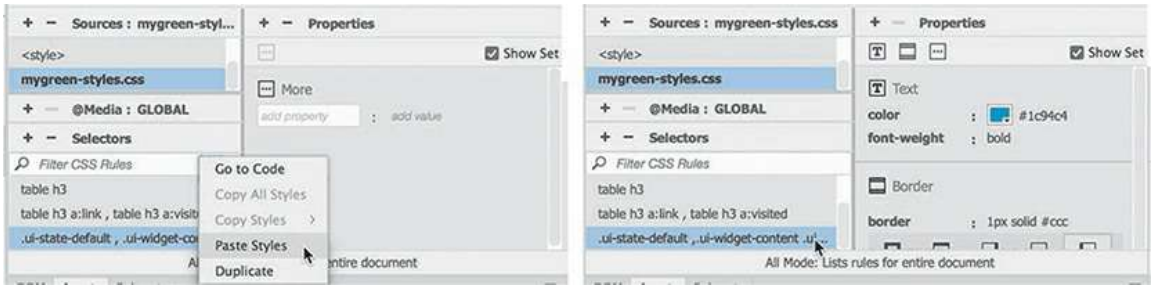
```
.ui-state-default,.ui-widget-content .ui-state-default, .ui-
widget-header .ui-state-default
```

- Right-click the rule, and select Copy All Styles from the context menu.
- In **mygreen-styles.css** select the rule

  Click here to view code image

```
.ui-state-default,.ui-widget-content .ui-state-default, .ui-
widget-header .ui-state-default
```

- Right-click the rule, and select Paste Styles from the context menu.



  The selector and all the styles from the original rule are now duplicated in your site style sheet.
- Save all files.

Bringing across all the properties enables you to identify the ones you need to modify. First,

we'll start with the background effect.

Applying a background effect to the accordion tab

Since some elements already feature the site theme, it's a simple matter to grab styling from another element using the CSS Designer.

In this exercise, you'll use the footer styling to format the accordion tab.

- Right-click the footer rule in **mygreen-styles.css**.

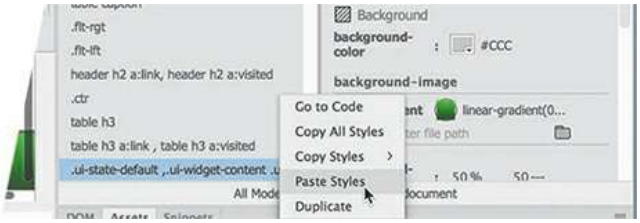
  Select Copy Styles > Copy Background Styles from the context menu.

- Right-click the rule

  Click here to view code image

  ```
  .ui-state-default,.ui-widget-content .ui-state-default,.ui-
  widget-header .ui-state-default in mygreen-styles.css.
  ```

- Select Paste Styles from the context menu.



  The background color and gradient properties are added to the new rule. The tabs now have the same gradient background as were applied to the footer. The text in the tabs is now hard to read.

- Edit the following property:

  ```
  color: #FFC
  ```

  When you change the property, nothing happens in the layout. That's because the text color is also being supplied by the rule styling the hyperlink. You changed the color applied to the heading, but you'll also need a new rule to format the <a> element.

- In **jquery.ui.theme.min.css**, select the rule

  Click here to view code image

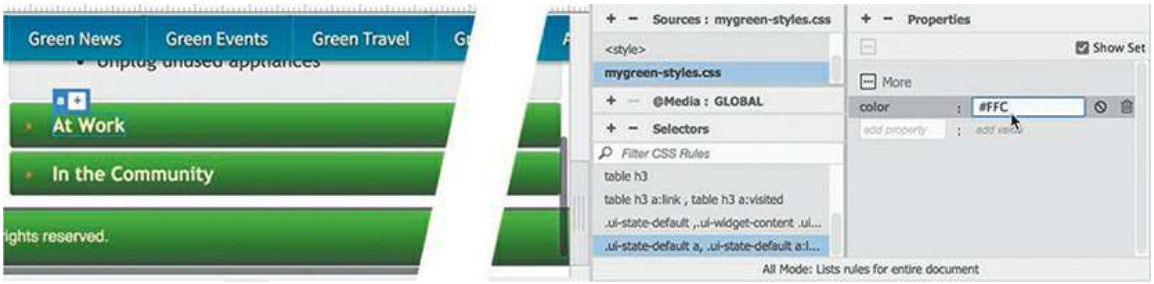
  ```
  .ui-state-default a,.ui-state-default a:link,.ui-state-default
  a:visited
  ```

- Double-click the selector and copy the name.

- In **mygreen-styles.css** create a new selector and paste the name copied in the previous step. Press Enter/Return as necessary to complete the selector.

- Create the following property:

```
color: #FFC
```



The text in the tab is now styled in pale yellow, a nice contrast to the green gradient.

- Save all files.

The new rule completes the formatting of the default tab state. You still need to style the open and hover states.

Formatting a conditional state for an accordion tab

In this exercise, you will identify and style the open state of the accordion tab.

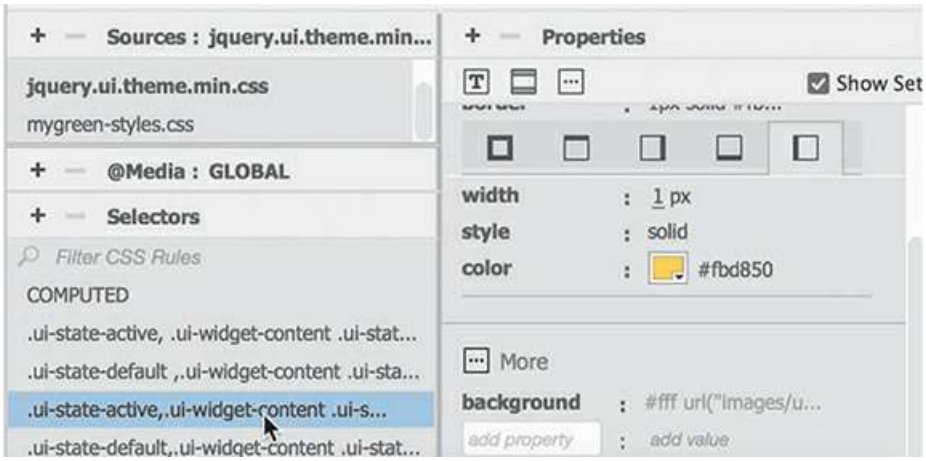
- If necessary, open **tips.html** in Split view.

  The file has a jQuery accordion in the layout with three panels. One panel is open by default. CSS Designer can identify the rules that affect this component and its various states.

- If necessary, click the first tab in Live view to open it.
- In Code view, insert the cursor in the text *At Home*.

  Click the Current button.

  Select the h3 tag selector.



You will sometimes find that the display in the CSS Designer is different when you use Code

view over Live view.

- Examine the rules affecting the `<h3>` element.

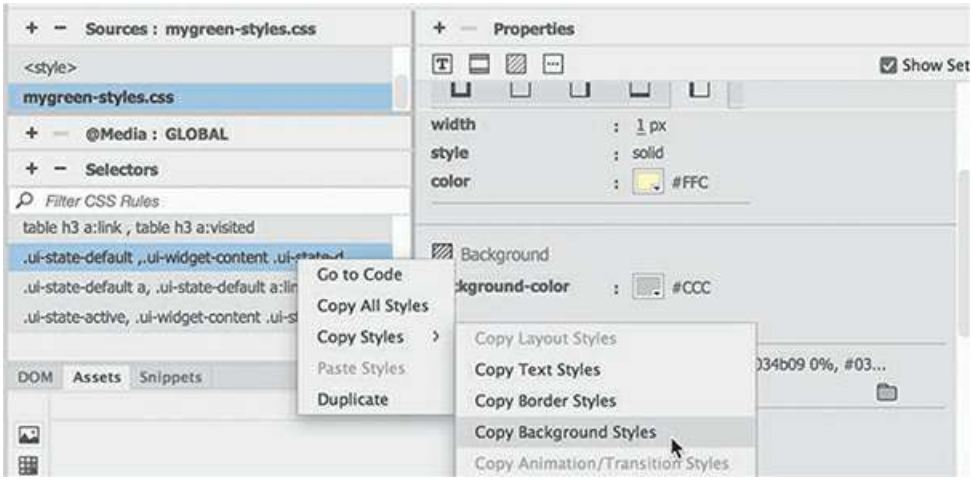
  Of the seven rules displayed, only one affects the *active* state of the tab. To change the styling of the open tab, you will need to duplicate this rule in your site style sheet and create alternative styling.

- Click the All button.
- In **jquery.ui.theme.min.css**, double-click the rule

  Click here to view code image

```
.ui-state-active,.ui-widget-content .ui-state-active, .ui-
widget-header .ui-state-active
```

- Copy the entire selector name.
- In **mygreen-styles.css** click the Add selector icon ✚, paste the name copied in step 7, and press Enter/Return to create the selector.
- Copy the background styles from the default state of the tab.



- Paste the styles on the rule created in step 8.

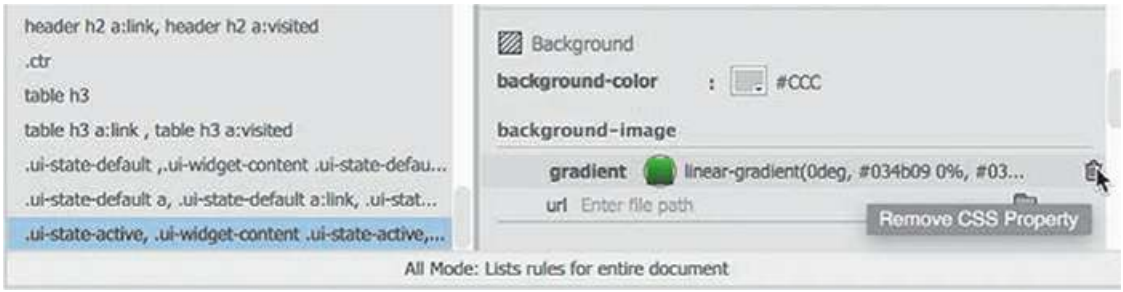
  The new rule has the same background as the default state of the tab. To reset the styling, you have to turn off the background gradient.

- In CSS Designer, click the Show Set option to disable it.
- Select the rule

  Click here to view code image

```
.ui-state-active, .ui-widget-content .ui-state-active, .ui-
widget-header .ui-state-active
```

- In the Background category, remove the background-image property.



There's no change to the styling, because the default state styling is still being inherited. For the open state, let's format it with a solid color. To do that, you have to turn off or reset the gradient being applied by the other rules.

- In the `background-image` property URL field, enter **`none`** and press Enter/Return.

When you press Enter/Return, the gradient is turned off and the tab turns gray. The gray color comes from the `background-color` property in the same rule. This is an important point to remember: one rule may interfere or override another.

- Change the background-color to **`#090`**

The open tab now displays a solid green color.

- In Live view, test the three tabs.

  As you click each tab, it opens and reformats. The open tabs display the solid green background; the closed tabs show the gradient.

- Save all files.

Now that you have addressed the default and active states, the last step is to format the hover state.

Using Live Code to identify dynamic styling

When you are working on a complex component like the jQuery accordion, it may be difficult to identify exactly where specific styling is coming from, especially in dynamic elements like accordion tabs.

Luckily, Dreamweaver provides an enhanced workflow that allows you to see what's happening behind the scenes and helps you track down just such styling questions.

. If necessary, open **tips.html** in Split view. Scroll down to view the accordion markup.

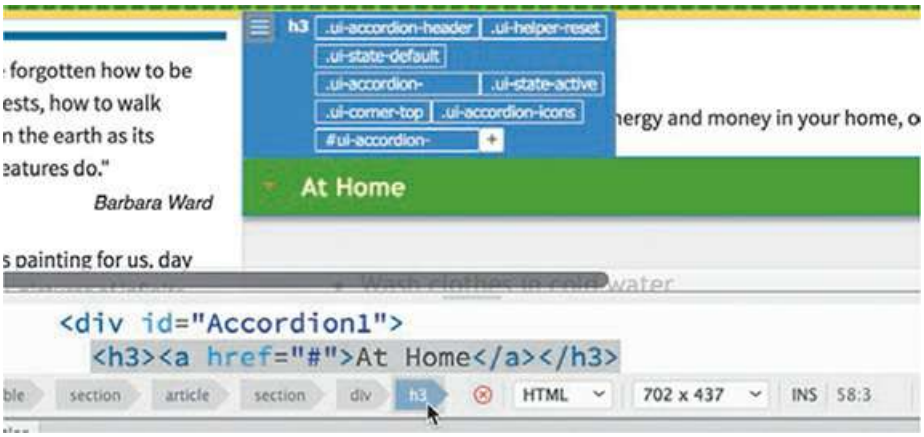
```
<div id="Accordion1">
  <h3><a href="#">At Home</a></h3>
  <div>
    <ul>
          <li>Wash clothes in cold water</li>
          <li>Hang clothes to dry</li>
```

On the surface, the accordion seems like a simple component. It is composed of four div elements and three h3 headings. It sports only a single id. The simplicity of the HTML belies the actual complexity of the final product. In most HTML editors, you would never be able to see the truth, but that's not the case with Dreamweaver.

Using Live view you can get a partial sense of the accordion's true nature.

. Click the first tab in Live view.

If necessary, select the `h3` tag selector.

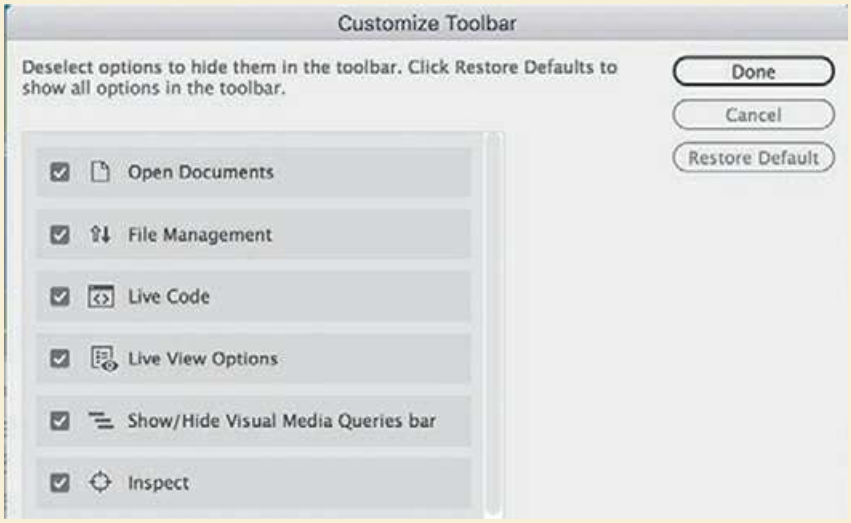


The Element Display appears focused on the h3 element and displays eight class names. But in Code view, there are no classes at all. So what gives?

Live view is based on the WebKit rendering engine, which is the same one used in the Safari browser. It doesn't just display the HTML that you can see in Code view; it also picks up the CSS and JavaScript, and then previews the page almost exactly the way it would appear in the browser. So although no classes appear in the code, Live view shows what's generated by jQuery in the background. Dreamweaver's Live Code feature can actually show you what's happening in Live view.

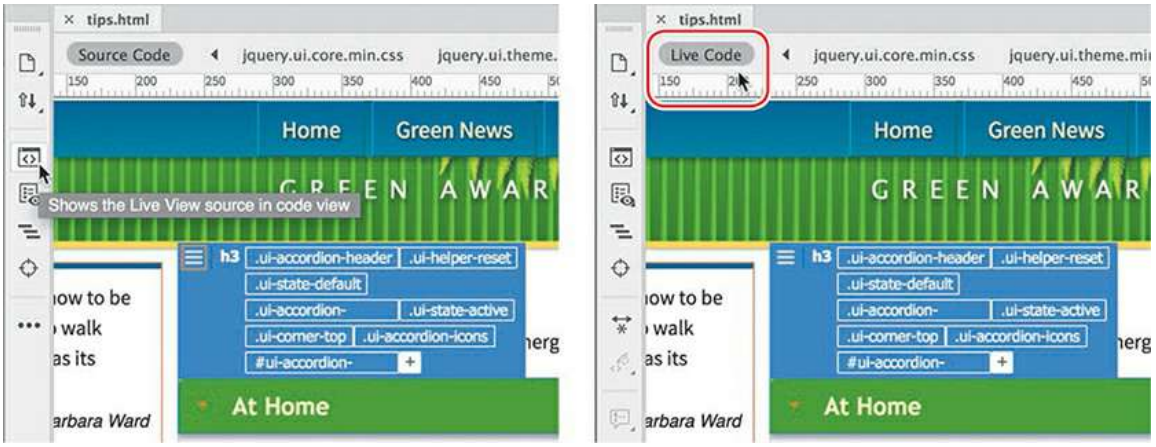
Missing tools?

To use Live Code fully, you may need to customize the Common toolbar. If any of the tools mentioned in the exercise are missing, you can add them to the toolbar by clicking the Customize Toolbar icon ••• and then selecting them in the Customize Toolbar dialog.



- In the Common toolbar, click to open the Live View Options pop-out menu.
- Select the option Hide Live View Displays to enable it.

- Click the Live Code icon .



In the Related Files interface, the Source Code reference changes to say Live Code. The Code view window now shows how the source code is being manipulated by the jQuery functions. The window will continue to update as you interact with the page.

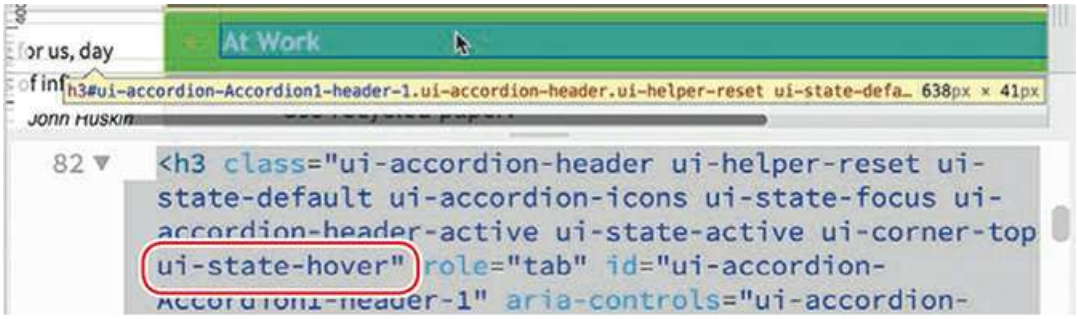
- In Code view, scroll down as needed to show the tab *At Home*.

With Live Code enabled, the markup has changed dramatically. You can now see all the classes and other attributes added by JavaScript. But this shows you only half the magic that's possible.

- Click the Inspect Mode icon .

Inspect mode turns your cursor into an interactive inspection tool. Position the inspection tool over components in your page, and Code view dynamically refreshes the code. At the same time, the CSS Designer displays the various rules and properties styling the targeted element.

- Position the cursor over each tab and observe the changes in Code view and in the CSS Designer.



As the cursor passes over the tab, you can see the changes to the markup. When the cursor is over the tab, you can see the class ui-state-hover added to the h3 element. When the cursor moves away, the hover class is removed. You can freeze the CSS display by clicking the element.

- Click the tab *At Work*.

The Selectors pane displays 14 rules affecting the targeted element. Some are coming from **mygreen-styles.css** and others are supplied by the jQuery framework. You can see that one of the selectors contains the class `.ui-state-hover`. To style the hover state of the tabs, you should duplicate this rule in **mygreen-styles.css**.

. Select the rule that contains the class `.ui-state-hover`

  The rule is contained in **jquery.ui.theme.min.css**.

. Click the All button.

. In **jquery.ui.theme.min.css**, copy the selector

  Click here to view code image

```
.ui-state-hover,.ui-widget-content .ui-state-hover, .ui-widget-
header .ui-state-hover,.ui-state-focus, .ui-widget-content .ui-
state-focus,.ui-widget-header .ui-state-focus
```

. Select **mygreen-styles.css**. Create a new selector.

  Paste the selector copied in step 12.

. Press Enter/Return as needed to create the new rule.

  You can use this rule to format the hover state of the tab.

. Add the following properties to the new rule:

  Click here to view code image

```
background-color: #0C0

background-image: none
```

  The accordion tabs are now fully styled, as a hyperlink would be. There are styles for each state: `link`, `visited`, `active`, and `hover`. However, the active state is out of order. For all the styling to be effective, the active class must appear after the hover class. The CSS Designer allows you to reorder rules by dragging them in the selector pane.

. In **mygreen-styles.css**, drag the hover rule above the active rule.

  Test the styling in Live view.

. Position the cursor over each tab.

  The new hover rule styles only the closed tabs. It's a behavior that will prompt visitors to click them to see what's inside.

The interface is still in Live Code mode. The mode is handy for troubleshooting, but you may find it unnecessary for most workflows. It also tends to slow down the other functions of the program. It's a good idea to turn it off until the next time you need it.

- Click the Live Code icon to turn off Live Code mode.
- In the Common toolbar, click to open the Live View Options pop-out menu.
- Select the option Hide Live View Displays to disable the option.
- Save all files.

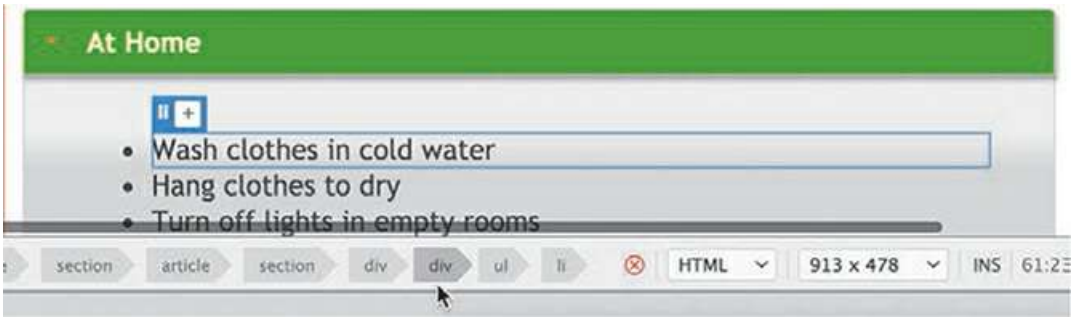
The last step in styling the accordion is to apply an appropriate fill color in the content area.

Styling the background of the accordion content

Each tab introduces a content area containing an HTML list. The default styling of the accordion applies a light-gray gradient background. In this exercise, you will apply a new background effect to the content area that better matches the site color scheme.

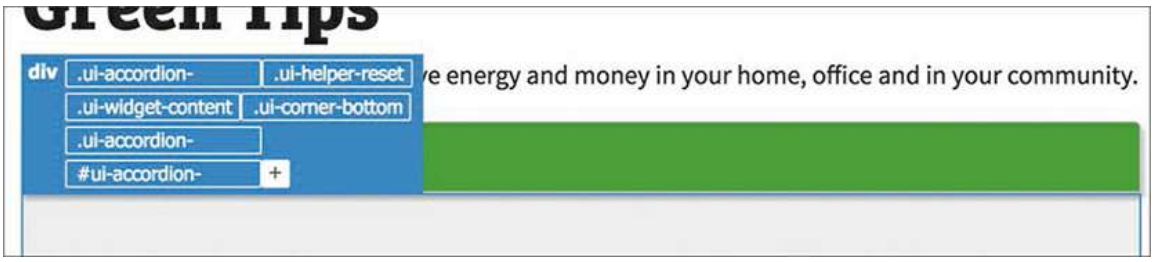
- If necessary, open **tips.html** in Live view.

  By default, one of the accordion tabs is always open.
- In Live view, select one of the bullets in the open content area. Examine the tag selector interface.



The first parent (the one closest to the ul element) of the HTML list is a `<div>` element. You need to identify the rule styling the content area.

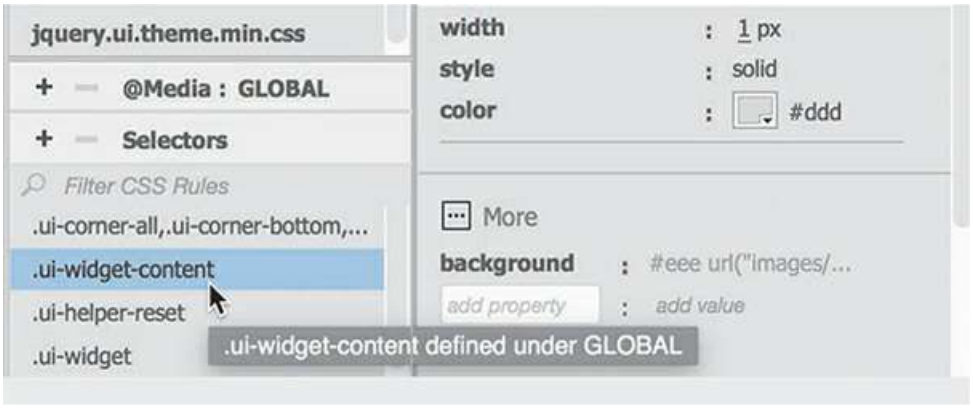
- Select the tag selector for the first parent div. Examine the Element Display.



The Element Display lists the classes applied dynamically to the content area. Sometimes the

class names scream out at you, “Here I am!” One of the classes assigned to the `<div>` is `.ui-widget-content`.

- Click the Current button in the CSS Designer. Select the rule: `.ui-widget-content` and examine the properties.



As expected, the rule applies the current background effect. You can use the same rule to reset it.

- Click the All button.
- In **mygreen-styles.css**, create the following selector:

```
.ui-widget-content
```

- Add the following properties to the new rule:

Click here to view code image

```
background-image: none
background-color: #FFC
```

The content area displays a pale yellow background. The accordion is fully styled.

- Save and close all files.

The accordion is just one of more than 100 jQuery and Bootstrap widgets and components offered by Dreamweaver. They allow you to incorporate advanced functionality into your website, while requiring little or no programming skill. All of these components can be accessed via either the Insert menu or the Insert panel.

Adding interactivity to your webpages opens new possibilities of interest and excitement for your visitors, engaging them in new ways. It can easily be overdone, but a wise use of

interactivity can help bring in new visitors and keep your frequent visitors coming back for more.

Review questions

1. What is a benefit of using Dreamweaver behaviors?
2. What three steps must be used to create a Dreamweaver behavior?
3. What's the purpose of assigning an id to an image before applying a behavior?
4. What does the jQuery accordion widget do?
5. What Dreamweaver tools are helpful in troubleshooting CSS styling on dynamic elements?

Review answers

1. Dreamweaver behaviors add interactive functionality to a webpage quickly and easily.
2. To create a Dreamweaver behavior, you need to create or select a trigger element, select a desired behavior, and specify the parameters.
3. The id is helpful for identifying specific images during the process of applying a behavior.
4. A jQuery accordion widget includes multiple collapsible panels that hide and reveal content in a compact area of the page.
5. The Current mode of the CSS Designer helps identify any existing CSS styling, and Live Code and Inspect mode can be used to troubleshoot dynamic CSS and JavaScript effects.

11 Publishing to the Web

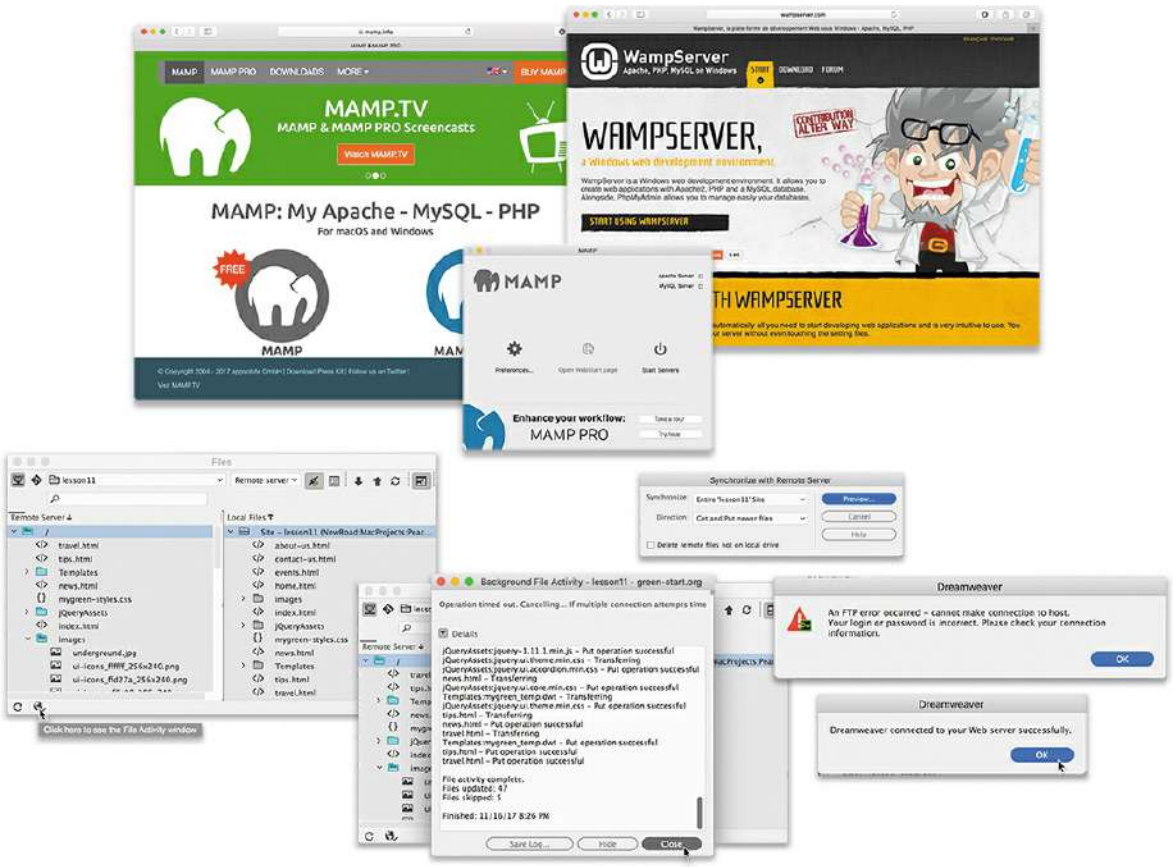
Lesson overview

In this lesson, you'll publish your website to the Internet and do the following:

- Define a remote site
- Define a testing server
- Put files on the web
- Cloak files and folders
- Update out-of-date links sitewide

This lesson will take about 1 hour and 15 minutes to complete. If you have not already done so, please log in to your account on peachpit.com to download the project files for this lesson asdescribed in the "Getting Started" section at the beginning of this book and follow the instructions under "Accessing the Lesson Files and Web Edition." Define a site based on the lesson11 folder.

Your Account page is also where you'll find any updates to the lesson files. Look on the Lesson & Update Files tab to access the most current content.



The goal of all the preceding lessons is to design, develop, and build pages for a remote website. But Dreamweaver doesn't abandon you there. It also provides powerful tools with which to upload and maintain a website of any size over time.

Defining a remote site

Dreamweaver's workflow is based on a two-site system. One site is in a folder on your computer's hard drive and is known as the *local site*. All work in the previous lessons has been performed on your local site. The second site, known as the *remote site*, is established in a folder on a web server, typically running on another computer, and is connected to the Internet and publicly available. In large companies, the remote site is often available only to employees via a network-based intranet. Such sites provide information and applications to support corporate programs and products.

Note

If you have not already downloaded the project files for this lesson to your computer from your Account page and defined a site based on this folder, make sure to do so now. See "Getting Started" at the beginning of the book.

Dreamweaver supports several methods for connecting to a remote site.

FTP (File Transfer Protocol)—The standard method for connecting to hosted websites.

SFTP (Secure File Transfer Protocol)—A protocol that provides a method to connect to hosted websites in a more secure manner to preclude unauthorized access or interception of online content.

FTP over SSL/TLS (implicit encryption)—A secure FTP (FTPS) method that requires that all clients of the FTPS server be aware that SSL is to be used on the session. It is incompatible with non-FTPS-aware clients.

FTP over SSL/TLS (explicit encryption)—A legacy-compatible, secure FTP method where FTPS-aware clients can invoke security with an FTPS-aware server without breaking overall FTP functionality with non-FTPS-aware clients.

Local/network—A local or network connection is most frequently used with an intermediate web server, known as a *staging server*. Staging servers are typically used to test sites before they go live. Files from the staging server are eventually published to an Internet-connected web server.

WebDav (Web Distributed Authoring and Versioning)—A web-based system also known to Windows users as Web Folders and to Mac users as iDisk.

RDS (Remote Development Services)—Developed by Adobe for ColdFusion and primarily used when working with ColdFusion-based sites.

Dreamweaver can now upload larger files faster and more efficiently and as a background activity, allowing you to return to work more quickly. In the following exercises, you'll set up a remote site using the two most common methods: FTP and Local/Network.

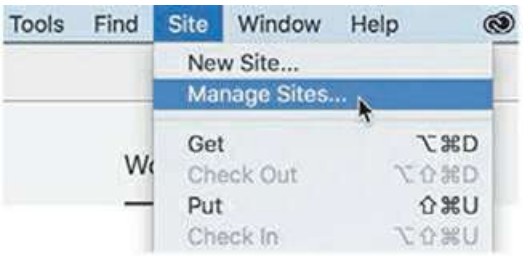
Setting up a remote FTP site

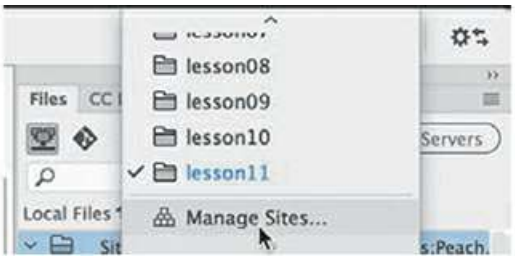
The vast majority of web developers rely on FTP to publish and maintain their sites. FTP is a well-established protocol, and many variations of the protocol are used on the web—most of which are supported by Dreamweaver.

Warning

To complete the following exercise, you must have a remote server already established. Remote servers can be hosted by your own company or contracted from a third-party web-hosting service.

- Launch Adobe Dreamweaver CC (2019 release) or later.
- Choose Site > Manage Sites, or choose Manage Sites from the site list dropdown menu in the Files panel.





In the Manage Sites dialog is a list of all the sites you may have defined.

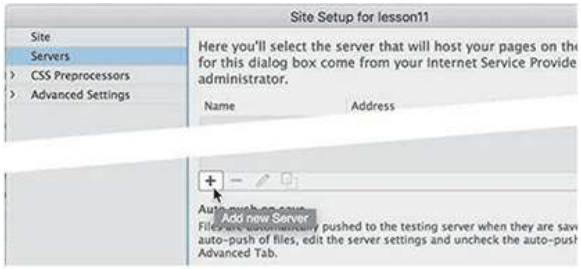
- Make sure that the current site, lesson11, is selected. Click the Edit icon .

- In the Site Setup dialog for lesson11, click the Servers category.

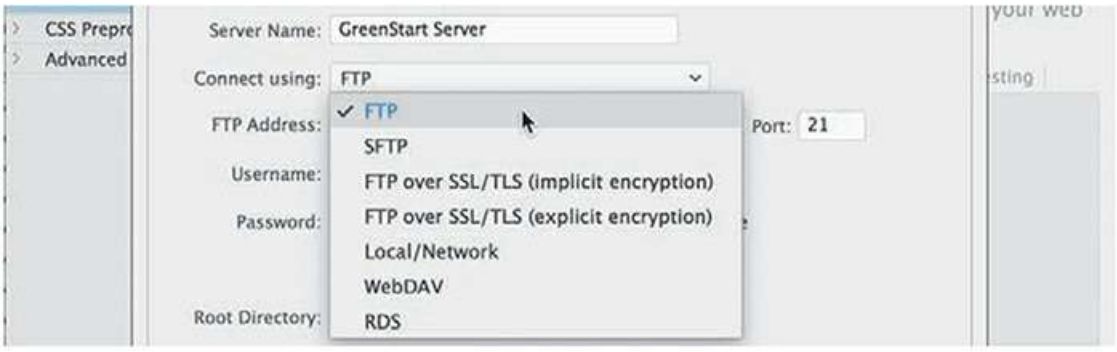
  The Site Setup dialog allows you to set up multiple servers so that you can test several types of installations, if desired

- Click the Add New Server icon .

  Enter **GreenStart Server** in the Server Name field.



- From the Connect Using pop-up menu, choose FTP.



Note

If necessary, select a different protocol to match your available server.

. In the FTP Address field, type the URL or IP (Internet protocol) address of your FTP server.

If you contract a third-party service as a web host, you will be assigned an FTP address. This address may come in the form of an IP address, such as 192.168.1.100. Enter this number into the field exactly as it was sent to you. Frequently, the FTP address will be the domain name of your site, such as **ftp.green-start.org**. But don't enter the characters *ftp* into the field.

Tip

If you are in the process of moving an existing site to a new Internet service provider (ISP), you may not be able to use the domain name to upload files to the new server. In that case, the IP address can be used to upload files initially.

. In the Username field, enter your FTP username.

In the Password field, enter your FTP password.

Usernames may be case sensitive, but password fields almost always are; be sure you enter them correctly. Often, the easiest way to enter them is to copy them from the confirmation email from your hosting company and paste them into the appropriate fields.

Note

The username and password will be provided by your hosting company.

. In the Root Directory field, type the name of the folder that contains documents publicly accessible to the web, if any.

Some web hosts provide FTP access to a root-level folder that might contain nonpublic folders—such as cgi-bin, which is used to store common gateway interface (CGI) or binary scripts—as well as a public folder. In these cases, type the public folder name—such as public, public_html, www, or wwwroot—in the Root Directory field. In many web-host configurations, the FTP address is the same as the public folder, and the Root Directory field should be left blank.

Tip

Check with your web-hosting service or IS/IT manager to obtain the root directory name, if any.

. Select the Save checkbox if you don't want to re-enter your username and password every time Dreamweaver connects to your site.

. Click Test to verify that your FTP connection works properly.

Tip

If Dreamweaver does not connect to your host, first check the username and password, as well as the FTP address and root directory for any errors.

Dreamweaver displays an alert to notify you that the connection was successful or unsuccessful.

. Click OK to dismiss the alert.

If Dreamweaver connects properly to the webhost, skip to step 14. If you received an error message, your web server may require additional configuration options.

. Click the More Options triangle to reveal additional server options.

Consult the instructions from your hosting company to select the appropriate options for your specific FTP server:

- **Use Passive FTP**—Allows your computer to connect to the host computer and bypass a firewall restraint. Many web hosts require this setting.
- **Use IPV6 Transfer Mode**—Enables connection to IPV6-based servers, which use the most recent version of the Internet transfer protocol.
- **Use Proxy**—Identifies a secondary proxy host connection as defined in your Dreamweaver preferences.
- **Use FTP Performance Optimization**—Optimizes the FTP connection. Deselect this option if Dreamweaver can't connect to your server.
- **Use Alternative FTP Move Method**—Provides an additional method to resolve FTP conflicts, especially when rollbacks are enabled or when moving files.

Once you establish a working connection, you may need to configure some advanced options.

Troubleshooting your FTP connection

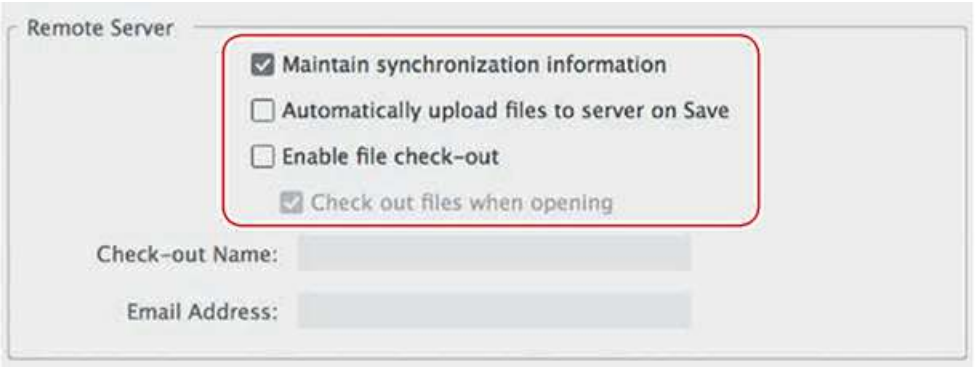
Connecting to your remote site can be frustrating the first time you attempt it. You can experience numerous pitfalls, many of which are out of your control. Here are a few steps to take if you have issues connecting:

- If you can't connect to your FTP server, double-check your username and passwordand re-enter them carefully. Remember that usernames may be case sensitive on some servers, while passwords frequently are. (This is the most common error.)
- Select Use Passive FTP and test the connection again.
- If you still can't connect to your FTP server, deselect the Use FTP Performance Optimization option and click Test again.
- If none of these steps enables you to connect to your remote site, check with your IS/IT manager or your remote site administrator or web-hosting service.

. Click the Advanced tab.

Select from the following options for working with your remote site:

- **Maintain Synchronization Information**—Automatically notes the files that have been changed on the local and remote sites so that they can be easily synchronized. This feature helps you keep track of your changes and can be helpful if you change multiple pages before you upload. You may want to use cloaking with this feature. You'll learn about cloaking in an upcoming exercise. This feature is usually selected by default.



- **Automatically Upload Files To Server On Save**—Transfers files from the local to the remote site when they are saved. This option can become annoying if you save often and aren't yet ready for a page to go public.
- **Enable File Check-Out**—Starts the check-in/check-out system for collaborative website building in a workgroup environment. If you choose this option, you'll need to enter a check-out name and, optionally, an email address. If you're working by yourself, you do not need to select this option.

It is acceptable to leave any or all these options unselected, but for the purposes of this lesson, select the Maintain Synchronization Information option, if necessary.

. Click Save to finalize the settings in the open dialogs.

The server setup dialog closes, revealing the Servers category in the Site Setup dialog. Your newly defined server is displayed in the window.

. The Remote radio button should be selected by default once the server is defined. If you have more than one server defined, click the Remote radio button for the GreenStart Server.

. Click Save to finish setting up your new server.

A dialog may appear, informing you that the cache will be re-created because you changed the site settings.

. If necessary, click OK to build the cache.

When Dreamweaver finishes updating the cache, click Done to close the Manage Sites dialog.

You have established a connection to your remote server. If you don't currently have a remote server, you can substitute a local testing server instead as your remote server.

Establishing a remote site on a local or network web server (optional)

If your company or organization uses a staging server as a "middleman" between web designers and the live website, it's likely you'll need to connect to your remote site through a local or network web server. Local/network servers are often used as testing servers to check dynamic functions before pages are uploaded to the Internet.

> **Warning**
>
> To complete the following exercise, you must have already installed and configured a local or network web server as described in the sidebar "Installing a testing server."

- Launch Adobe Dreamweaver CC (2019 release) or later.
- Choose Site > Manage Sites.
- In the Manage Sites dialog, make sure that lesson11 is selected.

  Click the Edit icon.

> **Note**
>
> You must install a local testing server on your computer or network before performing step 5.

- In the Site Setup for lesson11 dialog, select the Servers category.
- Click the Add New Server icon.

  In the Server Name field, enter **GreenStart Local**.

Installing a testing server

When you produce sites with dynamic content, you need to test the functionality before the pages go live on the Internet. A testing server can fit that need nicely. Depending on the applications you need to test, the testing server can simply be a subfolder on your actual web server, or you can use a local web server such as Apache or Internet Information Services (IIS) from Microsoft.

For detailed information about installing and configuring a local web server, check out the following links:

- Apache/ColdFusion—http://tinyurl.com/setup-coldfusion
- Apache/PHP—http://tinyurl.com/setup-apachephp
- IIS/ASP—http://tinyurl.com/setup-asp

Once you set up the local web server, you can use it to upload the completed files and test your remote site. In most cases, your local web server will not be accessible from the Internet or be able to host the actual website for the public.

. From the Connect Using pop-up menu, choose Local/Network.

. In the Server Folder field, click the Browse icon .

Select the local web server's HTML folder, such as C:\wamp\www\lesson11.

. In the Web URL field, enter the appropriate URL for your local web server. If you are using WAMP or MAMP local servers, your web URL will be something like http://localhost:8888/lesson11 or http://localhost/lesson11.

You must enter the correct URL or Dreamweaver's FTP and testing features may not function properly.

Note

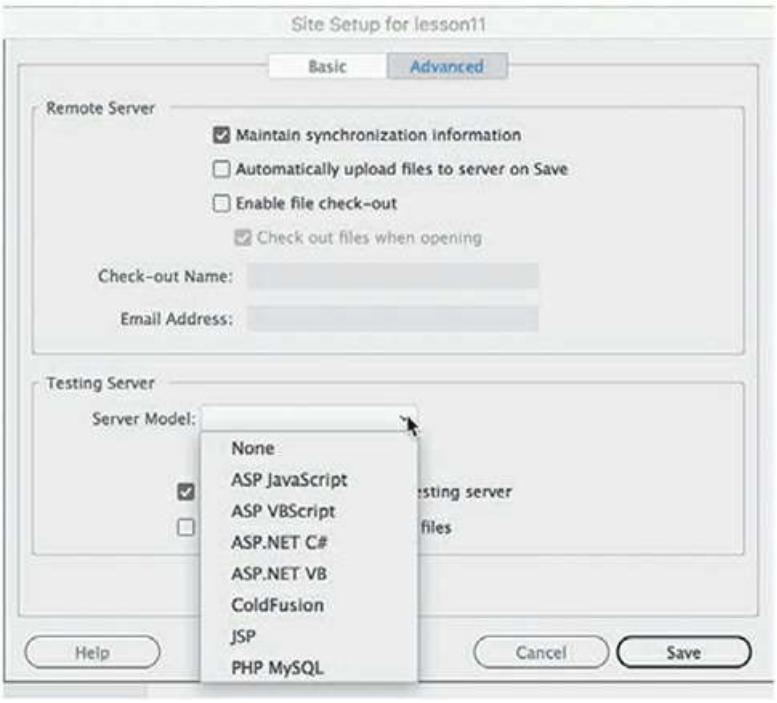
The paths you enter here are contingent on how you installed your local web

server and may not be the same as the ones displayed.

- Click the Advanced tab, and as with the actual web server, select the appropriate options for working with your remote site: Maintain Synchronization Information, Automatically Upload Files To Server On Save, and/or Enable File Check-Out.

  Although leaving these three options unselected is acceptable, for the purposes of this lesson select the Maintain Synchronization Information option if necessary.

- If you'd like to use the local web server as the testing server too, select the server model in the Advanced section of the dialog. If you are creating a dynamic site using a specific programming language, like ASP, ColdFusion, or PHP, select the matching Server Model from the drop-down menu so you'll be able to test the pages of your site properly.



Note

In the latest version of Dreamweaver, you cannot select one server to be both the remote and testing server.

- Click Save to complete the remote server setup.
- In the Site Setup dialog for lesson11, select Remote. Click Save.
- In the Manage Sites dialog, click Done.

  If necessary, click OK to rebuild the cache.

Only one remote and one testing server can be active at one time, but you may have multiple servers defined. Before you upload files for the remote site, you may need to cloak certain folders and files in the local site.

Cloaking folders and files

Not all the files in your site root folder may need to be transferred to the remote server. For example, there's no point in filling the remote site with files that won't be accessed or that will remain inaccessible to website users. Minimizing files stored on the remote server may also pay financial dividends, since many hosting services base part of their fee on how much disk space your site occupies. If you selected Maintain Synchronization Information for a remote site using FTP or a network server, you may want to cloak some of your local materials to prevent them from being uploaded. *Cloaking* is a Dreamweaver feature that allows you to designate certain folders and files that will not be uploaded to or synchronized with the remote site.

Tip

If disk space is not a concern, you might consider uploading the template files to the server as a means of creating a backup.

Folders you don't want to upload include the Templates and resource folders. Some other non-web-compatible file types used to create your site, such as Photoshop (.psd), Flash (.fla), or Microsoft Word (.doc or .docx), also don't need to be on the remote server. Although cloaked files will not upload or synchronize automatically, you may still upload them manually, if desired. Some people like to upload these items to keep a backup copy of them off-site.

The cloaking process begins in the Site Setup dialog.

- Choose Site > Manage Sites.
- Select lesson11 in the site list, and click the Edit icon.
- Expand the Advanced Settings category and select the Cloaking category.
- Select the Enable Cloaking and Cloak Files Ending With options, if necessary.

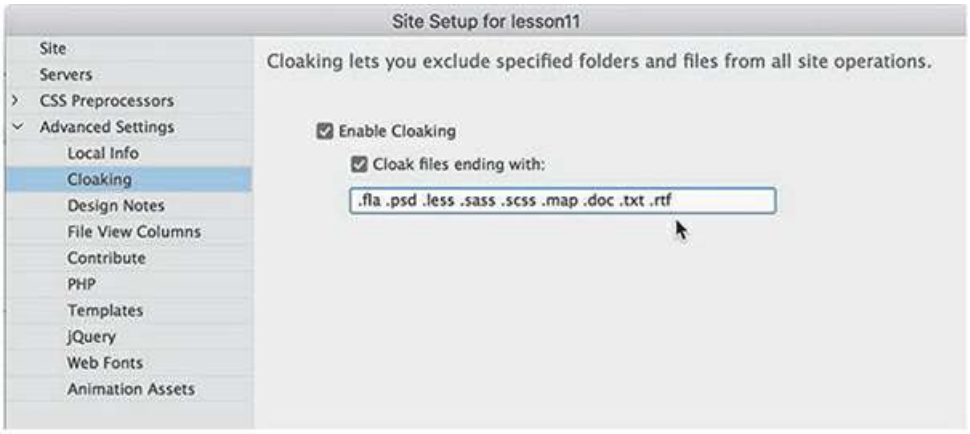
  The field below the checkboxes displays several extensions and may differ from those pictured.

Note

Add any extension you may be using as your own source files.

- Insert the cursor after the last extension, and insert a space, if necessary.

Type **.doc .txt .rtf** in the field.



Be sure to insert a space between each extension. By specifying the extensions of file types that don't contain desired web content, you prevent Dreamweaver from automatically uploading and synchronizing these file types no matter where they appear in the site.

. Click Save. If Dreamweaver prompts you to update the cache, click OK.

Then, click Done to close the Manage Sites dialog.

Although you have cloaked several file types automatically, you can also cloak specific files or folders manually from the Files panel.

Note

It should be mentioned that any resource uploaded to the server can be accessed by search engines and accessed by the public. Sensitive material or content should not be uploaded if you are concerned that it may be seen by the public.

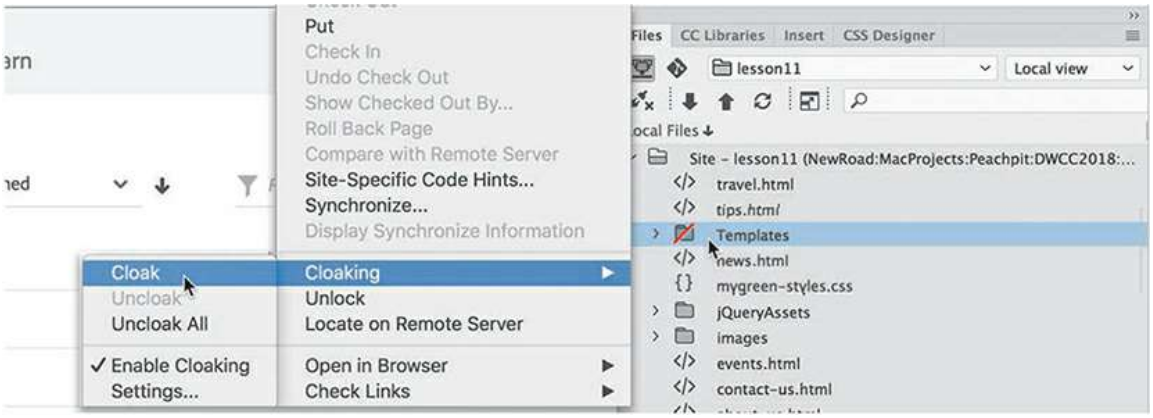
. Open the Files panel.

In the site list, you will see a list of the files and folders that make up the site. Some of the folders are used to store the raw materials for building content. There's no need to upload these items to the web. The Templates folder is not needed on the remote site, because your webpages do not reference these assets in any way. If you work in a team environment, it may be handy to upload and synchronize these folders so that each team member has up-to-date versions of each on their own computers. For this exercise, let's assume you work alone.

. Right-click the Templates folder.

From the context menu, choose Cloaking > Cloak.

. In the warning dialog that appears, click OK.



The selected folder shows a red slash, indicating that it is now cloaked.

Using the Site Setup dialog and the Cloaking context menu, you cloaked file types, folders, and files. The synchronization process will ignore cloaked items and will not automatically upload or download them.

Wrapping things up

Over the last 10 lessons, you have built an entire website, beginning with a starter layout and then adding text, images, movies, and interactive content, but a few loose strings remain for you to tie up. Before you publish your site, you'll need to create one important webpage and make some crucial updates to your site navigation.

The file you need to create is one that is essential to every site: a home page. The home page is usually the first page most users see on your site. It is the page that loads automatically when a user enters your site's domain name into the browser window. Since the page loads automatically, there are a few restrictions on the name and extension you can use.

Basically, the name and extension depend on the hosting server and the type of applications running on the home page, if any. Today, the majority of home pages will simply be named *index*. But *default*, *start*, and *iisstart* are also used.

Extensions identify the specific types of programming languages used within a page. A normal HTML home page will use an extension of .htm or .html. Extensions like .asp, .cfm, and .php, among others, are required if the home page contains any dynamic applications specific to that server model. You may still use one of these extensions—if they are compatible with your server model—even if the page contains no dynamic applications or content. But be careful—in some instances, using the wrong extension may prevent the page from loading altogether. Check with your server administrator or IT manager for the proper extension.

The specific home page name or names honored by the server are normally configured by the server administrator and can be changed, if desired. Most servers are configured to honor several names and a variety of extensions. Check with your IS/IT manager or web-server support team to ascertain the recommended name and extension for your home page.

. Create a new page from the site template.

Save the file as **index.html** or use a filename and extension compatible with your server model.

- Open **home.html** from the lesson11 site root folder in Design view.

  The file contains content for the sidebars and main content areas of the new home page.

- Insert the cursor in the heading *Go Green with Meridien GreenStart!*

  Select the `article` tag selector and copy the content.

Note

Moving content from one file to another is easier in Design view or Code view. Remember that you must use the same view in both the source and target documents.

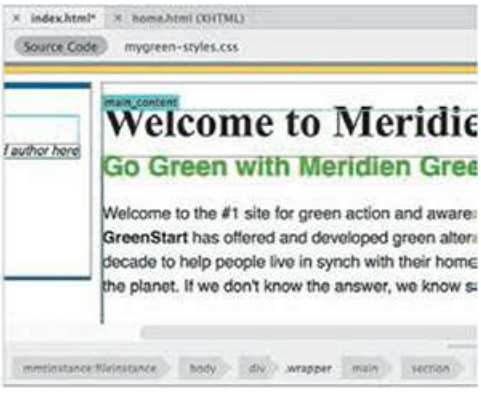
- In **index.html**, select Design view.

  Double-click to edit the heading *Add main heading here*.

  Type **Welcome to Meridien GreenStart**.

- Insert the cursor in the heading *Add article heading here*.

  Select the `article` tag selector. Paste to replace the selection.



> **Note**
>
> Pasting to replace an element with another works only in Design view and Code view.

The main content section in the new layout is replaced by the copied text and code.

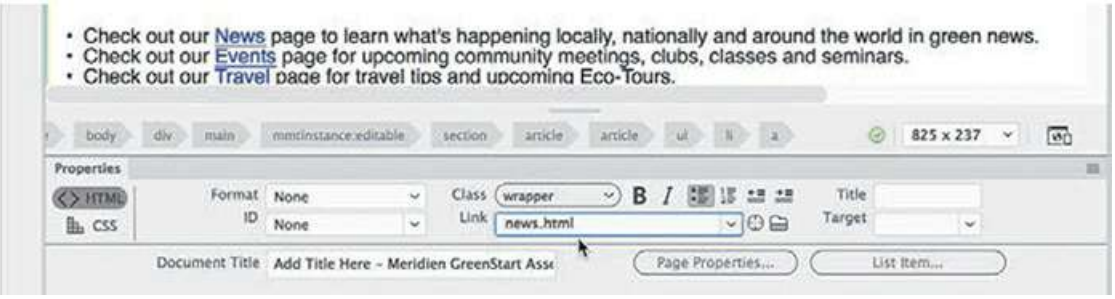
. In **index.html**, replace the quotation placeholder with the `<blockquote>` element in **home.html**.

. Replace the Sidebar 2 placeholder with the `<figure>` element in **home.html**.

Note the hyperlink placeholders in the `main_content` region.

. Insert the cursor in the *News* link in the `main_content` region.

In the Property inspector, browse and connect the link to **news.html**.



. Repeat step 7 with each link.

Connect the links to the appropriate pages in your site root folder.

. Save and close all files.

The home page is nearly complete. For example, the title and meta description placeholders still need to be updated. Feel free to update them with appropriate text.

In the meantime, let's assume you want to upload the site at its current state of completion. This happens in the course of any site development. Pages are added, updated, and deleted over time; missing pages will be completed and then uploaded at a later date. Before you can upload the site to a live server, you should always check for and update any out-of-date links and remove dead ones.

Prelaunch checklist

Take this opportunity to review all your site pages before publishing them to see whether they are ready for prime time. In an actual workflow, you should perform the following actions, which you learned in previous lessons, before uploading a single page:

- Spell-check (Lesson 7, "Working with Text, Lists, and Tables")

- Sitewide link check (Lesson 9, “Working with Navigation”)

Fix any problems you find, and then proceed to the next exercise.

Putting your site online (optional)

For the most part, the local site and the remote site are mirror images, containing the same HTML files, images, and assets in identical folder structures. When you transfer a webpage from your local site to your remote site, you are publishing, or *putting*, that page. If you *put* a file stored in a folder on your local site, Dreamweaver transfers the file to the equivalent folder on the remote site. It will even automatically create the remote folder or folders if they do not already exist. The same is true when you download files.

Warning

Dreamweaver does a good job trying to identify all the dependent files in a particular workflow. But, in some cases, it may miss files crucial to a dynamic or extended process. It is imperative that you do your homework to identify these files and make sure they are uploaded.

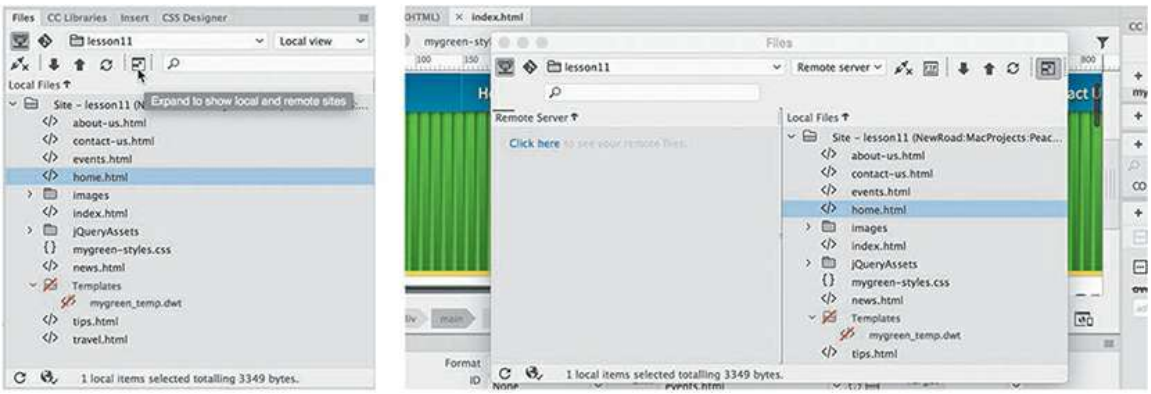
Using Dreamweaver, you can publish anything—from one file to a complete site—in a single operation. When you publish a webpage, by default Dreamweaver asks if you would also like to put the dependent files too. Dependent files are the images, CSS, HTML5 movies, JavaScript files, server-side includes (SSI), and other files necessary to complete the page.

Note

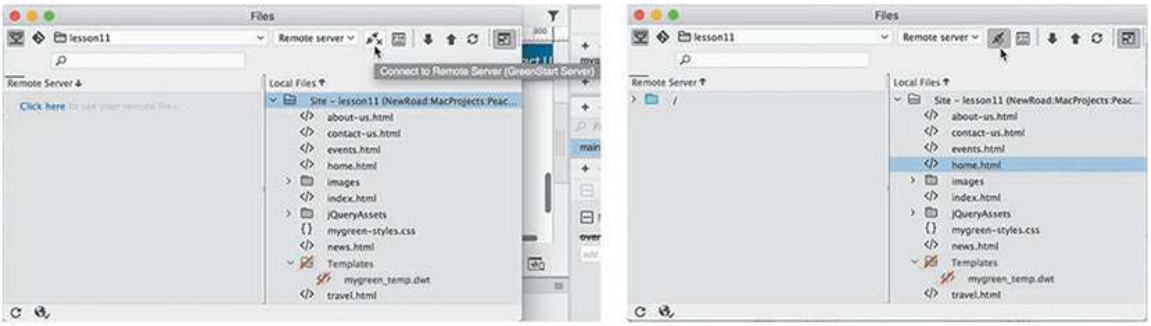
This exercise is optional, since it requires that you set up a remote server beforehand.

You can upload one file at a time or the entire site at once. In this exercise, you will upload one webpage and its dependent files.

. Open the Files panel and click the Expand icon  , if necessary.

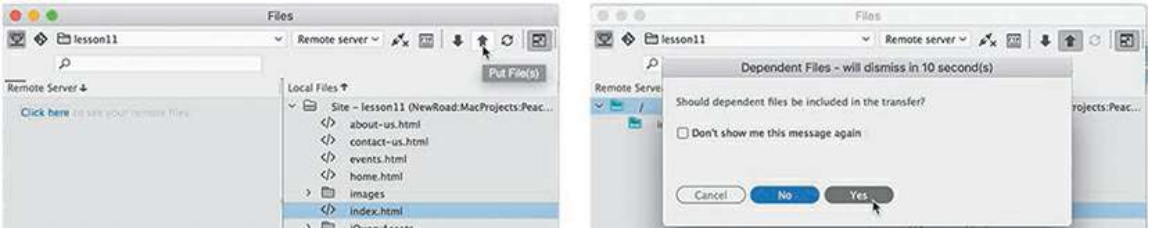


- Click the Connect To Remote Server icon to connect to the remote site.



If your remote site is properly configured, the Files panel will connect to the site and display its contents on the left half of the panel. When you first upload files, the remote site may be empty or mostly empty. If you are connecting to your Internet host, specific files and folders created by the hosting company may appear. Do not delete these items unless you check to see whether they are essential to the operation of the server or your own applications.

- In the local file list, select **index.html**.



- In the Files panel toolbar, click the Put icon .
- By default, Dreamweaver will prompt you to upload dependent files. If a dependent file already exists on the server and your changes did not affect it, you can click No. Otherwise, for new files or files that have had any changes, click Yes. There is an option within Preferences where you can disable this prompt, if desired.

Note

Dependent files include but are not limited to images, style sheets, and JavaScript used within a specific page and are essential to the proper display and function of

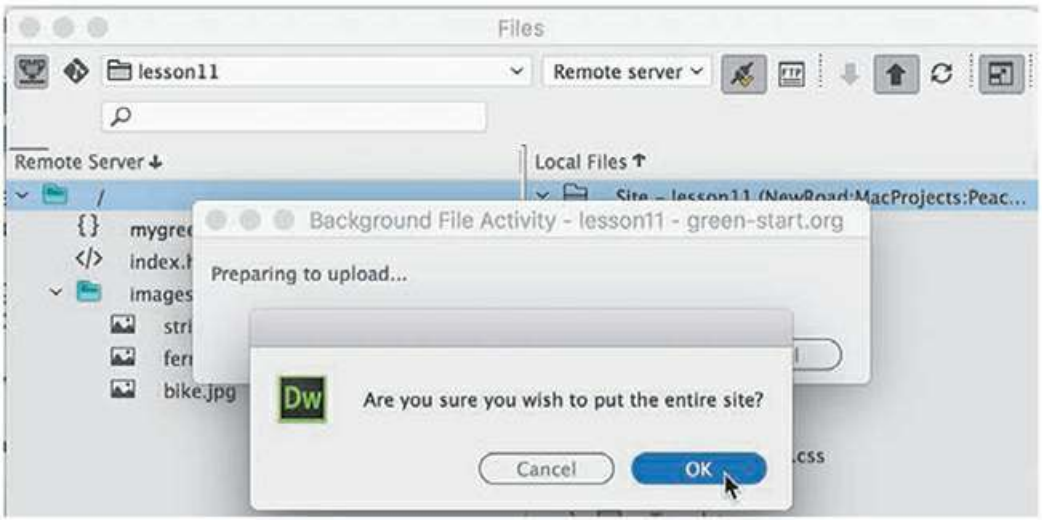
the page.

. Click Yes.

Dreamweaver uploads **index.html** and all images, CSS, JavaScript, server-side includes, and other dependent files needed to properly render the selected HTML file. Although you chose only one file, you can see that five files and one folder were uploaded.

The Files panel enables you to upload multiple files as well as the entire site at once.

. Select the site root folder for the local site and then click the Put icon ⬆ in the Files panel.



Dialogs appear, asking you to confirm that you want to upload the entire site.

. Click Yes or OK as appropriate.

Note

A file that is uploaded or downloaded will automatically overwrite any version of the file at the destination.

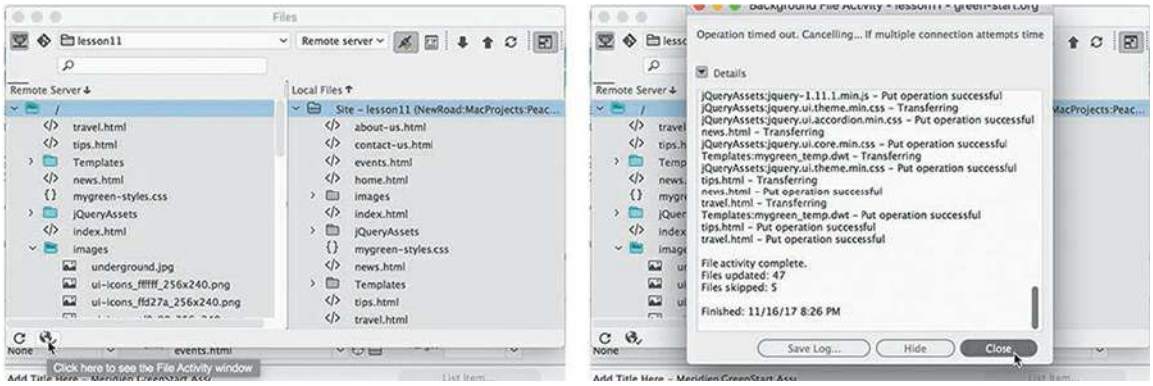
Tip

If you are using a third-party web-hosting service, be aware that they often create placeholder pages on your domain. If your home page does not automatically appear when you access your site, check to make sure that there is no conflict with the web host's placeholder pages.

Dreamweaver begins to upload the site. It will re-create your local site structure on the remote

server. Dreamweaver uploads pages in the background so that you can continue to work in the meantime. If you want to see the progress of the upload.

- Click the File Activity icon in the lower-left corner of the File panel.



When you click the File Activity icon, you will see a list featuring the filenames and the status of the selected operation. You can even save the report to a text file, if desired, by clicking the Save Log button in the Background File Activity dialog.

Note that neither the cloaked lesson folders nor the files stored within them were uploaded. Dreamweaver will automatically ignore all cloaked items when putting individual folders or an entire site. If desired, you can manually select and upload individually cloaked items.

- Right-click the Templates folder and choose Put from the context menu.

Dreamweaver prompts you to upload dependent files for the Templates folder.

- Click Yes to upload dependent files.

The Templates folder is uploaded to the remote server. The log report shows that Dreamweaver checked for dependent files but did not upload the files that had not changed.

Note that the remote Templates folder displays a red slash, indicating that it, too, is cloaked. At times, you will want to cloak local and remote files and folders to prevent these items from being replaced or accidentally overwritten. A cloaked file will not be uploaded or downloaded automatically. But you can manually select any specific files and perform the same action.

The opposite of the Put command is Get, which downloads any selected file or folder to the local site. You can get any file from the remote site by selecting it in the Remote pane and clicking the Get icon. Alternatively, you can drag the file from the Remote pane to the Local pane.

- If you were able to successfully upload your site, use a browser to connect to the remote site on your network server or the Internet. Type the appropriate address in the URL field—depending on whether you are connecting to the local web server or to the actual Internet site—such as: http://localhost/*domain_name* or http://www.*domain_name*.com.

Note

When accessing Put and Get, it doesn't matter whether you use the Local or

Remote pane of the Files panel. Put always uploads to Remote; Get always downloads to Local.

The GreenStart site appears in the browser.

. Click to test the hyperlinks to view each of the completed pages for the site.

Once the site is uploaded, keeping it up to date is an easy task. As files change, you can upload them one at a time or synchronize the whole site with the remote server.

Synchronization is especially important in workgroup environments where files are changed and uploaded by several individuals. You can easily download or upload files that are older, overwriting files that are newer in the process. Synchronization can ensure that you are working with only the latest versions of each file.

Synchronizing local and remote sites

Synchronization in Dreamweaver keeps the files on your server and your local computer up to date. It's an essential tool when you work from multiple locations or with one or more co-workers. Used properly, it can prevent you from accidentally uploading or working on out-of-date files.

At the moment, your local and remote sites are identical. To better illustrate the capabilities of synchronization, let's make a change to one of the site pages.

. Open **about-us.html** in Live view.

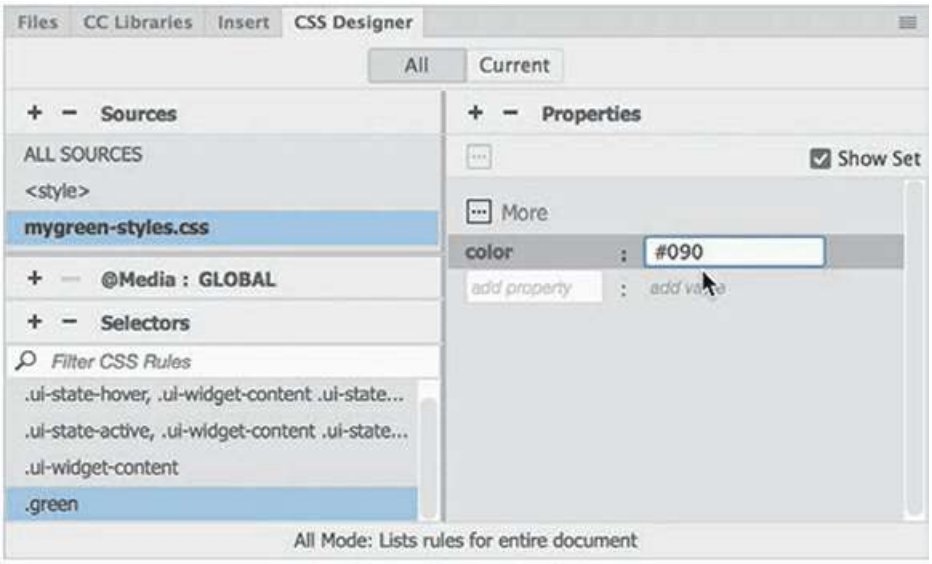
- Collapse the Files panel by clicking the Collapse icon , if necessary.

  Clicking the collapse button re-docks the panel on the right side of the program, if necessary.
- In the CSS Designer, click the All button. Select **mygreen-styles.css**.

  Create a new selector:

  `.green`
- Add the following property to the new rule:

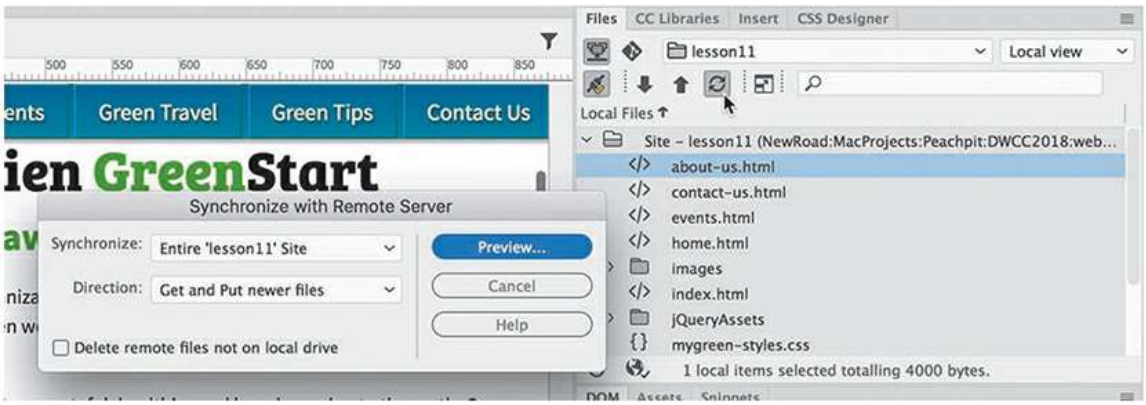
  `color: #090`



- In the main heading, drag the cursor across the word *Green* in the heading *About Meridien GreenStart.*
- Apply the `green` class to this text.

- Apply the `green` class to each occurrence of the word *green* anywhere on the page where the text is not already green.
- Save all files and close the page.
- Open and expand the Files panel.

  In the Document toolbar, click the Synchronize icon .



The Synchronize With Remote Server dialog appears.

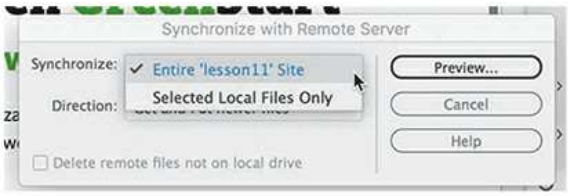
Note

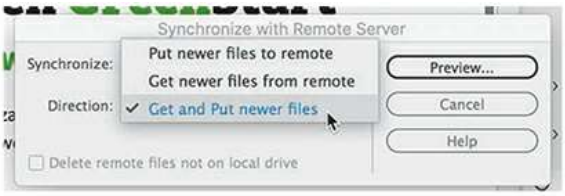
The Synchronize icon looks similar to the Refresh icon but is located on the upper-right side of the Files panel.

. From the Synchronize pop-up menu, choose the option Entire ‘lesson11’ Site. From the Direction menu, choose the Get And Put Newer Files option.

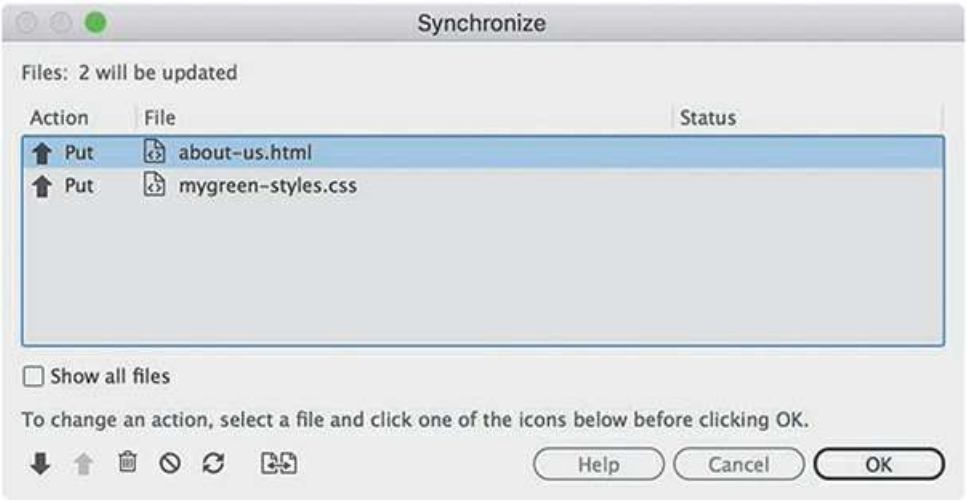
Note

Synchronize does not compare cloaked files or folders.





. Click Preview.



The Synchronize dialog appears, reporting what files have changed and whether you need to get or put them. Since you just uploaded the entire site, only the files you modified—**about-us.html** and **mygreen-styles.css**—should appear in the list, which indicates that Dreamweaver wants to put them to the remote site. If you see any other files listed, select them and click the Synchronize icon to tell Dreamweaver that these files are okay as is.

Synchronization options

During synchronization, you can choose to accept the suggested action or override it by selecting one of the other options in the dialog. Options can be applied to one or more files at a time.

Get—Downloads the selected file(s) from the remote site

Put—Uploads the selected file(s) to the remote site

Delete—Marks the selected file(s) for deletion

Ignore—Ignores the selected file(s) during synchronization

Synchronized—Identifies the selected file(s) as already synchronized

Compare—Uses a third-party utility to compare the local and remote versions of a selected file

. Click OK to upload the two files.

If other people can access and update files on your site, remember to run synchronization *before* you work on any files to be certain you are working on the most current versions of each file in your site. Another technique is to set up the check-out/check-in functionality in the advanced options of the server's setup dialog.

In this lesson, you set up your site to connect to a remote server and uploaded files to that remote

site. You also cloaked files and folders and then synchronized the local and remote sites.

Congratulations! You've designed, developed, and built an entire website and uploaded it to your remote server. By completing the exercises in this book to this point, you have gained experience in all aspects of the design and development of a standard website compatible with desktop computers. In the following lessons, you will learn some productivity tricks with HTML code and how to adapt your static, fixed-width site to work with cellphones, tablets, and other mobile devices.

Review questions

1. What is a remote site?
2. Name two types of file transfer protocols supported in Dreamweaver.
3. How can you configure Dreamweaver so that it does not synchronize certain files in your local site with the remote site?
4. True or false: You have to manually publish every file and associated image, JavaScript file, and server-side include that is linked to pages in your site.
5. What service does synchronization perform?

Review answers

1. A remote site is typically the live version of the local site stored on a web server connected to the Internet.
2. FTP (File Transfer Protocol) and local/network are the two most commonly used file transfer methods. Other file transfer methods supported in Dreamweaver include Secure FTP, WebDav, and RDS.
3. Cloaking the files or folders prevents them from synchronizing.
4. False. Dreamweaver can automatically transfer dependent files, if desired, including embedded or referenced images, CSS style sheets, and other linked content, although some files may be missed.
5. Synchronization automatically scans local and remote sites, comparing files on both to identify the most current version of each. It creates a report window to suggest which files to get or put to bring both sites up to date, and then it will perform the update.

12 Working with Code

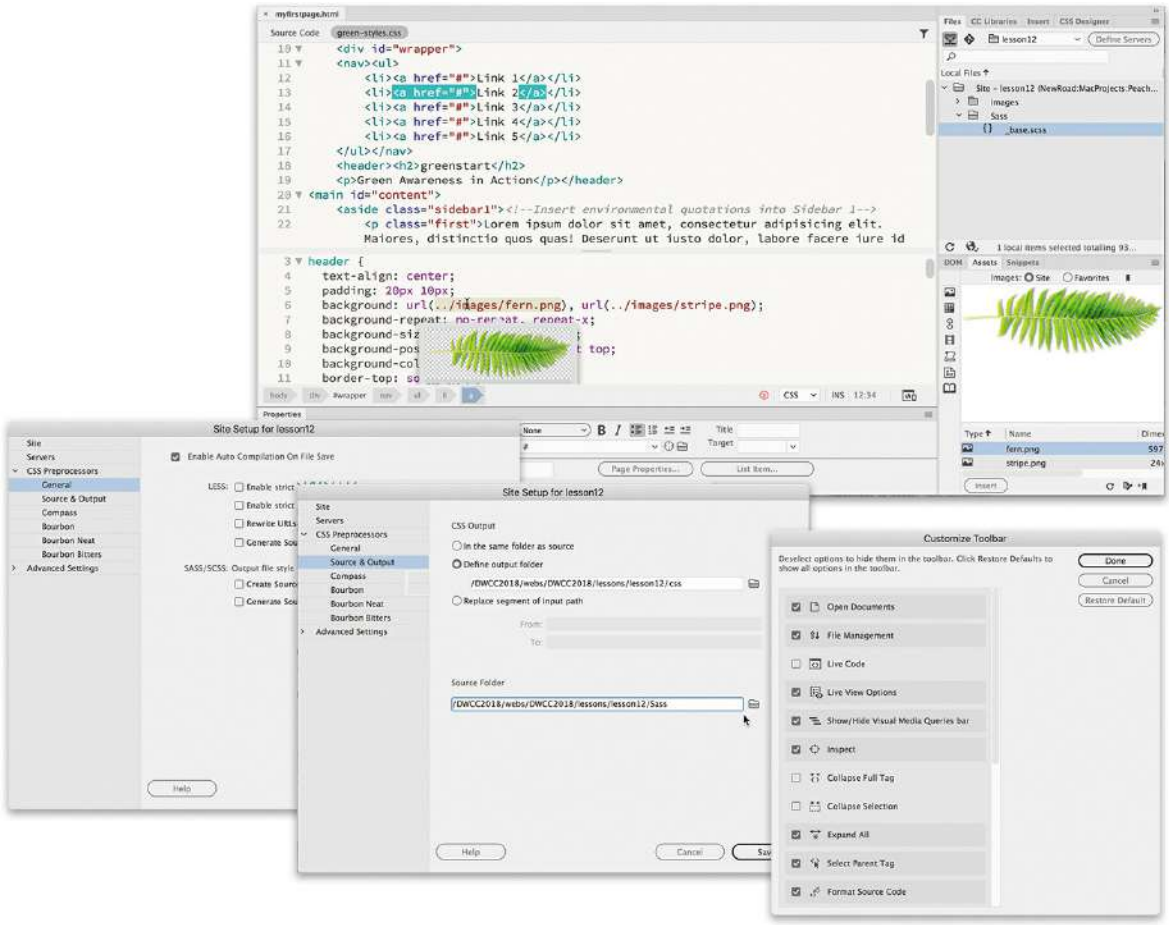
Lesson overview

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 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. If you have not already done so, please log in to your account on peachpit.com to download the project files for this lesson as described in the "Getting Started" section at the beginning of this book and follow the instructions under "Accessing the Lesson Files and Web Edition." Define a site based on the lesson12 folder.

Your Account page is also where you'll find any updates to the lessons or to the lesson files. Look on the Lesson & Update Files tab to access the most current content.



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 also a necessity.

Note

If you have not already downloaded the project files for this lesson to your computer from your Account page, make sure to do so now. See "Getting Started" at the beginning of the book.

Although Dreamweaver has always made it as easy to work with a page in Code view as it is in

Design view or Live view, some developers believe that the code-editing tools took a back seat to the visual design interface. Although in the past this was partially true, Dreamweaver CC (2019 release) is fully invested in the vastly improved tools and workflows for coders and developers that were brought to the program in the previous version. In fact, Dreamweaver CC can now unify your entire web development team as never before 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.

Note

Some tools and options are available only when Code view is active.

Writing code manually

If you completed the previous 11 lessons, you have had 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. The first step to experiencing Dreamweaver's code-writing and editing tools is to create a new file.

- Define a site based on the lesson12 folder downloaded from your account page, as described in the "Getting Started" section at the beginning of the book.
- Select Developer from the Workspace menu.

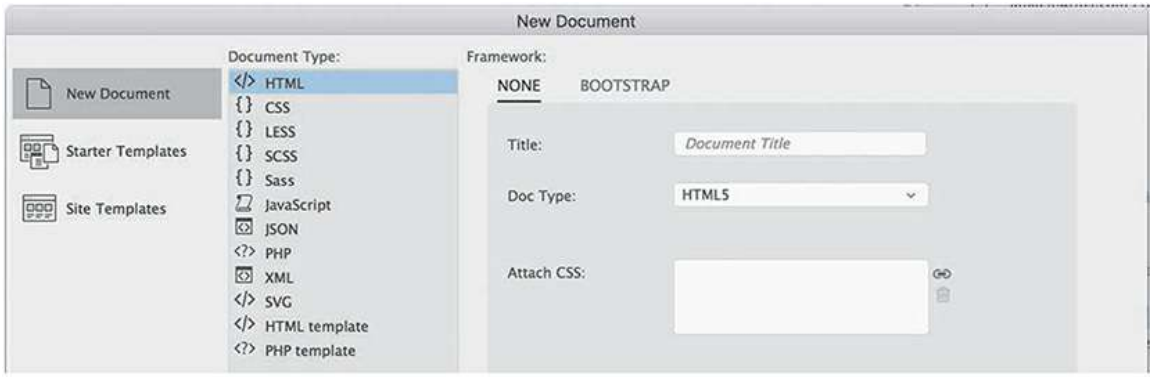
All the code-editing tools work identically in either workspace, but the Developer workspace focuses on the Code view window and provides a better experience for the following exercises.

- Choose File > New.

The New Document dialog appears.

- Choose New Document > HTML > None.

Click Create.



Dreamweaver creates the basic structure of a webpage automatically. The cursor will probably appear at the beginning of the code.

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.

Note

In all screen shots, we use the Solarized Light color theme, which can be selected in Preferences. See the "Getting Started" section at the beginning of the book for more details.

- Choose File > Save.
- Name the file **myfirstpage.html** and save it in the lesson12 folder.
- Insert the cursor after the `<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.

- 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.

- Press the Down Arrow.

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

- Continue pressing the Down Arrow until the tag `div` is highlighted.

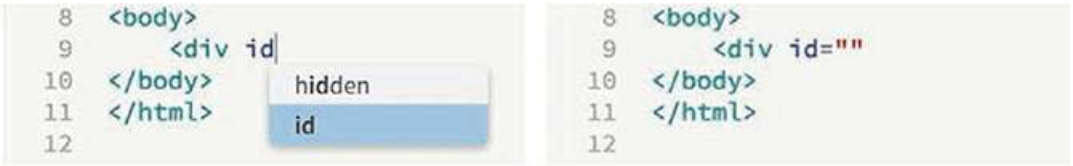
Press Enter/Return.

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.

- 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.

- Type **id** and press Enter/Return.



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

- Type **wrapper** and press the Right Arrow key once.

The cursor moves outside the closing quotation mark.

Note

Depending on the settings in your program, tags may close automatically, which means you can skip step 15. This behavior can be turned off or adjusted in the Code Hints section of Preferences.

. Type **></**

When you type the backslash (/), 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.

. Choose File > Save.

Writing code automatically

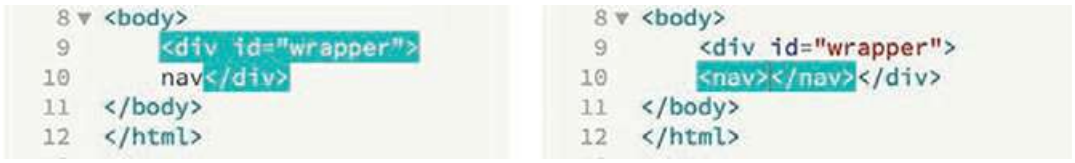
Emmet is a web-developer toolkit that was added to a previous version of Dreamweaver and enables you to supercharge your code-writing tasks. When you enter shorthand characters and operators, Emmet enables you to create whole blocks of code with just a few keystrokes. To experience the power of Emmet, try this exercise.

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

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

Emmet is enabled by default and works whenever you are typing in Code view. In the original site mockup, the navigation menu appears at the top of the page. HTML5 uses the `<nav>` element as the foundation of site navigation.

. 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 more than one element 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 (>).

. 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.

. 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.

. 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.

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

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

. 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.

. 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 (`$`).

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

Note

The cursor must be outside the brace before pressing 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 thing missing are targets for the `href` attributes. You could add them now using another Emmet phrase, but let's save this change for the next exercise.

- 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.

Note

Adding the new line makes the code easier to read and edit, but it has no effect on how it operates.

- 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. If you model this `header` on the one in the site completed in Lesson 5, "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.

- Type **`h2{greenstart}+p{Green Awareness in Action}`** and press Tab.

Click here to view code image

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.

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

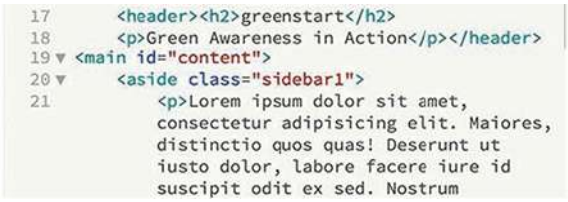
Emmet enables you to quickly build complex multifaceted parent—child structures like the navigation menu and the header, but it doesn't stop there. As you string together several elements with placeholder text, you can even add `id` and `class` attributes. To insert an id, start the name with the hash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

Note

The entire phrase may wrap to more than one line in Code view, but make sure there are no spaces within the markup.

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

Click here to view code image



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.

- Insert the cursor after the closing `</main>` tag.

  Create a new line.

  Type **`footer{Copyright 2019 Meridien GreenStart Association. All rights reserved.}`** and press Tab.

  Click here to view code image

```
29  </main>
30      footer{Copyright 2018 Meridien GreenStart
        Association. All rights reserved.}</div>
31  </body>
32  </html>
33
```

```
29  </main>
30      <footer>Copyright 2018 Meridien GreenStart
        Association. All rights reserved.</footer>
        </div>
31  </body>
32  </html>
```

- 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 any time 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 http://emmet.io to learn more about Emmet. Check out http://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? Another addition to Dreamweaver CC (2019 release) is 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.

- If necessary, open **myfirstpage.html** as it appears at the end of the previous exercise.

  The file contains a complete webpage with `header`, `nav`, `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 previous lessons, the hash mark (#) was used as placeholder content until the final link destinations could be added.

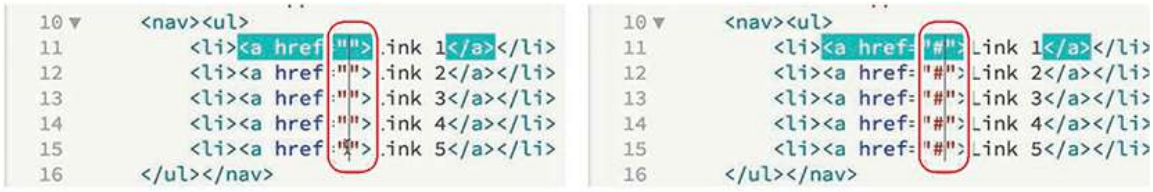
- Insert the cursor between the quotation marks in the `href=""` attribute in Link 1.

  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.

- 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.

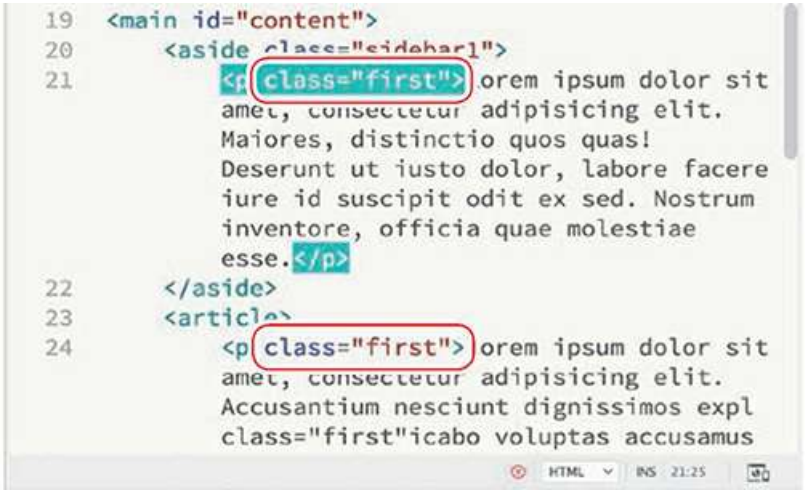
- Type **#**



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.

- 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.
- Press the spacebar to insert a space, and type **`class="first"`**



The class appears simultaneously in all three `<p>` tags.

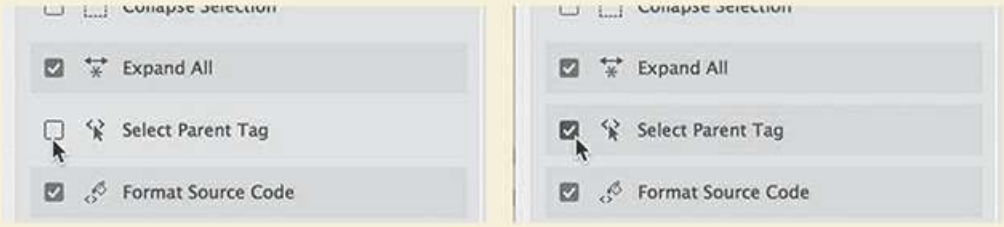
- Save the file.

Multicursor support can save tons of time in repetitive code-editing tasks.

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 new 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.



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 comments manually at any time, Dreamweaver has a built-in feature that can speed up the process.

- Open **myfirstpage.html** and switch to Code view, if necessary.
- Insert the cursor after the opening tag

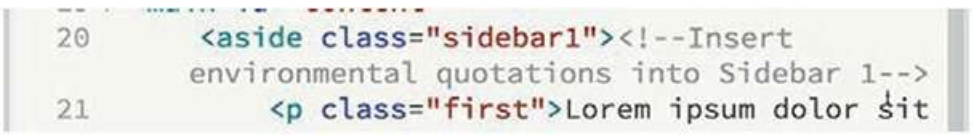
  `<aside class="sidebar1">`.
- Click the Apply Comment icon .

A pop-up menu appears with several comment options. Dreamweaver supports comment markup for various web-compatible languages, including HTML, CSS, JavaScript, and PHP.

- Choose Apply HTML Comment.

  An HTML comment block appears, with the text cursor positioned in the center.
- Type `Insert environmental quotations into Sidebar 1`

  Click here to view code image



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

. Insert the cursor after the opening tag

```
<aside class="sidebar2">.
```

. Type `Sidebar 2 should be used for content related to the Article section`

Click here to view code image

. Select the text created in step 7.

Click the Apply Comment icon .

A pop-up menu opens.

. 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 in the toolbar.

. Save all files.

You've created a complete basic webpage. The next step is to style the page. Dreamweaver CC (2019 release) now supports CSS preprocessors for LESS, Sass, and SCSS. 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 changes to the latest version of Dreamweaver was adding built-in support for LESS, Sass, and SCSS. These industry-standard CSS preprocessors are scripting languages that enable you to extend the capabilities of cascading style sheets with a variety of productivity enhancements and then compile the results 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 use a preprocessor with Dreamweaver as well as what advantages they offer compared to a regular CSS workflow.

Enabling a preprocessor

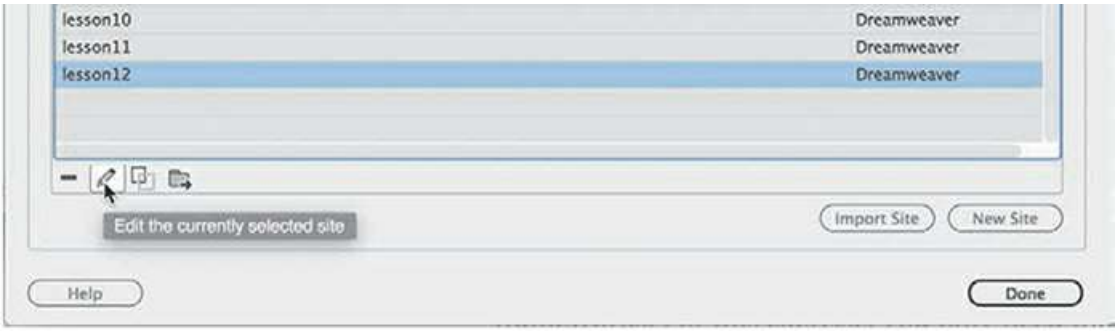
Support for CSS preprocessors is site-specific and must be enabled for each site defined in Dreamweaver, as desired. To enable LESS, Sass, or SCSS, you first define a site and then enable the CSS Preprocessors option within the Site Definition dialog.

- Select Site > Manage Sites.

  The Manage Sites dialog appears.

- Select **lesson12** in the Your Sites window.

  Click the Edit icon at the bottom of the Your Sites window.



  The Site Definition for lesson12 appears.

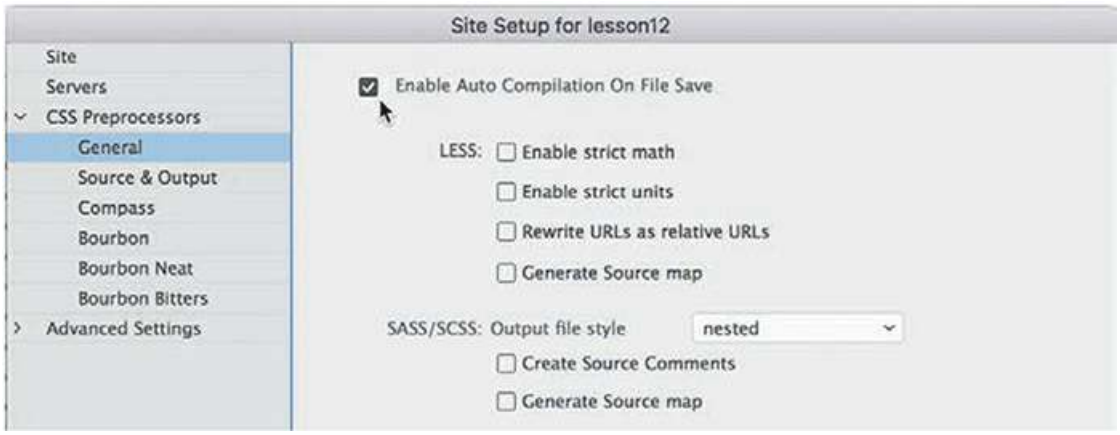
- Select the **CSS Preprocessors** option in the Site Definition dialog.

  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.

- 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.

- 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.

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 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:

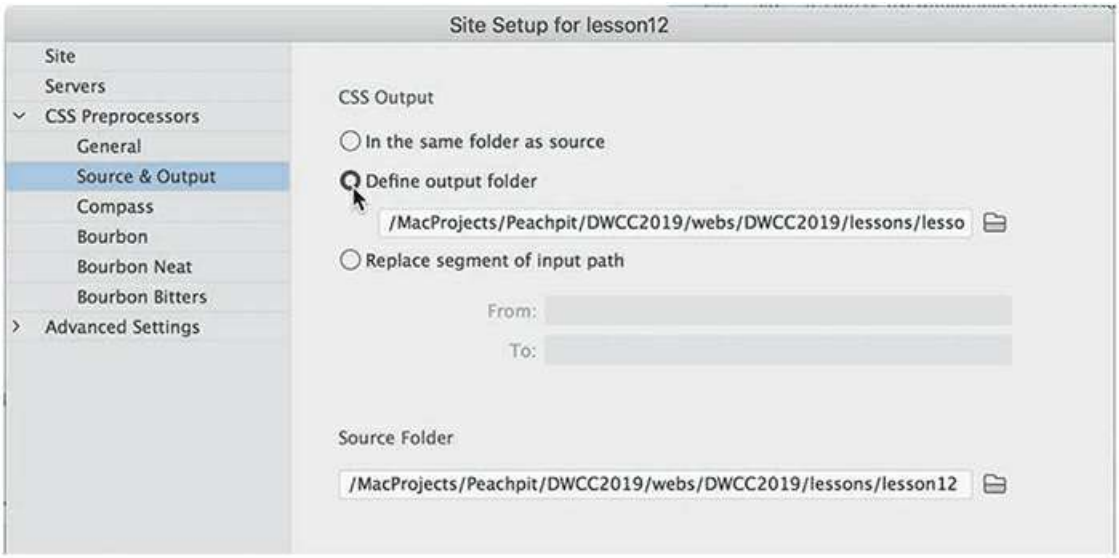
- blog.udemy.com/less-vs-sass/
- css-tricks.com/sass-vs-less/
- zingdesign.com/less-vs-sass-its-time-to-switch-to-sass/
- keycdn.com/blog/sass-vs-les

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

. 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.

. Select the Define Output Folder option.



- Click the Browse for Folder icon .

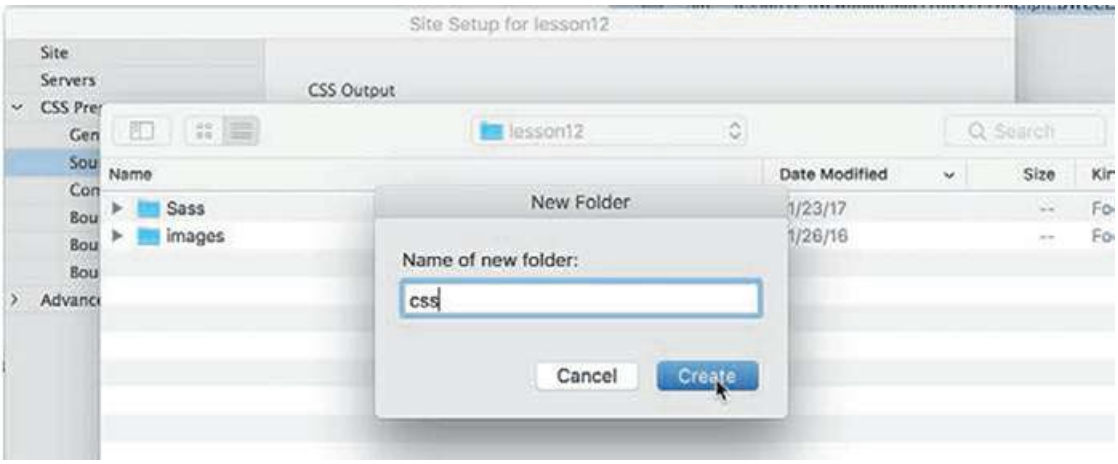
  A file browser dialog appears.

- Navigate to the Site Root folder, if necessary.

  Create a new folder.

- Name the new folder **css**.

  Click Create.



- Select the css folder and click Select Folder/Choose.
- Click the Browse for Folder icon beside the Source Folder field.
- Navigate to the Site Root folder.

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

- 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.

Creating the CSS source file

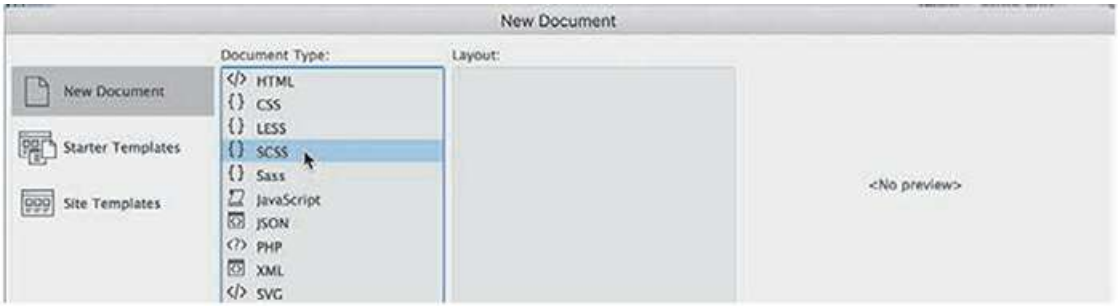
When using a preprocessor workflow, you do not write the CSS code directly. Instead, you write rules and other code in a source file that is then compiled to the output file. For the following exercise, you'll create a Sass source file and learn some of the functions of that language.

- Select Standard from the Workspace menu.
- Choose Window > Files to display the Files panel, if necessary. Select lesson12 from the Site List drop-down menu, if necessary.
- If necessary, open **myfirstpage.html** and switch to Split view.

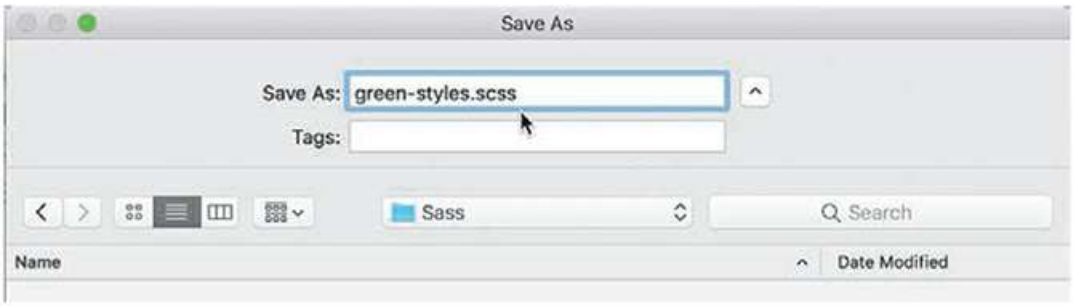
  The webpage is unstyled at the moment.
- Choose File > New.

  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.
- Choose New Document > SCSS.

  Click the Create button.



  A new blank SCSS document appears in the document window. SCSS is a flavor of Sass that uses a syntax similar to regular CSS that many users find easier to learn and work with.
- Save the file as **green-styles.scss** in the Sass folder you targeted as the Source folder in the previous exercise.



There's no 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 are updated automatically.

. Insert the cursor into line 2 of **green-styles.scss**.

Type `$logogreen: #090;` and press Enter/Return.

You've created your first variable. This is the main green color of the site theme. Let's create the rest of the variables.

. Type `$darkgreen: #060;`
`$lightgreen: #0F0;`
`$logoblue: #069;`
`$darkblue: #089;`
`$lightblue: #08A;`
`$font-stack: "Trebuchet MS", Verdana, Arial, Helvetica, sans-serif;` and press Enter/Return to create a new line.

Click here to view code image

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.

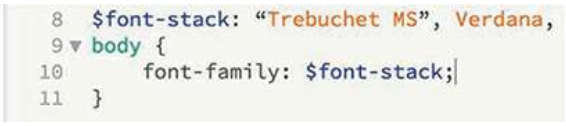
Let's start the style sheet with the base or default styling of the `body` element. SCSS markup in most cases looks just like regular CSS, except in this case you'll use one of your variables to set the font family.

. 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 the closing brace moved to the following line. You can also use Emmet to enter the settings more quickly.

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



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

. Press Enter/Return to create a new line.

Type **c** and press Tab.

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

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

. Press Enter/Return to create a new line.

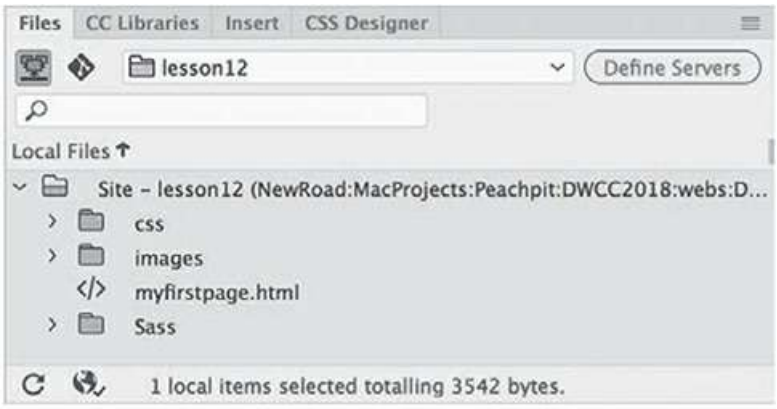
Type **m0** and press Tab.

The shorthand expands to `margin: 0;`. This property completes 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.

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

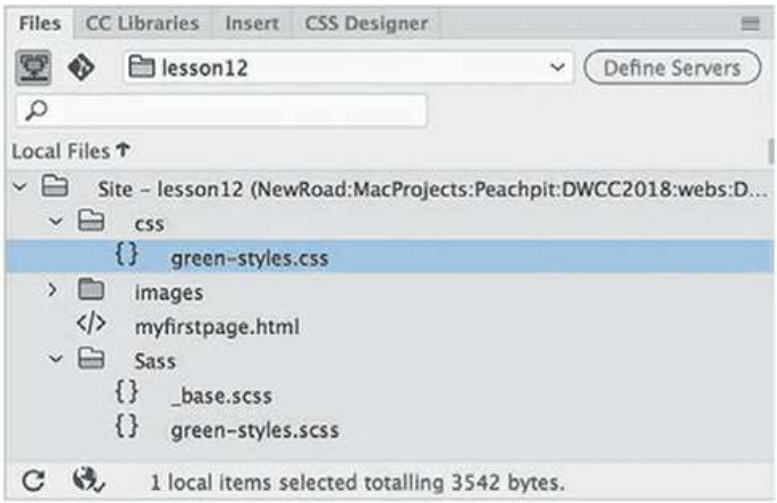


The site consists of one HTML file and three folders: css, images, and Sass.

. Expand the view of the css and Sass folders.

Note

The **green-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.



The Sass folder contains **green-styles.scss** and **_base.scss**. The css folder contains **green-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.

. Select the document tab for **myfirstpage.html**.

Switch to Split view, if necessary.

The page shows only default HTML styling.

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

The hinting menu appears. You'll link the webpage to the generated CSS file.

- Type **`href`** and press Enter/Return.

The complete `href=""` attribute appears, and the hinting menu changes to display the Browse command and a list of pathnames to folders available in the site.

- Press the Down Arrow to select the path `css/` and press Enter/Return.

The hinting menu now displays the path and filename to **green-styles.css**.

- Press the Down Arrow to select

`css/green-styles.css` and press Enter/Return.

The URL to the CSS output file appears in the attribute. The cursor is moved outside the closing quotation mark and is ready for the next entry. For the style sheet reference to be valid, you need to create one more attribute.

. Press the spacebar, and then type **`rel`** and press Enter/Return.

Select `stylesheet` from the hinting menu.

. Move the cursor outside the closing quotation mark. Type **>** to close the link.

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 **green-styles.css** displayed in the Related Files interface.

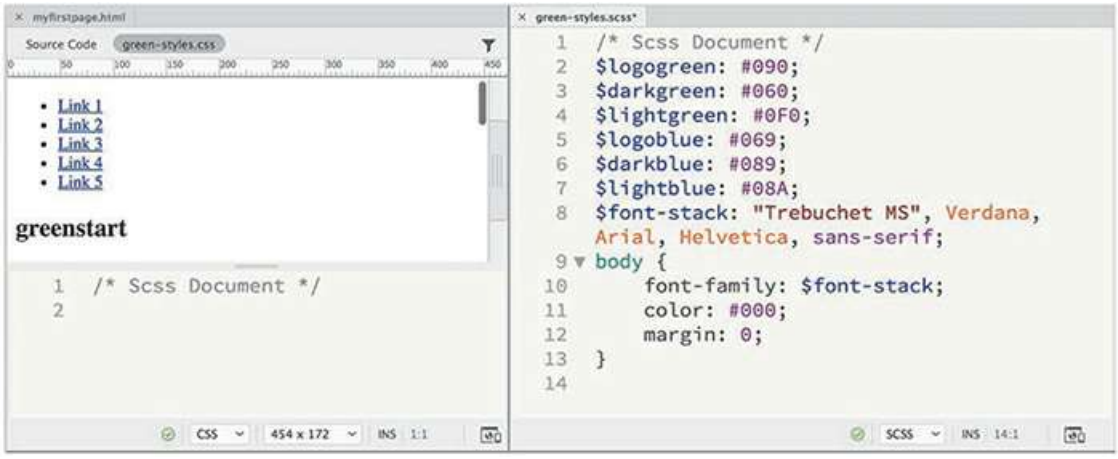
Note

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.

. Select **green-styles.css** in the Related Files interface.

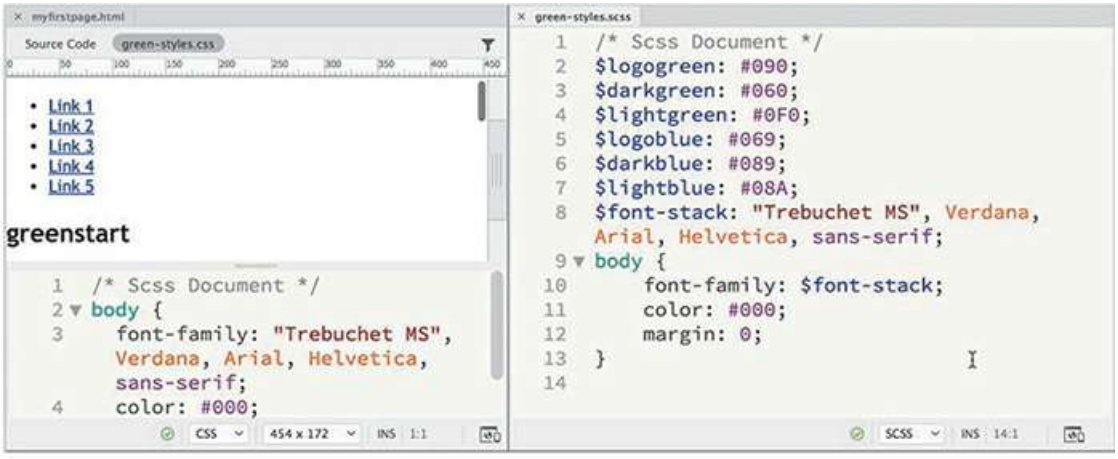
Code view displays the contents of **green-styles.css,** showing only the comment entry `/* Scss Document */`. An asterisk appears next to the filename in the document tab for **green-styles.scss**, indicating that the file has been changed but not saved.

. Choose Window > Arrange > Tile.



The webpage and the source file appear side by side in the program window.

. Insert the cursor anywhere in the **green-styles.scss** document window and choose File > Save.



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 **green-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.

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

Note

Make sure you are working in the SCSS file.

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

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

. Type **`bg$logoblue`** and press Tab. Press Enter/Return.

The shorthand expands to `background: $logoblue`, which is the first variable you created in the SCSS source file. This will apply the color `#069` to the nav element.

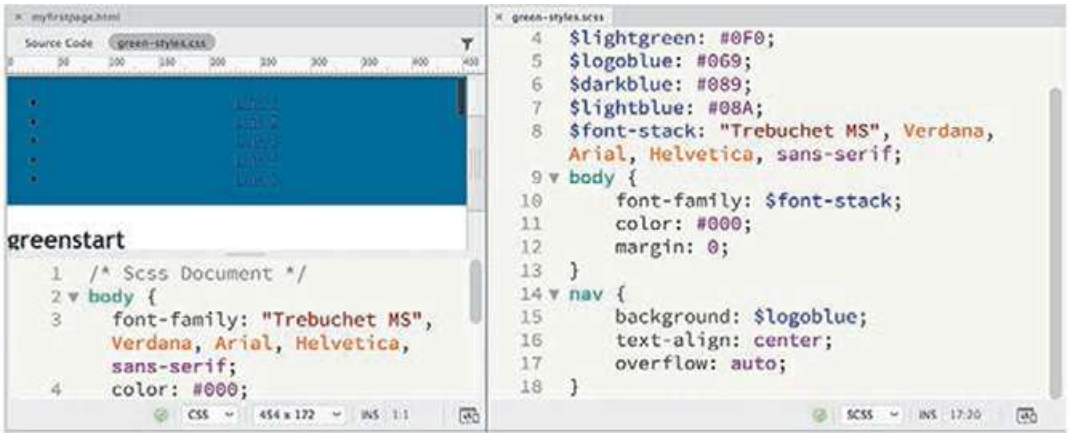
- Type **`ta:c`** and press Tab. Press Enter/Return.

  The shorthand expands to `text-align: center`.

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

  The shorthand expands to `overflow: auto`.

- Save the source file.



The `<nav>` element in **myfirstpage.html** displays the color `#069`. 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.

- Type **`ul {`** and press Enter/Return.

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

- Create the following properties:

  `list-style: none; padding: 0;`

  These properties reset the default styling of the unordered list, removing the bullet and indent. Next, you'll override the styling of the list items.

- Press Enter/Return and type **`li {`** Press Enter/Return again.

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

- Create the property **`display: inline-block;`** and press Enter/Return.

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

- Type **`a {`** and press Enter/Return. Create the following properties:

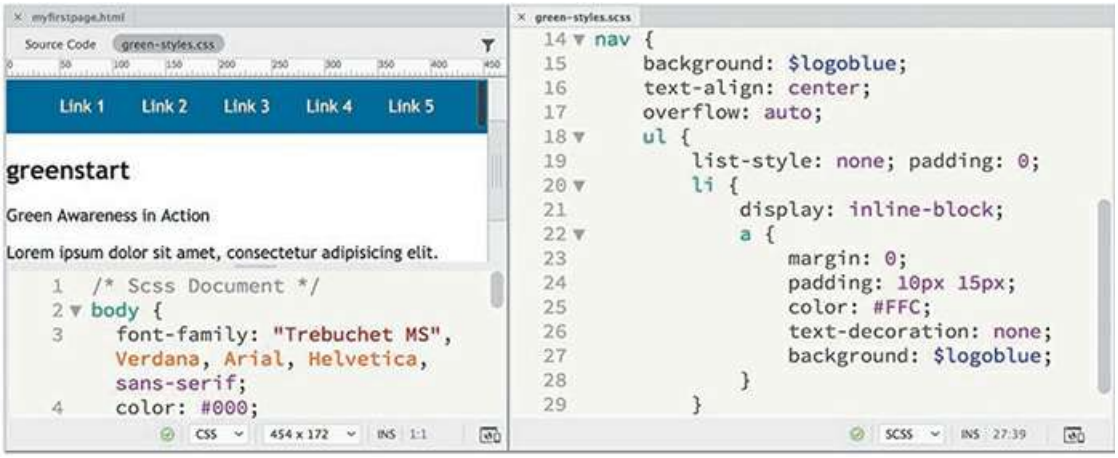
  Click here to view code image

  `margin: 0;`

```
padding: 10px 15px;
color: #FFC;
text-decoration: none;
background: $logoblue;
```

The rule and declaration for `a` appear 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. And more surprising, 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 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.

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 in one file.

- Open **myfirstpage.html** and switch to Split view, if necessary.

  Open **green-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.

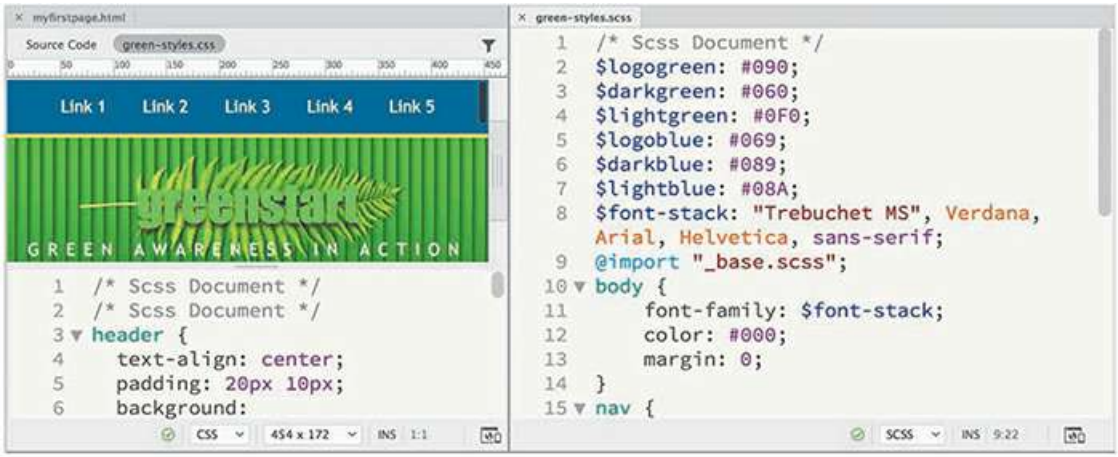
- In **myfirstpage.html,** click **green-styles.css** in the Related Files interface.

  Code view displays the content of **green-styles.css**. It contains the output of rules written in the SCSS Source file.

- In **green-styles.scss**, insert the cursor before the `body` rule. 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 **green-styles.scss** has not been saved yet.

- Save **green-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. If you examine **green-styles.css**, you will see that several rules were inserted before the `body` rule. Imported content will be added starting at line 2, the position of the `@import` command. 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.

. 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.

Learn more about preprocessors

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

Beginning CSS Preprocessors: With SASS, Compass.js, and Less.js, by Anirudh Prabhu, Apress (2015), ISBN: 978-1484213483

Instant LESS CSS Preprocessor How-to, by Alex Libby, Packt Publishing (2013), ISBN: 978-1782163763

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

Linting support

Dreamweaver CC (2019 release) 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.

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

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

. Type **`<h1>Insert headline here`**

Note

Dreamweaver may create the opening and closing tags at once. If so, delete the closing **`</h1>`** tag before proceeding to step 4.

. 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.

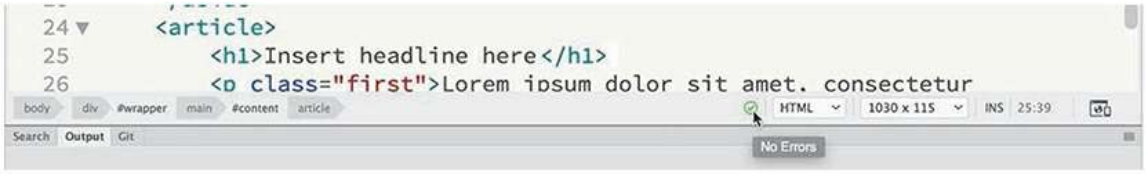
. Click the X icon .

Note

You may need to click the Refresh button to display the Linting report.

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.

. Double-click the error message.



Dreamweaver focuses on the section in the Code view window that it identifies as containing the error. Since Dreamweaver is looking for the closing tag for the `<h1>` element, the first closing tag it encounters is `</article>` and flags it, which is incorrect. This behavior will get you close to the error, but often you will have to track down the actual issue yourself.

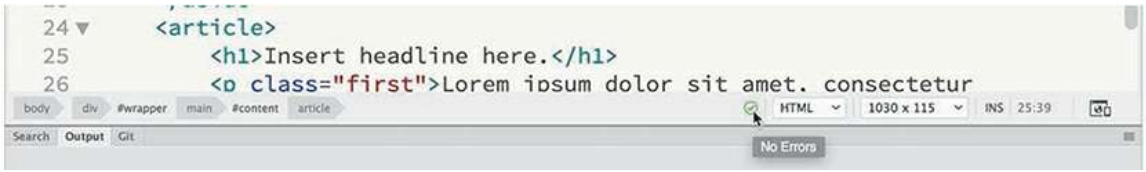
. Insert the cursor at the end of the code `<h1>Insert headline here.` Type **</**

Note

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

Dreamweaver should close the `<h1>` tag automatically. If not, go ahead and finish it properly.

. Save the file.



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

- 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.

- Open **myfirstpage.html**, if necessary, and switch to Code view.
- Scroll down and locate the `<nav>` element (around line 11).
- 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.
- Click the line number beside the `<nav>` tag.

  The entire line is selected within the window.
- 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 have frequently done in previous lessons.

- Scroll down and locate the following code:

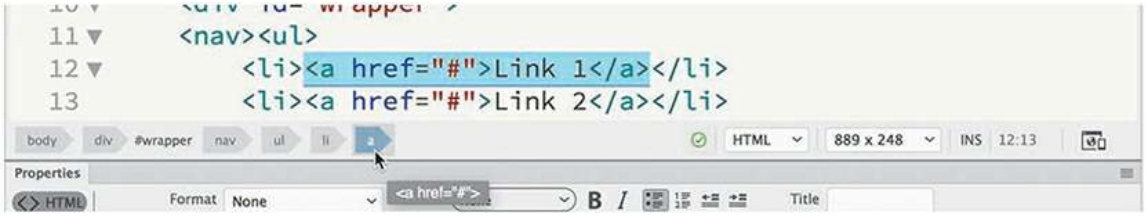
  `<a href="#">Link 1</a>`

- 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.

- 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.

- Select the `<ul>` tag selector.

  The code for the unordered list is entirely selected.

- Select the `<nav>` tag selector.

  The code for the entire menu is selected.

- 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 tags

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

- Choose Window > Toolbar > Common to display the Common toolbar, if necessary.
- Insert the cursor anywhere in the text `Link 1`.
- In the Common toolbar, click the Select Parent Tag icon .

The entire `<a>` element is highlighted.

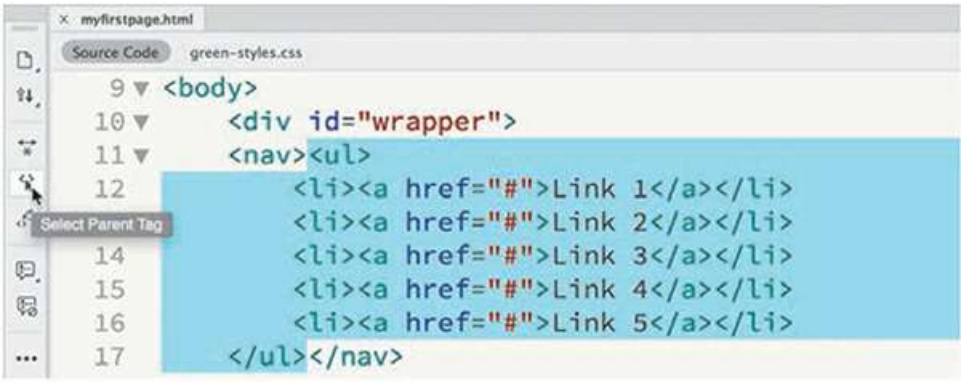
Note

The Select Parent Tag icon may not be displayed by default in the Common toolbar. Click the Customize Toolbar icon and enable the tool before proceeding to step 3, if necessary.

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

  The entire `<li>` element is selected.

- Click the Select Parent Tag icon.



  The entire `<ul>` element is selected.

- Press Ctrl+[/Cmd+[until `<div#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.

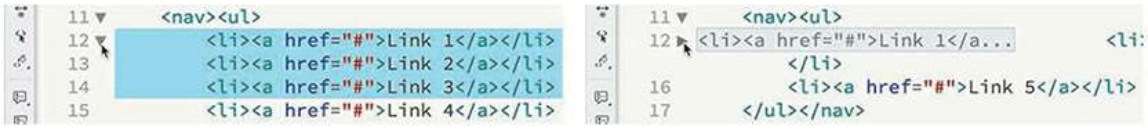
Collapsing code

Collapsing code is a productivity tool that makes a simple process out of copying or moving 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.

- Select the first three Link items in the `<nav>` element.

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

- 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.

- 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.

- 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.

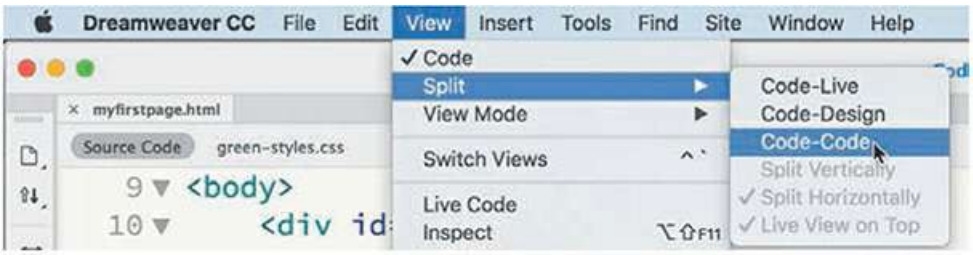
- Click the Expand icon ▸ beside the line for the <li> elements.

All collapsed elements are now expanded.

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.

- If necessary, switch to Code view.
- Choose View > Split > Code-Code.

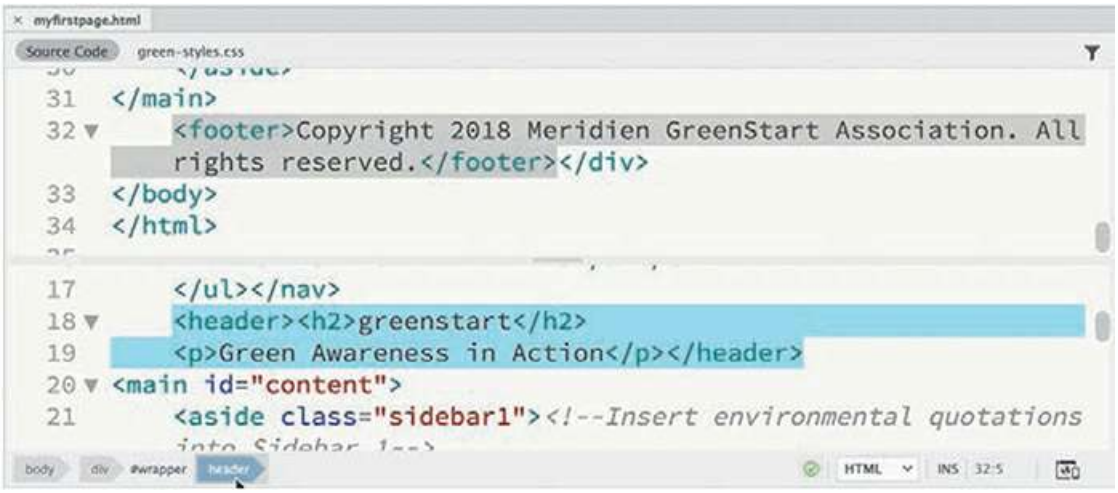


The document displays two Code view windows, both focusing on **myfirstpage.html**.

- 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.

- 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.

- In the Related Files interface, select **green-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 why 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.

- 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 **green-styles.css**.

- Click **green-styles.css** in the Related Files interface.

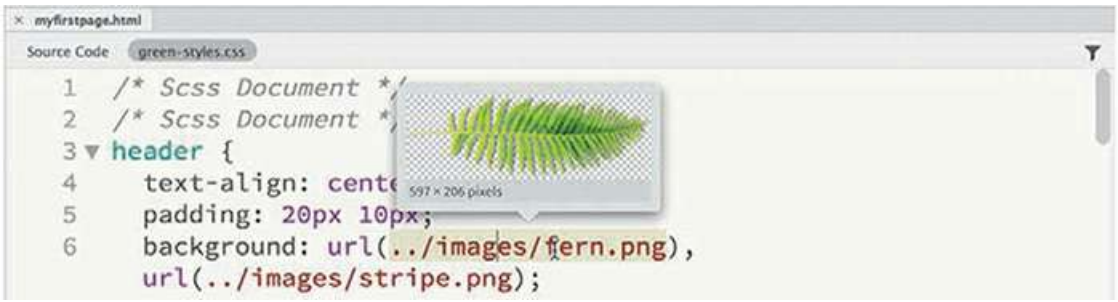
  The style sheet appears in the window. Although it's fully editable, don't waste your time making any changes to it. Since the file is the output of the SCSS source file, any changes you make will be overwritten the next time the file compiles.

- Locate the `header` rule (around line 3).

  The `header` consists of two text elements and two images. You should be able to see the

image references in the `background` property.

- Position the cursor over the markup `url(../images/fern.png)` in the `background` property (line 6).

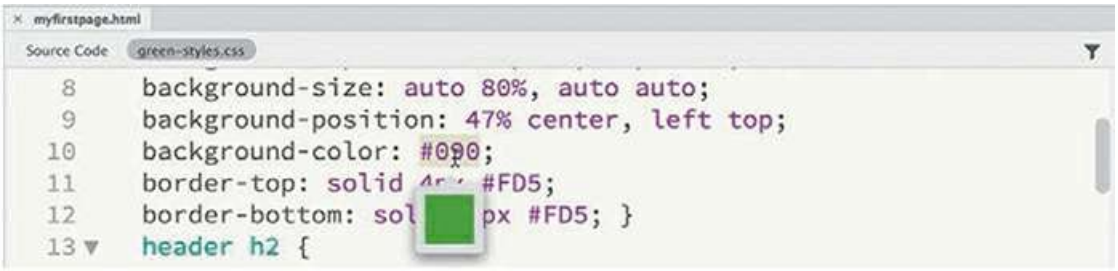


A miniature preview of the fern image appears below the cursor.

- Position the cursor over the markup `url(../images/stripe.png)` in the `background` property.

A miniature preview of the stripe image appears below the cursor. The preview function also works with color properties.

- Position the cursor over the markup `#090` in the `background-color` 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 code without compromises.

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. What do you have to install to create a LESS, Sass, or SCSS workflow in Dreamweaver?
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 most linked files?

Review answers

1. Dreamweaver provides code hinting and auto-completion for HTML tags, attributes, and CSS styling as you type, along with support for ColdFusion, 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 stored on your local hard drive.

13 Designing for Mobile Devices

Lesson overview

In this lesson, you'll learn how to do the following:

- Access and configure Dreamweaver's tools and interface for mobile design
- Create a mobile-friendly page layout based on a web framework
- Insert and format new content and components into a Bootstrap-based layout
- Apply a mobile template to existing site pages

This lesson will take about 1 hour to complete. If you have not already done so, please log in to your account on peachpit.com to download the project files for this lesson as described in the "Getting Started" section at the beginning of this book and follow the instructions under "Accessing the Lesson Files and Web Edition." Define a site based on the lesson13 folder.

Your Account page is also where you'll find any updates to the lesson files. Look on the Lesson & Update Files tab to access the most current content.

The trend toward designing sites to respond automatically to mobile devices and smartphones continues to grow exponentially. Dreamweaver has powerful tools to get your site mobile ready.

Responsive design

The Internet was not designed for smartphones and tablets. In the ‘90s, the most difficult challenge a programmer or developer faced was accounting for the size and resolution differences between 13- and 15-inch computer monitors. For years, resolutions and screen sizes only got *larger*.

Today, the odds that some or all of your visitors are accessing your site with a smartphone or tablet are increasing daily. Statistics show that starting in 2014 more people used mobile devices to access the Internet than used desktop computers, and that number has been increasing steadily ever since.

Mobile-first design

As the number of people favoring phones and tablets over desktop computers con-tinues to grow, the logical evolution is toward a philosophy known as *mobile-first* design. It assumes that sites will shed users and traffic if they aren’t optimized for phones and tablets.

Mobile-first design starts with a design for mobile devices, such as smartphones, and then adds content and structure for larger devices and computers. In some cases, the site’s content is

actually minimized so that it loads and performs in an optimal fashion, and then additional content is *injected,* using JavaScript and online databases, for computers and more powerful devices.

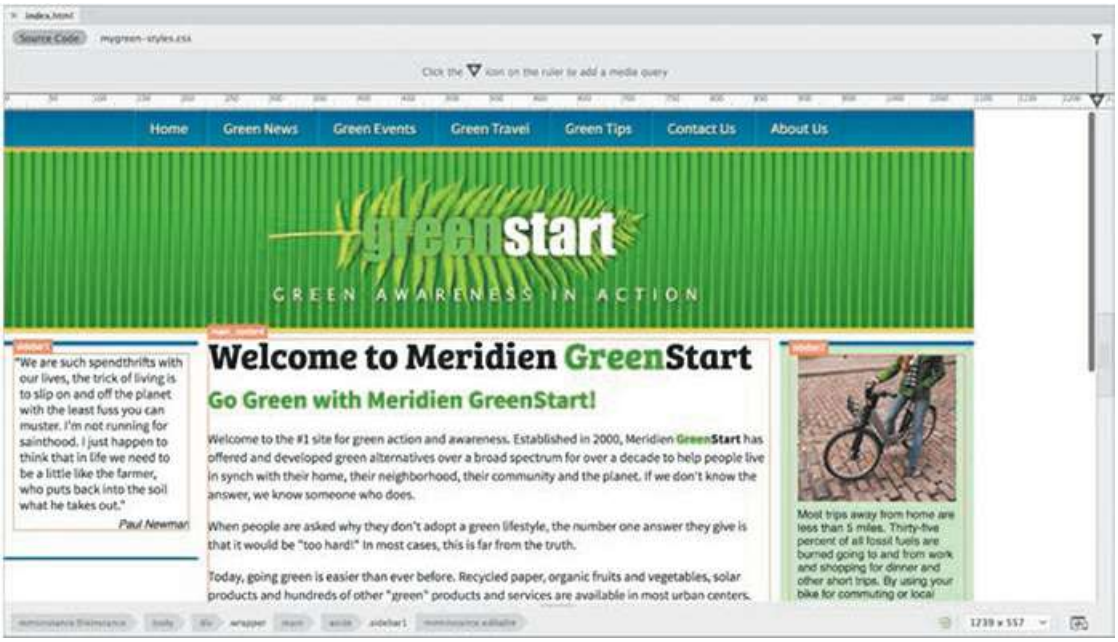
How your site deals with smartphones and mobile devices depends on whether you're adapting an existing site or developing a new one from scratch. So, what does all this mean to the website you built over the last 12 lessons? Let's take a look.

Testing responsiveness in Dreamweaver

You could upload the GreenStart site to the Internet, as you learned how to do in Lesson 11, "Publishing to the Web," and test the various pages on desktop computers, smartphones, and tablets. But most of the information you need can be found right inside Dreamweaver itself. You have learned that Live view was designed to preview the HTML, CSS, and JavaScript creating your page content exactly as it would appear on the web. This functionality also supports responsive design techniques.

- Launch Dreamweaver CC (2019 release or later).
- If necessary, select lesson13 from the sites dropdown menu in the Files panel.
- Maximize the size of the program interface to fill the entire computer display. Maximize the size of the document window in the interface.

  Many mobile designs are keyed into the display size and width of the browser window. It's essential to make sure the Dreamweaver document is as large as the computer display will allow (at least 1200 pixels, if possible).
- Open **index.html** from the site root in Live view.



This page was created from the site template in Lesson 11. It represents the basic design of the

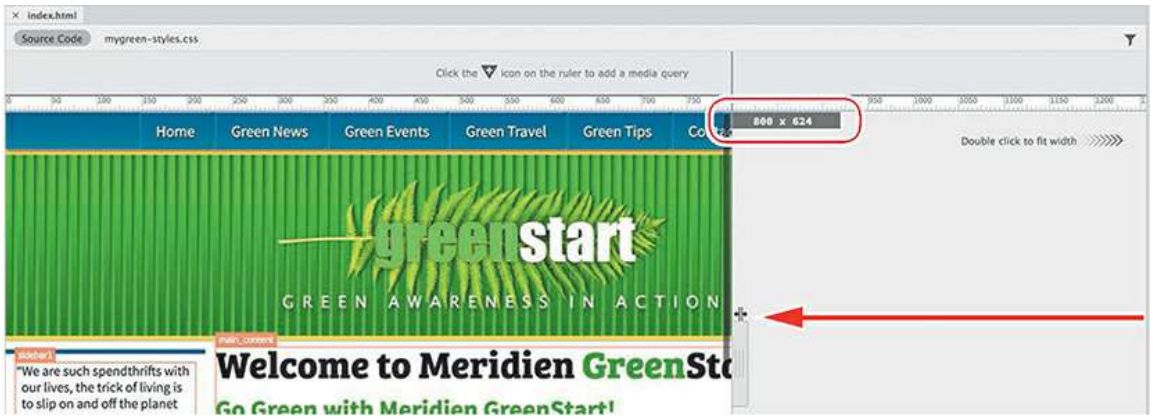
various pages in the GreenStart website to this point. The Dreamweaver interface provides several features that allow you to test and control the responsiveness of a webpage.

Before Dreamweaver CC, you had to resize the whole document window to test the responsiveness of a particular webpage. The scrubber tool was added to Live view to enable you to interact with pages without changing the size of the program interface. It appears by default on the right edge of the Live view window.

Tip

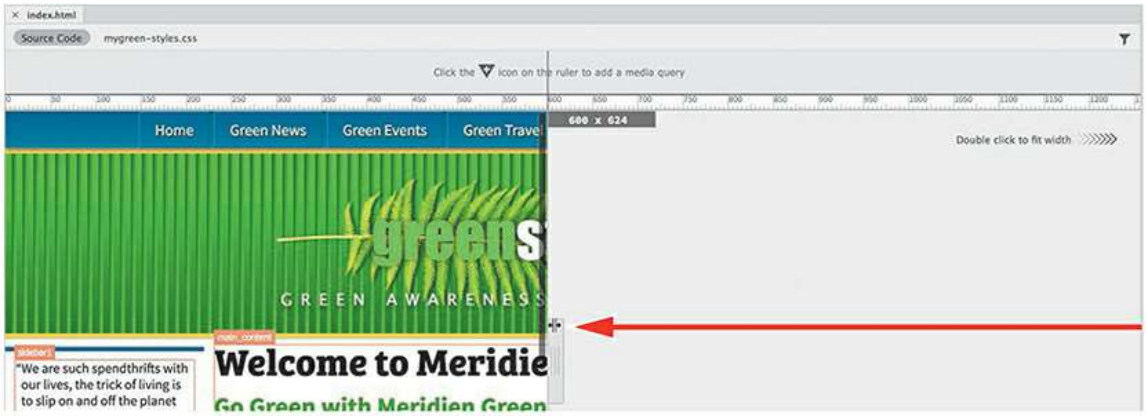
As you drag the scrubber, notice that the width and height of the document window is displayed on the right side of the scrubber.

- Drag the scrubber to the left to set the width of the document window to 800 pixels.



As the window decreases in width, Dreamweaver simulates what the page would look like if it were loaded in a browser or device 800 pixels in width.

- Drag the scrubber to set the document window to 600 pixels.

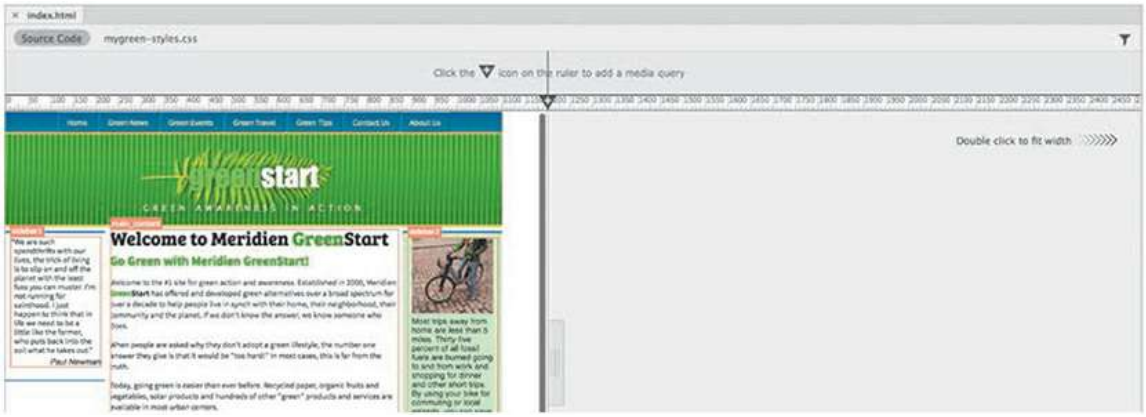


Once the width of the window decreases from full size, the right side of the page is obscured

and unusable. The page does not respond in any way to the changing width. Since the site template is based on a fixed-width design, it was never intended to adapt automatically to smaller screens. Webpages like this react to different screen sizes in one of two ways.

On desktop displays, the page may display at actual size and simply scroll off the screen to the right, as you see it now in Dreamweaver. The other option, which will occur on smartphones and tablets, is that the page may simply scale in size to fit the available screen. You can simulate that effect in the document window by zooming out.

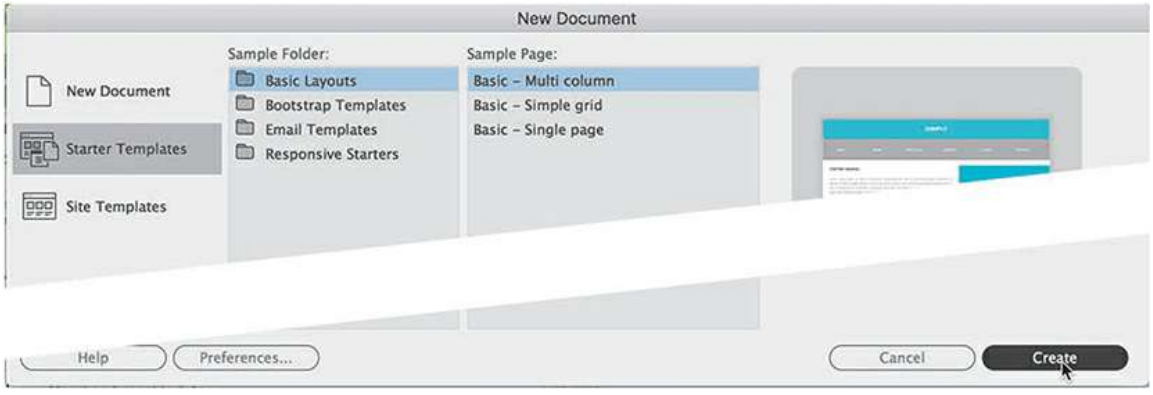
- Press Ctrl+–(minus)/Cmd+– (minus) twice to zoom out.



The webpage scales smaller to fit into the reduced space.

Whether the article scrolls off the screen or scales down to fit, the result is a webpage that is useless to many visitors. The alternative to a fixed-width design is one that can adapt automatically to different screens and devices. Dreamweaver provides some built-in examples of such designs.

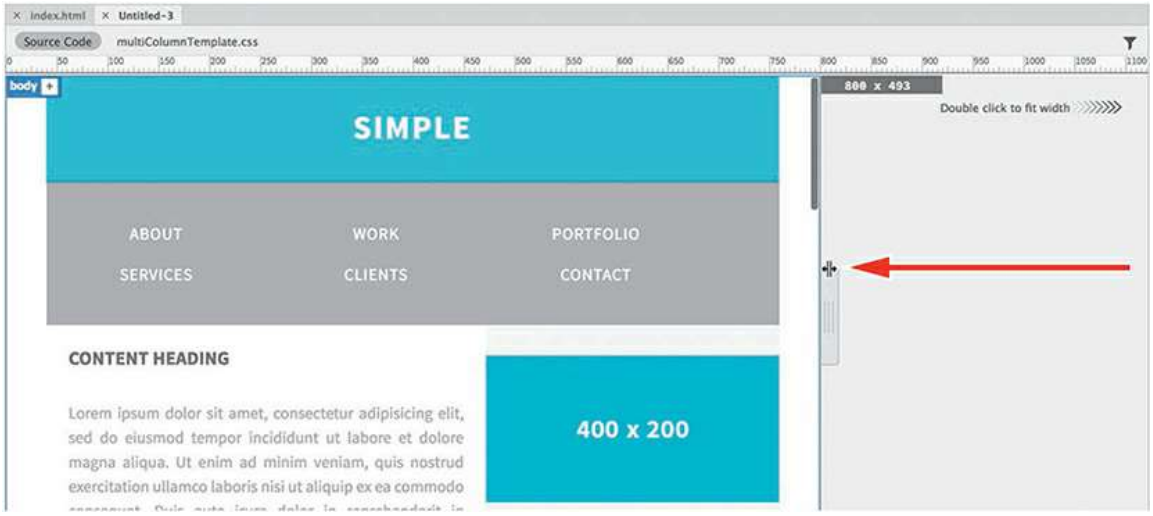
- Select File > New.
- In the New Document dialog, select Starter Templates > Basic Layouts > Basic – Multi Column and click Create.



The document window displays a new untitled page based on the selected starter template. All the new Dreamweaver templates provide various levels of responsiveness.

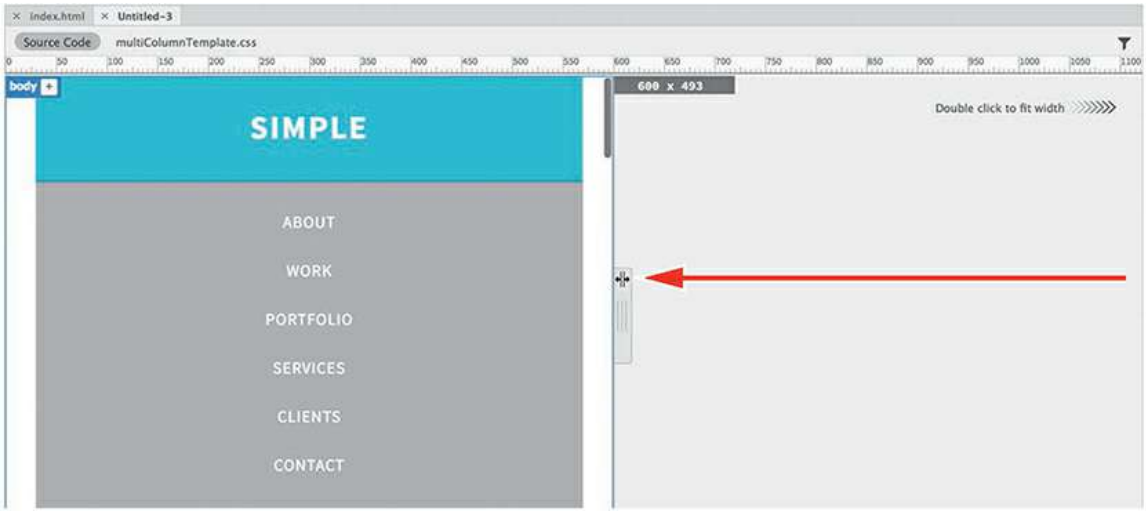
- Drag the scrubber to the left to 800 pixels.

  Observe the changes to the page content.



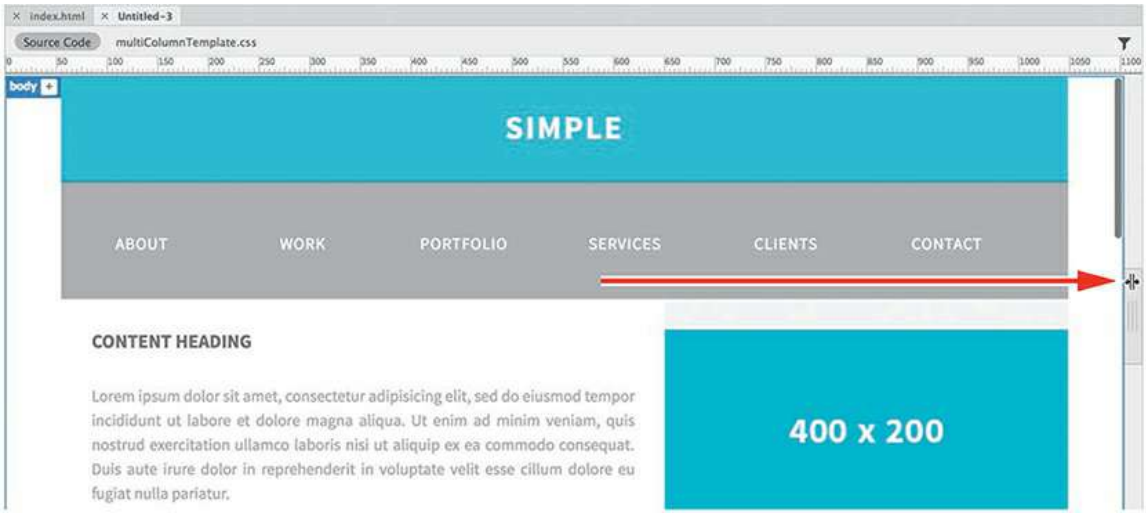
  As the width decreases, the page content automatically reacts to changes. The navigation menu switches from a single row to two rows. The text and image placeholders resize to share the remaining space. At no time while you are resizing the window does any of the content scroll off to the right. It all remains within the visible area and accessible.

- Drag the scrubber to 600 pixels.



  As you drag the scrubber, you will see the content continue to adapt. Eventually, the content shifts to form a single column. The original horizontal navigation now stacks in a single vertical column. This behavior also works in reverse.

- Drag the scrubber to the right.



As you drag the scrubber the document reformats on the fly, putting everything back where it originally came from. This isn't magic; it's responsive design. It's based on some concepts you already know and some you are about to learn.

No matter whether you decide to design for desktop first or mobile first, you still need to learn how to make webpages that can respond automatically to devices of any size. To start, you have to learn about two basic CSS parameters: *media type* and *media query*. These parameters enable browsers to identify what size and type of device is accessing the webpage and then load the appropriate style sheet, if one exists.

Media type properties

The media type property was added to the CSS2 specifications and adopted in 1998. It was intended to address the proliferation of non-computer devices that were able to access the web and web-based resources at that time. The media type is used to target customized formatting to reformat or optimize web content for different media or output.

In all, CSS includes ten individually defined media types, as shown in **Table 13.1**.

Table 13.1 Media type properties

PROPERTY	INTENDED USE
`all`	All devices. "All" is the default media type if one is not specified in the code.
`aural`	Speech and sound synthesizers.
`braille`	Braille tactile feedback devices.
`embossed`	Braille printers.
`handheld`	Handheld devices (small screen, monochrome, limited bandwidth).
`print`	Documents viewed onscreen in print preview mode and for printing applications.
`projection`	Projected presentations.
`screen`	Primarily for color computer screens.

`tty`	Media using a fixed-pitch character grid, such as Teletypes, terminals, or portables.
`tv`	Television-type devices (low resolution, color, limited-scrollability screens, sound available).

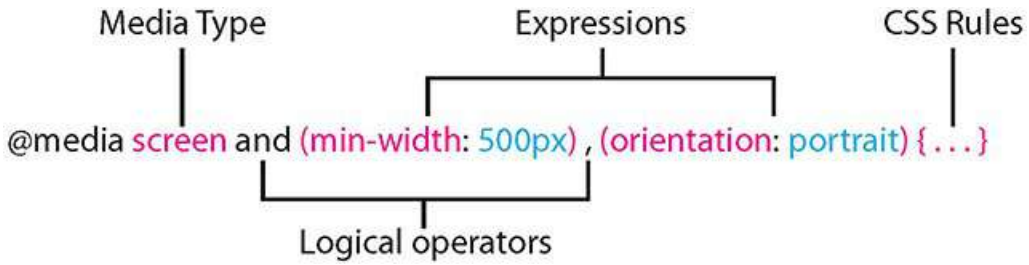
Although the media type property works fine for desktop screens, it never really caught on with browsers used on cellphones and other mobile devices. Part of the problem is the sheer variety of devices available in all shapes and sizes. Add to this smorgasbord an equally diverse list of hardware and software capabilities and you've produced a nightmare environment for the modern web designer. But all is not lost.

Media queries

A media query is a newer CSS development that enables a webpage to interactively determine what formatting to use based not only on what kind of device (media type) is displaying the page but also on what dimensions and orientation it's using. Once the browser knows the type or size of the device it has encountered, it reads the media query to know how to format the webpage and content. This process is as fluid and continuous as a precision dance routine, even allowing the visitor to switch orientations during a session and have the page and content adapt seamlessly without other intervention. The key to this ballet is the creation of style sheets optimized for specific browsers, specific devices, specific orientation, or all three.

Media query syntax

Like the CSS it controls, a media query requires a specific syntax to work properly in the browser. It consists of one or more media types and one or more expressions, or media features, which a browser must test as true before it applies the styles it contains. Currently, Dreamweaver supports 22 media features. Others are being tested or are still under development and may not appear in the interface, but you can manually add them to the code, if necessary.



The media query creates a set of criteria to determine whether a specific set of rules contained within it is applied in a webpage.

You can create media queries in a variety of ways. For example, they can be designed to work exclusively (by completely resetting the existing styling) or in tandem (by inheriting some styles and modifying specifications only as necessary). The latter method requires less CSS code and is typically more efficient. We will favor that method in the upcoming exercises and for the new responsive site design.

To learn more about media queries and how they work, check out

www.w3schools.com/cssref/css3_pr_mediaquery.asp.

Working with the Visual Media Queries interface

One of the best aspects of working with Dreamweaver is that it provides visual tools to help you style and troubleshoot various CSS properties. The Visual Media Queries (VMQ) interface is one such tool. It allows you to identify and interact with media queries instantly using the cursor.

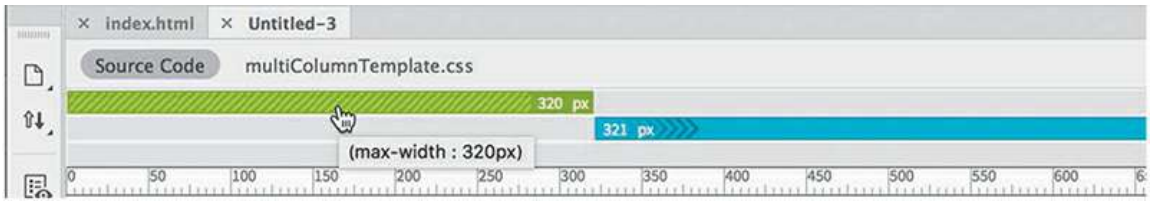
Note

The following exercise requires the sample document created from the Basic – Multi Column starter template earlier.

. If necessary, click the Toggle Visual Media Queries Bar icon in the toolbox on the left side of the screen.

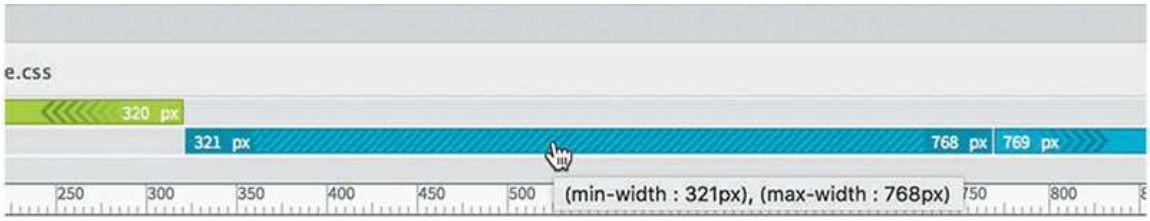
Depending on the width of your screen, the toolbar displays four media queries applied to this document.

. Position the cursor over the first media query.



A tooltip appears, identifying that the media query is set for a max-width of 320 pixels. In other words, the styling controlled by this media query applies only to screens 320 pixels wide or narrower.

. Position the cursor over the second media query.



The second media query is set for a min-width of 321 pixels and a max-width of 768 pixels. By using both min-width and max-width, this media query targets screens from 321 pixels to 768 pixels. The interface displays the media query settings using colored bars labeled with the pixel dimensions so that you can see the specifications instantly.

You can use the cursor to activate or switch the document view between each media query.

VMQ enables you to see the differences

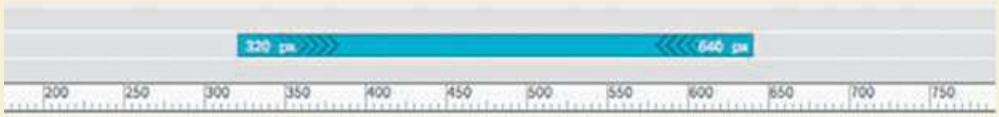
Media queries enable your webpage and its content to adapt to a variety of different types of screens and devices. They do this by loading custom style sheets created for specific screen sizes, devices, or even orientations. Later, you'll learn more about media queries, how they work, and how to create them. For now, let's review how the Visual Media Queries interface works.

The VMQ interface identifies all media queries defined on the page or within style sheets linked to it. The media queries are displayed in color based on their specifications.

Media queries that define only a minimum width are displayed in purple.

Media queries that define only a maximum width are displayed in green.

Media queries that define both minimum and maximum widths are displayed in blue.



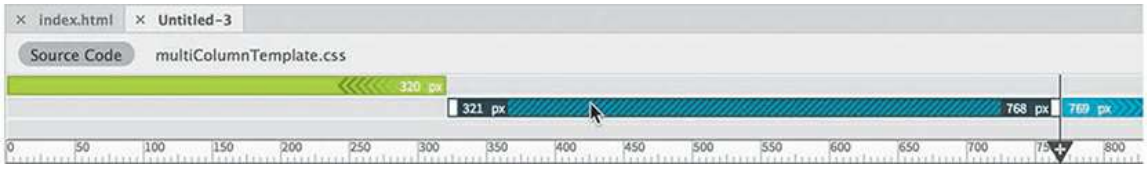
The Dreamweaver workspace is fully responsive and will display the specific CSS styling appropriate to the screen size and orientation within CSS Designer. To display the styling associated with a specific media query, simply click the media query

notation in the @Media pane of CSS Designer.

Note

Don’t click directly on the number itself, because it will open a field that allows you to edit the width setting.

- Click the cursor to the right of the number 321 px in the VMQ interface.

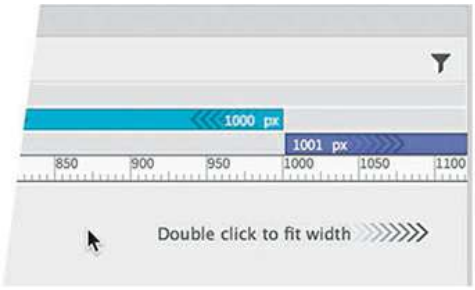


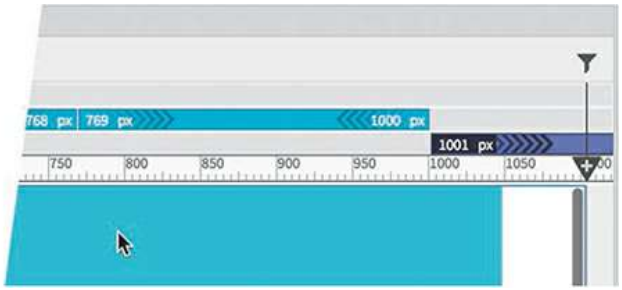
The document window resizes instantly to 768 pixels. The content adapts to the window size based on the applicable CSS styles.

- Click the same media query again.

The document window narrows to 321 pixels. Clicking the media query a second time toggles the display between its beginning and end. To return to full size, drag the scrubber to the right edge of the window or double-click in the gray area to the right of the scrubber.

- Double-click anywhere in the gray area on the right side of the scrubber.



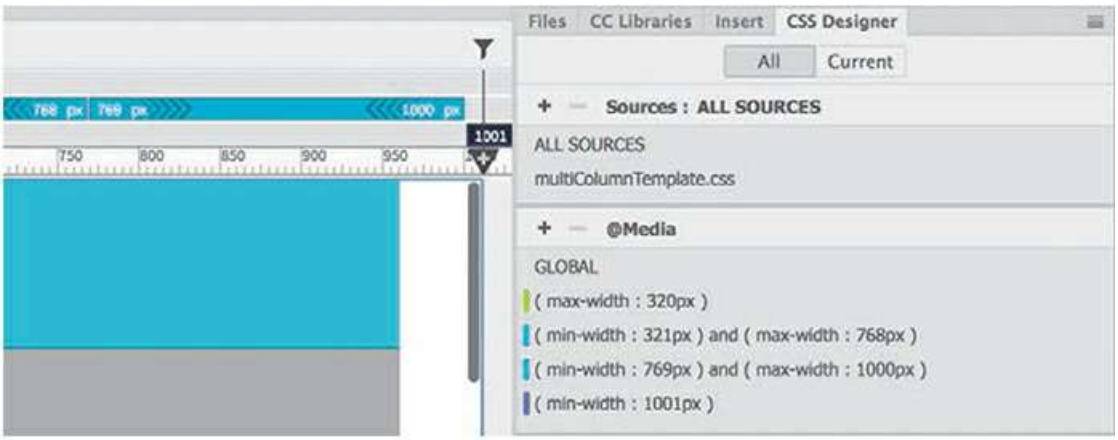


The document window opens to the full size of the workspace.

The Visual Media Query interface display is integrated closely with CSS Designer and reflects the various specifications defined there.

- If necessary, select Window > CSS Designer to display the panel.

Open the @Media pane to display the settings defined within it.



In the @Media pane, you will see the GLOBAL attribute and four predefined media queries. Within each of these queries are sets of CSS rules that are geared to style and reformat the page on different screen dimensions. To see the CSS rules assigned to each you merely have to click on each media query.

Switch between the two open documents and compare the display of the VMQ interface and CSS Designer. The main difference between the two is obviously the lack of media queries and the associated rules defined within them. The advantage of using the starter layouts in Dreamweaver is that many of them are based on powerful web frameworks.

Introducing web frameworks

If you look through the style sheet and at each media query displayed in CSS Designer in the Untitled document, you will see the various ways this starter template is formatted for different screen sizes. Creating CSS rules this way, to handle all types of content for every type of screen and device, can be a daunting task. Even experienced web designers think twice about building responsive designs from scratch. Because the number and types of mobile devices have proliferated exponentially in the last few years adapting content successfully to every device and screen size is nearly impossible without a little help. That's why developers have created a number of front-end frameworks to make that task much easier.

A framework is a collection of predefined HTML and CSS code—often including JavaScript—that permits the rapid development and deployment of webpages and web applications. Most frameworks are built from the ground up to display content responsively on a wide range of devices. That way, you can concentrate on what the content says rather than how, or whether, it displays.

As of this writing, Dreamweaver CC (2019 release) supports three web frameworks out of the box: jQuery UI, jQuery Mobile, and Bootstrap. The jQuery accordion used in Lesson 10, "Adding Interactivity," is part of the jQuery UI framework, which provides a set of GUI widgets, animated visual effects, and themes. It's not designed to build an entire site but can add useful interactivity to existing pages.

jQuery Mobile is a touch-enabled web framework specifically optimized for smart-phones and

tablets. It’s geared for building mobile-only websites or for integration with frameworks like PhoneGap or Worklight.

Created by Twitter, Bootstrap was released to the public in 2011 and quickly became one of the most popular frameworks in use. It was incorporated into Dreamweaver in a previous version and provides a powerful set of tools for building complete websites from the ground up that are fully optimized for mobile devices and desktop screens. One advantage of using Bootstrap is that there are plenty of resources and add-ons available that extend its capabilities independent of Adobe and Dreamweaver. In the 2019 release, Dreamweaver support was updated to the latest version, Bootstrap 4.

In the upcoming exercises, you’ll learn how to build a custom layout using the Bootstrap framework to replace the current non-responsive site template and then update the existing site pages to support all size screens and devices.

Identifying the Bootstrap structure

Underlying all the power and flexibility of Bootstrap is a basic grid system of rows and columns. This concept harks back to the earlier days of web design, before the advent of CSS, when we would use tables to cobble together our layouts. The only way to impose order on our webpages was to organize our text and pictures into table rows and cells.

No, we’re not going back to the bad old days. Tables were not responsive. Although they could scale up or down in size, they would not automatically adapt to the screen and knew nothing about mobile devices. Instead, the rows and columns of Bootstrap have been carefully engineered to work in most modern browsers and devices.

The first step in Bootstrap is to identify the basic grid structure you need to build, or impose, on the proposed site design. This should be quite easy when you reexamine the site mockup.

. If necessary, open **GreenStart_mockup.html** in Live view.

To study the site mockup, start by marking up the content that would be grouped together in rows, as in the following figure.

Next, identify the columns within the content. Remember that the columns are divvied up in the rows you already created.

. Close all open files. Do not save any changes.

Once you have the basic plan established, building the Bootstrap structure is a simple procedure using Dreamweaver.

Creating a Bootstrap layout

In this exercise, you will build the basic structure of the new template using the Bootstrap framework and the Insert panel.

. Select File > New.

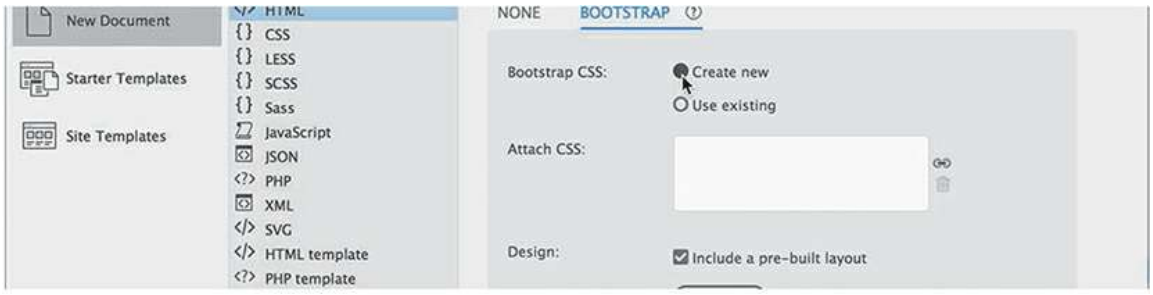
  The New Document dialog appears.

. Select the New Document tab in the New Document dialog.

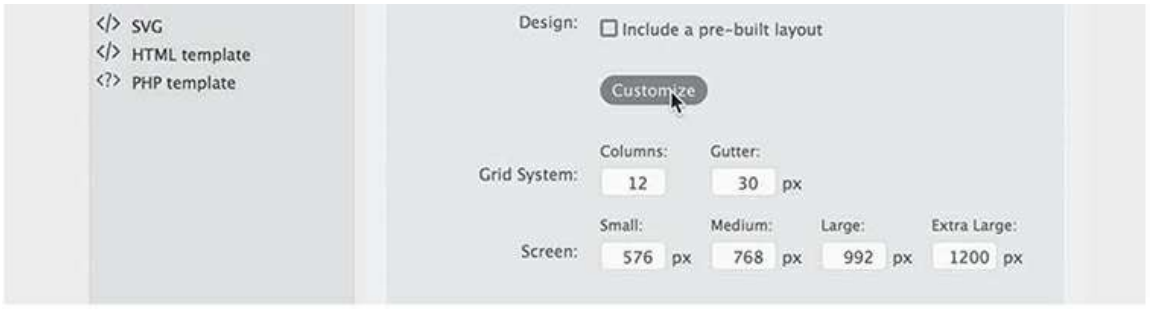
. Select HTML in the Document Type window.

. Click the Bootstrap tab.

- For Bootstrap CSS, select Create New.



- Deselect the Design option Include A Pre-built Layout.
- Click the Customize button.



The Customize options allow you to change the number of columns, the gutter width, or the predefined screen sizes. For this layout, you will leave the default settings as they are, but if you use Bootstrap in the future, you should make sure these numbers reflect the needs of your site and its visitors.

- Click Create.

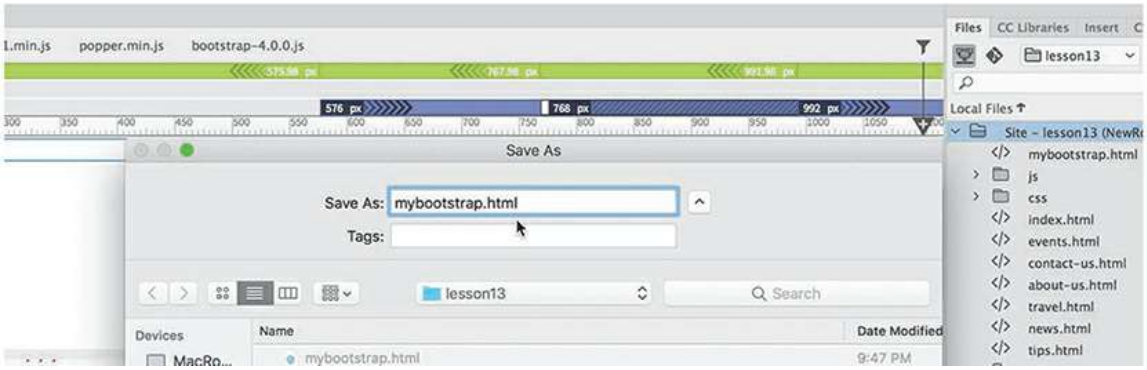
A new, untitled document appears in the document window.

- Select File > Save.

Name the file **mybootstrap.html** and save it in the lesson13 folder.

Note

The supporting JavaScript and CSS files for various frameworks, like Bootstrap, are updated from time to time and may be different from the ones shown in this lesson.



Although the file seems to be entirely empty, lots of things are already going on. You can see that the VMQ interface displays at least six media queries and that the Related Files interface shows one CSS and three JavaScript files. But you've only started.

Adding Bootstrap components

The new layout is set up to support the Bootstrap framework. The next step is to add some basic structures. In this exercise, you will add the basic page skeleton.

- If necessary, open **myBootstrap.html** in Split view.

  Click in the Live view window.

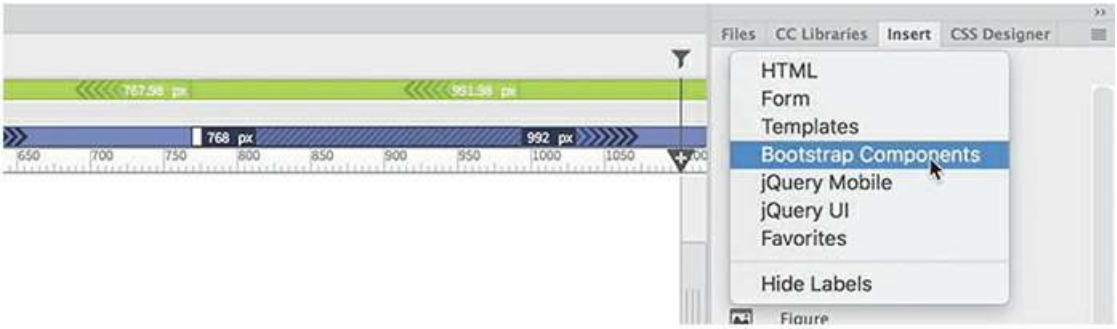
  The `<body>` element should be selected in the window.

  If neccessary, click the body tag selector.

- Display the Insert panel.

  If it's not visible on the screen, you can select it from the Window menu.

- In the Insert panel, select **Bootstrap Components** from the dropdown menu.

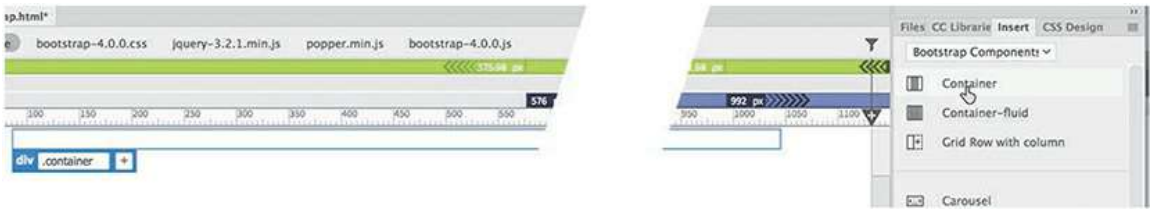


The panel displays a list of 26 main items and more than 80 subitems supported by the framework. Although the list is not exhaustive of all the possible Bootstrap components and widgets, it's a good start. And whatever you can't find in the Insert panel can always be manually added by using the Code window.

. Click in the Live view document window.

  The Element Display appears focused on the `<body>` element.

. Click the **Container** item at the top of the Insert panel.



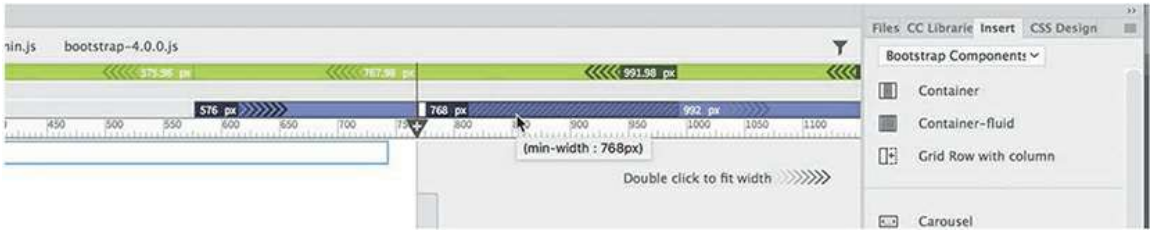
This option inserts a fixed-width `<div>` element in the page. In Bootstrap it is, by default, 1170 pixels wide in a full-screen browser on a desktop computer. On smaller screens or devices, this container will display at various smaller fixed widths or scale as necessary to fit the screen.

Once you've established your overall container, you can start creating the row and column scheme devised earlier, but first you will target the default page width.

Note

To see all the media queries, the document window will have to be at least 1100 pixels in width.

. Click the *Medium* (purple) media query (min-width 768 pixels) in the VMQ interface.

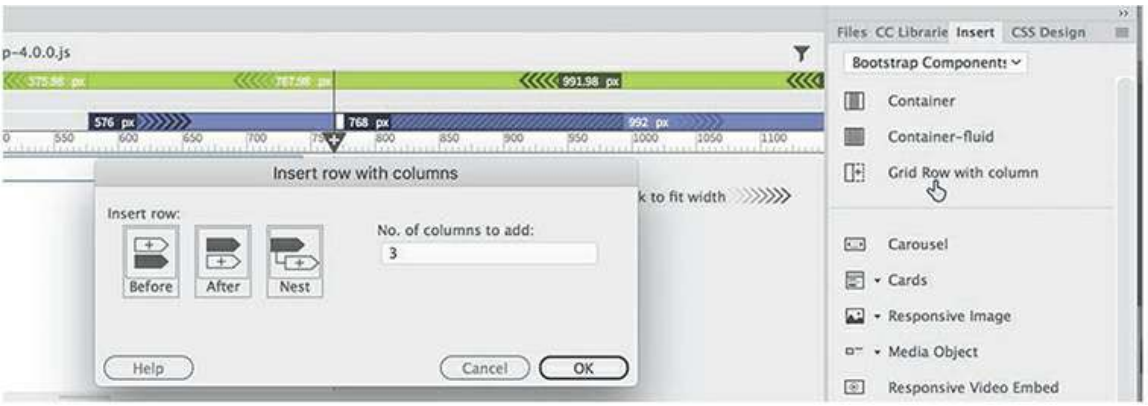


The document window resizes to match the dimensions of the media query. Targeting a specific width first determines what classes the Bootstrap framework automatically assigns to the components when you add columns to the rows.

Note

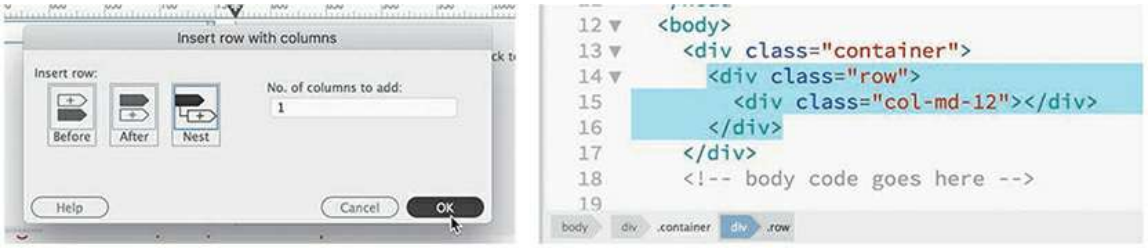
The `div` element should be selected in the document window by default.

. Click the **Grid Row With Column** item in the Insert panel.



The Insert Row With Columns dialog appears.

. Click the **Nest** option, which will insert the row inside the container element created in step 5. Enter **1** in the No. Of Columns To Add field. Click OK.



In the Code window, you can see that Dreamweaver inserted two `<div>` elements, one with a class of row and the other with a class of `col-md-12`, nested one inside the other in the initial container. Since the *Medium* media query was targeted in step 6, the class says md. *Small* classes will say `sm`, and *Large* classes will say `lg`.

The structure doesn't look very remarkable, but this is the key to the power of Bootstrap. These classes apply predefined styles to the elements that will allow them to adapt to different screens and devices. By adding more classes or manipulating the existing ones, you can provide different types of formatting and behaviors as desired.

As complex and elaborate as Bootstrap might be, one aspect of this scheme is easy to understand. If you remember what you saw in the New Document dialog, the Bootstrap specifications called for 12 columns in the grid. The class col-md-12 speaks to this grid by telling the `<div>` to be *12 columns* wide on *medium* devices. A medium device is considered to be a tablet at least 768 pixels wide. But don't let that fool you. It's important to know that some Bootstrap classes, like this one, are based on inheritance theory and format elements even on larger devices. In other words, this class will continue to format the page unless another class overrides the styling.

As we work through this layout and the upcoming online lessons, you will learn how to add to the main components other Bootstrap classes that will specify their behavior in each target environment you want to support. The first row will hold the main site navigation menu. Let's continue building this layout by adding a new row for the page header next. The first row should still be selected.

Warning

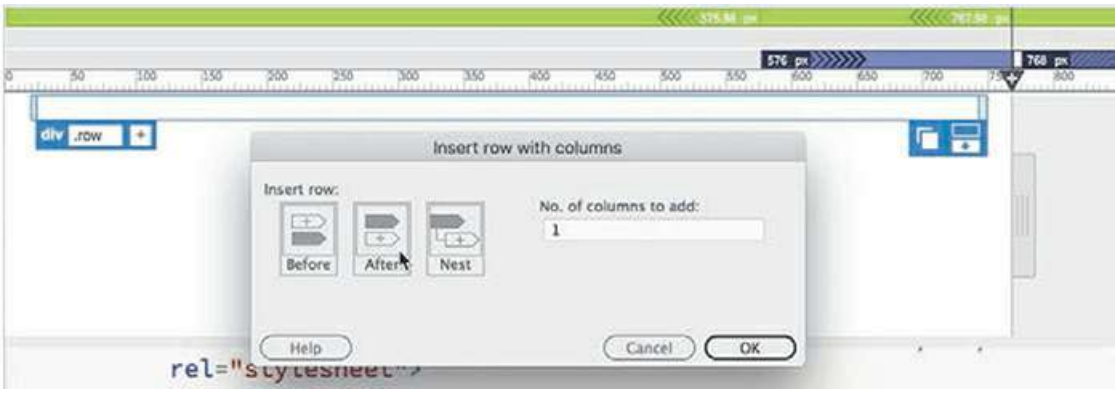
Make sure the class on the new element says `col-md-12`. Dream-weaver assigns the class based on the media query selected in the VMQ.

. Click the **Grid Row With Column** item again.

The Insert Row With Columns dialog appears.

. Enter **1** for the number of columns.

Select **After**, which will insert a row after the current one. Click OK.

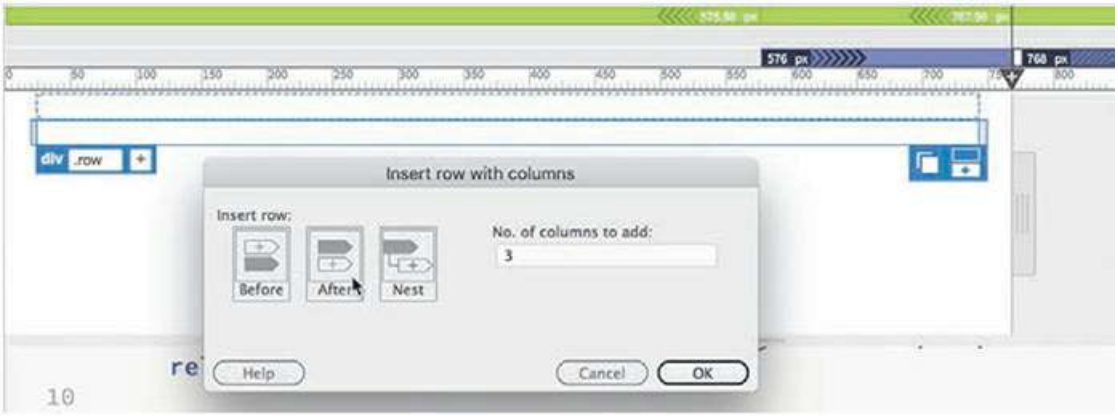


A new row is created, inserted after the first. The second is a duplicate of the first and will eventually hold the header and company logo. You'll add that later, but now let's create the next row.

Note

Pay close attention to where the new element appears in the responsive structure. Make sure the new row appears separate from and below the first but wholly inside the container element.

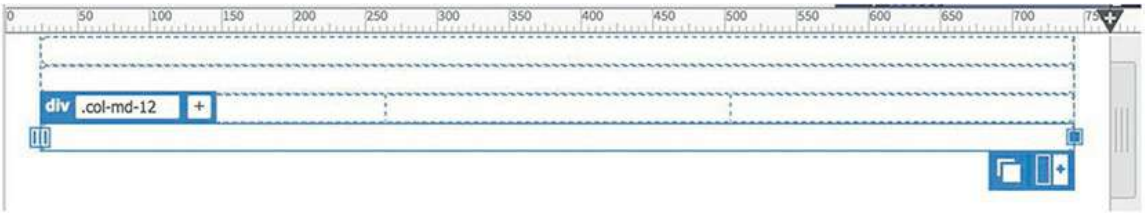
. Insert another Grid Row With Column, as in step 7. In the Insert Row With Columns dialog, enter **3** this time for the number of columns and select After. Click OK.



A new row appears with three nested `<div>` elements. The new elements have a class of `col-m-4`. Because 12 divides into 4 three times, the new elements form three columns that divide the available space into three equal parts. Later, you will modify these classes to change the widths and the relationships of these elements to one another. But let's finish the layout first. There's one more row needed for the page footer.

```
13 ▼    <div class="container">
14 ▼      <div class="row">
15          <div class="col-md-12"></div>
16        </div>
17 ▼      <div class="row">
18          <div class="col-md-12"></div>
19        </div>
20 ▼      <div class="row">
21          <div class="col-md-4"></div>
22          <div class="col-md-4"></div>
23          <div class="col-md-4"></div>
24        </div>
25    </div>
```

- Repeat steps 9 and 10 to create a new row with one column.



A new row is added for the last row, which will hold the footer. The basic Bootstrap structure for the site design is now complete.

- Save the file. If you are continuing to the next exercise, leave **myBootstrap.html** open, but close any other open documents.

In the upcoming exercises, you will modify the basic layout to add HTML5 elements and content placeholders for the site template.

Adding semantic elements to Bootstrap

As you can see from the previous exercise, Bootstrap relies heavily on the `<div>` element. There's nothing wrong with this technique; the framework can't intuit the purpose of the elements in the rows and columns and automatically add the appropriate tags. But as a generic container, the `<div>` element conveys no semantic value, or other information, to search engines or other web applications.

Once you have created your basic structure it makes sense to go back and swap out these generic structures with HTML5 semantic elements that more closely match your intended usage or content model, as you did in the existing website earlier. Dreamweaver makes it easy to edit structural elements.

- If necessary, launch Dreamweaver CC (2019 release) or later.

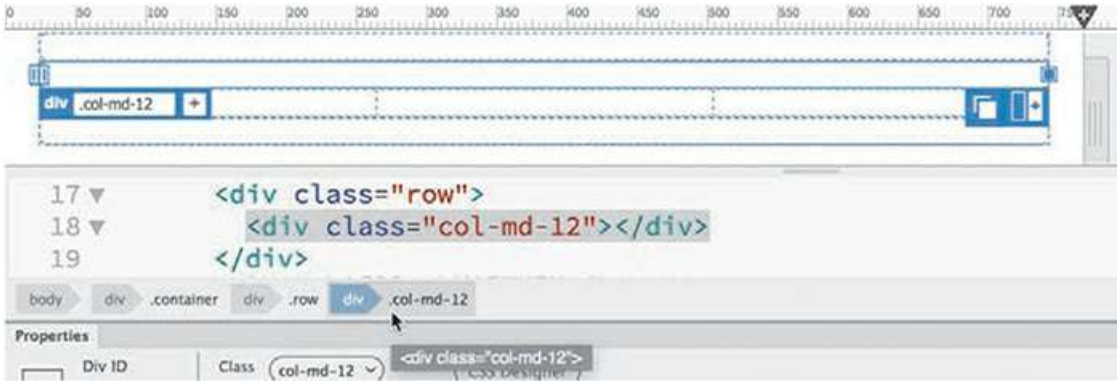
  Open **myBootstrap.html** from the lesson13 folder.

  The file has a basic Bootstrap structure containing four rows, three with one column and one with three columns.

- Select Split view so that the workspace displays the Code and Live view windows at the same time.

  The Bootstrap borders, rows, and columns should be visible in the Live view window as faint blue lines. Since we're going to add a dedicated navigation menu to the first row later, let's start on the second row.

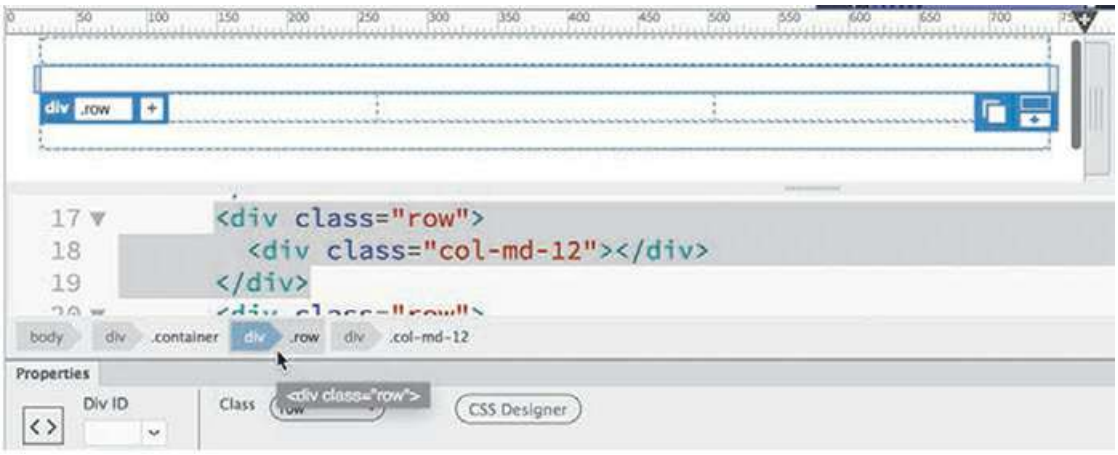
- Click in the second row of the layout and examine the tag selectors at the bottom of the document window.



One of the elements in the row is selected. The Element Display appears focused on one of the nested elements.

Depending on where or how you click, you might select the row itself or the column nested within it. You can determine which element is selected by looking at the class name displayed in either the Element Display or the tag selectors. If it displays `div.row`, it indicates you have selected a row, whereas `div.col-md-12` means you have a column selected. The selected element will be highlighted in the tag selectors interface.

. Click the tag selector for `div.row`.

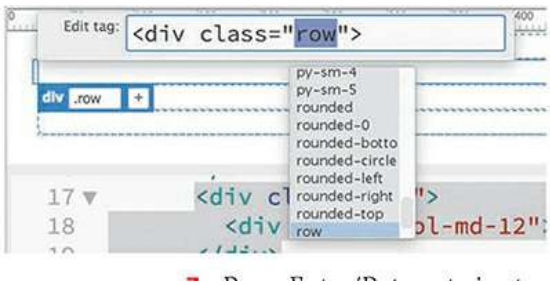


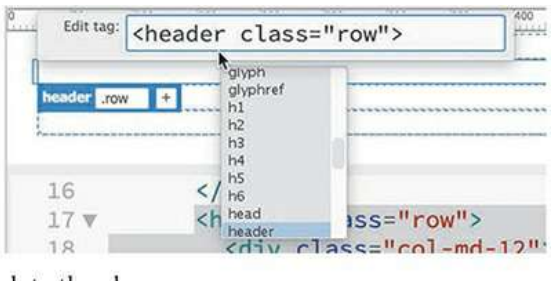
. Press Ctrl+T/Cmd+T to activate the Quick Tag Editor.

The Quick Tag Editor appears, populated by the code for the row element. As you might recall from the original page diagram, this element should be designated as an HTML5 `<header>`.

. Edit the element code as highlighted:

```
<header class="row">
```



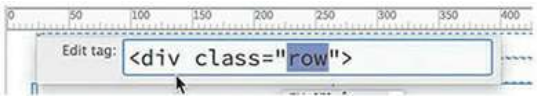


. Press Enter/Return twice to complete the change.

The structure is now updated to use the new header element.

. Click in the third row, then repeat steps 4 through 7 to edit the third row as highlighted:

```
<main class="row">
```



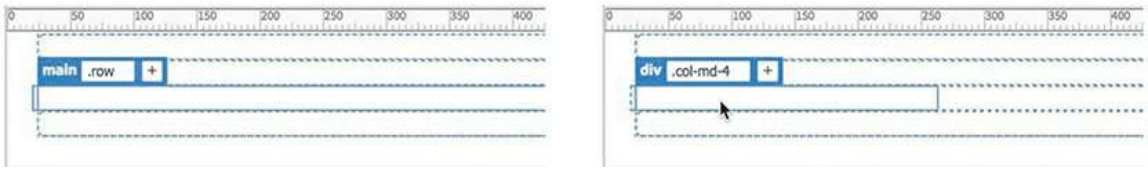
This row also contains three columns: one `article` element and two sidebars, or `aside` elements.

. If necessary, click the third row. The Element Display appears for main.row.

Tip

You can also use the DOM viewer to select the column element.

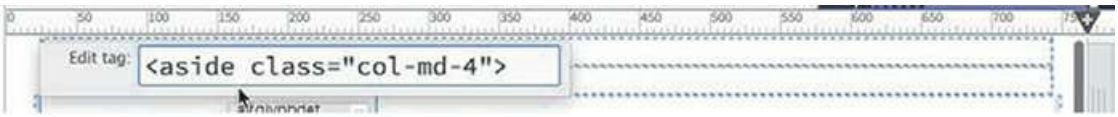
. Press the down arrow key once.



You can use the up and down arrows keys in Live view to change the selection focus on consecutive elements in the HTML code. The first `div.col-md-4` element in the row should be selected. We'll refer to this element as Sidebar 1 from this point on.

. Press Ctrl+T/Cmd+T.

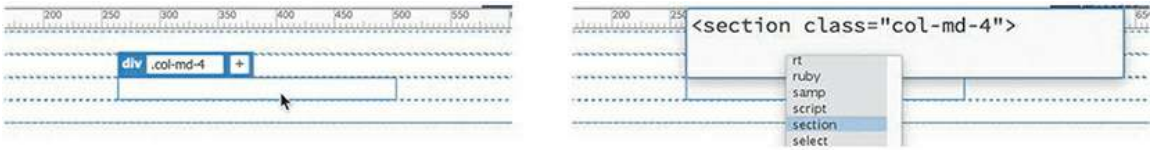
Edit the element as highlighted: `<aside class="col-md-4">`



This element will contain Sidebar 1.

. Click the third row and press the down arrow twice to select the second column `in main.row`. Edit the column element as highlighted:

`<section class="col-md-4">`



This element will contain the main content of each page.

. Edit the third column as highlighted: `<aside class="col-md-4">`

This element will contain Sidebar 2.

. Edit the fourth row as highlighted: `<footer class="row">`

. Save the file.

```
13 ▼    <div class="container">
14 ▼      <div class="row">
15          <div class="col-md-12"></div>
16        </div>
17 ▼      <header class="row">
18          <div class="col-md-12"></div>
19        </header>
20 ▼      <main class="row">
21          <aside class="col-md-4"></aside>
22          <section class="col-md-4"></section>
23          <aside class="col-md-4"></aside>
24        </main>
25 ▼      <footer class="row">
26          <div class="col-md-12"></div>
27        </footer>
28      </div>
```

The Bootstrap layout has now been updated to use HTML5 semantic elements. In the next online lesson, you'll learn how to convert this file into an alternate site tem-plate that will then be applied to the existing site pages, as well as to format various elements. See the "Getting Started" section at the beginning of the book for instruc-tions on how to obtain these lessons.

Note

Lesson 14, "Adapting Content to Responsive Design," is in your online resources. See the "Getting Started" section at the beginning of the book for specific instructions on how to obtain these resources.

Review questions

1. What is responsive design?
2. How are web frameworks used in responsive design?
3. What does the Visual Media Queries (VMQ) interface do?
4. What do the colors in the VMQ interface signify?
5. How does the scrubber work in conjunction with the VMQ interface?
6. Why should you consider using Bootstrap for your next website?
7. True or false: You have to use one of the six predefined templates if you want to use Bootstrap.
8. Why should you replace the `<div>` elements created by Bootstrap with HTML5

semantic elements?

Review answers

1. Responsive design is a method for designing webpages so that they automatically adapt to any size screen or device.
2. A web framework is a set of predefined HTML and CSS code, often including JavaScript, designed to support responsive design from the ground up and enable designers to quickly and easily build webpages and applications.
3. The VMQ interface provides a visual representation of the existing media queries in a file and allows you to create new media queries and interact with them in a point-andclick interface.
4. The colors displayed indicate whether the media query is defined with min-width specifications, max-width specifications, or a combination of both.
5. The scrubber allows you to quickly preview the page design at varying screen sizes to test the predefined media queries and pertinent styling.
6. Bootstrap and other web frameworks use predefined CSS and scripting to provide built-in support for multiple screen sizes and mobile devices.
7. False. The six templates provide a quick way to jumpstart a Bootstrap design, but you can create your own Bootstrap layout from scratch at any time in Dreamweaver.
8. The `<div>` element is a generic container that passes no semantic information to search engines or other applications. Semantic elements like header, nav, article, and aside help search engines identify what type of content they contain.

APPENDIX

TinyURLs

PAGE	TINYURL	FULL URL
Lesson 3		
92	tinyurl.com/shorten-CSS	developer.mozilla.org/en-US/docs/Web/CSS/Shorthanc
Lesson 8		
252	tinyurl.com/pew-broadband-report	www.pewinternet.org/fact-sheet/internet-broadband/
Lesson 11		
358	tinyurl.com/setup-coldfusion	www.adobe.com/devnet/archive/dreamweaver/articles/s PID=4166869
358	tinyurl.com/setup-apachephp	www.adobe.com/devnet/archive/dreamweaver/articles/s PID=4166869
358	tinyurl.com/setup-asp	www.adobe.com/devnet/archive/dreamweaver/articles/s PID=4166869
Lesson 16 Online		
online	tinyurl.com/fluid-width-animation	css-tricks.com/NetMag/FluidWidthVideo/Article-FluidWi
online	tinyurl.com/video-HTML5-1	www.w3schools.com/html/html5_video.asp
online	tinyurl.com/video-HTML5-2	www.808.dk/?code-html-5-video
online	tinyurl.com/video-HTML5-3	www.htmlgoodies.com/html5/client/how-to-embed-videc html5.html
online	tinyurl.com/fluid-video	ulrich.pogson.ch/complete-responsive-videos-breakdov
online	tinyurl.com/fluid-video-1	css-tricks.com/NetMag/FluidWidthVideo/Article-FluidWi

online	tinyurl.com/do-not-host-video	www.wp101.com/10-reasons-why-you-should-never-ho videos/
online	tinyurl.com/video-hosting-overview	www.koozai.com/blog/social-media/video-marketing/vid vs-self-hosting-videos/

INDEX

SYMBOLS

Numbers

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

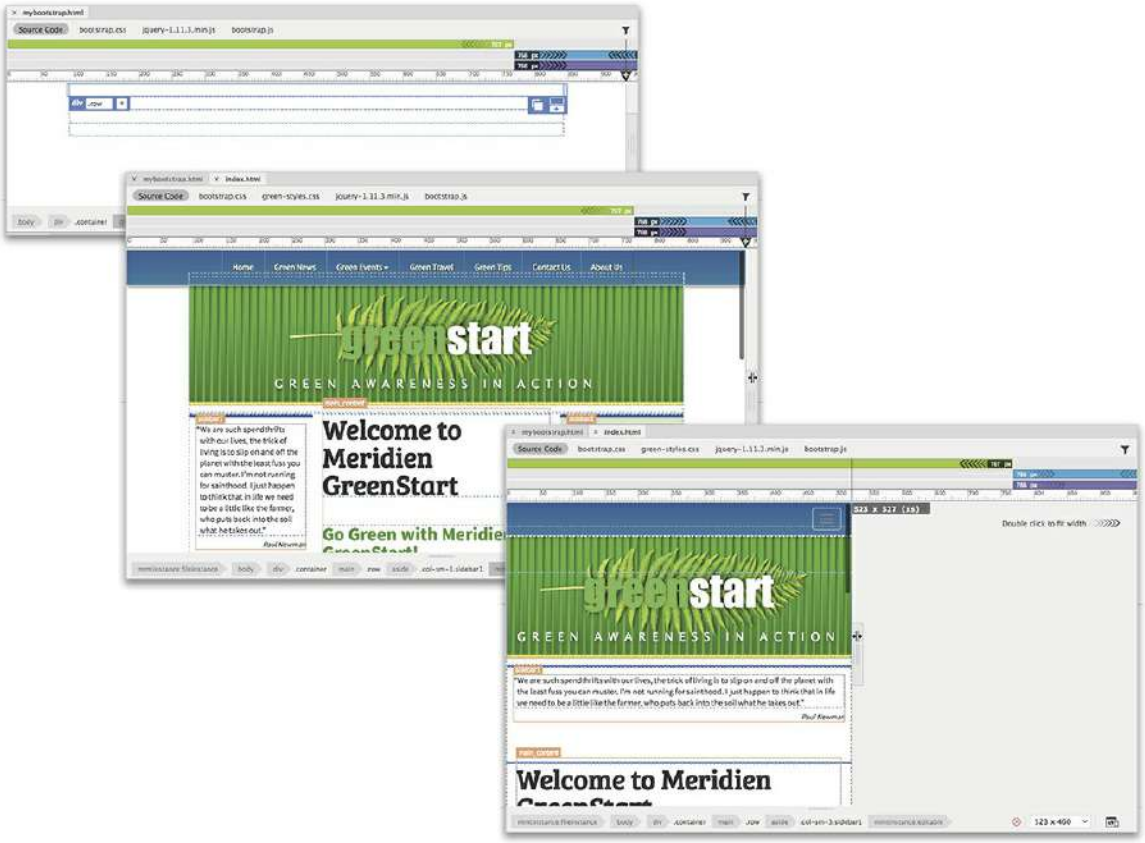
14 Working with a Web Framework

Lesson overview

In this lesson, you'll learn how to do the following:

- Add a Bootstrap navigation menu
- Edit and update a Bootstrap navigation menu
- Populate a Bootstrap layout with boilerplate and content placeholders
- Convert the Bootstrap layout to a Dreamweaver template
- Apply the Bootstrap template to the existing static site pages

This lesson will take about 3 hours to complete. If you have not already done so, please log in to your account on peachpit.com to download the project files for this lesson as described in the "Getting Started" section at the beginning of this book and follow the instructions under "Accessing the Lesson Files and Web Edition." Define a site based on the lesson14 folder.



Dreamweaver has incorporated many advanced functions and components from web frameworks such as jQuery and Bootstrap, speeding up and simplifying the process of developing fully functional, mobile-friendly websites—and you can take advantage of much of it without having to write a single line of code.

Creating a responsive site template

The Bootstrap-based layout you created in the previous lesson is fully responsive and will adapt automatically to any screen size or device. But at the moment that would be hard to prove because the layout is entirely devoid of content.

In this lesson, you will populate the various elements in the layout with the placeholder content that resides in the current GreenStart template. Once the layout matches the current design, you will save the file as the new responsive template for the site and then apply it to the existing pages in the site. In most cases, the existing content will adapt seamlessly to the new template. But some items will need custom CSS styling, will have to be physically modified, or will have to be fully replaced. The main navigation menu falls into the last category.

Building responsive menus

The original horizontal menu in the site template does not adapt automatically to different size

screens or devices. With a little custom CSS and some simple JavaScript, you could make the menu responsive. But the Bootstrap framework provides dozens of prebuilt widgets that provide fully responsive menus and other web components right out of the box. In this exercise, you will create a new responsive menu from one of the Bootstrap widgets.

- Open **mybootstrap.html** from the lesson14 folder in Live view.

  This document should be the same as the one you created in Lesson 13. It contains a three-column Bootstrap-based layout.

- Make sure the document window is displayed at a width of 1100 pixels or greater.

  Because the layout and all the components are designed to be responsive, the document window needs to be at least 1100 pixels to display all the elements properly.

Tip

If you are working with Dreamweaver on a smaller screen such as a laptop, to get the most out of the Dreamweaver interface you should consider using a second external monitor or display.

- Select the first row of the Bootstrap layout.

  The Element Display appears. Two elements compose the first row. It's important that you select the correct element when building the navigation menu.

- Select the `div.col-md-12` tag selector.

  This `<div>` element is inserted as a responsive Bootstrap element. The class `col-md-12` formats the element to occupy all 12 columns in the grid, or the entire width of the container. Since Bootstrap navbars are designed to be responsive, this element is redundant and may cause undesirable interference. You will replace this element with a Bootstrap navbar.

- Press Delete.

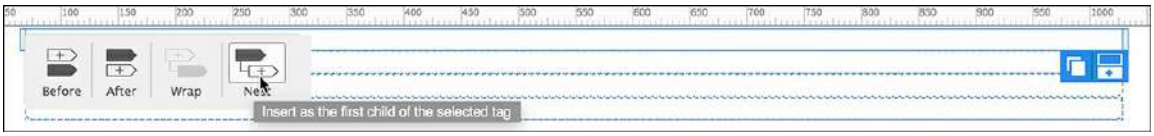
  The `<div.col-md-12>` is removed. When an element is deleted, Dreamweaver leaves the cursor at the position of that element. So this is the perfect time to insert the navbar.

- Display the Insert panel.

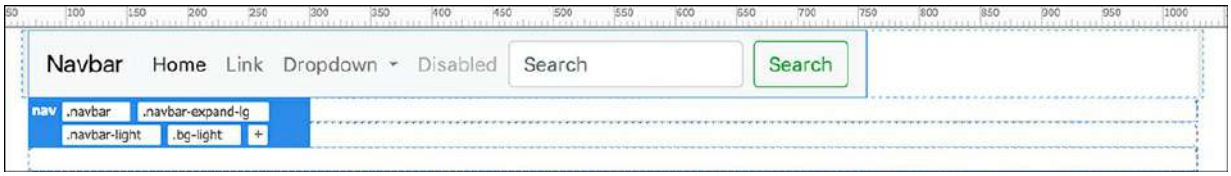
Select the Bootstrap Components category.

The Bootstrap Components category offers two types of navigational elements: complete navigational menu bars, or *navbars*, and standalone navigational menu components. In this situation, you'll use one of the complete navbars.

. In the navbar item dropdown menu, click **Basic Navbar**. The position-assist dialog appears.



. Click **Nest**.



A predefined Bootstrap navbar appears in the first row.

The navbar is composed of a navigation menu with four link placeholders, one of which has a dropdown component, a search field with a button, and a menu header. The GreenStart site design doesn't require all these items, so any element that's not needed should be deleted. The easiest way to select and delete HTML elements is via the tag selector interface. First, you'll remove the search field and button.

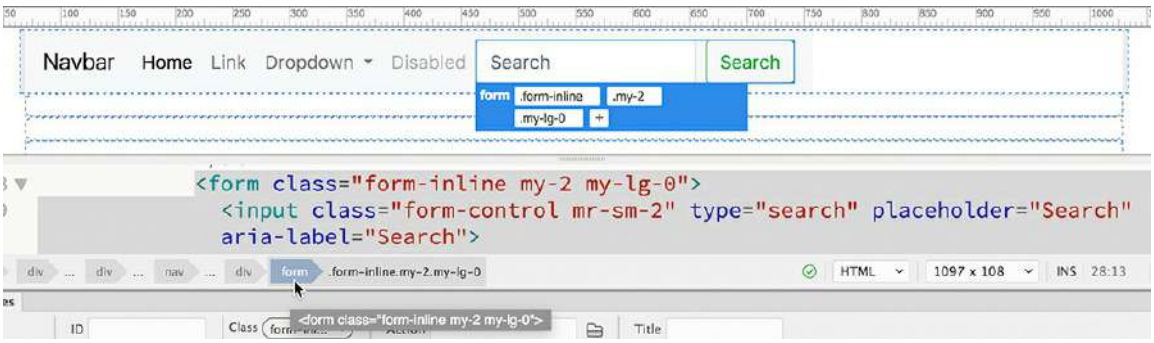
Tip

The DOM panel can also be used for deleting unneeded components.

. Click the search field or button.

The Element Display appears, identifying the selected element. If you examine the tag selectors, you should be able to track down the parent structure. It helps to know that search fields need to be inserted into an HTML `<form>` element.

. Select `form.form-inline.my-2.my-lg-0` in the tag selector interface.



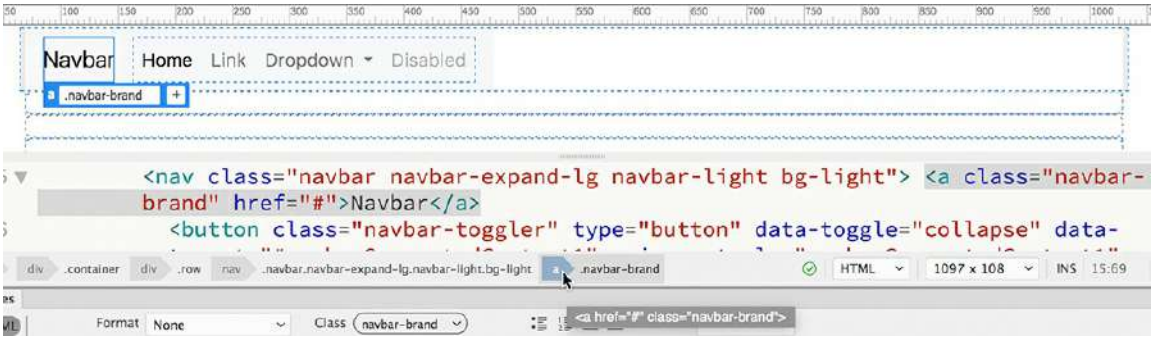
. Press Delete.

The search field and button are removed. The menu now contains four navigation links and one other item.

. Click the word *Navbar*.

If you examine the tag selector, you should see that the word "Navbar" is a standalone link with the class `.navbar-brand`. You could use the link for your company name to point back to your home page. Like the search box you deleted, it's unneeded in this layout. You need to remove the text as well as any link markup.

. Select the `a.navbar-brand` tag selector.



. Press Delete.

The Navbar link is removed. All the unneeded components have now been removed.

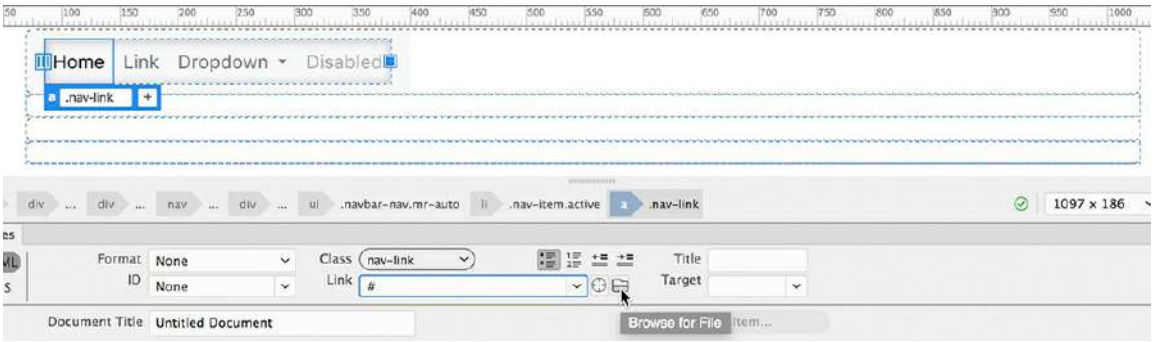
Note

When in editing mode, the text display may be quite distorted. Make sure the `<a>` element is still selected.

You may have noticed that the navbar features a dropdown menu. Although the current design doesn't have one, we will keep it because it provides an excellent way to target the two tables on the *Green Events* page.

The next step will be to replace the generic link placeholders with ones that match the current site design and content. The first link placeholder already says *Home*, which matches the existing GreenStart menu, so all we need to do is add a hyperlink to the appropriate page.

- Click the Home link. Click the `a` tag selector.
- In the Property inspector, click the Browse For File icon .



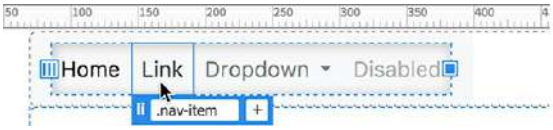
- Select **index.html** in the site root folder. Click Open.

The first menu item is complete.

- Double-click the next “Link” placeholder.

The orange text editing box appears.

- Type **Green News** to replace the Link text.



- Link the item to **news.html**.
- Double-click the text “Dropdown.”
- Type **Green Events** to replace the text “Dropdown.”

- Save the file.

The first two predefined links now match the menu items from the current GreenStart design, but you still need to edit the dropdown menu and create four additional items.

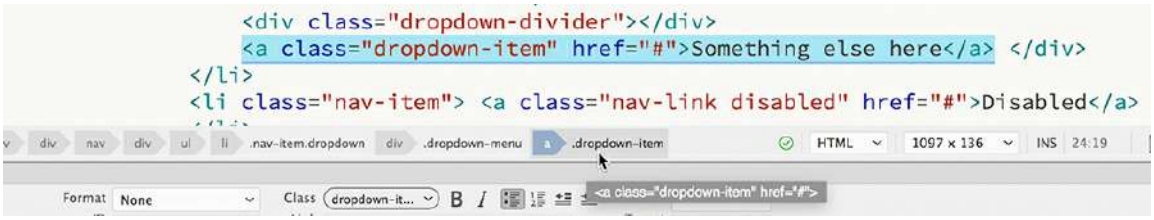
Editing a dropdown menu

Editing dynamic components, like the dropdown menu, can present some challenges in Live

view and Design view. Since the menu is designed to react to the cursor, it may be difficult to select or edit text in the various elements. With elements like this, Code view is often the preferred workspace.

- Switch to Split view, if necessary.
- Examine the structure of the dropdown menu in Code view.

  The dropdown menu is a complex element containing three separate sublinks as well as a divider component. Since you will need only two links to target the tables on the *Green Events* page, you can delete the unneeded markup.
- Select the `<a>` element containing the text *Something else here* (approximately line 32) in the dropdown menu, and press the Delete key.



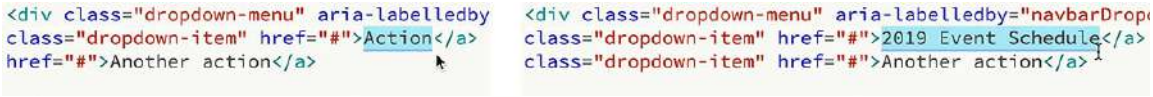
- Select and delete the element

  Click here to view code image

```
<div class="dropdown-divider"></div>
```

  Now only two links remain in the dropdown menu. You will link one to each of the tables on the page.
- Select the text *Action* in the first sublink.

  Type **`2019 Event Schedule`** to replace it.



  Code view provides an easy way to create links to local site files.
- Select the hash (`#`) symbol in the `href` attribute for the new link.
- Type **`events`**.

  As you type, Dreamweaver provides hinting for the local file structure. It will list any filename that matches the text you enter.

- When you see the name of the file you wish to link to, simply highlight the file with the keyboard or mouse and select it.
- Insert the cursor after the filename **events.html** and type **`#calendar`** to target the `id` attribute of the first table in the page.

Note

Make sure there is no space between the filename and the hash (#) symbol.

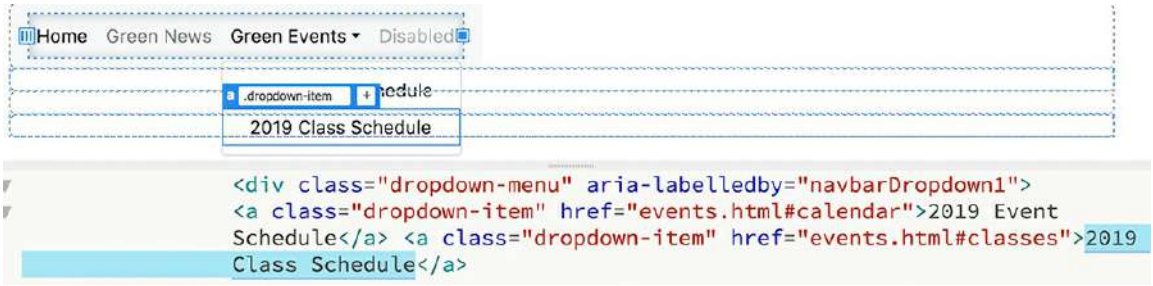
The sublink in the dropdown menu now targets the event schedule directly. You will use the remaining link to target the second table.

- In Code view, select the text *Another Action*.

  Type **`2019 Class Schedule`** to replace it.

- Select the hash symbol for the `link` attribute.

  Type **`events.html#classes`** to replace it.



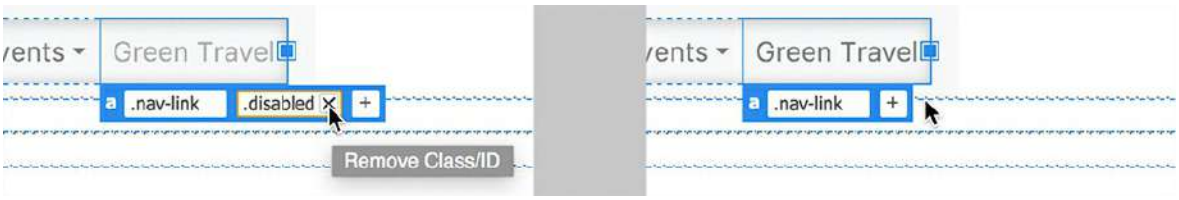
The Green Events menu item is complete. The menu has one last item left that needs to be edited, but you may notice that it is grayed out. That's a result of the class `.disabled`.

- Edit the text *Disabled* to say **Green Travel**.
- Point the link to **travel.html**.

  To remove the gray formatting, you need to remove the class `.disabled`.

- Select the link *Green Travel*.

  In the Element Display, click the Remove Class/ID icon to delete the `.disabled` class.



The *Green Travel* link now appears styled like the others. You have now completed the editing of the predefined link placeholders. Next, you will re-create the remaining three links from the original main menu.

Adding new items to a Bootstrap navigation menu

In Lesson 5, "Creating a Page Layout," you learned how to insert new items in a navigation menu built from an unordered list. Although the Bootstrap menu has a dropdown menu and is designed to be fully responsive, adding links to it can be done in the same manner.

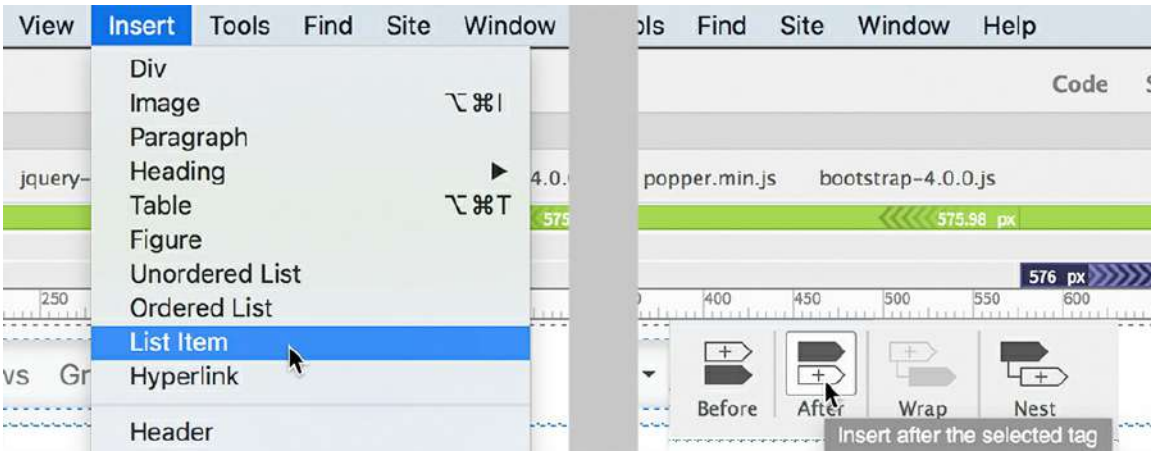
. In Live view, click the *Green Travel* link.

  Select the `li` tag selector.

. Choose Insert > **List Item**.

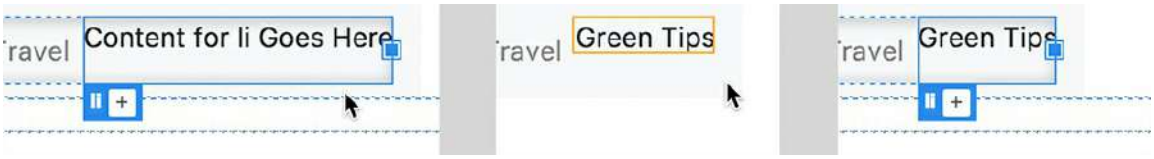
  The position-assist dialog appears.

. Click **After**.

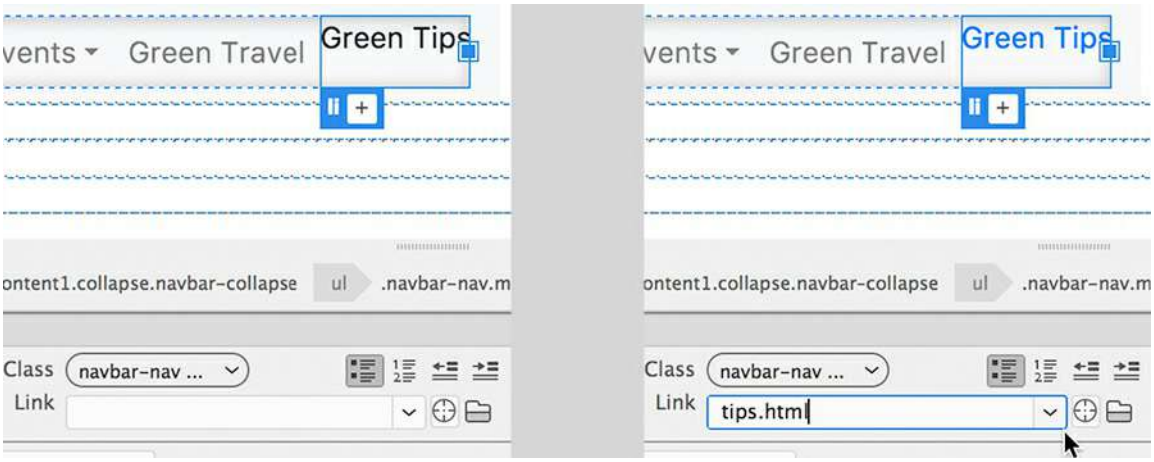


A new list item appears with placeholder text.

. Edit the placeholder text and type **Green Tips** to replace it.



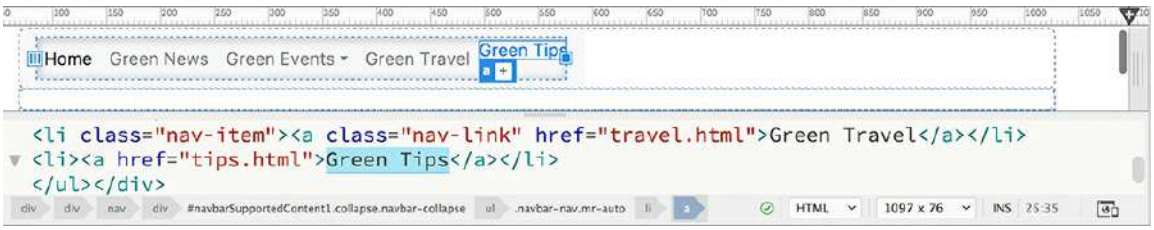
. Link the item to **tips.html**.



In Lesson 5, you learned that the menu items without hyperlinks were styled differently than those that had them. However, in this case, adding the hyperlink did not change the styling as expected. A closer look at the menu markup should provide the answer to the current differences.

. If necessary, switch to Split view.

Examine the markup of the Bootstrap menu.



The best way to identify the issue is simply to compare the last two menu items. You can quickly see that *Green Travel* has custom classes assigned to both the `<li>` and `<a>` elements. Let's add the same classes to the *Green Tips* menu item.

. If necessary, select the menu item *Green Tips*.

Select the `a` tag selector.

The Element Display appears around the `<a>` element.

. Click the Add Class/ID icon +.

. Type **`.nav-link`** and press Enter/Return to apply the class.

The styling for the link now looks identical to the other links, but don't stop now. You have one more class to apply.

. Select the `li` tag selector for *Green Tips*.

The Element Display appears around the `<li>` element.

. Click the Add Class/ID icon +.

. Type `.nav-item` and press Enter/Return to apply the class.

The new link is now structured identically to the others.

You added *Green Tips* using a menu option. New menu items can also be added by typing the code manually. You could type them one at a time, but don't forget you can use Emmet to supercharge any code-writing chores.

Using Emmet to create new menu items

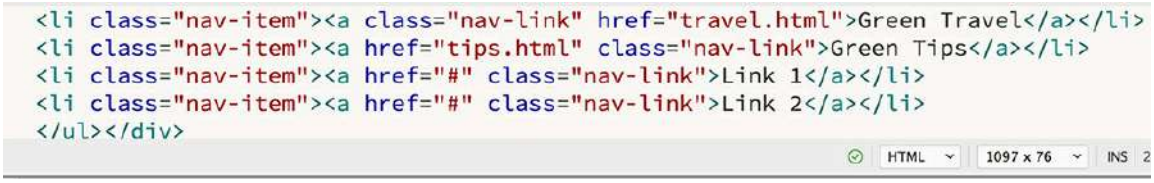
In this exercise, you will use Emmet to create the last two menu items.

. In Code view, insert the cursor after the closing `</li>` tag for the link *Green Tips*.

  Press Enter/Return to create a new line.

. Type `li.nav-item*2>a.nav-link[#]{Link $}`

. Press Tab.



  Two new list items with hyperlink placeholders are created automatically, along with the custom classes.

. Edit `Link 1` to say `Contact Us` and link it to **contact-us.html**.

. Edit `Link 2` to say `About Us` and link it to **aboutus.html**.

```
<li class="nav-item"><a class="nav-link" href="travel.html">Green Travel</a></li>
<li class="nav-item"><a href="tips.html" class="nav-link">Green Tips</a></li>
<li class="nav-item"><a href="contact-us.html" class="nav-link">Contact Us</a></li>
<li class="nav-item"><a href="about-us.html" class="nav-link">About Us</a></li>
</ul></div>
```

HTML 1097 x 76 INS

- Save the file.

Seven items are now in the menu. Before you format it, you'll have to correct some inconsistencies in its basic structure. These differences entail two class attributes: `active`, which modifies the styling of the *Home* link, and `sr-only`, which provides alternate text for screen reader devices. Let's discard both of them.

Cleaning up Bootstrap components

Your newly modified Bootstrap menu has some code attributes in the new menu that provide inconsistent styling. In this exercise, you'll clean up this markup.

- If necessary, open **mybootstrap.html**.

  Switch to Split view.

- Examine the code of the horizontal menu in the Code view window and the styling of the menu in Live view.

  The menu is constructed using an unordered list with seven list items. In Live view, you can see that the *Home* link is formatted differently from the rest. The most obvious difference between this link and the others is the class attribute: `.active`.

- Select and delete the attribute `active` from the *Home* item markup in Code view.

```
<ul class="navbar-nav mr-auto">
<li class="nav-item active"> <a c
only">(current)</span></a> </li>
<li class="nav-item"> <a class="n
<li class="nav-item dropdown"><a
```

```
<ul class="navbar-nav mr-auto">
<li class="nav-item"> <a class="n
(current)</span></a> </li>
<li class="nav-item"> <a class="n
<li class="nav-item dropdown"><a
```

  Once the class is deleted from the *Home* link, the formatting of the button will match the others. The last step is to remove the screen reader text.

- Select and delete the element `<span class="sr-only">(current)</span>` from the *Home* `<li>` markup.

```
<ul class="navbar-nav mr-auto">
<li class="nav-item"> <a class="nav-link" href="index.html">Home <span class="sr-only">
(current)</span></a> </li>
<li class="nav-item"> <a class="nav-link" href="index.html">Green News</a></li>
```

```
<ul class="navbar-nav mr-auto">
<li class="nav-item"> <a class="nav-link" href="index.html">Home</a> </li>
<li class="nav-item"> <a class="nav-link" href="index.html">Green News</a></li>
<li class="nav-item dropdown"><a class="nav-link dropdown-toggle" href="#"
```

Now all the differences between the menu items have been resolved.

- Save the file.

All the links in the menu are now formatted identically. It's common when using predefined or third-party components to have to modify the original structure and formatting to conform to your own content or project requirements.

Overriding Bootstrap styles

In the previous 13 lessons, you learned how to use the CSS Designer to create and edit CSS rules for a static website. There are aspects of the panel that you may not have seen or explored yet that will help you work with, and style, responsive components and webpages. It's vital to your role as a designer to understand and identify any existing structure and formatting of the layout so that you can effectively complete your tasks.

Tip

Not sure how wide your window is? The ruler should be visible at the top of the document window whenever Live view is selected.

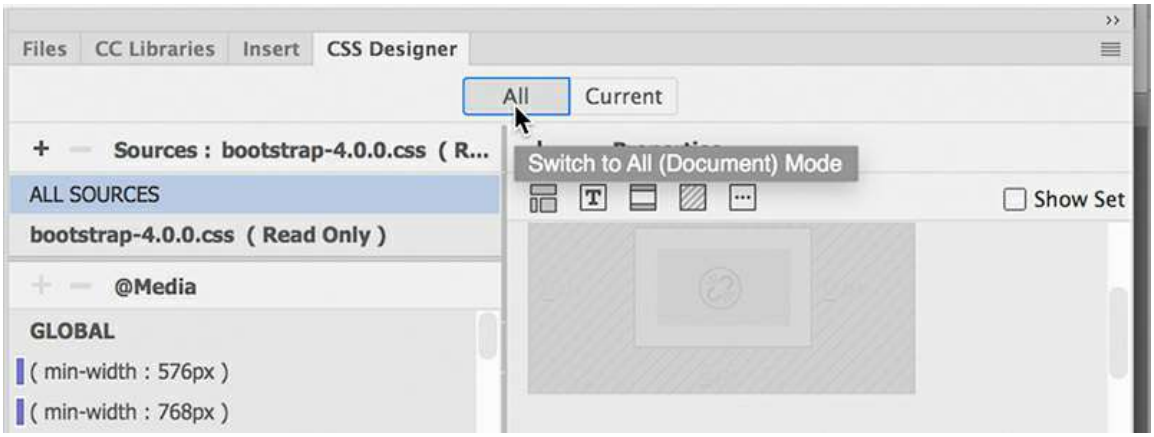
Tip

If the pane is collapsed, you can open it by clicking its name. You may also need to resize the individual panes to create a more effective display.

- Open **mybootstrap.html** from the lesson14 folder in Live view, if necessary.

  The document window should be set to a minimum width of 1100 pixels.
- Choose Window > CSS Designer to display it, if necessary.
- Click the All button in the CSS Designer.

  The Sources panel now displays all style sheets embedded or linked to the page. You should see two notations: ALL SOURCES and **bootstrap.css**.
- Select ALL SOURCES in the Sources panel.

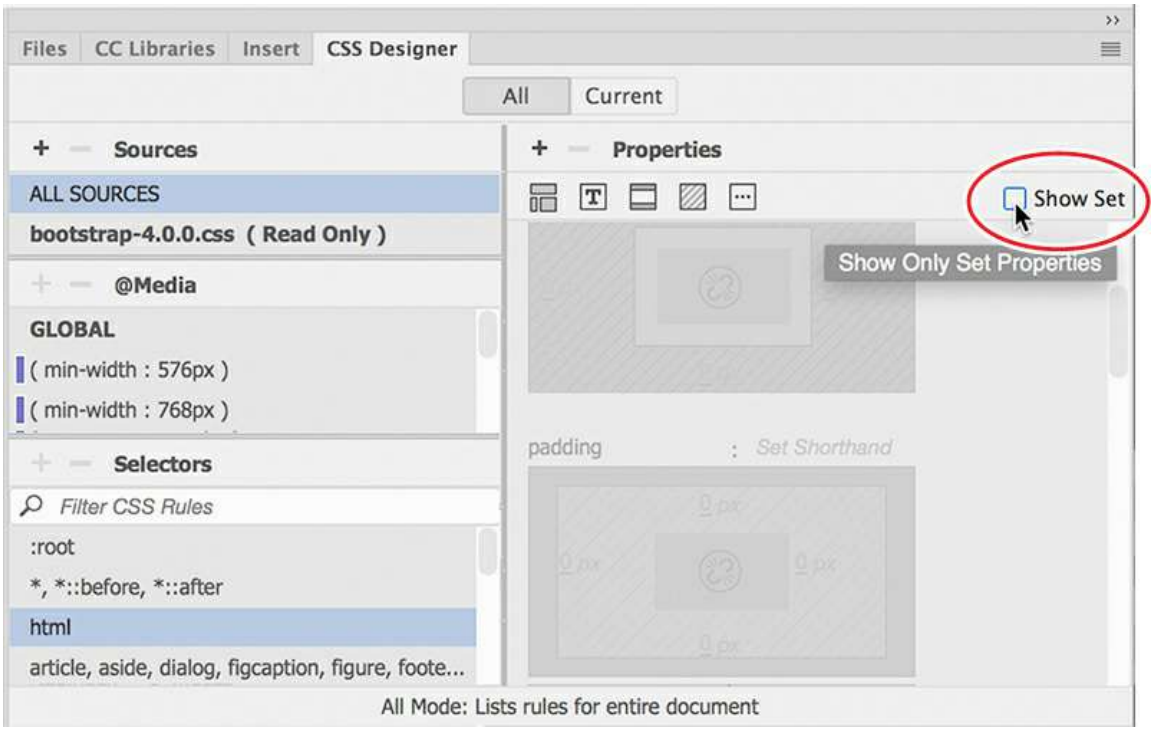


The @Media and Selectors panels display all the media queries and selector names defined in any listed style sheet. Notice that even though the current layout has only a single CSS source, it features almost 50 media queries and nearly 9000 lines of CSS code. This may sound like a lot, but modern responsive webpages are frequently styled by multiple style sheets and media queries just like this.

With the exception of the jQuery accordion used in Lesson 10, "Adding Interactivity," you have had to contend with only global CSS rules: rules that apply universally to the entire page and all screen types. But once you enter the world of Bootstrap and responsive web design, one of the hardest tasks you will encounter is often just tracking down what one specific rule does and where it is defined. Luckily, whenever you select an item in any panel, CSS Designer will identify its location by highlighting its source in bold.

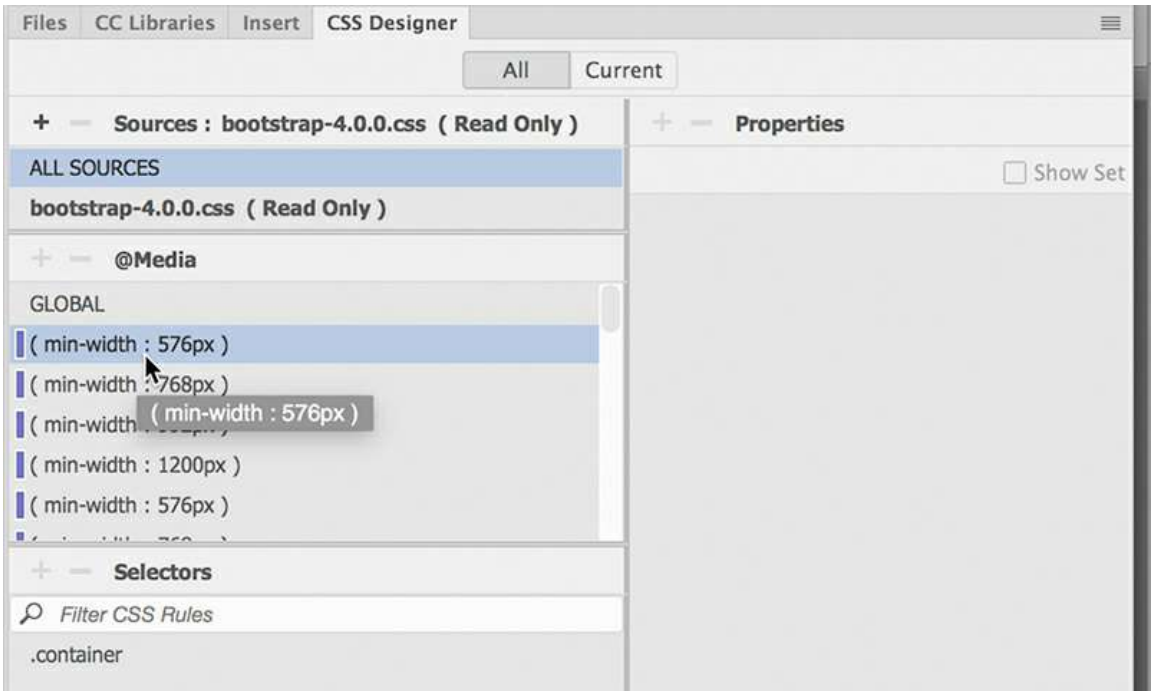
. Select the rule `html` in the Selectors panel.

In the Properties panel, deselect the Show Set option, if necessary.



Note that the name **bootstrap.css** in the Sources panel and GLOBAL in the @Media pane are now bolded. The bolding indicates that the selected rule is defined in the **bootstrap.css** style sheet as a global rule. This behavior works even in the other panels.

- Select the first `(min-width : 576px)` media query in the @Media window. Observe the changes in the CSS Designer display.



Note that **bootstrap.css** is still selected, but the Selectors panel now shows only one name: `.container`. This indicates that only one rule is defined within the selected media query.

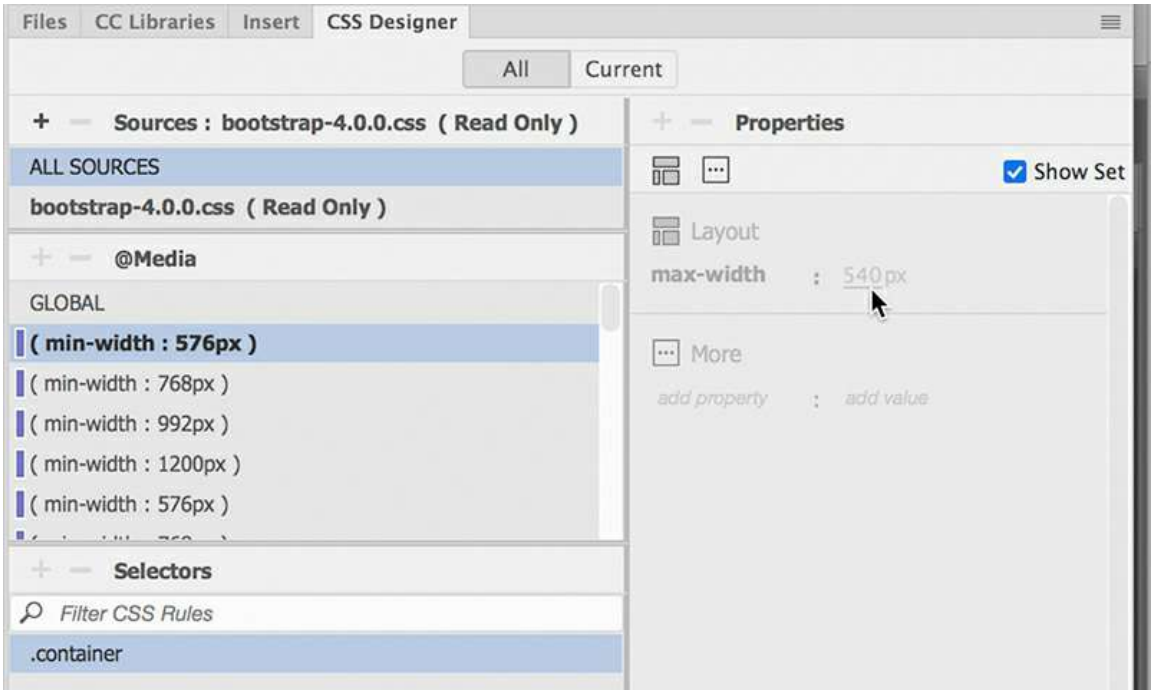
- Select the `.container` rule in the Selectors panel.

The Properties panel displays the CSS properties defined in the rule. Depending on its configuration, you may not see which property or properties are set.

Tip

Some users have reported that the Bootstrap style sheet does not display the Read Only label. This does not change the warning to avoid editing this style sheet.

- If necessary, select the Show Set option in the Properties panel.



When Show Set is enabled, the Properties panel displays only the properties modified by the rule. In this case, the `.container` rule sets the `max-width` property to 540 px.

You may also notice that the Properties panel and the settings are grayed out. This indicates that the properties are not editable. If you look at the Sources panel, you should see that the **bootstrap.css** style sheet is marked as `(Read Only)`.

Since the page is based on the Bootstrap framework, Dreamweaver prevents you from modifying and potentially damaging its predefined styling. The styling contained within it is complex and full of interdependencies. It's recommended that any changes or overrides be made in your own custom style sheet.

At the moment, there is no custom style sheet. Before you can style the structure or content of the new page and site, you'll have to add a new editable style sheet. You can create the style sheet directly in the CSS Designer.

Reading, no writing

The Bootstrap style sheet is formatted as a read-only file to prevent you from making accidental changes to the framework's complex styling. From time to time as you work in your pages, a warning message may appear at the top of the screen indicating that the file is read-only and prompting you to make it *writable*.

You can dismiss the message by clicking the Close icon on the right side. It also provides an option to make the file writable. You're advised to resist the temptation.

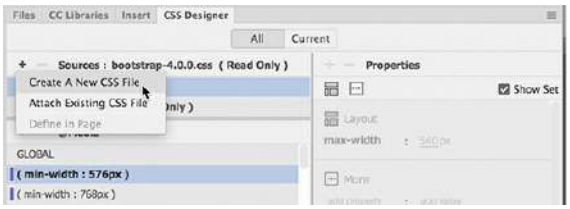
- Click the Add CSS Source icon in the CSS Designer.

  A dropdown menu appears that allows you to create a new CSS file, attach an existing CSS file, or define a style sheet embedded within the page code.

- Choose **Create A New CSS File** from the dropdown menu.

  The Create A New CSS File dialog appears.

- Type **green-styles.css** in the Create A New CSS File dialog. Click OK to create the style sheet reference.



Tip

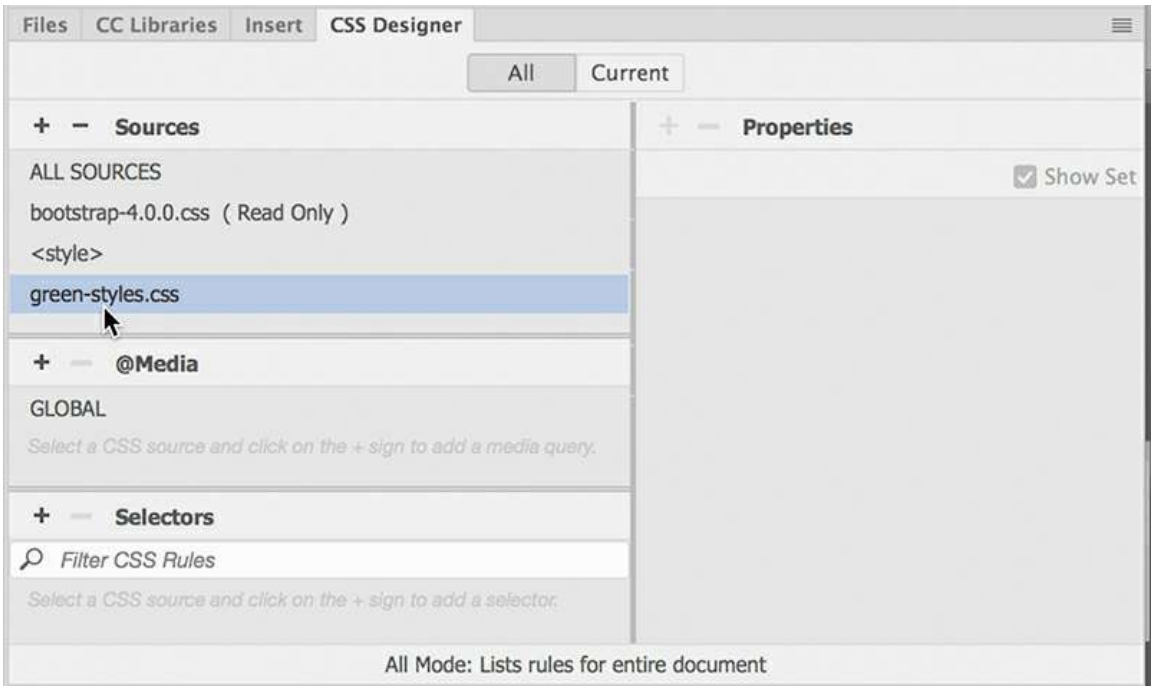
It is essential to note here that the CSS file has not been created yet, and will not be until you create a CSS rule and save the file. If Dreamweaver crashes before that happens, you will have to create the file in a separate operation.

When you click OK, a reference to the new style sheet is added to the CSS Designer Sources panel. The CSS file has not actually been created yet, but a link has been added to the `<head>` section of the page, and the file will be created automatically as soon as you create your first

custom rule and save the file.

Be sure you don't name the new file **mygreen-styles.css**. That is the name of the style sheet formatting the existing static pages for the GreenStart site. You will be able to reuse many of the specifications in this file, so you don't want to accidentally delete or overwrite it.

. Click **green-styles.css** in the Sources panel.



The @Media and Selectors panes are both empty. This means there are no CSS rules or media queries defined in this file. You have a blank slate on which you can make any design additions or modifications. Since you will not change the Bootstrap CSS directly, this style sheet will be the means you use to make its structure and content bend to your wishes.

. Select File > Save all.

The changes to the layout and the new style sheet are saved.

Creating a responsive template

Once you create your new style sheet, you can start formatting the Bootstrap layout. In the following exercises, you will transfer both the placeholder content and the relevant styling from the current static GreenStart template to the new responsive layout.

. If necessary, open **mybootstrap.html** in Live view.

. Open **mygreen_temp.dwt** in the Templates folder.

The template is the basis of the current GreenStart site design. Most of the styling could be obtained from any of the site pages, but because the editable regions are locked, you will have

full access only to elements inside the template itself.

Styling the background of a Bootstrap navbar

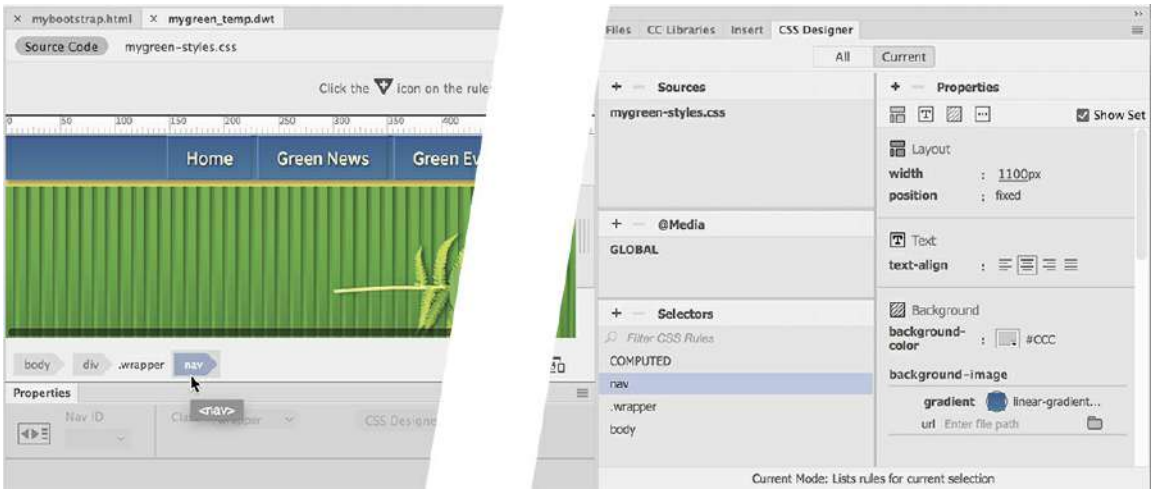
Let’s work down from the top of the page. The horizontal navigation menu appears at the top of the template. We won’t need the old menu, but we can definitely grab the styling. The menu is composed of several layered elements, each of which contributes to the overall appearance. The first step is to pull the background styles.

- In **mygreen_temp.dwt**, click the *Green News* menu item.

  You may not see the Element Display in the template, but the tag selectors should reflect your selection.

- Select the `nav` tag selector.

  In the CSS Designer, click the Current button.



  The Selectors panel displays the CSS rules affecting the element. In the Properties panel, you can see the settings for the background gradient. Once you identify the source of the styling, it’s a simple matter to transfer those specifications to the new layout.

- Right-click the `nav` selector in the CSS Designer.

  Select Copy Styles > Copy Background Styles from the context menu.

. Switch to **mybootstrap.html**.

Applying the background styles in the new layout is a two-step process. First, you have to identify any rules currently formatting the same element in the new file. Then, you have to create a new rule in the custom style sheet to apply the specifications.

. Click *Green News* in the responsive menu.

The Element Display appears focused on the `li` or `a` tag. The Bootstrap file is not a template yet, so it may respond differently. Examine the tag selectors.

The background effect was applied to the `nav` element in the other file, so it makes sense to find a similar element or an element serving the same purpose. In this case, the responsive menu is also using a `nav` element for its basic structure.

. Select the `nav` tag selector.

The Element Display appears focused on

Click here to view code image

```
nav.navbar.navbar-expand-lg.navbar-light.bg-light.
```

Note

The CSS Designer reacts to the size of the document window and position of the scrubber. The window must be 1100 pixels or wider to see the proper display of rules.

Since the Current button is still active, the Selectors panel should be displaying the CSS rules pertaining to the `nav` element. Pay close attention to the syntax of the selectors. The new rules should match or exceed the specificity of any existing rule. Since the Bootstrap CSS file is locked, you will create the new rule in **green-styles.css**.

Conflicts and crisis

You may be wondering why you aren't just linking the old GreenStart style sheet to the new responsive layout. Wouldn't that be faster than transferring each rule one at a time?

Yes and no.

Yes, it would be faster. But there's no telling what conflicts you'd be creating by simply dumping the old rules into the new layout. It might be impossible to identify and troubleshoot all the issues that could arise.

By adding the rules and specifications one by one, you can instantly see any problems and address them immediately. It might take a little longer, but the result will be more predictable and controllable.

. Click the All button.

You may find that **green-styles.css** is selected by default when you click the All button, but it is essential that all new rules are added only to this file and to the proper media queries defined within it, as needed. It's very easy to add rules and specifications to the wrong place, and the results can be tragic. Always be aware of where your new rules are being inserted.

. Select **green-styles.css** in the Sources panel.

Click the Add Selector icon  .

A new selector appears.

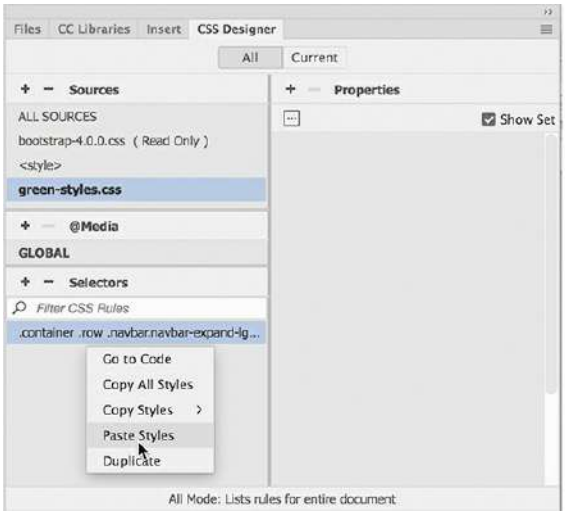
. Press the up or down arrow to create the following selector:

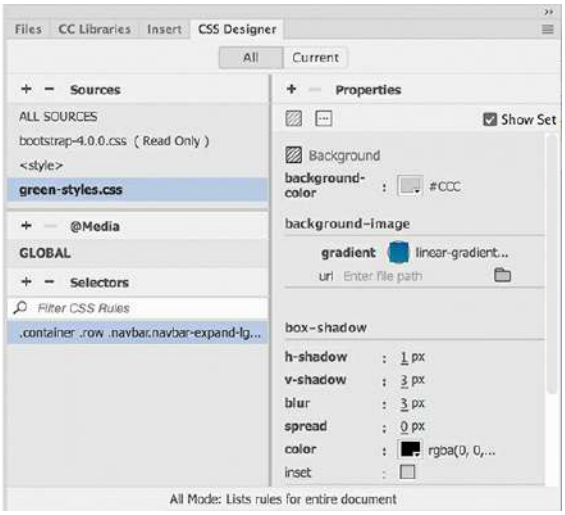
Click here to view code image

```
nav.navbar.navbar-expand-lg.navbar-light.bg-light.
```

. Right-click the new selector.

Select Paste Styles from the context menu.





The responsive menu now features the same gradient background as from the static page. Next, you'll bring over the text styling.

Styling the text in a Bootstrap navbar

In this exercise, you will bring over the text styling of the navigation menu.

- Switch to **mygreen_temp.dwt**.
- Click the menu item *Green News*.

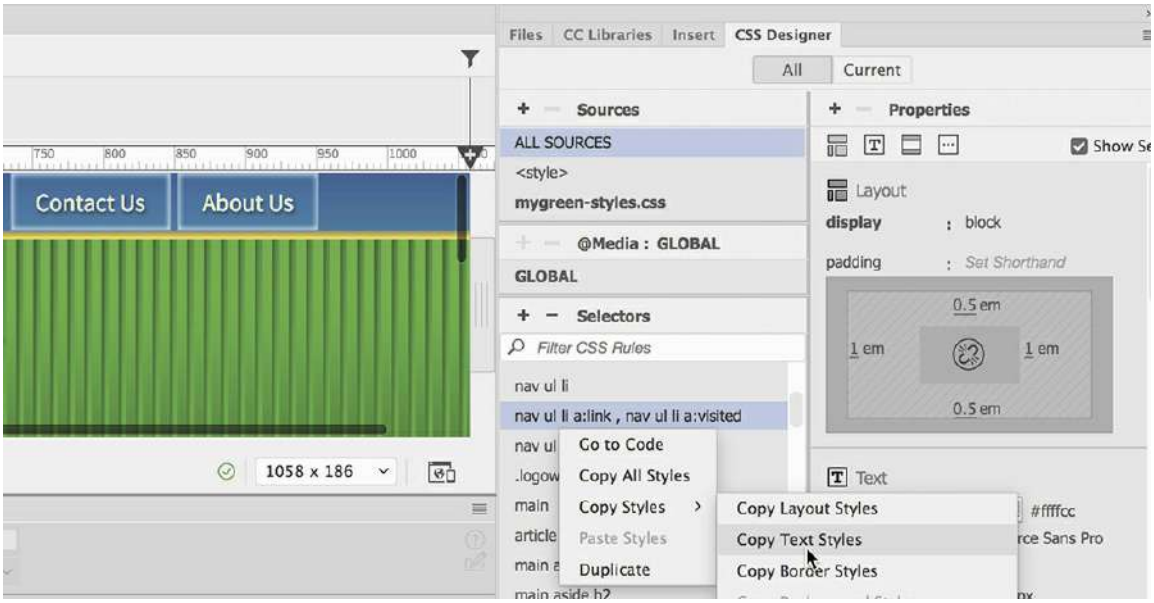
  Click the Current button in the CSS Designer.

  The text styling for the horizontal menu is applied via rules targeting the `a:link`, `a:visited`, and `a:hover` pseudoclasses. You will need to bring over the styling for all these states to complete the basic menu.
- Right-click the following rule:

  Click here to view code image

```
nav ul li a:link , nav ul li a:visited
```

- Select Copy Styles > Copy Text Styles from the context menu.

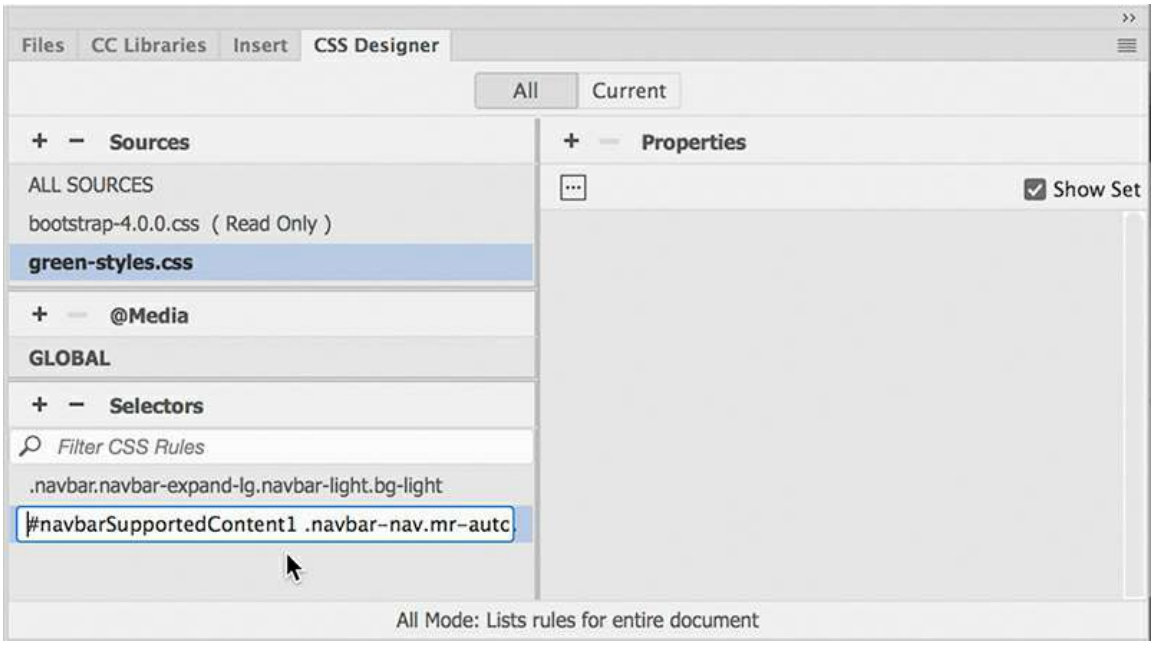


- Switch to **mybootstrap.html**.
- Select Green News in the horizontal menu.

  Click the `a` tag selector, if necessary.
- Click the All button in the CSS Designer.

  Select **green-styles.css** in the Sources panel.
- Click the Add Selector icon ➕.

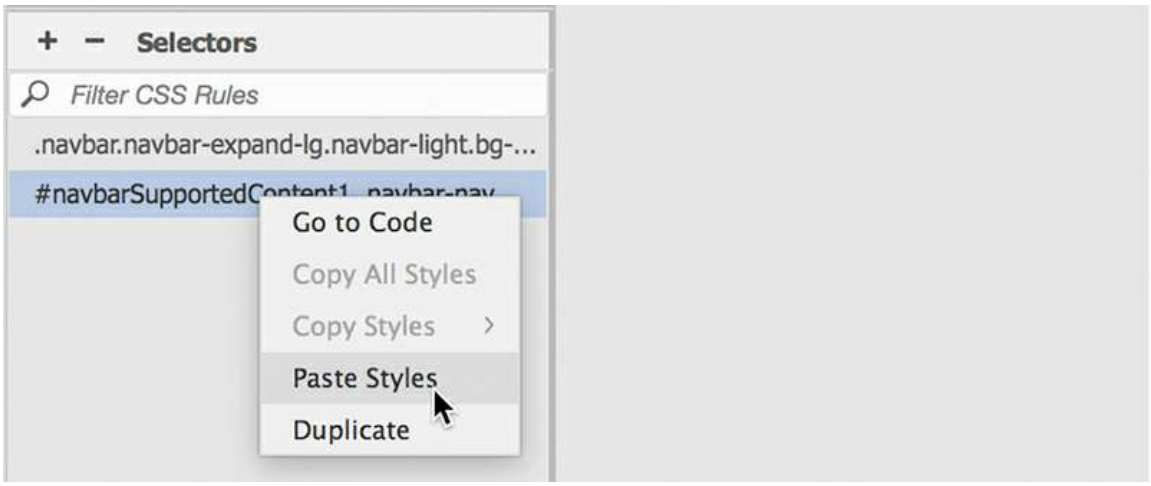
  An automatic selector name appears in the Selector pane.
- Press the down arrow as needed to create the following selector: `#navbarSupportedContent1 .navbar-nav.mr-auto .nav-item .nav-link`



The selection will have to style both the default link state and the visited state.

. Press Enter/Return to complete the selector.

. Right-click the new rule and select Paste Styles from the context menu.



The new rule styles the default state of the text. But when a hyperlink appears in a menu like this, most designers want to style the `a:visited` state the same way.

. Double-click the new rule to edit it.

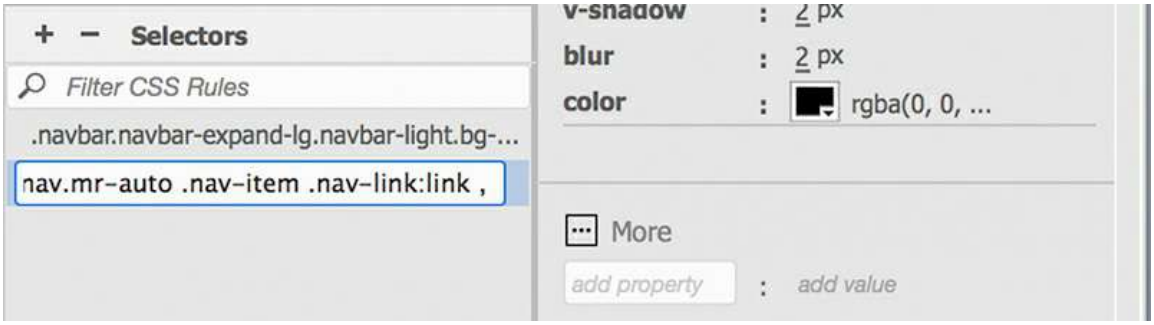
. Select the entire selector name and copy it.

Note

Don't forget the comma (`,`). The rule will not work without it.

- Insert the cursor at the end of the selector.

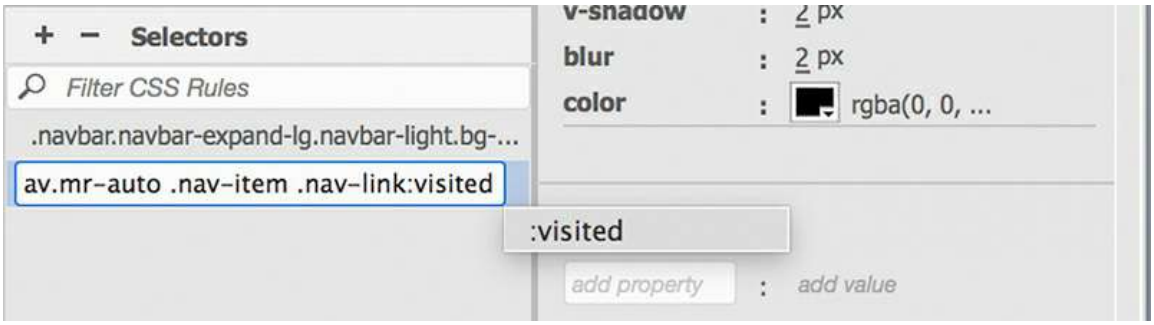
  Type `:link,` and press Ctrl+V/Cmd+V.



  A copy of the selector name appears after the comma.

- Insert the cursor at the end of the selector.

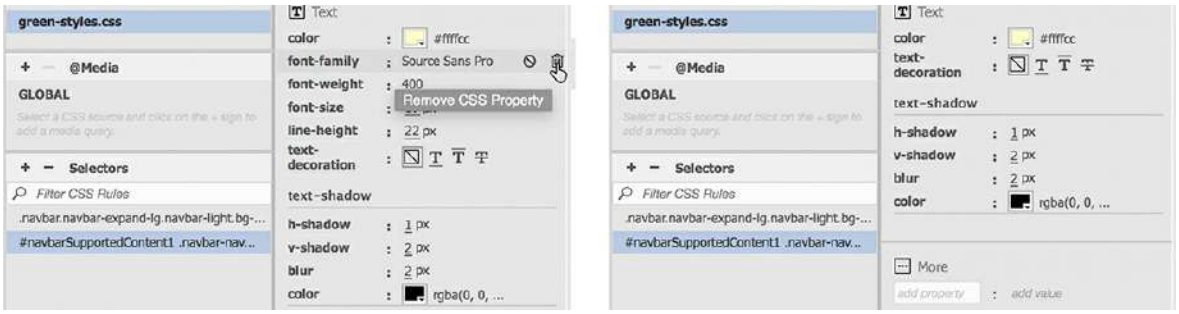
  Type `:visited` and press Enter/Return to complete the selector.



  The text styling was extracted from the Photoshop mockup in Lesson 5 and uses formatting most web designers try to avoid. It's best to deal with this issue while you are working with the pertinent rule.

- Delete the following properties from the rule:

```
font-family: Source Sans Pro
font-weight: 400
font-size: 17px
line-height: 22px
```



Later in this lesson, you will learn how to properly build site-wide font styling.

But first, let's continue working on the menu items by grabbing the border styling.

Styling borders on a Bootstrap navbar

The borders on the menu items help distinguish one item from another. In this exercise, you will transfer the border styling from the GreenStart template.

. Switch to **mygreen_temp.dwt**.

The borders are defined in the same rule that styles the text.

. In the CSS Designer, right-click the following rule:

Click here to view code image

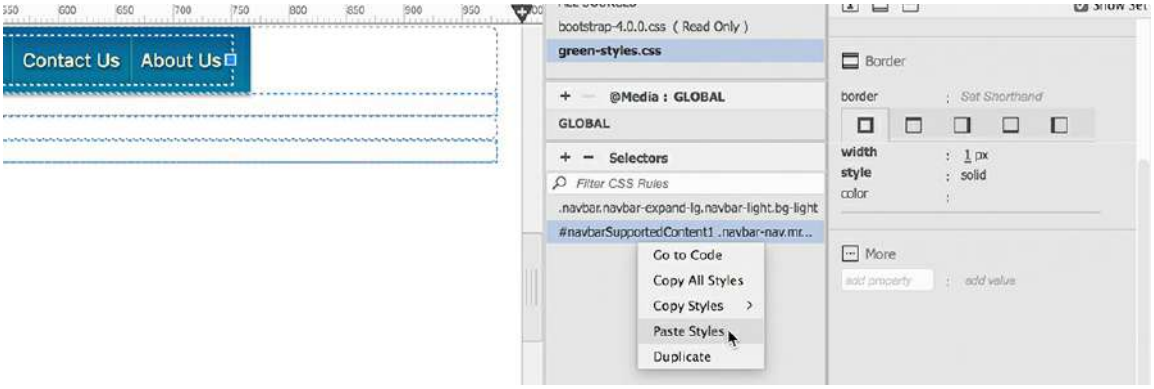
```
nav ul li a:link , nav ul li a:visited.
```

. Select Copy Styles > Copy Border Styles from the context menu.

. Switch to **mybootstrap.html**.

. Right-click the rule you created in the previous exercise.

. Select Paste Styles from the context menu.



The menu items now display border styling that gives them a 3D visual effect.

. Save all files.

This completes the default styling for the menu items. Next, you will style the `a:hover` state.

Styling interactive Bootstrap menu effects

The `a:hover` state should always follow the default and visited state in the style sheet.

. In CSS Designer, select the following rule:

Click here to view code image

```
#navbarSupportedContent1 .navbar-nav.mr-auto .nav-item
.nav-link:link , #navbarSupportedContent1 .navbar-nav.mr-auto
.nav-item .nav-link:visited
```

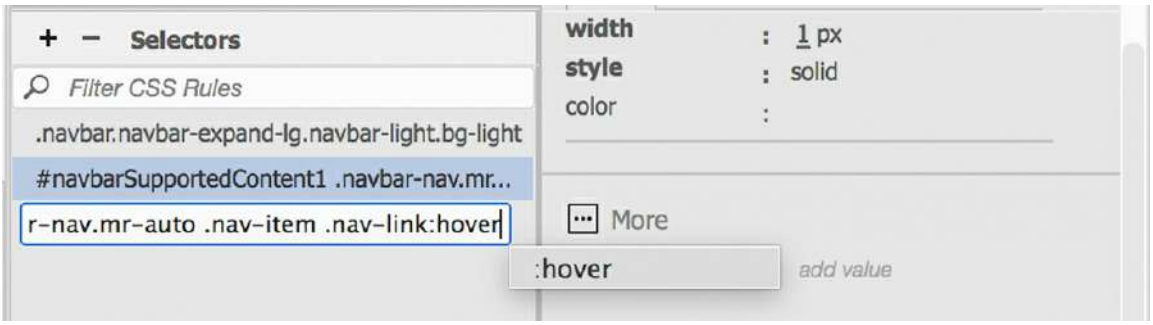
- Click the Add Selector icon ✚.

  A new selector is inserted after the selected rule. It is essential that the new selector appear after the `:link` and `:visited` rule.

- Create the following selector:

  Click here to view code image

```
#navbarSupportedContent1 .navbar-nav.mr-auto .nav-item
.nav-link:hover
```



- Create the following property in the new rule:

```
color: #FFF
```

  This property changes the color of the link text to white when the cursor interacts with the menu item.

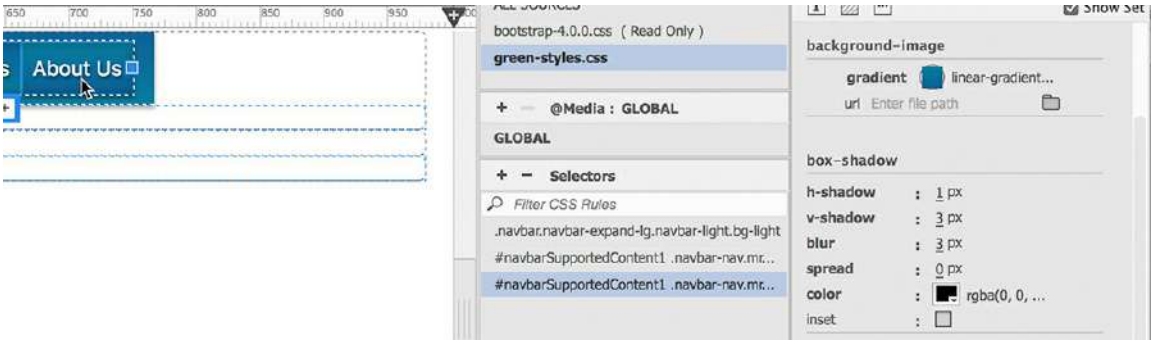
- Position the cursor over any menu item.

  The link text changes to white. In the original styling, the hover state also applied a contrasting gradient background. You can also grab that styling from the GreenStart template.

- Switch to **mygreen_temp.dwt**.
- Copy the background styles from the following rule:

  `nav ul li a:hover`.

- Switch to **mybootstrap.html**.
- Paste the styles on the `:hover` rule created in step 3.
- Position the cursor over any menu item.

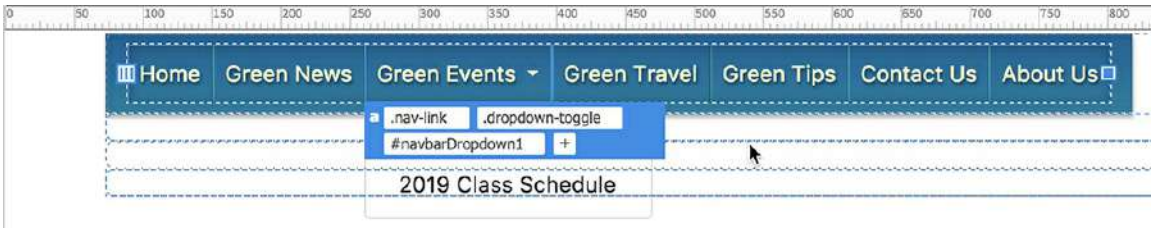


The background of the menu item reverses as the text turns white. The styling is complete in the regular part of the menu, but all is not well. There is a problem lurking in the dropdown component.

. Click the *Green Events* dropdown menu to display the sublinks.

The submenu opens. The links in the dropdown menu are still using the default Bootstrap styling. Notice that the text is formatted in black.

. Move the cursor away from the *Green Events* menu item.



The dropdown menu remains open. It's clear that you will need some additional styles to adapt the sublinks to the GreenStart site scheme.

. Save all files.

In the next exercise, you will identify the pertinent styles for the dropdown menu and override them.

Styling a Bootstrap dropdown menu

The current styling for the horizontal menu controls all the regular menu items and their rollover effects, but it doesn't cover the items within the dropdown menu. In this exercise, you will conform the submenu to the site design scheme.

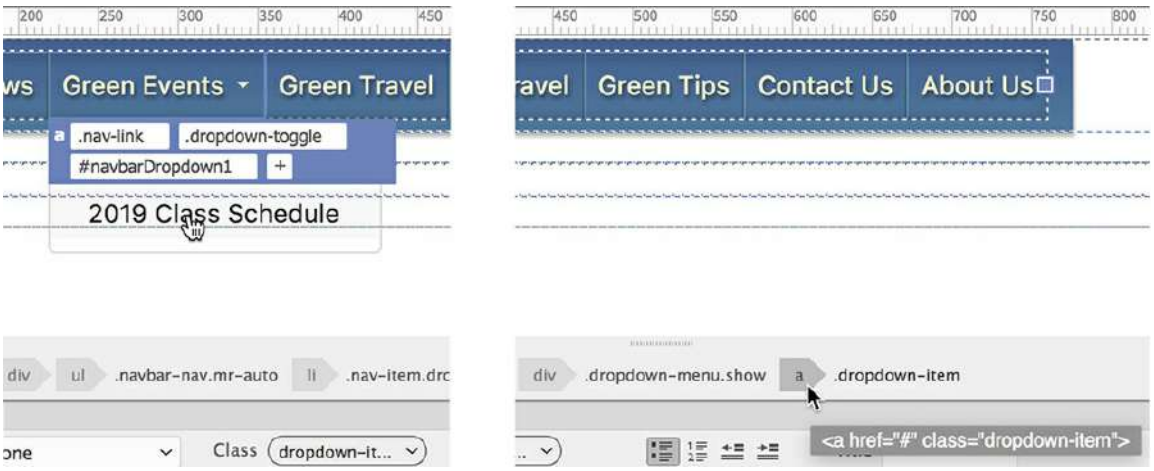
. Open **mybootstrap.html** in Live view, if necessary.

. Click the Current button in the CSS Designer.

. Click to open the *Green Events* submenu, if necessary.

Click the subitem *2019 Class Schedule*.

Examine the tag selectors.



When you click the dropdown menu it will close, but the tag selectors should change to show the structure of the item. Notice that the menu subitem link has the class `.dropdown-item` assigned to it. CSS Designer displays the rules that are styling the open menu. At the moment, all the rules styling the menu item are in the Bootstrap style sheet.

- In the CSS Designer, select the All button.

  Select **green-styles.css**.

  Select the last rule displayed.

- Click the Add Selector icon ✚.

  Press the up arrow to create the following selector:

  ```
  .dropdown-menu .dropdown-item
  ```

- Press Ctrl+A/Cmd+A to select the entire selector.
- Press Ctrl+C/Cmd+C to copy the selector.
- Modify the selector as highlighted:

  Click here to view code image

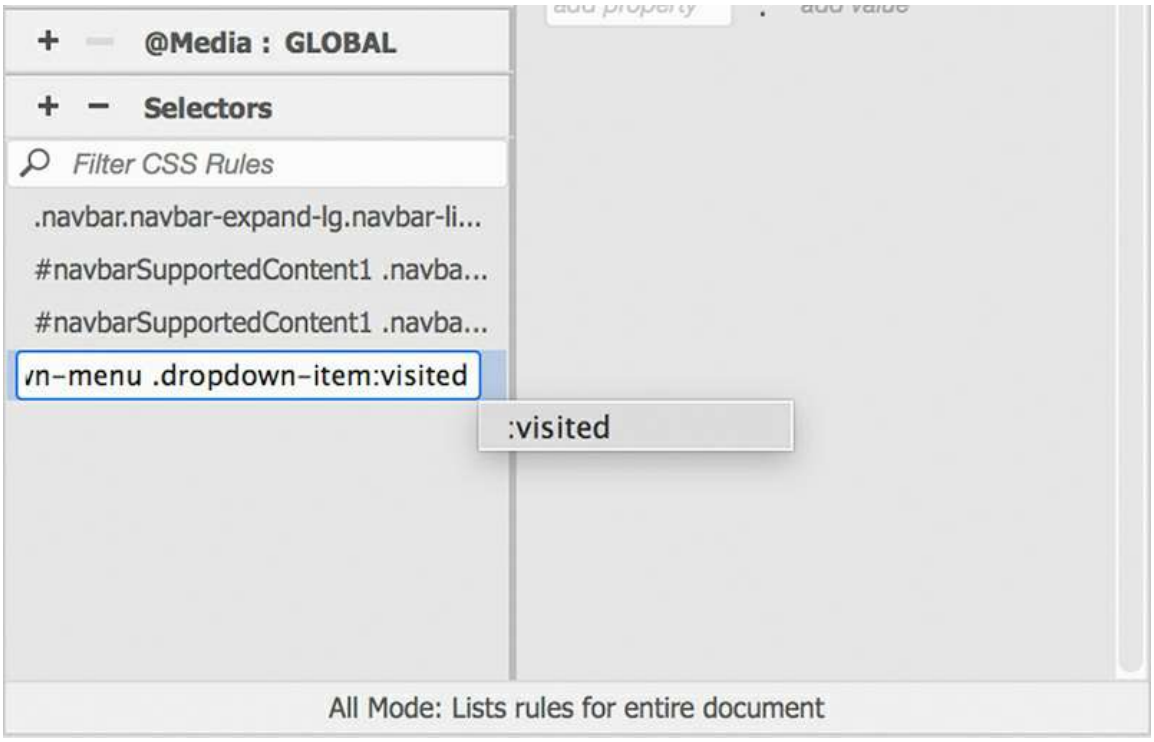
  ```
  .dropdown-menu .dropdown-item:link ,
  ```

  This selector will style the default link state.

- Press Ctrl+V/Cmd+V to paste the selector again.
- Modify the selector as highlighted:

  Click here to view code image

  ```
  .dropdown-menu .dropdown-item:link ,
  .dropdown-menu .dropdown-item:visited
  ```



The changes will now style the default and visited states of the link.

- Press Enter/Return to complete the selector.

The dropdown menu should be styled identically to the regular menu. You can grab the styling from the existing rules. Since the selector name is still in memory, this is a good time to create the selector for the `hover` state.

- Click the Add Selector icon ✚.

Press Ctrl+A/Cmd+A, and then paste to insert the new selector name.

- Modify the selector as highlighted:

Click here to view code image

```
.dropdown-menu .dropdown-item:hover
```

Once the selectors are created, you can copy and paste the appropriate styles in CSS Designer.

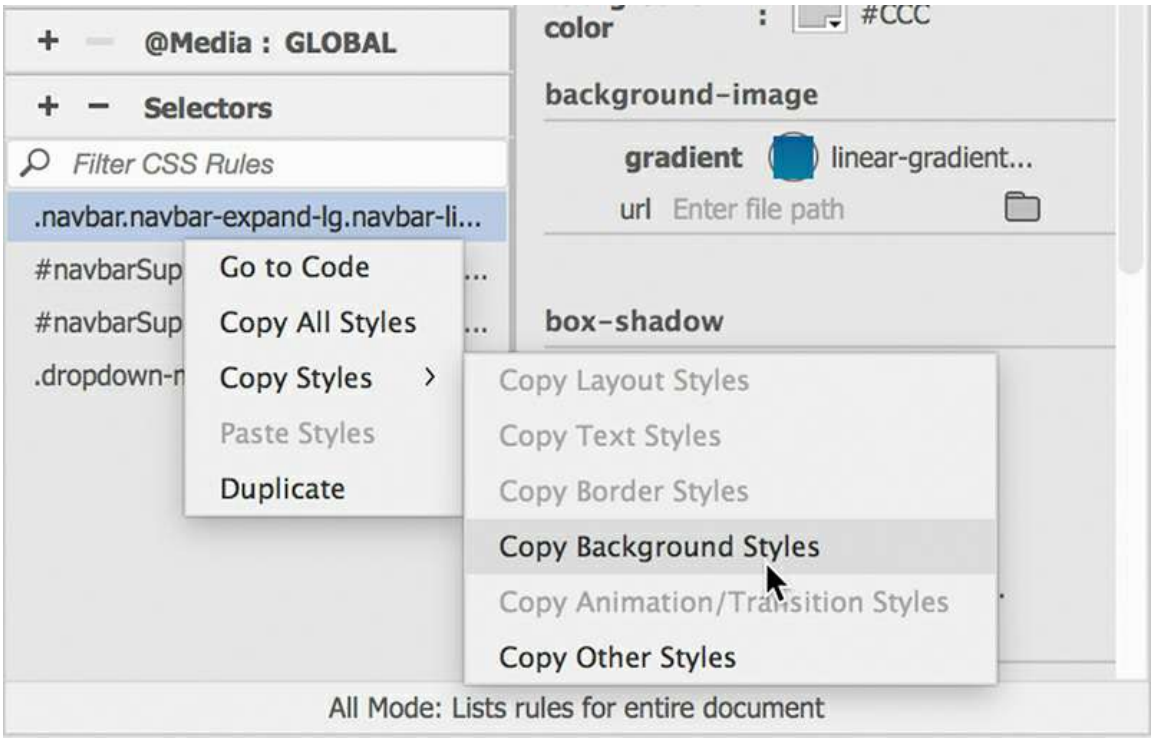
- Right-click on the rule

Click here to view code image

```
.navbar.navbar-expand-lg.navbar-light.bg-light.
```

This rule applies the background styles for the main menu.

- Select Copy Styles > Copy Background Styles from the context menu.



- Right click the rule

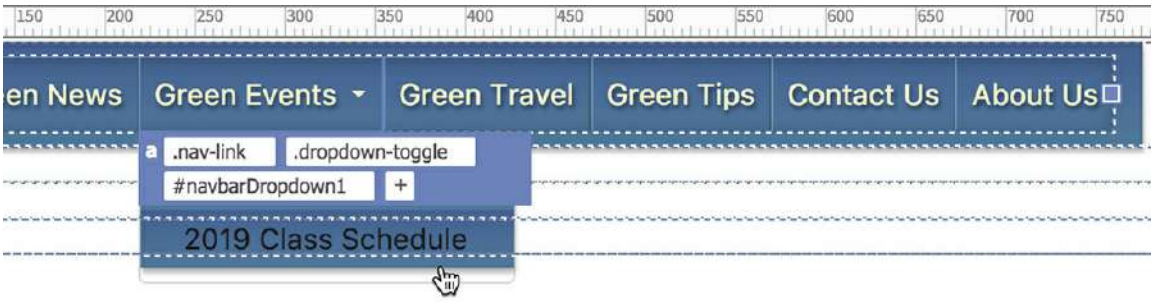
  Click here to view code image

```
.dropdown-menu .dropdown-item:link ,
.dropdown-menu .dropdown-item:visited.
```

- Select Paste Styles from the context menu.

  The dropdown menu items should be formatted now.

- Click the Green Events dropdown menu.



  The background of the menu is now formatted, but the text is not. That styling came from a different rule.

- Right-click the rule

  Click here to view code image

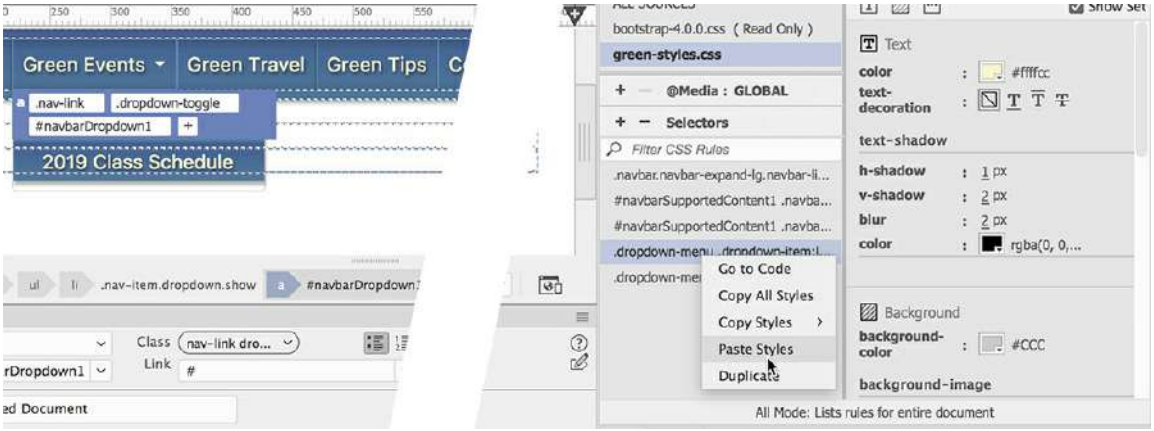
```
#navbarSupportedContent1 .navbar-nav.mr-auto .nav-item
.nav-link:link , #navbarSupportedContent1 .navbar-nav.mr-auto
.nav-item .nav-link:visited.
```

. Copy all styles.

. Paste the styles on

Click here to view code image

```
.dropdown-menu .dropdown-item:link ,
.dropdown-menu .dropdown-item:visited.
```



The text in the dropdown menu now looks the same. Next, let’s apply the hover styles.

. Copy all styles from the rule

Click here to view code image

```
#navbarSupportedContent1 .navbar-nav.mr-auto .nav-item
.nav-link:hover.
```

. Paste the styles on the rule

Click here to view code image

```
.dropdown-menu .dropdown-item:hover.
```

The styling of the dropdown menu should be complete.

. Save all files. Test the dropdown menu.

The entire navigation menu is now styled to match the site theme, but it appears off to the left side of the layout. To match the original GreenStart design, you'll need to center the responsive menu in the layout.

Centering a responsive menu

Bootstrap enables you to create complex menu and navigation components with a minimum of effort, but it doesn't provide unlimited styling options. For one thing, the framework offers two basic alignment options for horizontal menus: aligned to the left or justified across the entire structure. Aligning the menu to the center of the layout, as in the original site design, will require you to step away from the framework defaults and create custom styling.

The first step is to set a fixed width on the menu.

. In the CSS Designer, select **green-styles.css**.

  Create the following rule:

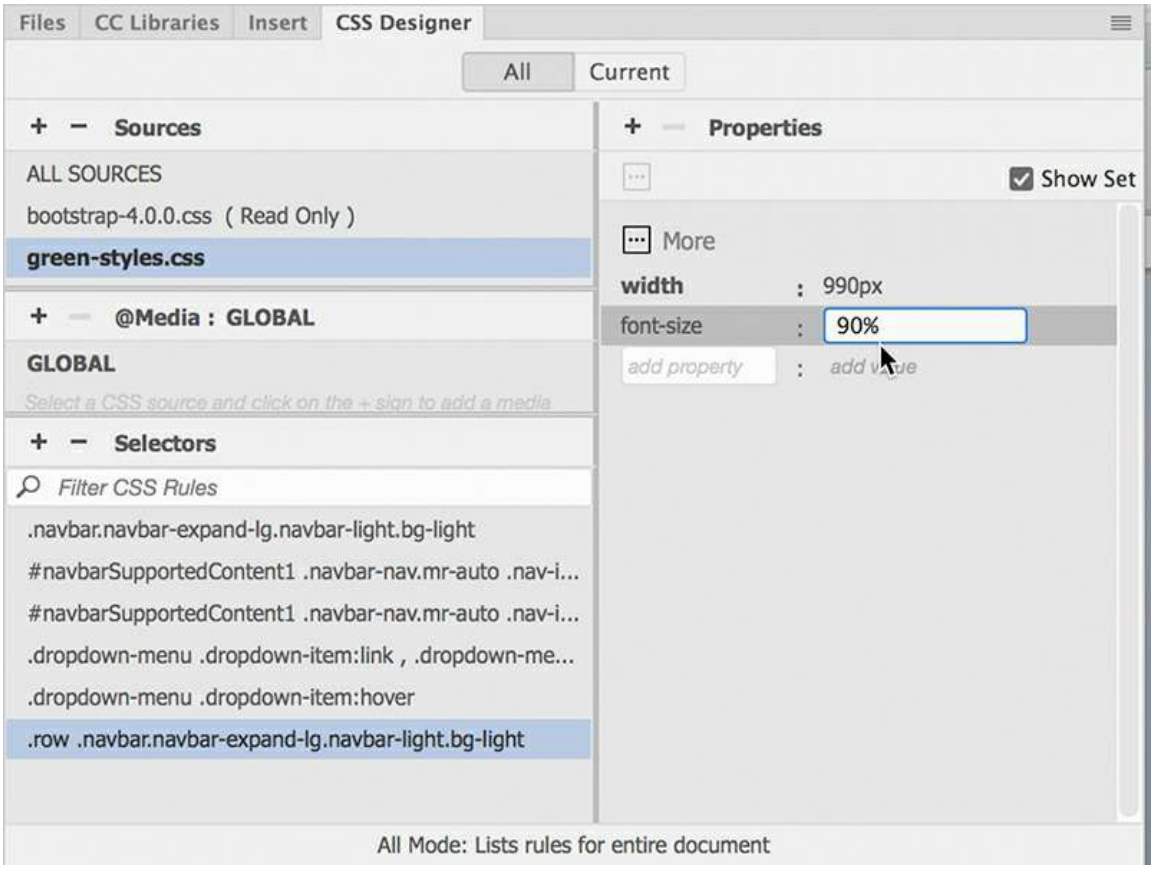
  Click here to view code image

```
.row .navbar.navbar-expand-lg.navbar-light.bg-light
```

> **Note**
>
> The menu width is derived from the Bootstrap CSS, which sets a maximum of 992 pixels for the menu before it collapses to an icon.

. Create the following properties:

```
width: 990px
font-size: 90%
```



Note

The width entered should keep the menu items on one line. If your menu breaks to two lines, increase the width slightly until it displays on one line.

This width allows the menu to fit on one line but still function properly in the responsive structure.

Now that you have set the width of the entire navbar, you can center the menu using a simple CSS trick. First, you'll have to create a custom rule to target it.

. Click any menu item.

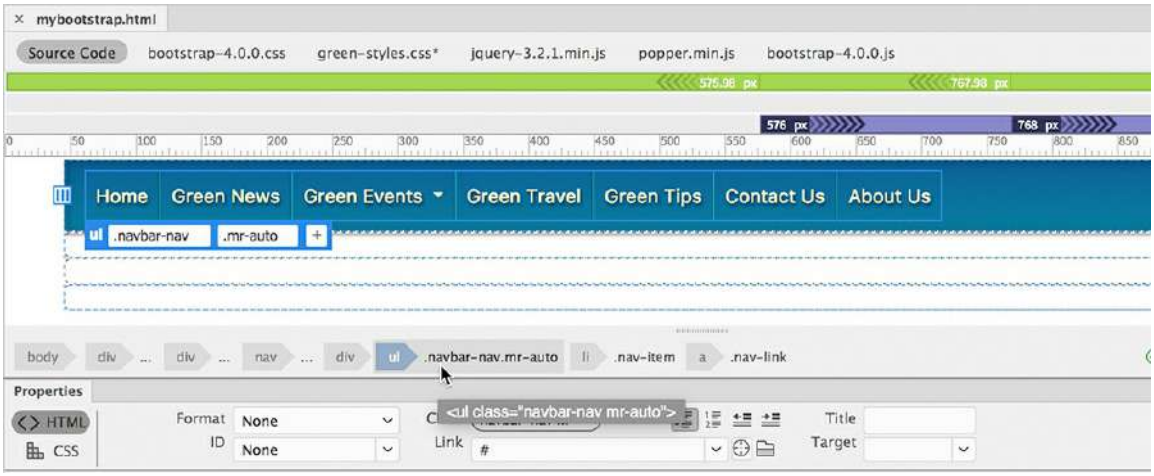
Examine the menu structure in the tag selector interface.

The menu is composed of a `<ul>` element with `<li>` and `<a>` elements as children.

. Select the `ul.navbar-nav.mr-auto` tag selector.

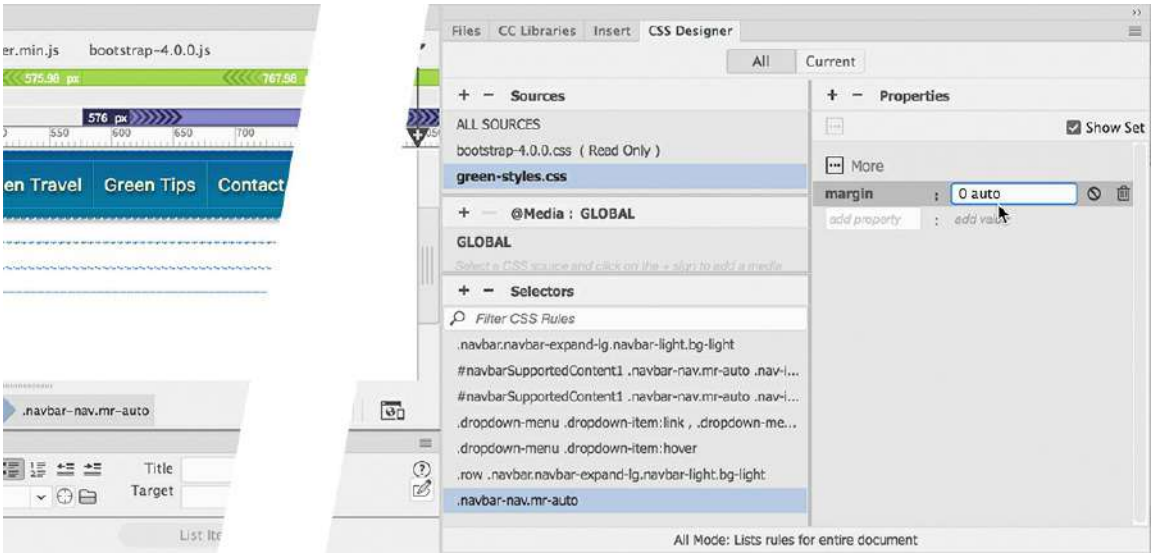
. Create the following selector:

```
.navbar-nav.mr-auto
```



. Add the following property to `.navbar-nav.mr-auto`:

```
margin: 0 auto
```



The menu centers in the navbar. The shorthand applies zero pixels of margin to the top and bottom of the menu and equal amounts of spacing to the left and right.

. Choose File > Save All.

The next task is to fill in the content area with the template placeholders.

Transferring content to a responsive template

Once the responsive menu is complete and styled properly, you can start populating the rest of the layout. In most cases, you will be able to use the content and styling from the old template without major modifications.

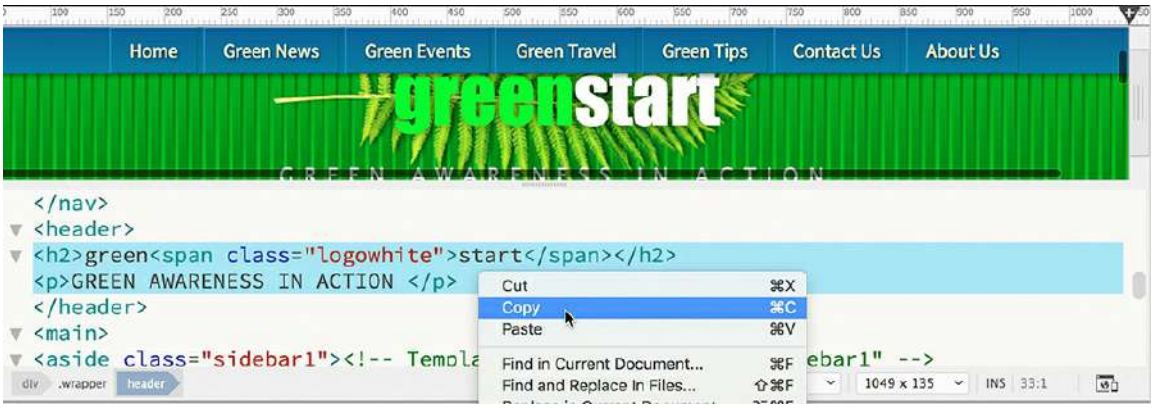
Note

In the following exercise, you will be copying and pasting HTML code and CSS styling using Split view and the Related File interface.

Creating a responsive header

Working from the top down, the header element is the next task on your list.

- If necessary, open **mybootstrap.html** and **mygreen_temp.dwt** in Split view. Make sure the width of the document window is at least 1100 pixels.
- In **mygreen_temp.dwt**, select and copy the `<h2>` and `<p>` elements in the `<header>` element in the Code view window.



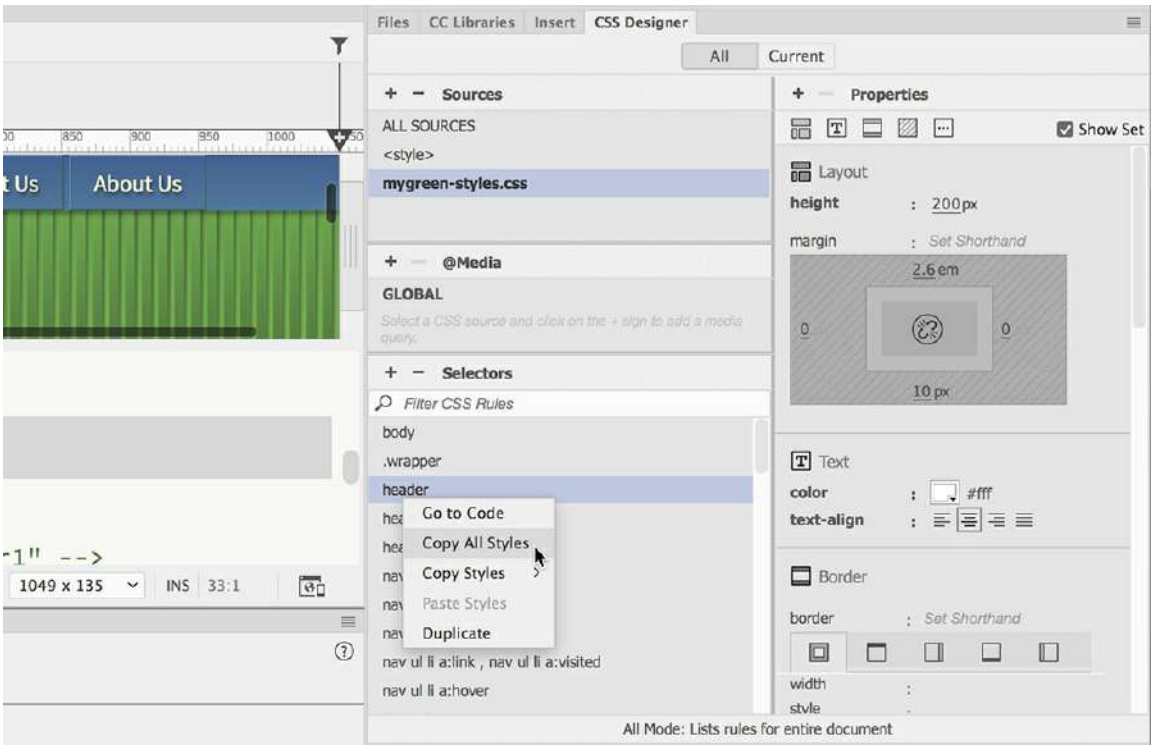
- Switch to **mybootstrap.html**. In the Code view window, scroll down and insert the cursor in `<div class="col-md-12">` in the `<header>` element.
- Paste the `<h2>` and `<p>` elements.

The GreenStart name and motto appear within the second row of the Bootstrap layout. At the moment, the text and header are unstyled.

. Switch to **mygreen_temp.dwt**.

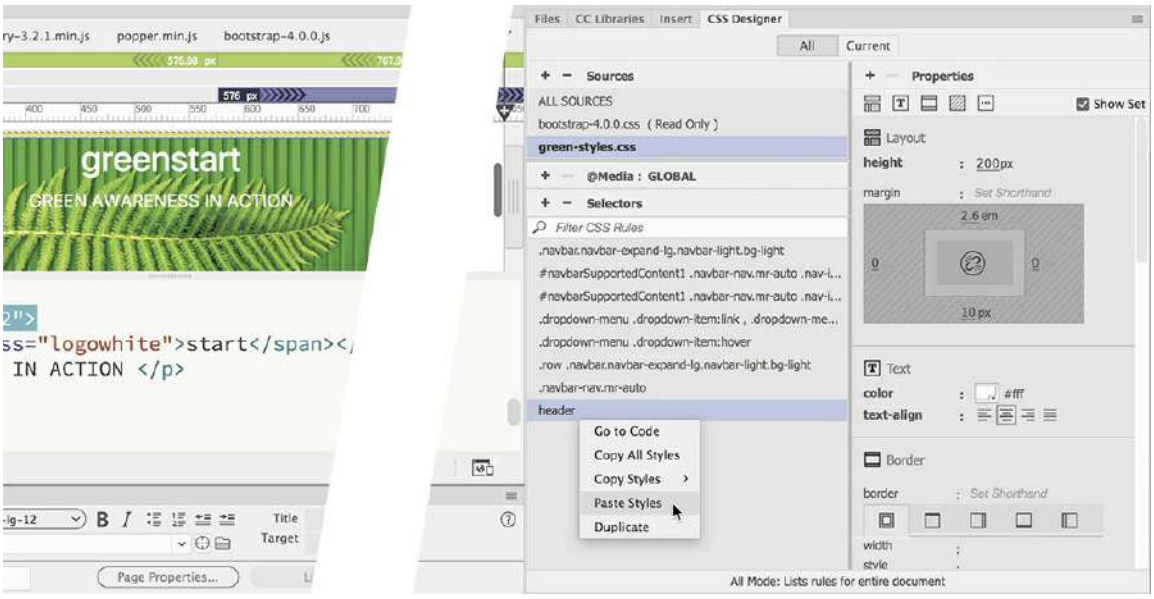
In the CSS Designer, right-click the `header` rule.

. Select Copy All Styles from the context menu.



. In **mybootstrap.html**, create the rule **`header`** in **green-styles.css**.

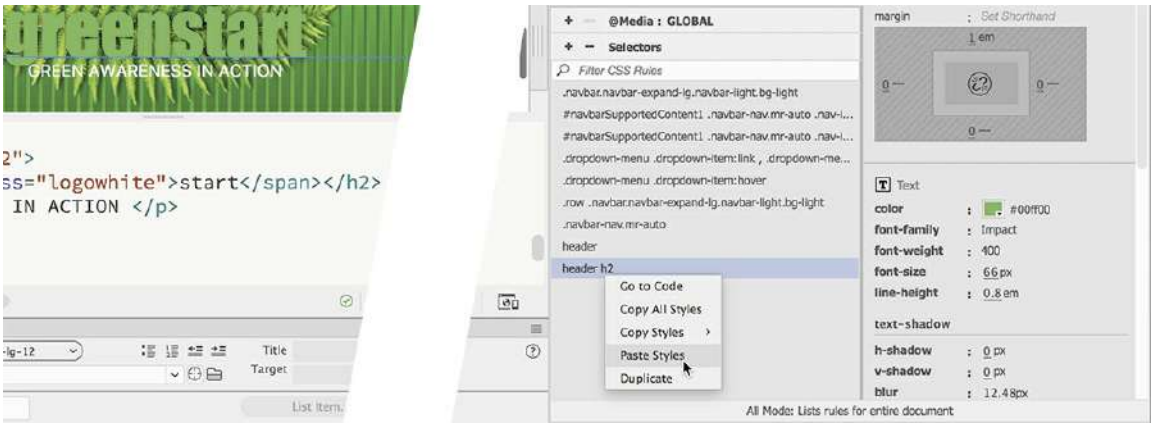
. Right-click the new rule and select Paste Styles from the context menu.



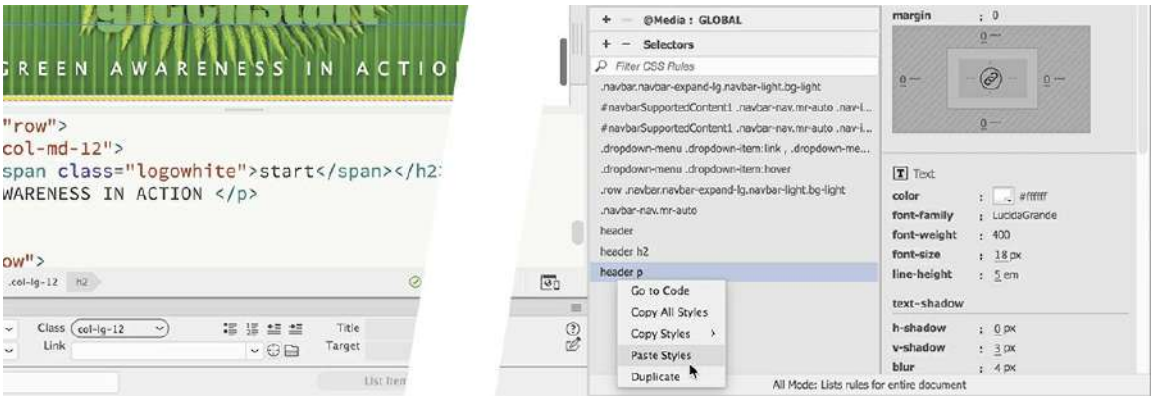
The background of the header displays the vertical stripes and fern images. Space was also added above the header.

You may remember that the space was needed to accommodate the horizontal menu when it was formatted to be non-scrolling in Lesson 9, "Working with Navigation." You'll fix this later, but first let's bring over the formatting for the header text.

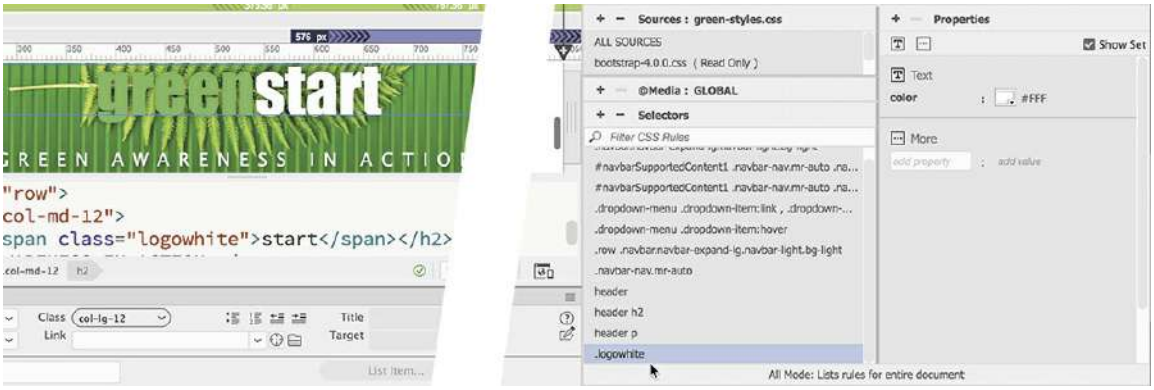
- In **mygreen_temp.dwt**, copy all styles from the rule `header h2`.
- In **mybootstrap.html**, create the rule **`header h2`** in **green-styles.css** and paste the styles.



- In **mygreen_temp.dwt**, copy all styles from the rule `header p`.
- In **mybootstrap.html**, create the rule **`header p`** in **green-styles.css** and paste the styles.



- In **mygreen_temp.dwt**, copy all styles from the rule `.logowhite`.
- In **mybootstrap.html**, create the rule **`.logowhite`** in **green-styles.css** and paste the styles.



. Save all files.

The styling for the responsive header is now complete and matches the GreenStart template.

Creating responsive placeholders

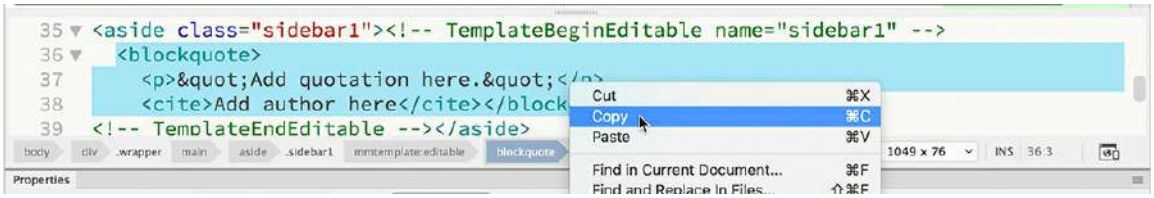
Note

You can perform most tasks in any document view, but remember to copy and paste in the same view.

Transferring the content and styling from the GreenStart template to the Bootstrap layout is pretty straightforward. You identify the content. You copy the needed elements and paste them into the corresponding structures in the new layout. Then, you assess the pertinent CSS rules that format all the components and create similar rules in the destination style sheet. In this exercise, you will move the content and styling for the main content placeholders. Be aware that you will not be able to use all the styling from the GreenStart template.

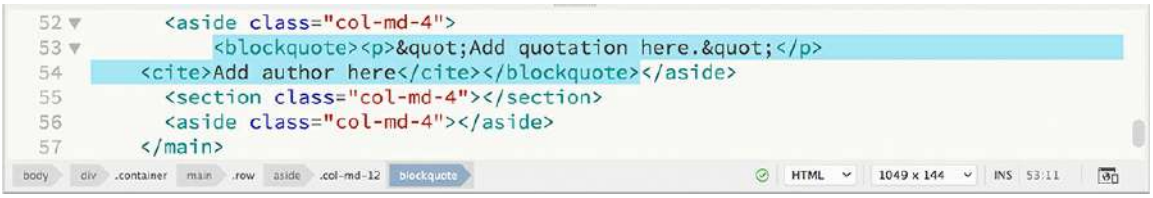
. If necessary, open **mybootstrap.html** and **mygreen_temp.dwt** in Live view. Make sure the width of the document window is at least 1100 pixels.

. In **mygreen_temp.dwt**, select and copy the `blockquote` element in sidebar 1.



The `blockquote` element contains the quotation and citation placeholders.

. Paste the `blockquote` placeholder in `aside.col-md-4` in **mybootstrap.html**.



- Create the following rules in **green-styles.css**:

```
.sidebar1 blockquote
.sidebar1 blockquote p
.sidebar1 blockquote cite
```

- Copy and paste all styles from the corresponding rules in **mygreen_styles.css**.

  When you are finished creating the three rules, you will notice that the quotation placeholders do not display any of the styling. Can you identify the issue?

  All the rules are based on the class `sidebar1`. That class does not exist in the Bootstrap layout. You could delete the class from each selector name, but then the rules would apply to all `blockquotes`. Instead, you will simply add the class to the appropriate element.

- In **mybootstrap.html**, click the quotation placeholder in Live view. Select the `aside.col-md-4` tag selector.

  The Element Display appears focused on the `aside` tag.

- Click the Add Class/ID icon +.

  Type `.sidebar1` and press Enter/Return to add the new class.

  As soon as the new class is added, the placeholder text displays the proper formatting. However, the `<aside>` element is missing the top and bottom borders that appear in the original design.

  If you examine the template and style sheet, you will discover that there is a single rule that formats both `<aside>` elements, providing the borders as well as some layout specifications. This is a perfect example of a situation in which you would not bring over all the styles. Instead, it would be easier to create the needed rule from scratch.

- Switch to **mybootstrap.html**, if necessary.

  Create the following rule in **green-styles.css**:

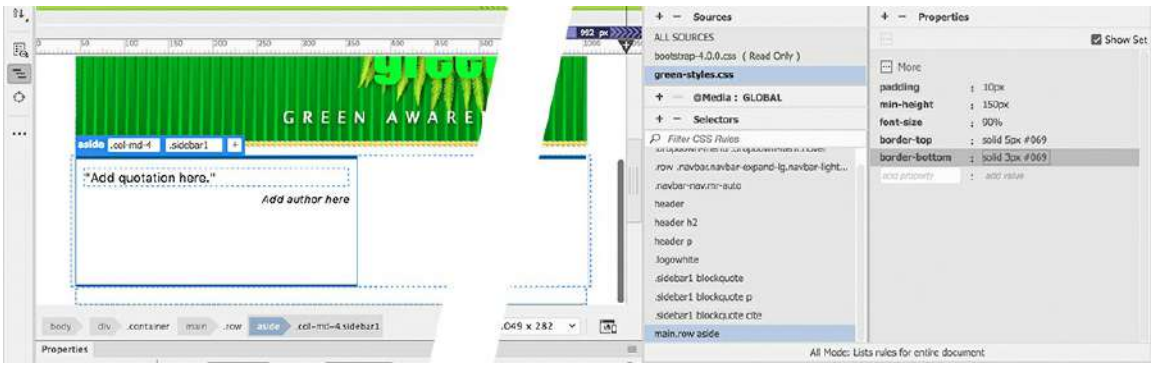
```
main.row aside
```

  This rule will target only the `aside` elements appearing in the `main` element. Notice that the

new selector is slightly different from the original one. That's because the predefined Bootstrap rules are already formatting aspects of these elements and are more specific than the original. Adding the class to the selector allows the rule to override the existing specifications.

- Create the following properties:

```
padding: 10px
min-height: 150px
font-size: 90%
border-top: solid 5px #069
border-bottom: solid 3px #069
```



When moving CSS settings over, keep an eye out for any specifications that set the width or height of an element. Bootstrap controls all the dimensions and interactions of your page structures. Since the current GreenStart site is a static, fixed-width design, you don't want to use any specifications that would interfere with the responsive behavior of the new template.

The next step is to bring over the placeholders from the second column.

- In **mygreen_temp.dwt**, select and copy the `<h1>` and `<article>` elements.
- In **mybootstrap.html**, paste the elements in `section.col-md-4`.

Once you have the placeholders in position, you will create the rules needed to style them. But in this case you will not bring over the old specifications. That's because the GreenStart design uses

Adobe web-hosted fonts, and simply copying and pasting the specifications will not support the use of those fonts properly. In this instance, you will have to redefine those custom font stacks from scratch.

Setting up web-hosted fonts

The GreenStart site design calls for the use of Adobe web-hosted fonts. You can't simply copy and paste the settings from one site to the other.

In this exercise, you will redefine the custom font stacks that will be used throughout the new responsive design. Whenever you set up a font scheme, always start with the base font specification for the site.

- Switch to **mybootstrap.html** in Live view, if necessary.

  Typically, the base font for a site is defined in the `body` rule.

- In the CSS Designer, select **green-styles.css**.

  Create a new selector named **`body`**

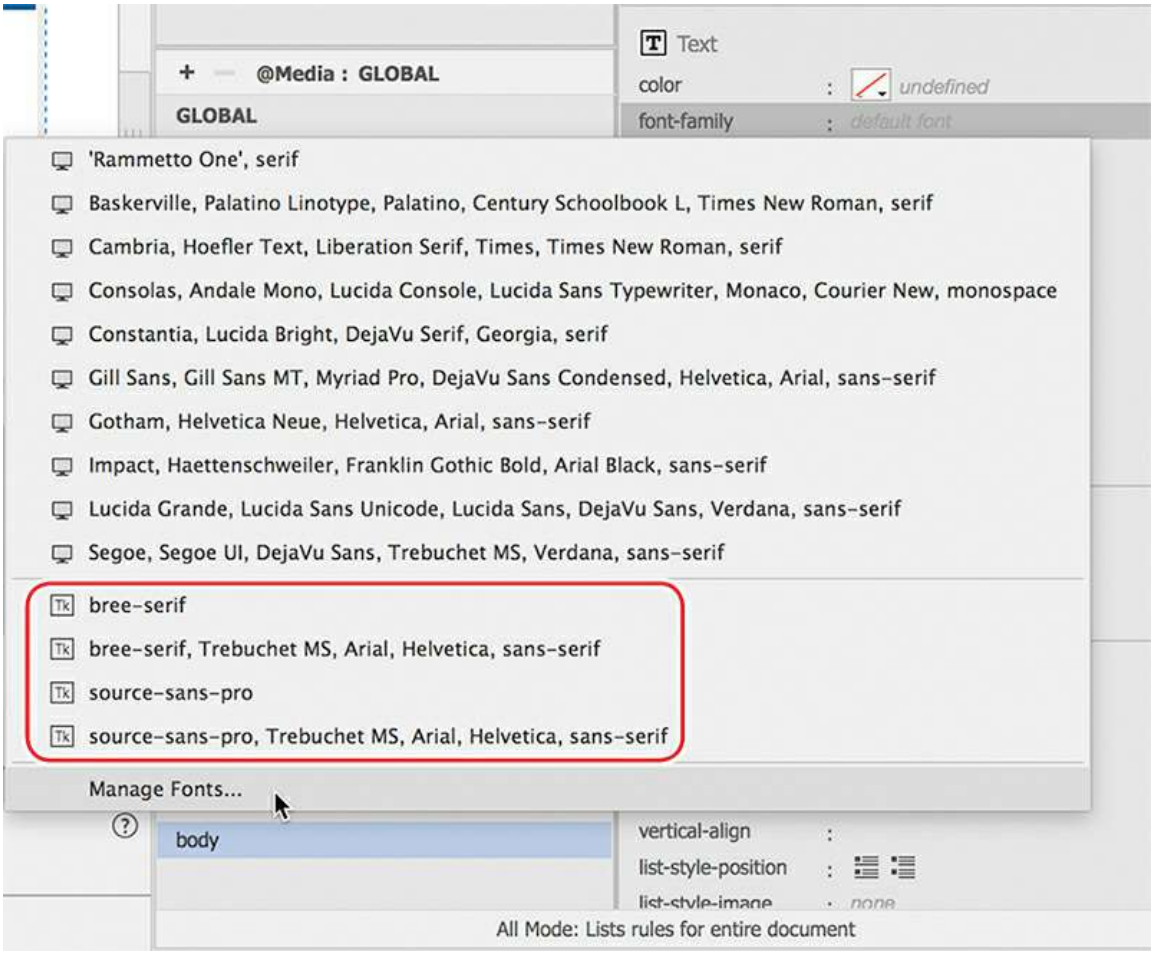
- In the Properties panel, deselect the Show Set option, if necessary.

  The panel now displays all CSS specifications.

- Click the Text category icon  .

  The panel display focuses on CSS Properties for text.

- Click to open the `font-family` property.



A popup window appears displaying the custom font stacks defined in Dreamweaver. Examine the list of font stacks and custom web fonts to see if Source Sans Pro and Bree Serif appear in the list.

. If Source Sans Pro appears in the custom font stack, skip to step 14. Otherwise, click Manage Fonts.

The Manage Fonts dialog appears. The panel has three tabs, for Adobe Edge Web Fonts, Local Web Fonts, and Custom Font Stacks. The dialog defaults to Adobe Edge Web Fonts. In the GreenStart design, the base font was Source Sans Pro.

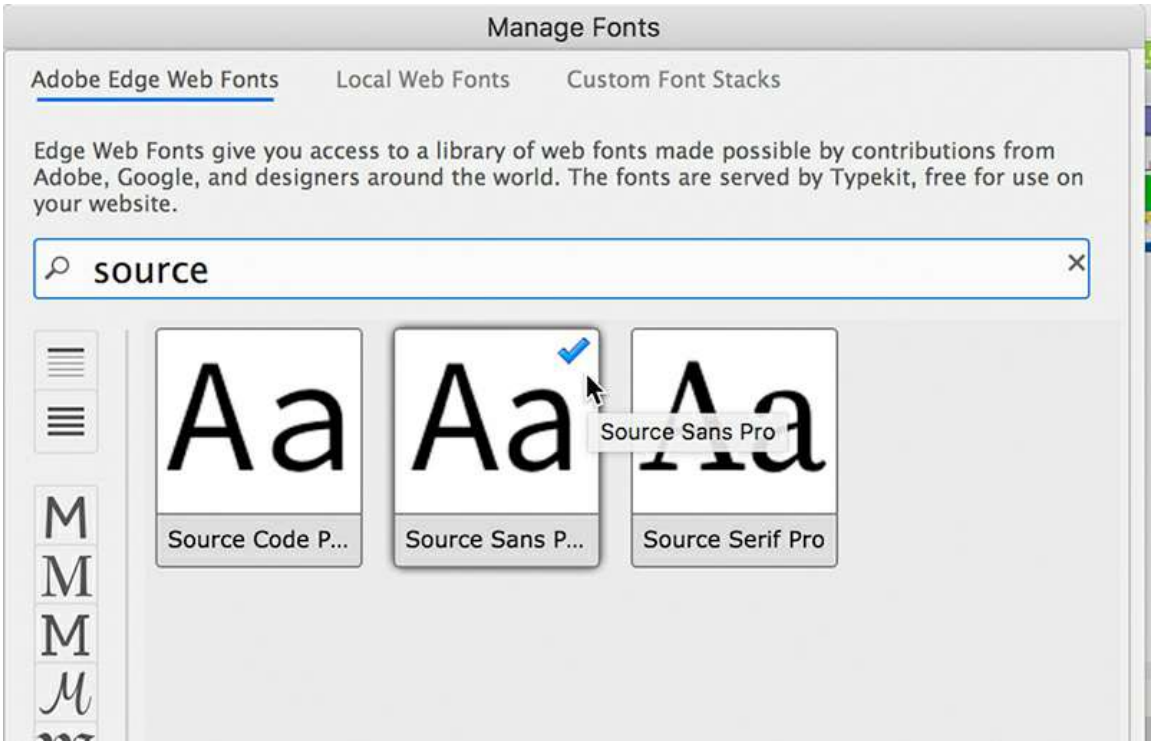
. In the search field, type **source**.

The window filters the list of fonts showing items that contain the word “source.”

Note

Source Sans Pro may still be selected from the original static site. If that is the case, you do not have to select it a second time.

- If necessary, select Source Sans Pro from the items displayed.

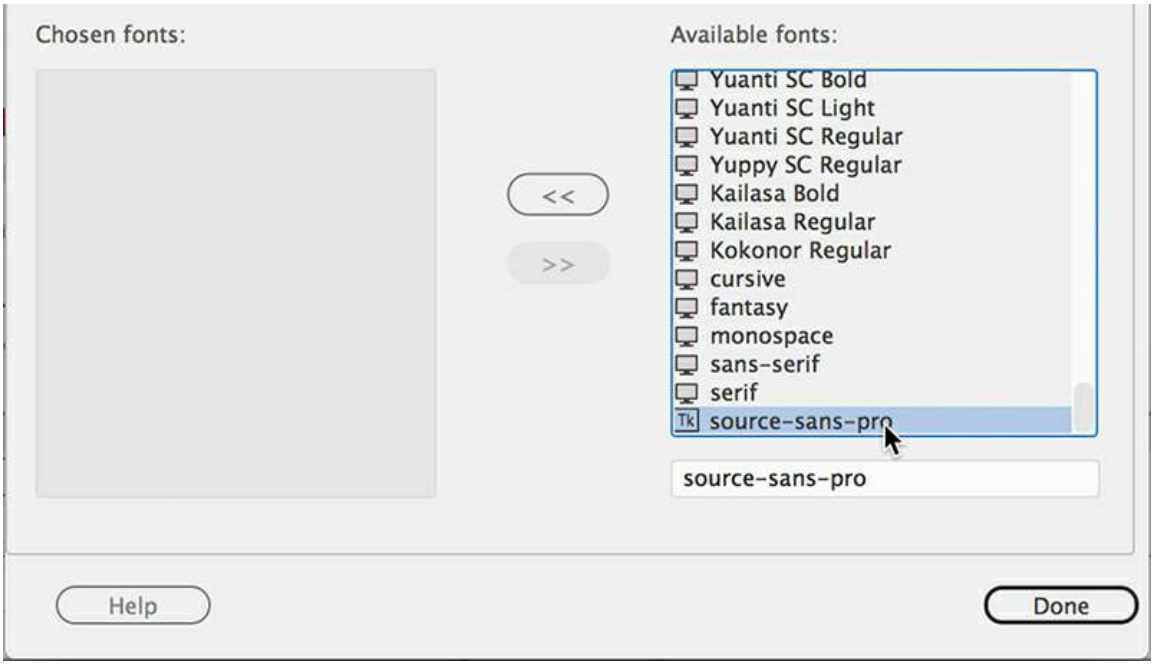


  A blue checkmark appears on the selected font. You could use this font immediately, but it's a good idea to build a complete custom stack. That way, if for some reason Source Sans Pro doesn't load, you can specify what font or fonts will be displayed instead.

- Click the tab Custom Font Stacks.

  Source Sans Pro is now displayed at the bottom of the Available Fonts window in the dialog.

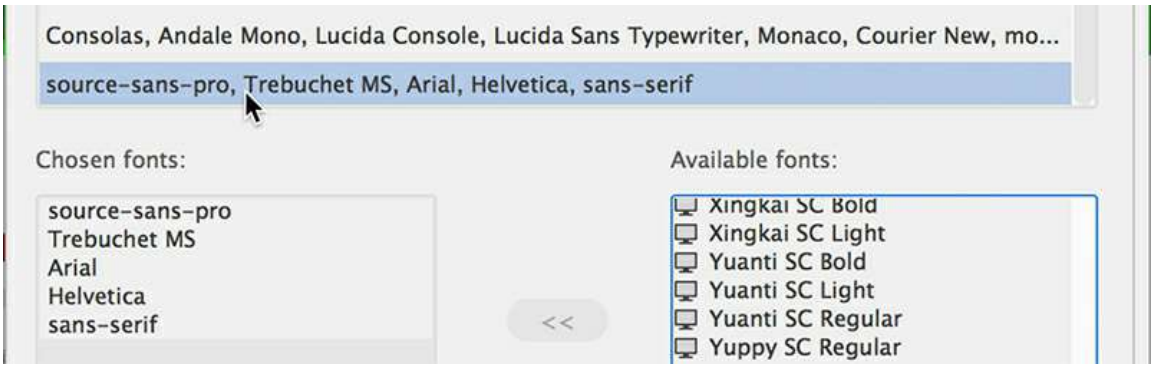
- Select `source-sans-pro` and click the << button to move the font into the Chosen Fonts window.



The font appears in the Chosen Fonts window. You have started your custom font stack. Next, you should add the fonts you want to load if the first one doesn't, and so on.

. Add the following fonts:

```
Trebuchet MS
Arial
Helvetica
sans-serif
```



A new custom font stack with your select fonts appears in the dialog.

. Click Done to close the panel.

. In the CSS Designer, select **green-styles.css**.

Select the `body` rule.

Click the `font-family` property.

When the Font Stack dialog opens, you will see `source-sans-pro` listed twice, once by itself and a second time as part of your new custom font stack.

. Select the custom font stack.

Once you select the new font stack, Dreamweaver adds the font to the CSS Designer interface and writes any code that is needed in your page to use it in your CSS specifications. You can see the specific markup in Code view.

. Switch to Code view.

Scroll to the `<head>` section of the code.

In the `<head>` section, you will see two script tags and an HTML comment explaining that the scripts download font from the Adobe Edge Web Font server. Notice that the script names the font you selected.

Note

You will see the web-hosted fonts in Live view only when you have a live connection to the Internet. Also remember that many of these fonts are provided via your Creative Cloud subscription. If your subscription expires, the fonts may fail to load on your site.

. Switch to Live view.

. In CSS Designer, select **400** in the `font-weight` property.

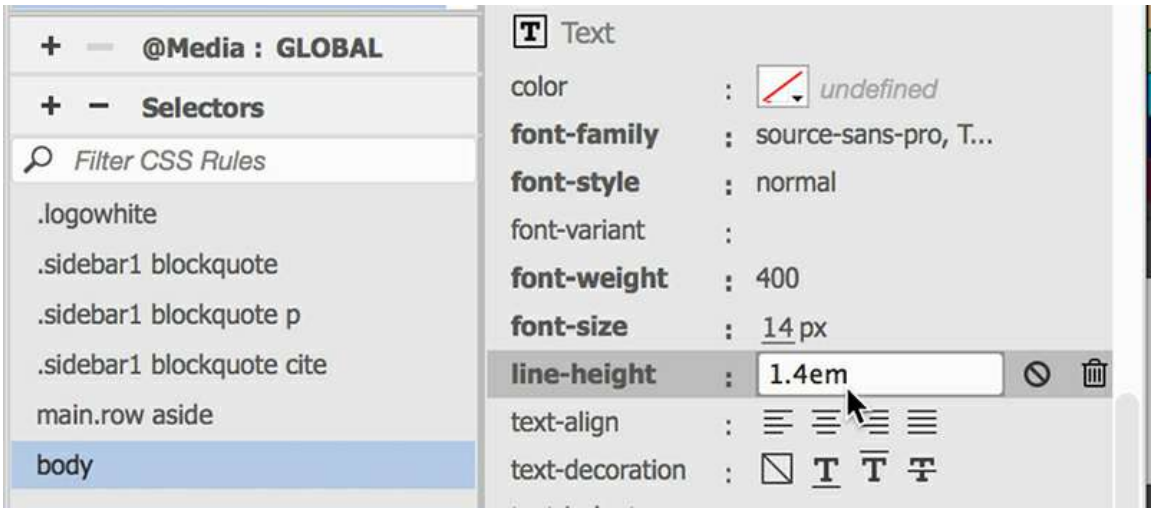
. Set the `font-size` property to **14px**.

Body and soul

In most websites, the `body` rule is used to set basic relationships between the various text elements used within the content. Since all visible content is contained within the `<body>` element, any style applied to it is automatically inherited by the other elements. That means text appearing in paragraphs, lists, and blockquotes is directly based on the `body` styles. Headings start with the `body` styles but then layer size and weight variations.

In modern style sheets, the font size set in the `body` rule is often specified in pixels and then by percentages or ems in all other rules. By avoiding fixed sizes, a design can automatically adapt when the `body` or default size changes. Remember this as you build pages and add content. Each new rule should modify only the undesirable inherited attributes. This reduces the total amount of code that you need to write and that your visitors need to download.

. Set the `line-height` property to **`1.4em`**.



Note

Edge Web Fonts and other hosted fonts require the use of JavaScript. But some visitors may turn off JavaScript in their browser.

In most cases, the changes in the layout will be instantaneous. The entire page, both headings and paragraph text, should now display Source Sans Pro.

If you don’t see the new font, you may not have a live connection to the Internet at this moment. Because Edge Web Fonts are hosted on the Internet, you won’t be able to see them

until you establish a live connection or upload this page to a live web server.

Some visitors change the default settings of their browsers, including font size. Setting the font size in the `body` rule in pixels is a common practice for many web designers. This practice is designed to reset the base font to a size that should be optimum for most visitors.

You also need to set up custom fonts for the main page heading.

. Create the following rule in **green-styles.css**:

```
main.row section h1
```

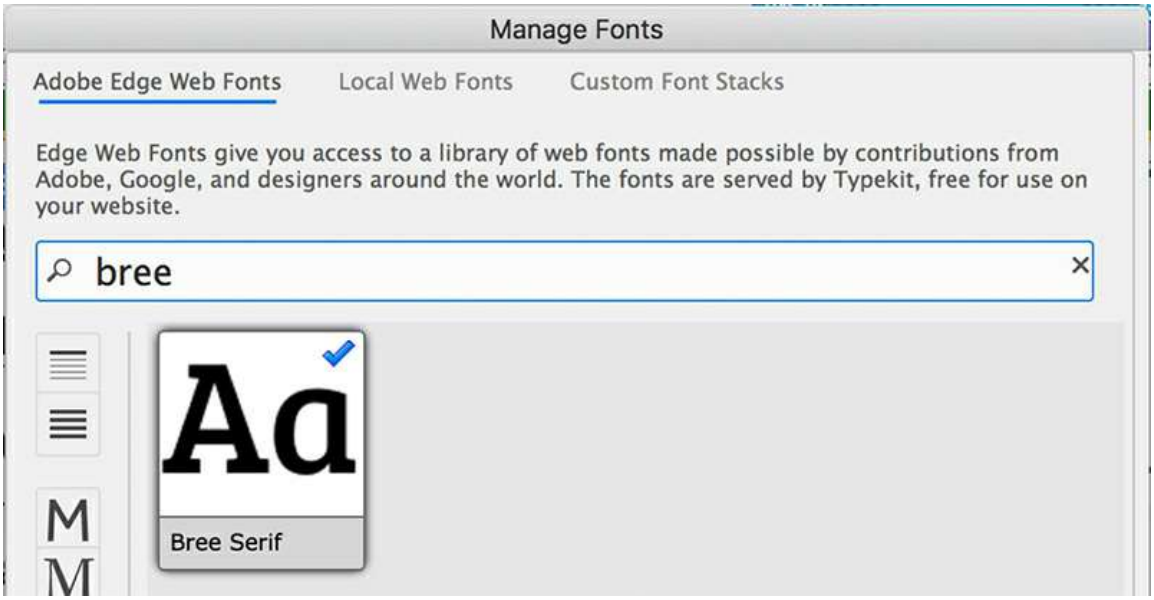
Since this heading uses web-hosted fonts too, you will define this style manually.

. Click to open the `font-family` property.

If Bree Serif appears in the custom font stacks, skip to step 24; otherwise, click Manage Fonts.

. In the search field type **Bree**.

. Select Bree Serif in the dialog.



A blue checkmark appears on the selected font.

. Click the tab Custom Font Stacks.

. Create a custom font stack with the following fonts:

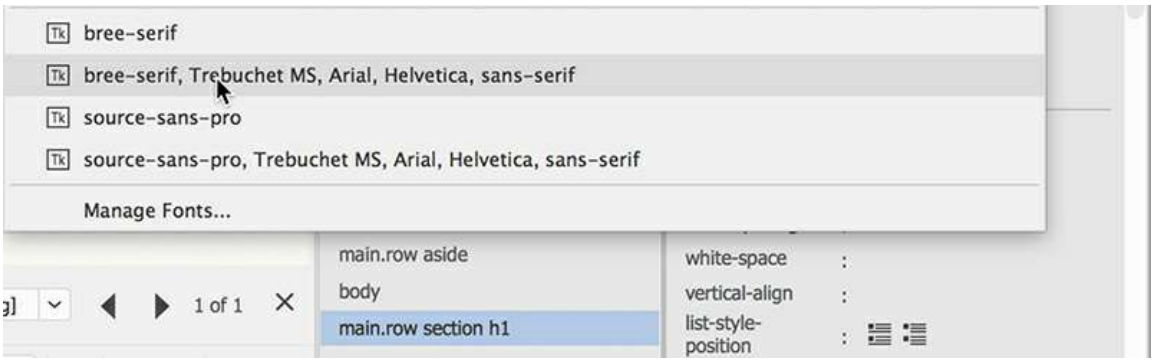
```
bree-serif
Trebuchet MS
Arial
Helvetica
sans-serif
```

. Click Done.

Note

When web-hosted fonts are selected, you may notice some properties are applied automatically.

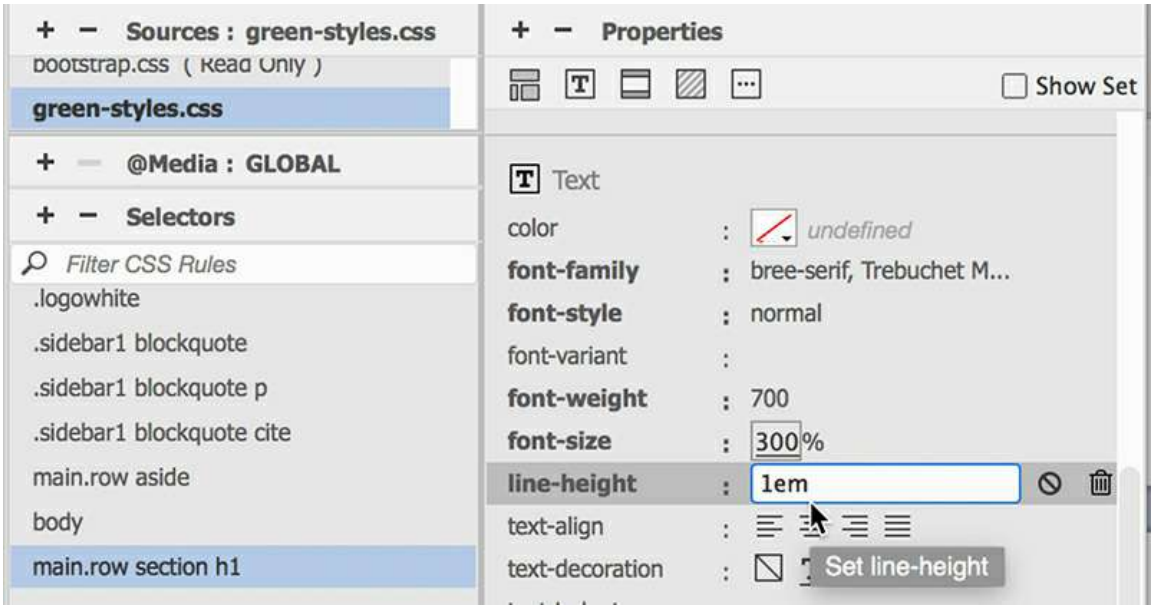
. In the rule `main.row section h1`, select the new custom font stack.



A reference to Bree Serif should now be added to the script loading the web-hosted fonts in your `<head>` section.

. In the rule `main.row section h1`, add the following properties:

```
margin-top: 0

margin-bottom: .5em

font-weight: 700

font-size: 300%

line-height: 1em
```



. Save all files.

The web-hosted fonts are now defined properly. Keep an eye out for any references to these two fonts in any rules you copy over from the old style sheet. The old settings may use fixed sizes or a different spelling of the font names. If they do, be sure to replace them with the proper spelling and size specified in percentages or ems.

Finishing the responsive placeholders

The remaining placeholders in the second column are in place and are still waiting to be styled. You will bring over the styling from the GreenStart template but edit the specifications as necessary.

. If necessary, open **mybootstrap.html** and **mygreen_temp.dwt** in Live view. Make sure the width of the document window is at least 1100 pixels.

. Switch to **mygreen_temp.dwt**.

Copy all the styles for the `main section article h2` rule.

This rule styles the headings in the `article` element.

. Switch to **mybootstrap.html**.

Create the following rule in **green-styles.css**:

```
main.row section article h2
```

. Paste the styles on the new rule.

Note

Fonts on your screen may appear different from the screenshots.

The pasted settings contain some unneeded and undesirable values. That's because the original settings were extracted from the Photoshop mockup. These should be replaced.

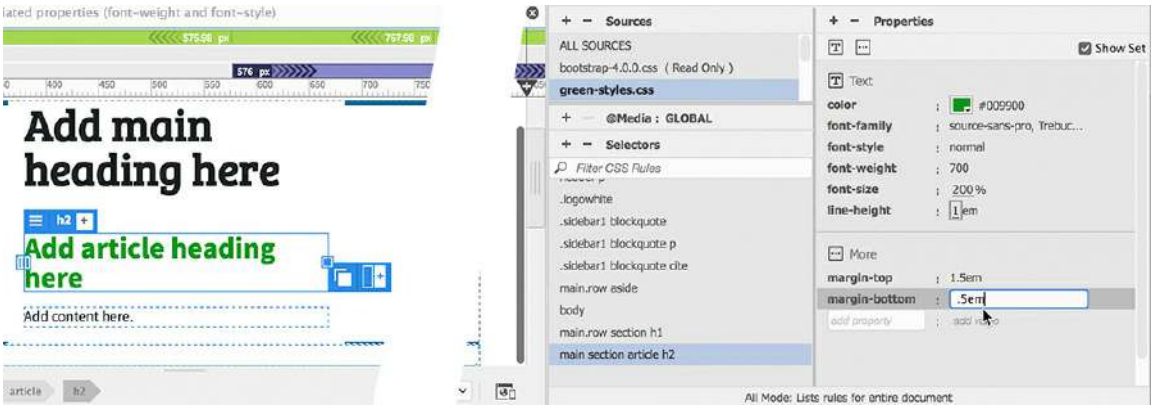
- Select the Show Set option.
- In the rule `main.row section article h2` edit the following properties:

Click here to view code image

```
font-family: source-sans-pro, Trebuchet MS, Arial, Helvetica,
sans-serif;
font-size: 200%
line-height: 1em
```

- In the rule `main.row section article h2` add the following properties:

```
margin-top: 1.5em
margin-bottom: .5em
```



The article heading is now complete. Since the paragraph text inherits the styling from the body rule, no further styling needs to be done for the `<p>` element. You can move on to Sidebar 2.

- In **mygreen_temp.dwt**, select and copy the `<p>` element in Sidebar 2.

  Notice that Sidebar 2 has a light-green background color.

- In **mybootstrap.html**, paste the element in `aside.col-md-4` in the third column.

Note

You can use Code view or Live view to copy the placeholders. Be sure to use the same view when pasting.

- Switch to **mygreen_temp.dwt**.

  Copy the background styles for the `.sidebar2` rule.

- Switch to **mybootstrap.html**.

  Create the following rule in **green-styles.css**:

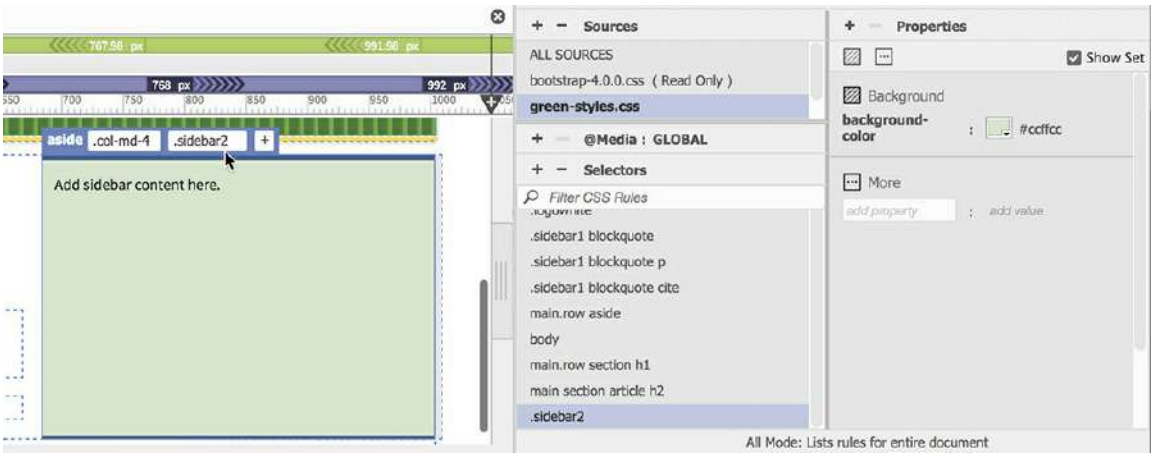
  `.sidebar2`

- Paste the styles on the new rule.

  As with Sidebar 1, you have to add the new class to the `<aside>` element.

- Select the `aside.col-md-4` tag selector for Sidebar 2.
- Click the Add Class/ID icon .

  Type `.sidebar2` and press Enter/Return to add the new class.



The background of the `aside` element is filled with light green.

The last element to move over is the footer.

- In **mygreen_temp.dwt**, select and copy the `<p>` element in the `<footer>` element.

  Notice that the `footer` has a gradient background color.
- In **mybootstrap.html**, paste the `<p>` element in `div.col-md-12` in the footer.
- Switch to **mygreen_temp.dwt**.

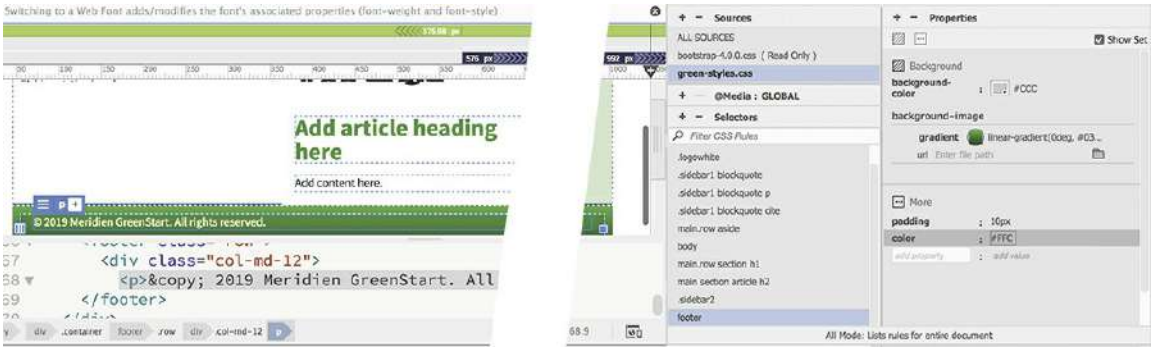
  Copy the background styles for the `footer` rule.
- Switch to **mybootstrap.html**.

  Create the following rule in **green-styles.css**:

  ```
  footer
  ```
- Paste the styles on the new rule.

  The footer displays the gradient background. The text is still black and seems a bit crowded.
- Add the following properties to the rule `footer`:

  ```
  padding: 10px
  color: #FFC
  ```



  The footer displays a little more space around the text, which now appears in a pale yellow.
- Save all files.

The boilerplate and placeholder text are now all in place. Before you convert this layout into the new responsive template, you'll need to adjust the widths of the columns. In the current layout, all three columns are identical in width. In the original design, the left and right columns were narrower than the middle one. Changing the widths is a simple process in Bootstrap.

Managing components in Bootstrap

The Bootstrap framework has built-in features that fulfill almost any need in a responsive website design. The developers and contributors built the framework to allow websites and web applications to scale easily and efficiently from desktops to phones to tablets. Most of the magic is enabled via CSS classes and media queries that also engage JavaScript in the process. By learning how to manipulate these classes, you can work miracles.

Controlling component width in Bootstrap

Bootstrap is based on a 12-column vertical grid system. Elements inserted into a layout conform to this grid by occupying some fraction of it. The amount of space an element uses is typically represented by a number that appears in the `class` attribute. For example, the three columns in the third row of the layout all have a class of `col-md-4`. Since 4 divides into 12 three times, each column occupies one-third of the screen. By adjusting these class names, you should be able to change the width of each element.

Tip

To achieve the expected results in this exercise, the document window should be at least 1100 pixels wide.

- Open **mybootstrap.html** in Live view, if necessary.

  At the moment, all three columns have the same class name and are equal in width.

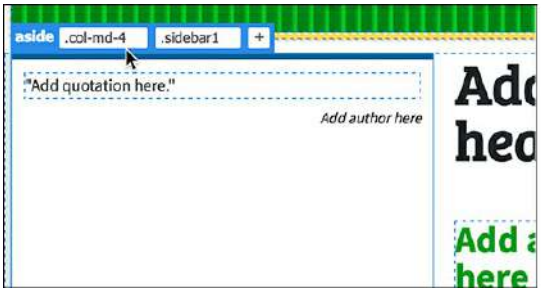
- Click Sidebar 1, in Live view.

  The Element Display appears focused on one of the elements in Sidebar 1.

- Click the `aside.col-md-4.sidebar1` tag selector.

  The Element Display focuses on `aside.col-md-4.sidebar1`. You can edit the class name directly in the Element Display.

- Click the class name `.col-md-4` in the Element Display. Change the class name to **`.col-md-3`** and press Enter/Return.



Note

Be careful not to delete the existing class name when editing it.

The width of the first column narrows. The other columns shift to the left, leaving open space on the right side of the third row. The Element Display is still focused on the first column.

- Click Sidebar 2.

  The Element Display focuses on one of the elements in `aside.col-md-4.sidebar2`.

- If necessary, select the tag selector for `aside.col-md-4.sidebar2`.

- Change the class name to **`.col-md-3`** and press Enter/Return.

  The right column narrows. In total, the entire row is now occupying only 10 of the 12 columns in the grid. You can leave these settings this way, add more space between the columns, or simply add the space to the main content section.

- Click the heading *Add main heading here*.

  The Element Display focuses on the `h1` element.

- Click the tag selector for `section.col-md-4`.

  Change the class name to **`.col-md-6`** and press Enter/Return.

  The center column widens to take up the empty space. The three columns are now using the entire width of the main container.

- Save all files.

You have now completed the layout and have populated it with the boilerplate text and content placeholders. But there's still one thing you need to do before you convert the layout into the new site template. The main navigation menu on the current GreenStart site is styled to remain visible at the top of the browser window as all the other content scrolls underneath it. The menu

in the new layout still scrolls with the rest of the page. Luckily, creating a fixed menu is one of the behaviors anticipated and supported by the Bootstrap framework.

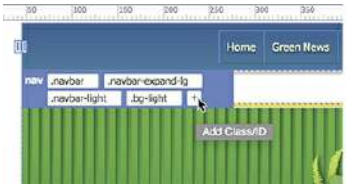
Freezing a navigation menu in Bootstrap

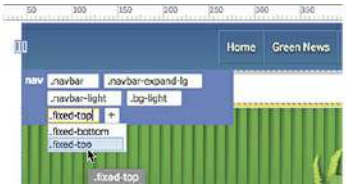
The new horizontal menu was built using a Bootstrap navbar widget. It provides a dynamic dropdown menu and automatically supports various screen sizes and devices. Currently, the menu scrolls with the rest of the webpage content. Freezing a menu to the top of the screen—like controlling component width—can be done by simply adding a single CSS class.

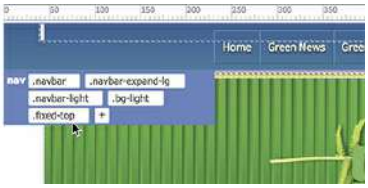
- Open **mybootstrap.html** in Live view, if necessary.
- Click any of the links in the menu.

  Select the `nav` tag selector.
- Click the Add Class/ID icon +.

  Type `.fixed-top` and press Enter/Return.





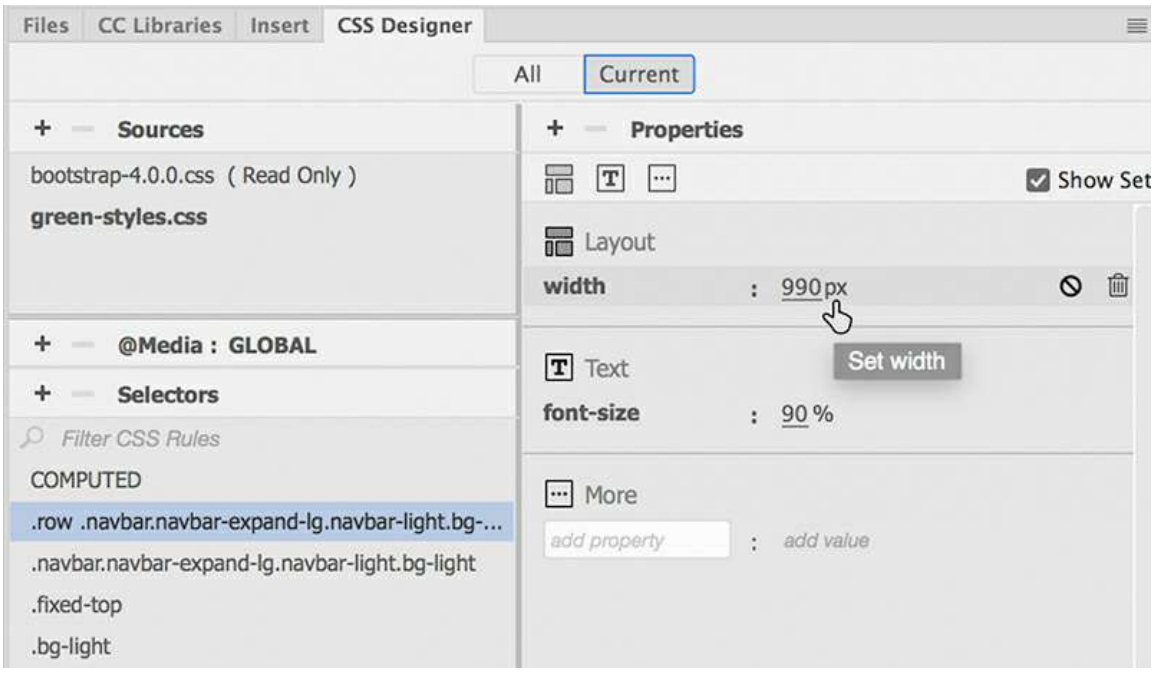


  As you type, you will see the hinting list display any matching classes. Feel free to select the class when it appears. This is a predefined class in the Bootstrap style sheet, which formats these kinds of menus.

  The horizontal menu is now fixed to the top of the page and removed from normal page flow. The menu will now remain visible as the content of the page scrolls underneath it.

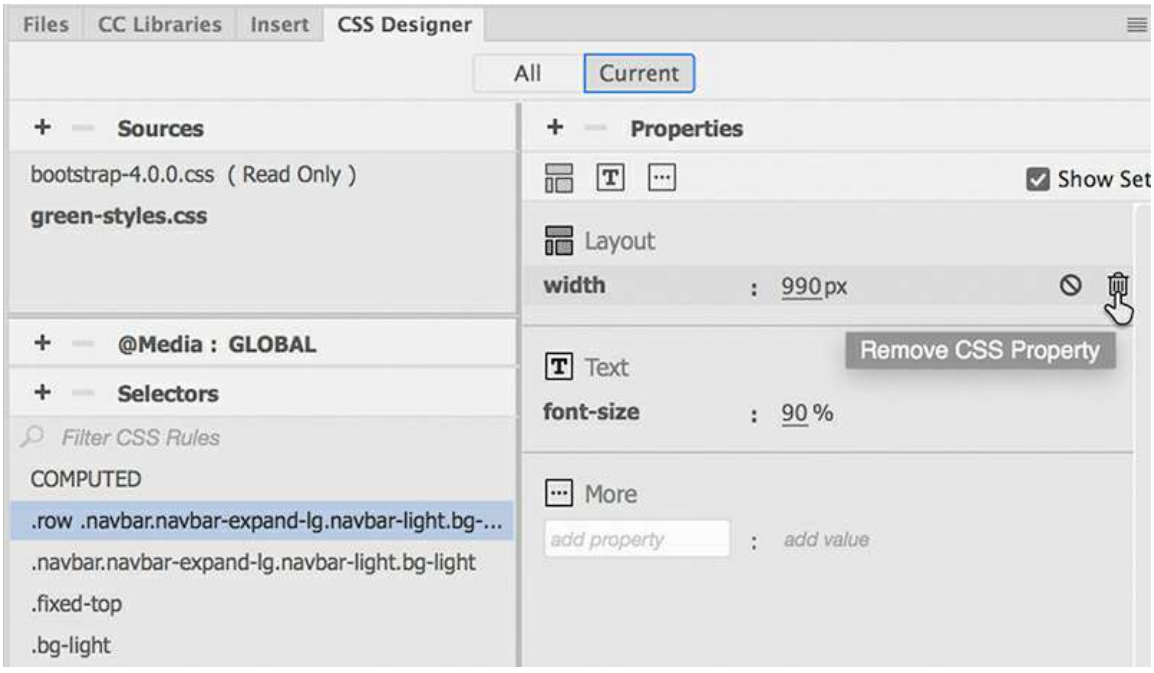
  Because the header styling you brought over already anticipated a fixed menu, the previous gap between the navbar and header is now gone and everything snugs up nicely together. The only thing left to tweak is the width of the menu. For some reason, it touches the left side but doesn't stretch all the way across to the right. Let's track down the reason for the glitch.
- Click any of the items in the horizontal menu.

  Select the `nav.navbar.navbar-expand-lg.navbar-light.bg-light` tag selector.
- In the CSS Designer, click the Current button and examine the properties applied to the menu.



You will see that the menu has a fixed width, which was assigned earlier to help center the menu items. Removing the fixed width should fix the issue.

. Click the Remove CSS Property icon for the Width value.



Once the value is removed, the navbar stretches all the way across the screen. The new responsive layout is complete.

. Save all files.

Converting a responsive layout into a Dreamweaver template is no different than working with a normal HTML file.

Creating a responsive template

The new responsive layout is nearly identical to the current GreenStart template. The main difference is that the file is based on a Bootstrap structure that will automatically support desktop computers and mobile devices.

Adding editable regions to a Bootstrap layout

In this exercise, you will add editable regions to the layout, save the file as a Dreamweaver template, and apply the template to existing site pages.

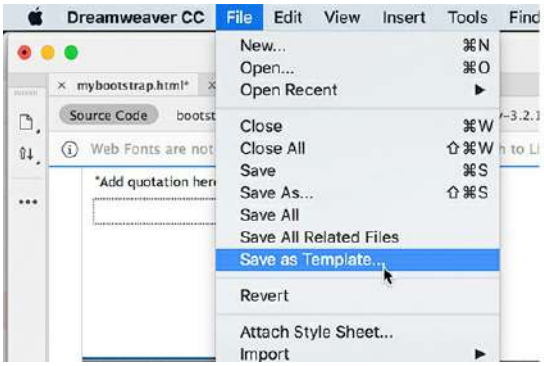
- Open **mybootstrap.html** in Live view, if necessary.

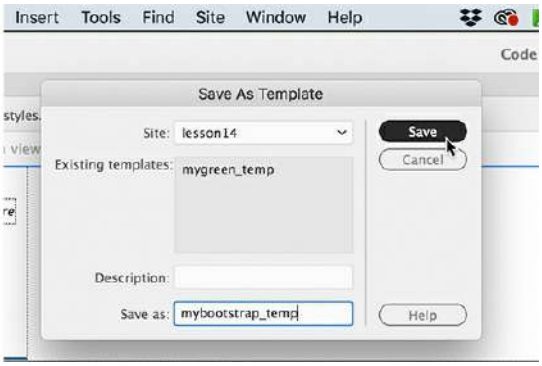
  As of this writing, the template workflow functions only in Design view and Code view.

- Switch to Design view.

  Much of the styling and responsive nature of the layout is not supported in Design view.

- Select File > Save As Template.
- In the Save As Template dialog, name the template **mybootstrap_temp** and click Save.





- Click Yes to update links.

  Once the file is saved as a template, you can add the editable regions. The new regions have to be named exactly the same way as in the GreenStart template so that the new design can be applied to the existing pages.

- Click the quotation placeholder.

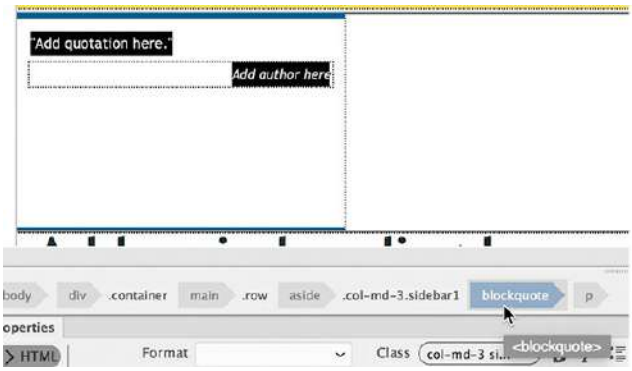
  Select the `blockquote` tag selector.

  The `<blockquote>` element is highlighted in the layout, showing that it is selected.

- Select Insert > Template > Editable Region.

The New Editable Region dialog appears.

. Enter **sidebar1** in the Name field and click OK.



A blue tab appears above the `<blockquote>` element, showing the name `sidebar1`. Next, you will create the region for the main content.

. Click the text *Add main heading here*.

Select the `h1` selector.

The `<h1>` element is highlighted in the layout, showing that it is selected.

. Hold the Shift key and click at the end of the text "Add content here."

The content placeholder is now completely selected.

. Select Insert > Template > Editable Region.

. Enter **main_content** in the Name field and click OK.

. Click the text placeholder in Sidebar 2.

Select the `p` tag selector.

. Select Insert > Template > Editable Region.

. Enter **sidebar2** in the Name field and click OK.

. Save all files.

All the editable regions have been added to the new template. The editable regions in the new template match the ones in the original template. This will allow you to apply the new Bootstrap-based template to all the existing site pages to make them responsive. The template is ready to be applied to the existing pages in the GreenStart site.

Optional exercise: Adding metadata placeholders

The main editable regions are now in place, but if this were a real site template you'd also move the other template-based items. There are two metadata elements in the static site template: one for the page title and the other for the metadescription. To fully complete the new responsive template, you should move those to the responsive template now.

```
6       <meta name="viewport" content="width=device-width, initial-scale=1">
7       <!-- TemplateBeginEditable name="doctitle" -->
8       <title>Add Title Here - Meridien GreenStart Association</title>
9       <!-- TemplateEndEditable -->
```

The page title is typically displayed in the browser tab.

```
14    <!-- TemplateBeginEditable name="head" -->
15    <meta name="description" content="Meridien GreenStart Association - add
      description here.">
16    <!-- TemplateEndEditable -->
```

The metadescription is often used as the summary of the page contents in search engine results.

Applying a new template to existing pages

The existing GreenStart pages were built using a static, fixed-width Dreamweaver template. The pages do not resize or react to different size screens or devices. The new template is built using the Bootstrap framework, which was designed to support all screens and devices by default. In this exercise, you will apply the new template to the existing pages and adjust them as necessary to work properly.

. Open **index.html** from the lesson14 folder in Live view.

This page is the home page of the GreenStart website. Before you apply the new template, you can see how the current design responds to different size screens and devices.

- Drag the scrubber to the left to reduce the width of the document window. Observe the changes to the page and its content.

The page does not react to the changing window size. As the window gets smaller, the right side of the page is obscured.

- Drag the scrubber to the right until the window is fully open.

To apply a new template, you have to switch to Code view or Design view.

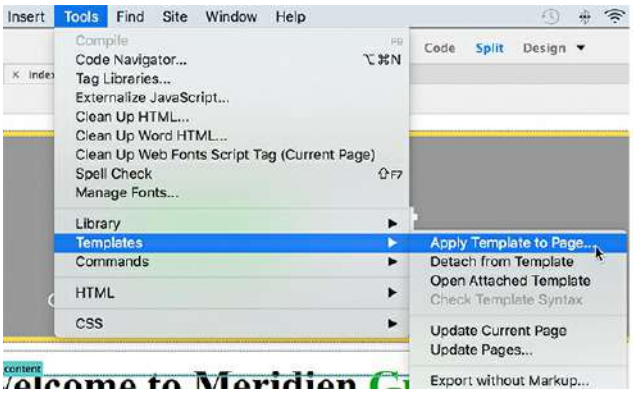
- Switch to Design view.

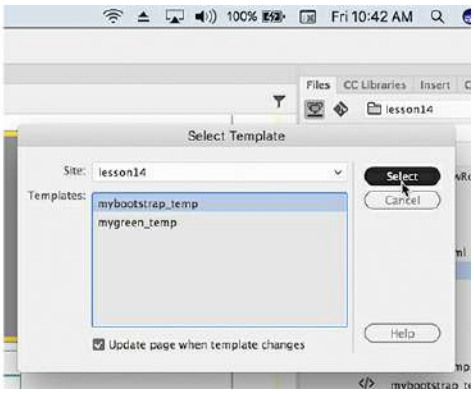
In Design view most of the page styling is gone, which also affects the location of the elements. In Design view, editable regions are identified by blue tabs.

- Select Tools > Templates > Apply Template To Page.

The Select Template dialog appears. The dialog lists both templates saved in the site.

- Select **mybootstrap_temp** and click the Select button.





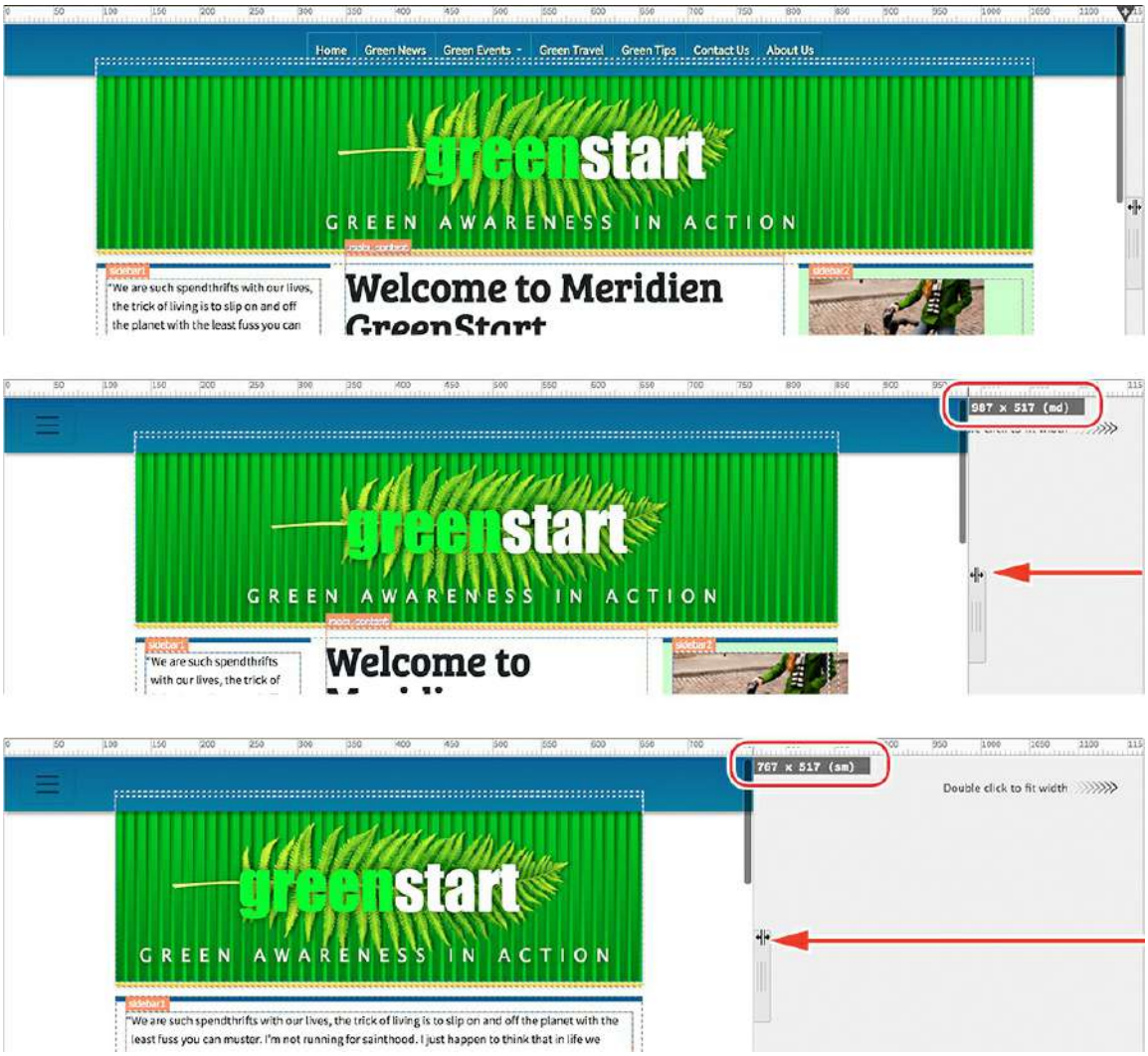
The new template is applied to the page instantly. In the top-right corner of the page, you will see the name of the new template displayed in a yellow box. The page is now based on the Bootstrap layout.

. Save the file.

The page is now ready to preview.

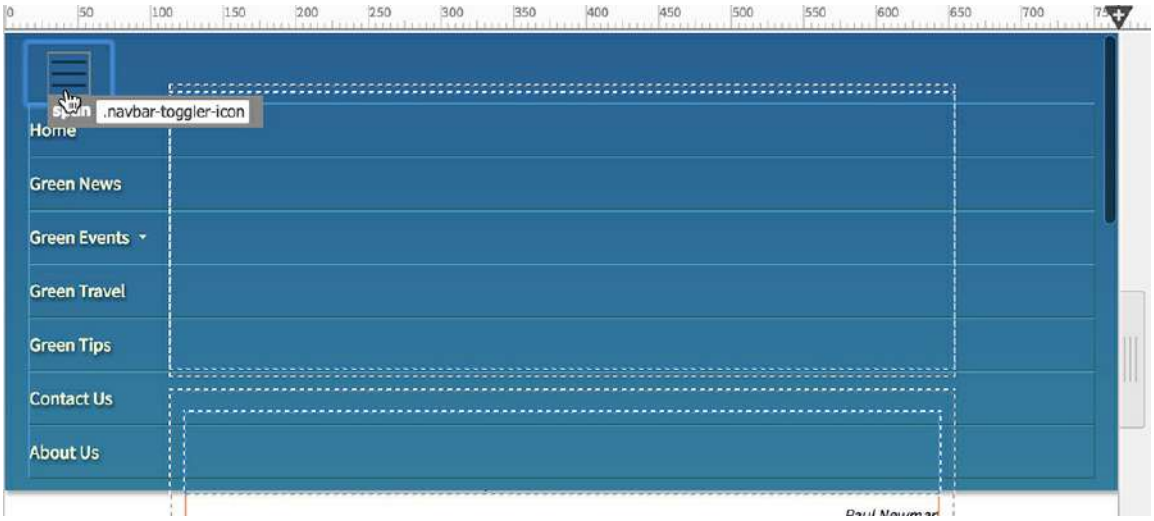
. Switch to Live view.

. Drag the scrubber to the left.



As the document window narrows, the layout begins to adapt. When the window drops below 992 pixels, the horizontal navigation menu is replaced by a blue bar with a sandwich icon. Below 768 pixels, the three-column layout converts to a single column with the three editable regions stacked one atop the other.

. Click the sandwich icon in the navigation bar.



The blue bar expands to show the menu. The seven menu items are stacked vertically.

. Click the sandwich icon again.

The menu closes, showing the blue bar again.

. Drag the scrubber to the right to open the document window.

Once you drag the scrubber past 768 pixels, the single-column layout converts back to three columns. The new responsive design will support all screen sizes and devices.

Now you have to apply the template and make the same changes to all the other pages in the site.

. Open **news.html**, **tips.html**, **events.html**, **travel.html**, **about-us.html**, and **contact-us.html** in Design view.
. Apply **mybootstrap_temp.dwt** to all pages.
. Save and close all files.

The responsive template is now applied to all pages, but you're not done yet. Although the basic layout is responsive, not all of the content is. In the next lesson, you will review the basic site design for any instances where the responsive design is failing. Once you review the basic structure, you will then review each page individually to make sure the content adapts appropriately to every screen size.

Review questions

1. What is the advantage of using Bootstrap components in your layout?
2. Can you edit the styling of a Bootstrap element directly?
3. How can you reproduce styling in an existing style sheet?

4. How can you control the width of Bootstrap elements?
5. Can web-hosted fonts be used in a custom font stack?

Review answers

1. Bootstrap components are built to be responsive out of the box, allowing you to create complex structures that support a variety of screen sizes and devices with minimum effort.
2. No. The Bootstrap style sheet is locked, but you can easily create styles in your own style sheet to format or override default styling.
3. The CSS Designer enables you to copy some or all of the styles from an existing rule and then paste them into a new rule or style sheet.
4. Bootstrap controls the width of HTML elements by using predefined CSS classes. Layouts are based on a default 12-column grid, and the classes specify what portion of that grid is taken up by the element.
5. Yes. Once the web-hosted font is selected in the Manage Fonts dialog, it can be added to a custom font stack and used throughout a site.

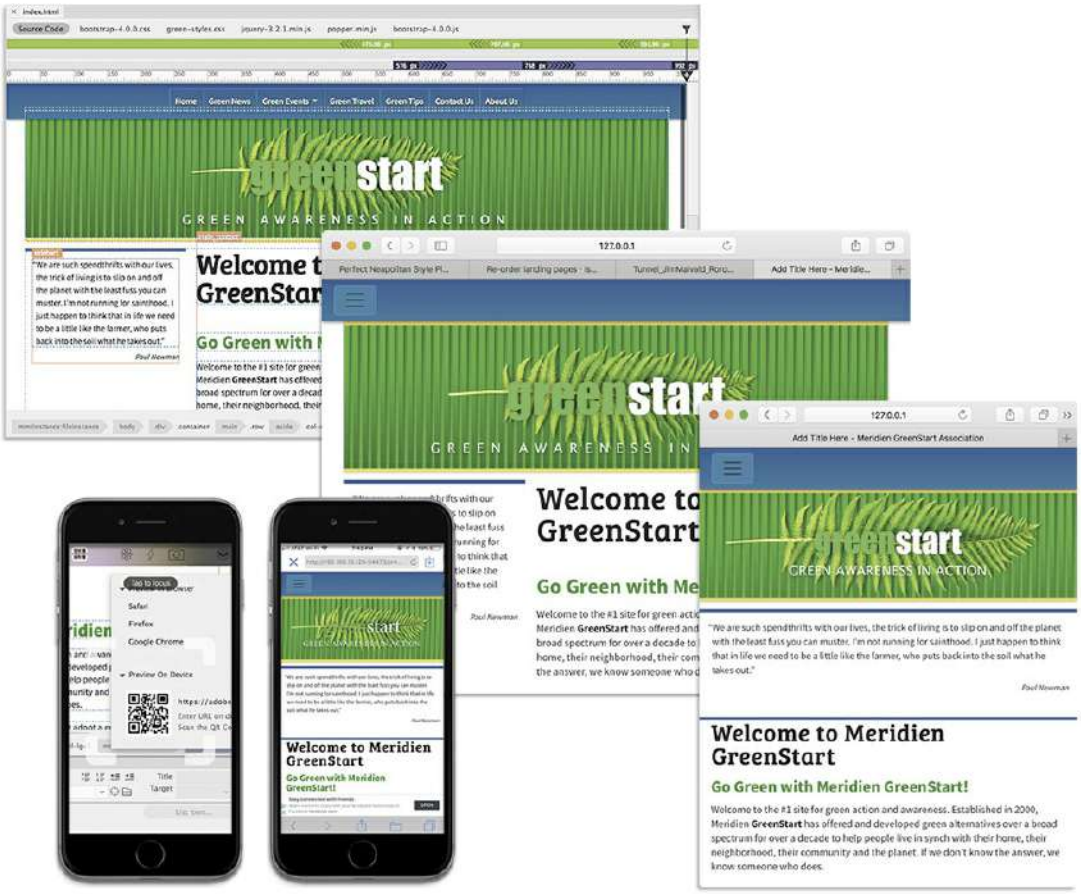
15 Adapting Content to Responsive Design

Lesson overview

In this lesson, you'll learn how to do the following:

- Review content to identify styling issues based on screen size
- Create custom media queries to target styling to specific screen sizes
- Create alternate Bootstrap column configurations
- Create custom table styling to support small screens
- Preview pages in various browsers and devices using Real-Time Preview

This lesson will take about 3 hours and 15 minutes to complete. Please log in to your account on peachpit.com to download the project files for this lesson, as described in the "Getting Started" section at the beginning of this book. Follow the instructions under "Accessing the Lesson Files and Web Edition." Define a site based on the lesson15 folder.



In most cases, text, images, and other HTML components will adapt automatically to a Bootstrap layout. When they don't, you'll have to know some techniques to get them to look right and work correctly.

Updating responsive design

Nothing in life is perfect. Although you have created a responsive template based on a powerful responsive web framework, you will still have issues with the basic design as well as with the content you try to deploy. The purpose of this lesson is to walk you through the process of identifying these issues and to demonstrate some techniques for adapting the layout and content to various screens and devices.

In some cases, the issues you will see are intrinsic to the layout. Others will be based on specific content or components used within the layouts. And, finally, some issues you will discover affect only a specific screen size or device. There is no way to find all the issues within this short lesson. Some will be discovered only after weeks of use and testing. Others may be reported by conscientious visitors or staff. But don't get complacent or lazy. It's easy to think that your HTML is fine because it works on your own computer and devices.

Testing and correcting errors is an ongoing and seemingly never-ending process. And what's worse is that HTML, CSS, JavaScript, and the Internet itself are constantly changing and

evolving. What works today may not work next week or next year. The teams of developers working to improve the web always try to make their changes backward compatible, but that's not always possible or practical. There's only one thing you can be certain of: things will break.

Inspecting the current site

The first step in troubleshooting is to put eyes on all the pages and review all the content. Yes, all of it. I can guarantee from personal experience that the first time you think that an item is okay and that you don't have to test it, it will be broken or display badly. And worse, it will often be your boss or client who finds the problem first. You never want to be in that position.

You must test every page and review them on multiple screen sizes and devices. I work on a Mac laptop, but I've installed Windows on my Mac and I even have a separate desktop computer with Windows installed on it. I own an iPhone and an iPad, but I went out and bought a used Android phone and tablet so I could test all my sites in both worlds. I try to test every page and every component in all those environments and in multiple browsers. And don't forget portrait and landscape orientations either. But even with all that effort you will still miss something. Be ready to jump on any report of a page or content that is misbehaving.

In the following exercises, we'll take look at some techniques for reviewing your pages and content.

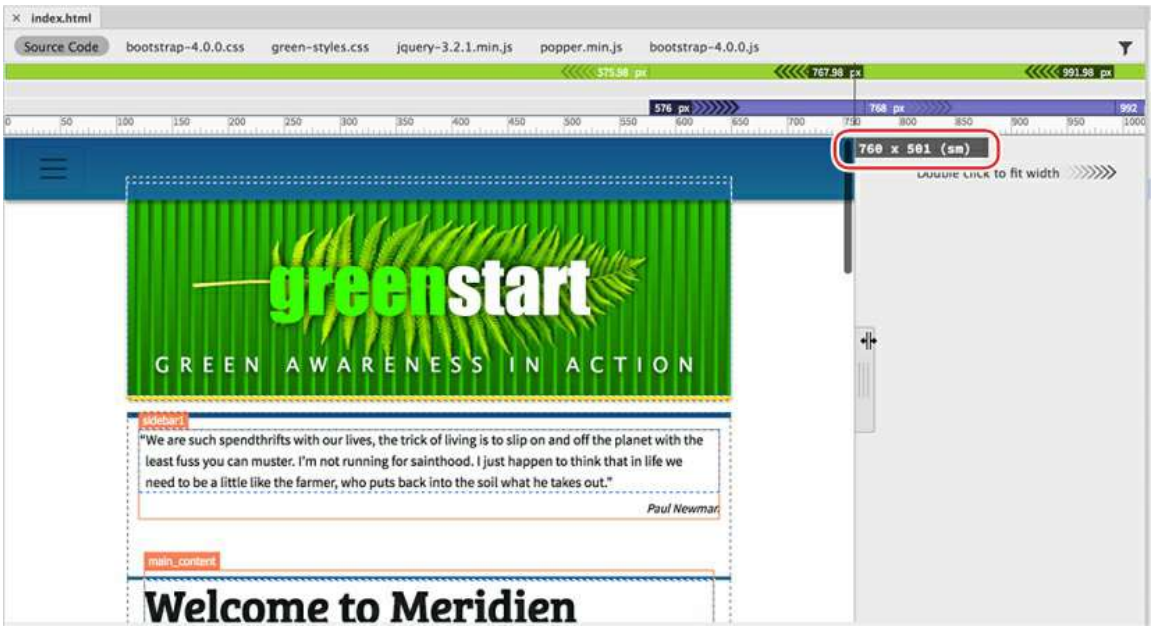
- Open **index.html** in Live view from the lesson15 folder.

  Make sure the document window is open to a minimum width of 1100 pixels.

- Scroll down the page to observe the footer. Scroll back to the top of the page.

  The home page contains various kinds of text and an image. Scan the page carefully, looking over all the elements and styling. Try to note the spacing for margins and padding, as well as line height and the spacing between paragraphs. If a page is longer than your screen height, make sure you scroll all the way to the bottom. When reviewing content, it's important to test the pages at different screen sizes too.

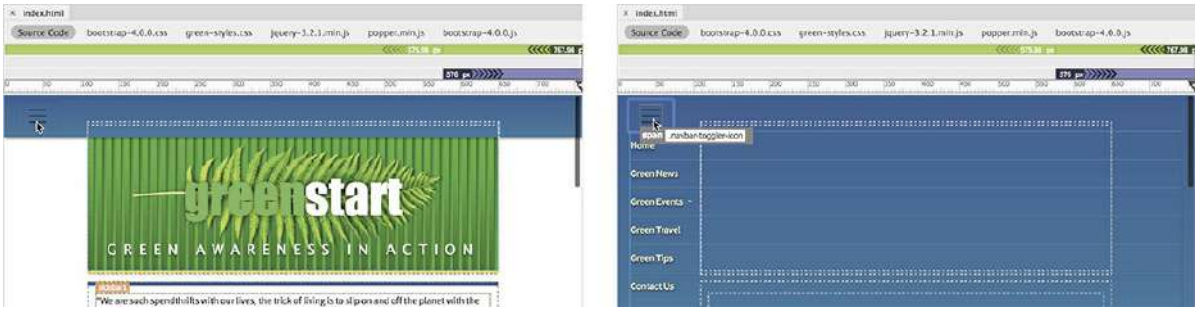
- Drag the scrubber to the left to simulate the width of a screen 768 pixels wide or less.



Note how the three-column layout reacts to the narrowing screen. Pay close attention to the widths and balance of each column.

As the screen narrows, the columns resize to share the space evenly until the screen width reaches 768 pixels. As soon as the screen drops below 768 pixels, the three-column design converts to a one-column design, with all the content stacking vertically. At that moment, the horizontal menu switches from seven individual buttons to a solid blue bar with a sandwich icon appearing in the left corner.

- Click the sandwich icon to open the menu.

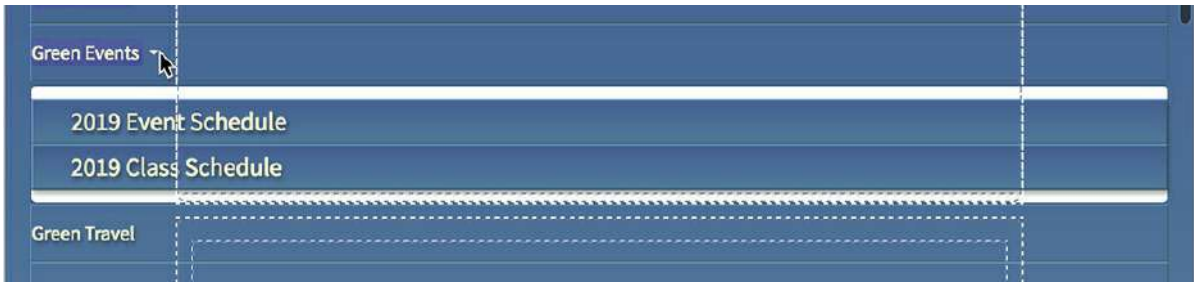


When you click the sandwich icon, the navigation menu drops down, showing all seven links stack vertically. Note that the *Green Events* menu item shows an arrow icon, indicating that it hosts its own dropdown menu.

- Position the cursor over each menu item and note the behavior of each item and the appearance of the cursor.

The menu items react with the hover behavior as the cursor passes over them.

- Click the *Green Events* menu item.



The submenu opens, showing the two sublinks to the event calendar and class schedule. Note the styling of the sublinks and the gap above and below them.

- Click the *Green Events* menu item a second time to close it.
- Drag the scrubber all the way to the right side.

  Observe how the page reacts to the screen as it widens.

How many issues did you see in the basic layout or design? Did you catch them all? Let's take a look at the major issues and see how you can correct them.

Identifying site design issues

Unfortunately, the site did not pass the first test. The design has some major problems. But don't worry—all the issues observed are normal and expected in a new responsive site. Luckily, everything can be fixed pretty easily with some simple CSS tweaks.

The first item you should have noticed is that the top border of the header element is hidden beneath the navigation menu. When you built the responsive menu in the previous lesson, it aligned perfectly. Something happened to it, and now the border is no longer visible. Things like this happen all the time when you are building pages and adding and styling content. You have to be constantly vigilant.

- In CSS Designer, click the All button.

  Select **green-styles.css** in the Sources panel.

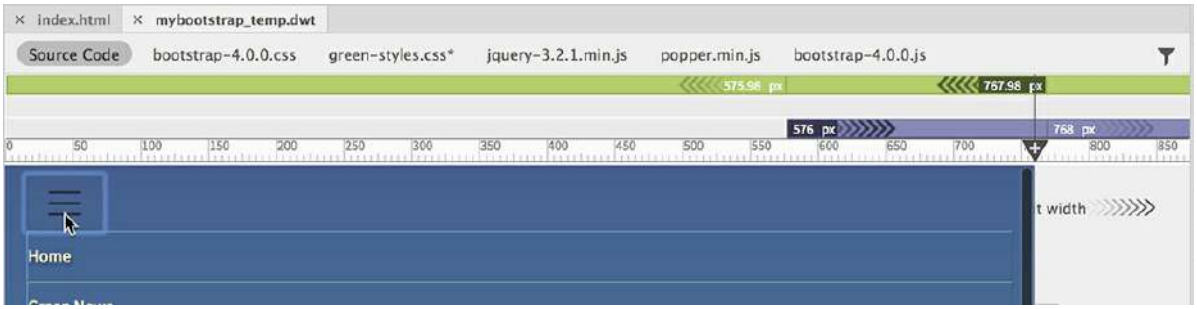
  The Selectors panel displays all the rules defined within the style sheet. The margin setting that controls the placement of the border is defined in the rule `header`.

- Select the rule `header`.

  Edit the following property: `margin-top: 4em`

This added spacing fixes the `header` display. The next issue we can correct is the gray border on the sandwich icon of the navigation menu when the screen is less than 768 pixels.

. Drag the scrubber to a width of 768 pixels or less.



Note

The border appears light blue when the element is selected.

You can see that the border around the sandwich icon is dark gray. The background color of the sandwich is the same as the rest of the menu bar. The whole color scheme makes it difficult to see amid the dark blue background. A lighter color scheme would make it easier to see. Normally, you could use the Current button in the CSS Designer to test and identify components on the page, but the horizontal menu is part of the locked portion of the site template. As of this writing, the CSS Designer ignores elements in the locked areas. To track down the pertinent rules, you'll need to work with the template itself.

. Open **mybootstrap_temp.dwt** in Live view from the lesson15/templates folder.

In the Dreamweaver templates, all elements are selectable and editable. You should be able to identify the specific rules formatting the sandwich icon and other components within the editable regions.

. Drag the scrubber to 760 pixels.

. In the CSS Designer, click the Current button.

Click the sandwich icon and observe the Selectors pane.

The Selectors panel lists the rules supplying some form of styling to the sandwich icon. Start at the top of the list and examine each until you find the rule, or rules, formatting the borders and other aspects of the sandwich icon. What you may notice is that there are two different components that compose the menu icon: `navbar-toggler-icon` and `navbar-toggler`. Both items set border properties, but only `navbar-toggler` sets a border color. To override the border and other non-scheme colors, you need to create new matching rules in your custom style sheet. Note the exact name of the existing rule.

. Click the All button.

Select **green-styles.css** and create the following rule:

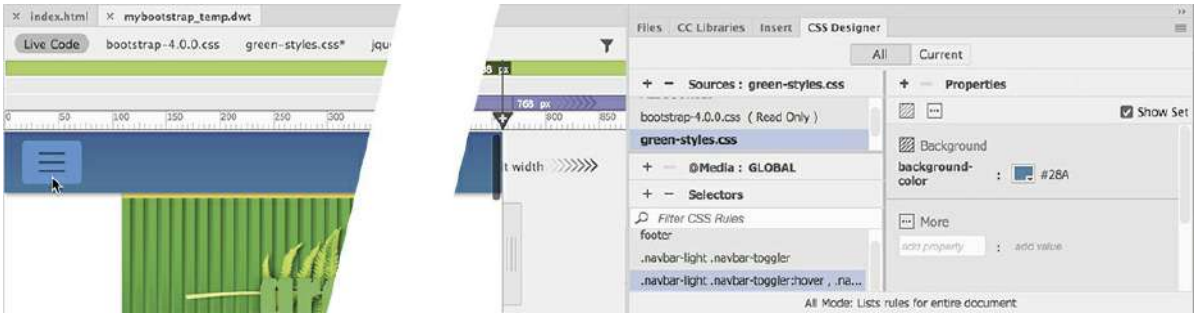
```
.navbar-light .navbar-toggler
```

. Add the following property to the new rule:

```
border: solid 1px #29C
```

This rule resets the borders around the outside of the icon. But the darker background color makes it hard to see the bars within the icon.

. Add the following property: `background-color: #28A`



The lighter background color now makes the three bars within the sandwich icon easier to see. Let's add a hover behavior to indicate the nature of the interactivity.

. Create the following rule:

Click here to view code image

```
.navbar-light .navbar-toggler:hover ,
.navbar-light .navbar-toggler:focus
```

Note

Don't forget the comma between the two parts of the selector.

This rule targets the `:hover` and `:focus` states of the sandwich icon.

. Add the following property: **`background-color: #29D`**

That addresses all the styling issues with the sandwich icon, but you may remember that the submenu also had some styling issues.

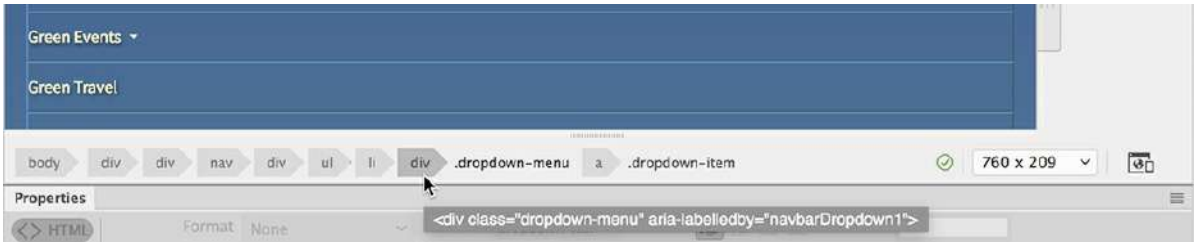
. Click to open the horizontal menu.

Click to open the *Green Events* sub-menu.

Notice the gap above and below the submenu items. One clue to help you find the appropriate rule to reset is the border radius visible on the corners of the dropdown menu.

. Click one of the subitems in the dropdown menu.

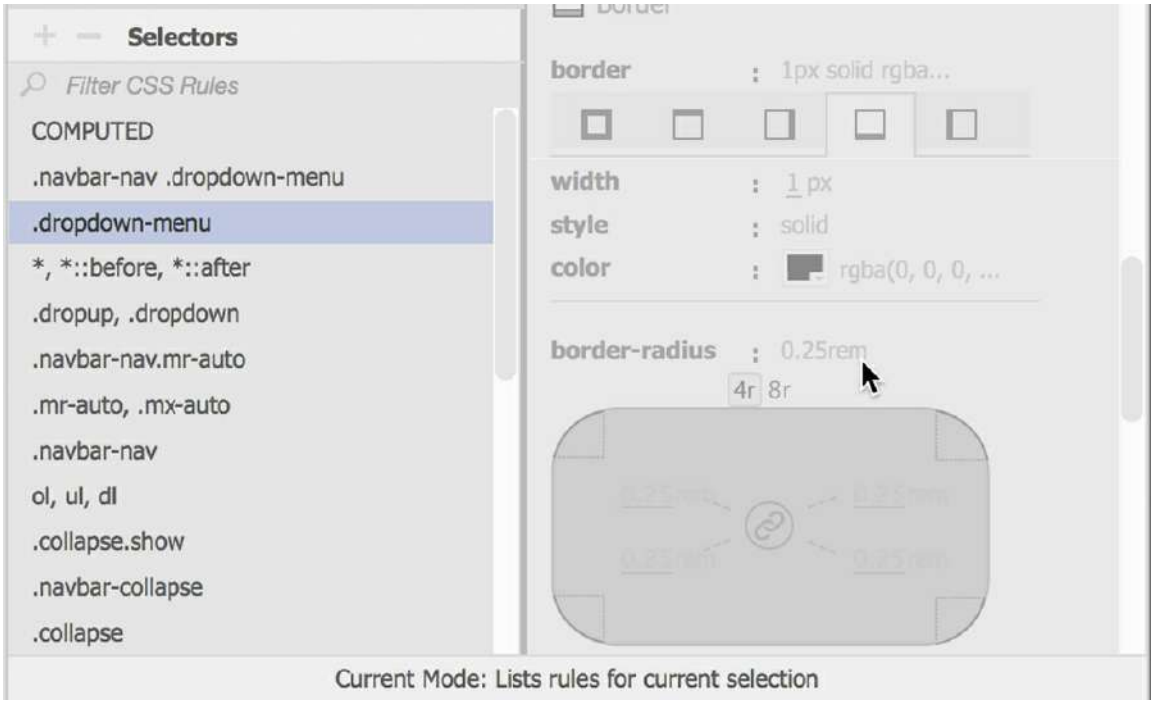
Examine the tag selectors.



The dropdown menu is constructed using two `<a>` elements wrapped by a `<div>`.

. Click the tag selector `div.dropdown-menu`.

Click the Current button and examine the rules styling the menu.



It's helpful when an HTML element has some sort of obvious styling. The border radius on the dropdown menu makes it easy to identify the specific rule styling it.

. Click the All button.

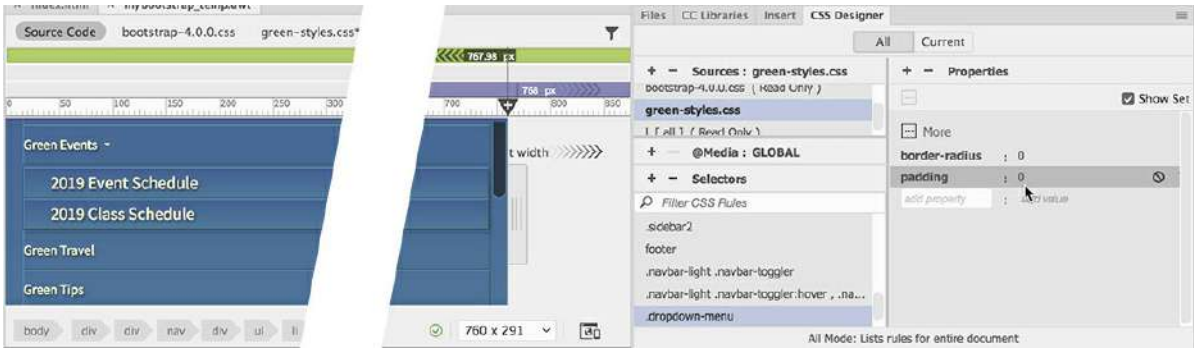
. Select **green-styles.css** and create the following rule:

```
.dropdown-menu
```

. Create the following properties:

```
border-radius: 0
```

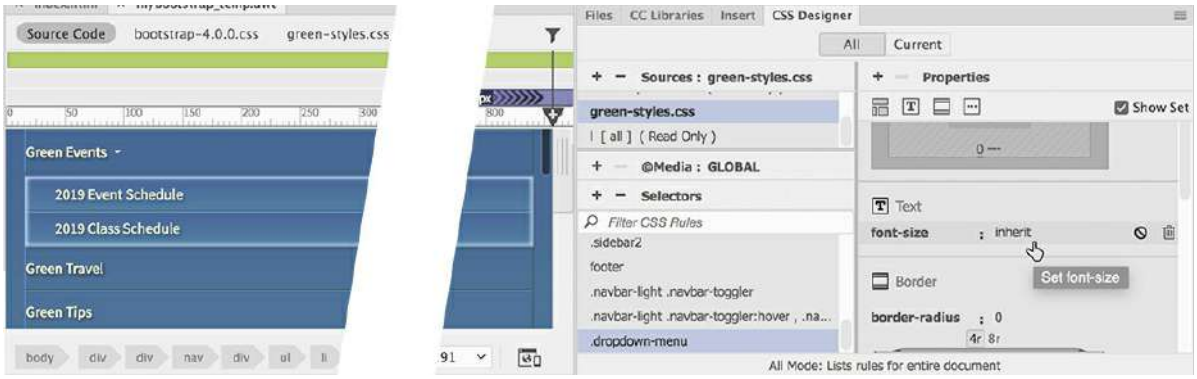
```
padding: 0
```



The new rule eliminates the gap above and below the submenu. You may also notice that the text in the submenu is slightly larger than the other menu items.

. Create the following property:

```
font-size: inherit
```



Using the inherit setting allows the styling applied to the other menu items to pass through to the submenu. That way, if you change size of the main menu, the submenu will update automatically.

- Save all files.

The changes in this exercise have brought the styling of the navigation menu within the site's design and color scheme. Close the menu and drag scrubber to the left and review the rest of the layout to identify any other styling issues.

Below 768 pixels in width, the content starts to display in a single column. If you watch carefully, at a certain point the company name and logo are too large for the available space and the motto breaks into two lines. You can see the second line displayed in the middle of the quotation.

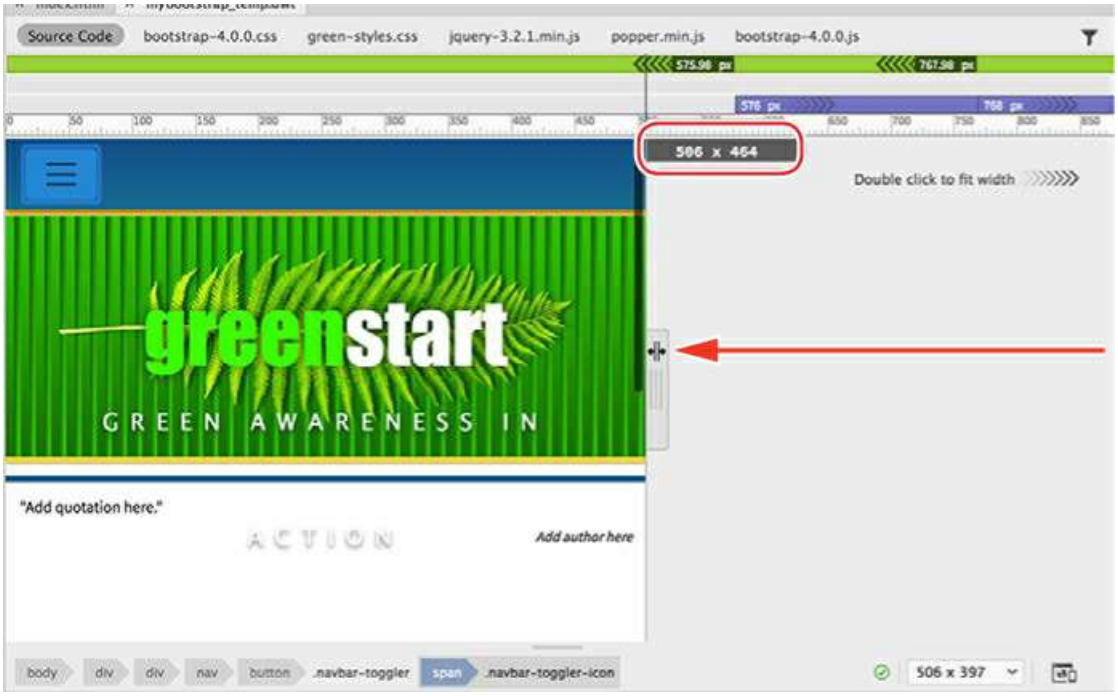
As you can plainly see, when the screen gets too small, the styling of the header is not scaling to fit the available space. There are several solutions to this issue, but the simplest method is to change the way the size of the logo and motto is applied. Instead of applying the font sizes as a global specification, you will apply them based on the size of the screen. To apply CSS styles based on a specific screen width, you first need to learn how to create a custom media query in your style sheet.

As you learned earlier, a media query is a tool you can use to target styling to specific screen sizes, orientations, and devices. In most cases, to deal with conflicts with your page content or with specific design requirements, you can simply add custom media queries and CSS rules to your own custom style sheet.

Adapting the header to smaller screens

If the global styling cannot serve the purpose of applying the needed styling, you will often need to create custom media queries and rules. In this exercise, you will create another media query to adapt the header to smaller screens.

- Open **mybootstrap_temp.dwt** in Live view, if necessary.
- Drag the scrubber left and right to identify the exact screen width where the motto breaks to two lines.



On my computer, the text breaks around 520 pixels or so. The exact width will depend on many factors, including, but not limited to, operating system, screen type, browser type, and version. In other words, don't count on whatever number you see on *your* system at the moment. Always add some extra space so that your new media query doesn't break right out of the box.

. Drag the scrubber to 530 pixels.

. Click the Add Media Query icon at the top of the scrubber.

Note

The purpose of the media query is to keep the motto on one line. Your pixel position may differ from the one described here or pictured in subsequent screenshots. Substitute your measurement in the following exercises, as necessary.

The Media Query Definition dialog appears. The `max-width` field is populated with the current scrubber position (530 px). The `min-width` field is grayed out, and **green-styles.css** is already selected in the Source menu. The new media query will apply its styles only when the screen is 530 pixels or narrower.

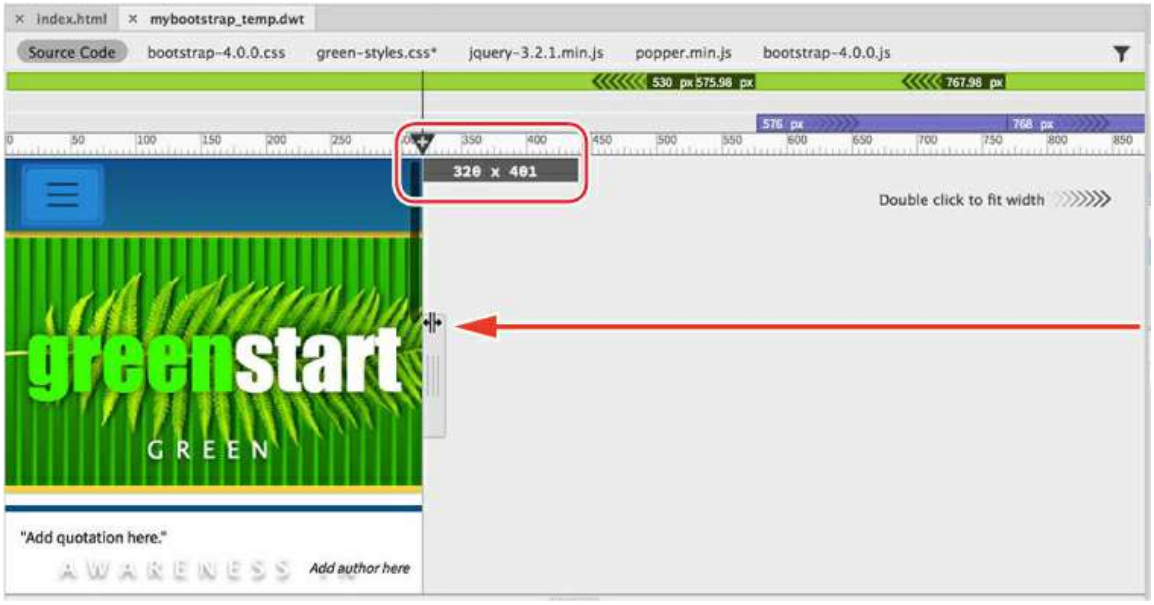
Note

If **green-styles.css** does not appear in the pull-down menu automatically, select it manually.

. Click OK to create the media query.

A new media query appears in the VMQ. The query is active immediately, but at this width, the motto is currently displayed on one line. It would help when adjusting the styling to set the scrubber at the smallest screen size you would need to support before making any new rules.

. Drag the scrubber to 320 pixels.



The width of 320 pixels is the size of the original iPhone and should be the smallest device you have to worry about. Notice that the logo extends off the edges of the screen and that the motto is showing a single word. Let's deal with the motto first.

- Switch to Split view, if necessary.
- Click the motto in Live view.

  If the Element Display does not appear highlighting the motto in Live view, insert the cursor in the motto text in the Code view window.

- Select the `p` tag selector.

  When you create new rules in a media query, there is a specific way to do it to make sure it gets added to the right place in the style sheet. The CSS Designer is integrated closely with the document window and the items selected therein. If you use the following method, your rules will always be added properly.

- Click the All button in the CSS Designer.

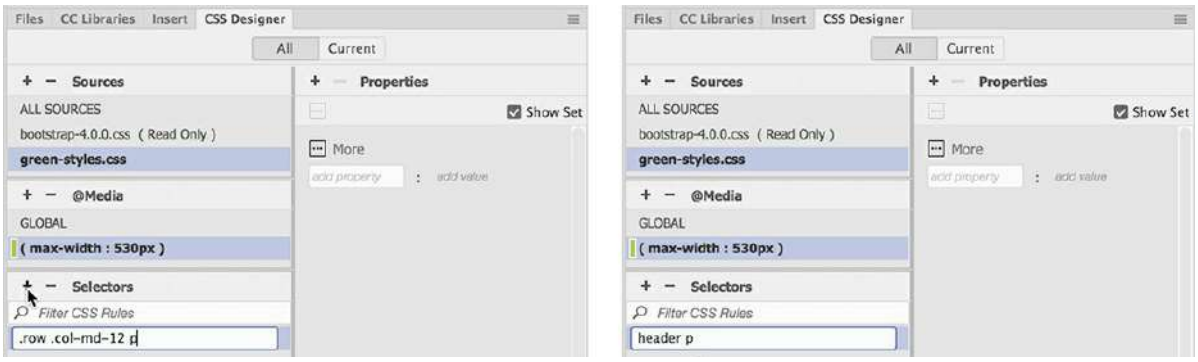
  Select **green-styles.css** in the Sources pane.

  Select `(max-width: 530px)` in the @Media pane.

  Click the Add Selector icon ✚.

  A new selector, `.row .col-sm-12 p`, is created in the Selector panel. It is based on the Bootstrap structure and does not match the original rule you created. To reset styling, it is essential that the selectors be identical.

- Create a new selector. Edit the selector name to **`header p`**



- Create the following properties:

  `letter-spacing: 0`

  `line-height: 2.5em`

The motto should now appear on one line, although you may need to refresh the display in Live view. The new styling will work on screens smaller than 531 pixels.

. Drag the scrubber to 600 pixels.

The motto shows no letter spacing until the scrubber passes 530 pixels, and then it returns to its original styling. Next, you will adjust the size of the logo.

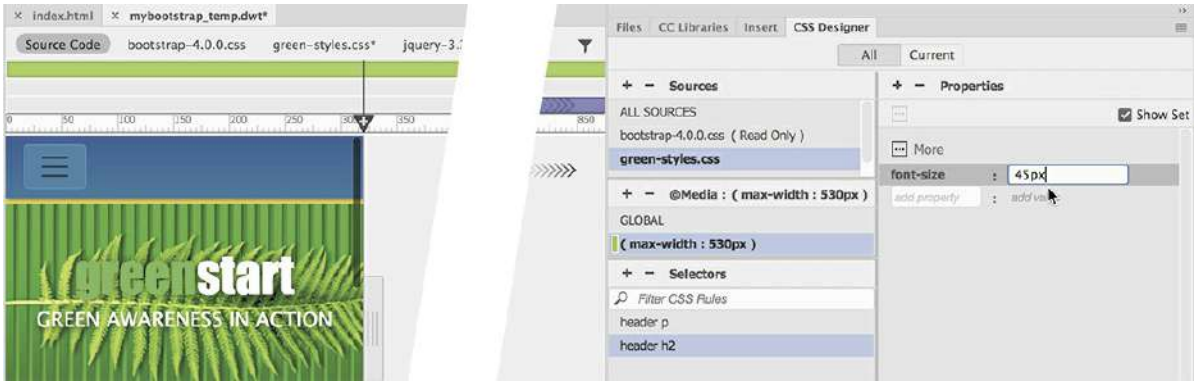
. Drag the scrubber to 320 pixels.

The logo is composed of the word *greenstart* and the fern image. You will have to reset the rules for both elements.

. Select **green-styles.css** > `(max-width: 530px)` in the CSS Designer.

Create the following selector: **`header h2`**

Create the following property: **`font-size: 45px`**



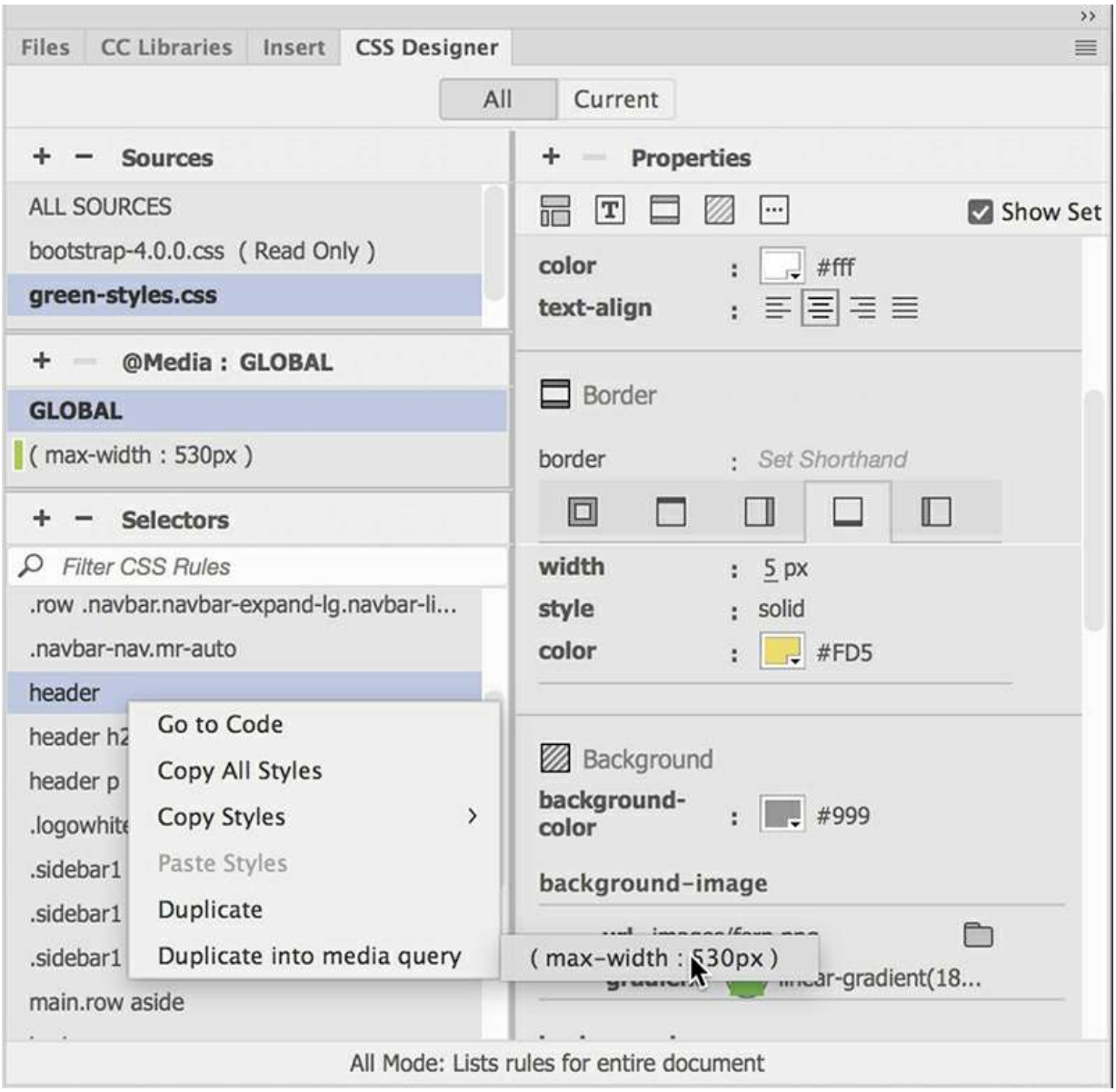
The heading reduces in size to fit the space better. The fern image is part of a complex background specification. The best way to modify a specification like this is to duplicate the entire rule into the media query and edit the appropriate settings.

Note

You may need to refresh Live view to see the changes.

- Select **green-styles.css** > `GLOBAL` in the CSS Designer.
- Right-click the rule `header`.

  Select Duplicate Into Media Query > `(max-width: 530px)` from the context menu.



  The entire rule is duplicated in the new media query. Since the background specification is so complex, the best place to edit it is in Code view.

- Select **green-styles.css** > `(max-width: 530px)` in the CSS Designer.

  Right-click the rule `header` and select Go To Code from the context menu.

  The document window switches to Split view, if necessary, and loads **green-styles.css** focused on the `header` rule. Since all the styles are coming from a global rule, you need to keep only the specifications that will reset portions of the background. The goal of every web designer

should be to minimize code whenever possible. You should delete any specification that is unneeded.

- Delete the following properties:

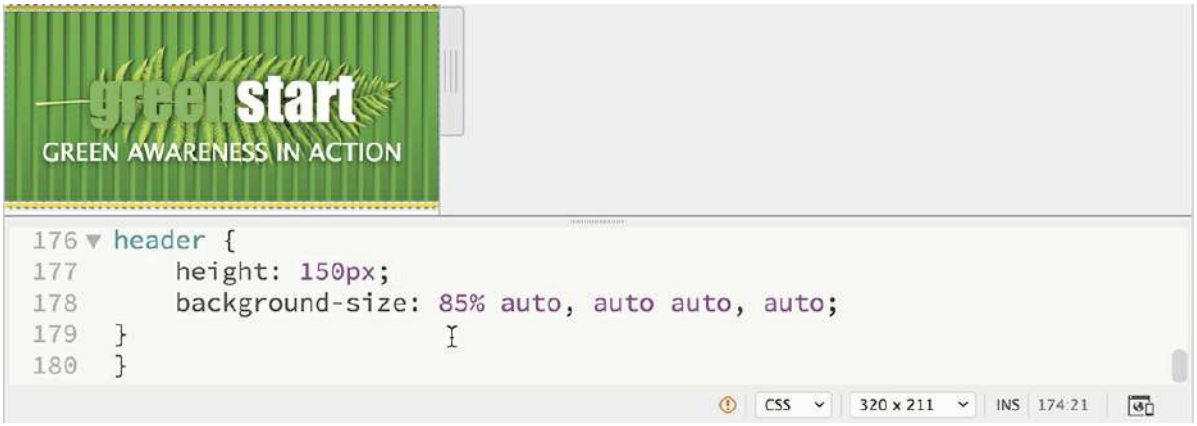
Click here to view code image

```
text-align: center;
background-color: #999;
color: #fff;
border-bottom: 5px solid #FD5;
border-top: 5px solid #FD5;
margin-bottom: 10px;
background-image: url(images/fern.png) , url(images/stripe.
png) , -webkit-linear-gradient(270deg,rgba(0,153,0,1.00)
0%,rgba(0,204,0,1.00) 100%);
background-image: url(images/fern.png) , url(images/
stripe.png) , linear-gradient(180deg,rgba(0,153,0,1.00)
0%,rgba(0,204,0,1.00) 100%);
-webkit-box-shadow: 0px 0px 3px 0px rgba(0, 0, 0, 0.55);
box-shadow: 0px 0px 3px 0px rgba(0, 0, 0, 0.55);
background-repeat: no-repeat , repeat-x;
background-position: 45% center, 0 0;
margin-top: 4em;
```

When these properties are deleted, you should see no changes in the header display. The existing properties are set as global settings and are inherited by the header element at all screen sizes. The only properties you need to keep are those that will be reset.

- Edit the following properties as highlighted:

```
height: 150px;
background-size: 85% auto, auto auto, auto;
```



These changes have greatly reduced the header footprint, which is appropriate for smartphones and other mobile devices. It’s also a good idea to reduce the size of the regular content.

. Select **green-styles.css** > `(max-width: 530px)` and create the following selector:

```
main.row section h1
```

. Create the following property:

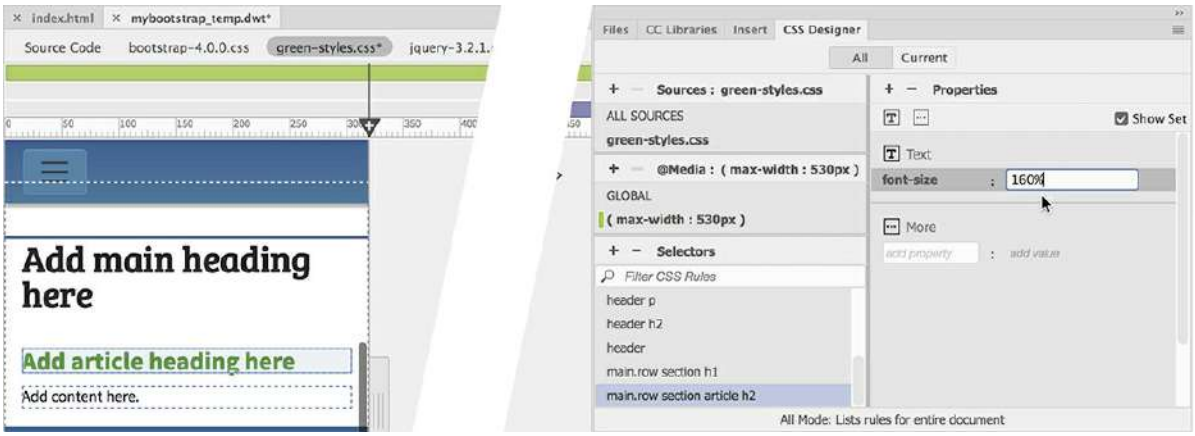
```
font-size: 225%
```

. In **green-styles.css** > `(max-width: 530px)` create the following selector:

```
main.row section article h2
```

. Create the following property:

```
font-size: 160%
```



Once you have updated the CSS, you should test the styling.

Note

If the Update Template Files dialog appears, go ahead and click the Update button.

. Save all files. Switch to Live view.

. Drag the scrubber to the right and left and observe how the content adapts to the screen width.

The text and background effects in the header and the headings in the main content change sizes seamlessly as you increase and decrease the width of the document window. Feel free to adjust any of the specifications to meet your own tastes and needs.

. Close all files.

This should take care of the styling for the basic content on this page for smaller screens. There are still other pages you need to review.

Adapting content to responsive design

There are seven pages in the GreenStart site. Each page contains text, images, and other types of HTML components. Converting the layout to a responsive design was only your first step. In the next few exercises, you will open each of the pages and address various issues to adapt the content to the new responsive design.

Re-creating float and indentation styles

On the *Contact Us* page, you created custom styling to highlight and indent the staff profiles. In this exercise, you will review the *profile* elements at multiple screen sizes and adapt them as needed.

. Open **contact-us.html** from the lesson15 folder in Live view.

Make sure the document window is at least 1100 pixels in width.

. Examine the page and content at full width.

Do you see any immediate issues? Since we're not using the GreenStart style sheet, the classes `.flt-lft`, `.flt-rgt`, and `.profile` do not exist. That means that the text in each profile is not wrapping around the staff photo. The first step is to re-create these styles.

Note

Don't forget to select the GLOBAL option in the @Media pane.

. Select the All button in the CSS Designer.

Select **green-styles.css** > `GLOBAL`.

. Create a new selector: **`.flt-lft`**

. Create the following properties:

```
float: left
margin-right: 10px
```

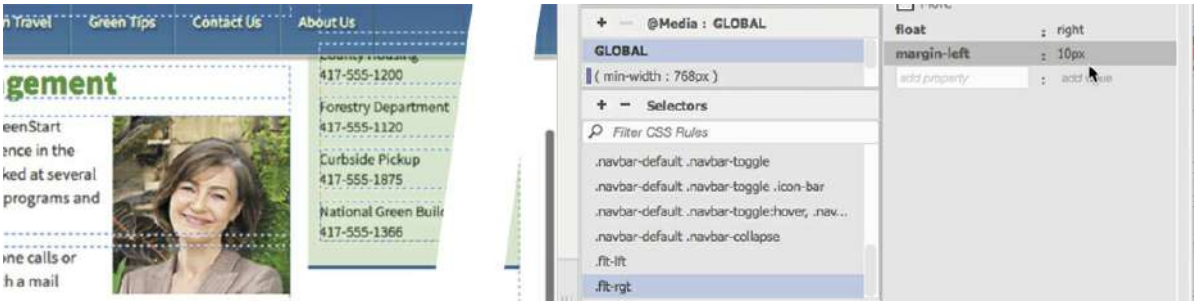
As soon as you finish the properties, the images styled with the class `.flt-lft` appear correctly formatted.

. Select **green-styles.css** > `GLOBAL`.

Create a new selector: `.flt-rgt`

. Create the following properties:

```
float: right
margin-left: 10px
```



The remaining images appear correctly formatted. The next step is to create the `.profile` rule.

. Select **green-styles.css** > `GLOBAL`.

Create a new selector: `.profile`

Create the following properties:

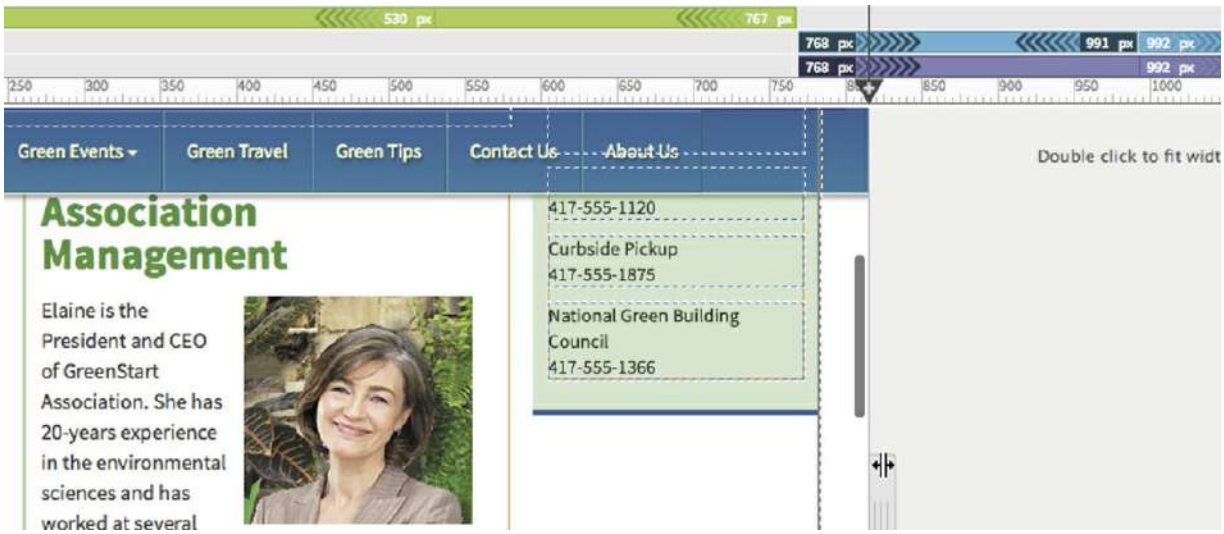
Click here to view code image

```
margin: 0 25px 15px 25px
padding-left: 10px
border-left: solid 2px #BDA
border-bottom: solid 10px #BDA
```

The original styling is complete and displayed in the layout. Let's test the formatting to see how it responds to the new responsive layout.

. Save all files.

. Drag the scrubber to the left to make the document window narrower. Test the layout down to a width of 320 pixels. Observe how the profile structure responds to the changing width.



The `.profile` section looks fine until you get down to widths between 768 and 991 pixels. In that range, the profile sections start to seem very cramped, with the text wrapping around the images. At 767 pixels, the content stacks in one column. It looks fine down to a width of 400 pixels. Below that, the left indent and border waste too much space. The interaction of the floated images and indented text with the changing width would best be fixed by first adjusting the basic layout. For the smallest screens, you'll provide alternate styling for the indents and floated images.

Creating alternate Bootstrap layouts

As the layout adapts to smaller screens, you may have noticed that the columns scale down to share the available space. At a certain width, the layout switches, or breaks, from three columns into a single column. This *breakpoint* is specified in the Bootstrap settings established when you first created the basic layout in Lesson 13, "Designing for Mobile Devices."

Unfortunately, the three-column design, which looks fine at full size, doesn't work very well at the intermediate sizes between 768 and 991 pixels. In this range, the layout would be better split into two columns instead of three. Luckily, Bootstrap makes it easy to create alternate layouts by simply changing some of the classes assigned to the layout elements.

In this exercise, you will create an alternate two-column layout by editing the existing classes and adding new classes to the Bootstrap structural elements.

. If necessary, open **contact-us.html** from the lesson15 in Live view.

  Make sure the document window is at least 1100 pixels in width.

  The underlying Bootstrap framework features five basic breakpoints: extra-small, small, medium, large, and extra-large. At this moment, the layout breaks only at a width of 767 pixels (medium).

  If you look at the code, you will see the classes assigned to the basic column elements. Each class is typically broken into three parts, such as `col-md-6`. The part "`col`" refers to

"column," the part "md" refers to "medium," and the part "6" refers to the number of grid sections. That means, in the current layout, the class `col-md-6` causes the element to span six grid divisions when the screen is 768 pixels or larger.

To change the layout from three to two columns, you first need to change the existing classes. The classes you need to change are outside the editable regions, so you will have to make the changes in the template.

- Open **mybootstrap_temp.dwt** in Split view.

Make sure the document window is at least 1100 pixels in width.

You want to change the layout from three columns to two columns between

768 and 991 pixels. It's always a good idea to set up the workspace to that environment so that you can see the changes visually.

- Drag the scrubber to 900 pixels.

The scrubber is positioned within the medium breakpoint. You'll find that Live view doesn't respond properly with the template markup in place. The following changes will have to be made in the Code view window.

- Locate the `<aside class="col-md-3 sidebar1">` element in the Code view window (around line 58).
- Edit the class as highlighted: `.col-md-4`

This change causes the first column to span four grid divisions. Sidebar 2 doesn't have enough space to appear on the same line and drops down below the first two columns. Now you will style the second column to use the remaining space.

- Locate the `<section class="col-md-6">` element (around line 63).
- Edit the class as highlighted: `.col-md-8`

The main content area now occupies the remaining space, creating a two-column layout. Sidebar 2 appears below the two columns, but it is aligned to the left. A preferable design would be to move it under the main content and allow it to fill the second column. Bootstrap enables you to push and pull columns using another class.

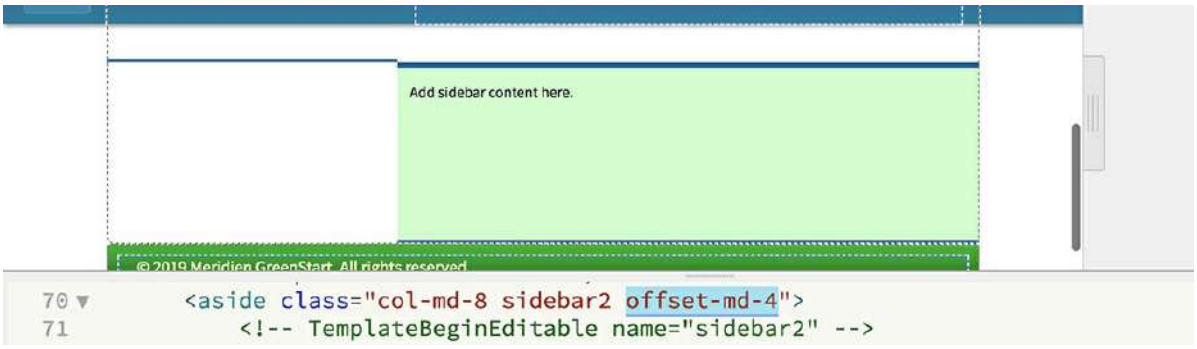
- Locate the `<aside class="col-md-3 sidebar2>` element.
- Edit the class name as highlighted: `.col-md-8`

The class change affects the width of the element. But to shift it over to the right, you will need to add another class.

- Edit the class for Sidebar 2 as highlighted:

Click here to view code image

```
<aside class="col-md-8 sidebar2 offset-md-4">
```



The class shifts Sidebar 2 under the main content area. The two-column layout is complete, but you now have to re-create the three-column layout for larger screens.

- Drag the scrubber to fully open the document window.

When fully open, the two-column layout scales to use the entire width of the screen. The classes you changed format the medium breakpoint. If no other classes exist, the larger breakpoints simply use the existing styles. To restore the three-column design, you'll need to add new classes to the same elements. In this case, you'll restore the three-column layout for the medium breakpoint.

- Locate the `<aside class="col-md-4 sidebar1">` element.
- Edit the class as highlighted:

Click here to view code image

```
<aside class="col-md-4 sidebar1 col-lg-3">
```

The new class reapplies the original width to the first column on large screens. Next, you'll restore the second column to the original width.

. Locate the `<section class="col-md-8">`element.

. Edit the element class as highlighted:

Click here to view code image

```
<section class="col-md-8 col-lg-6">
```

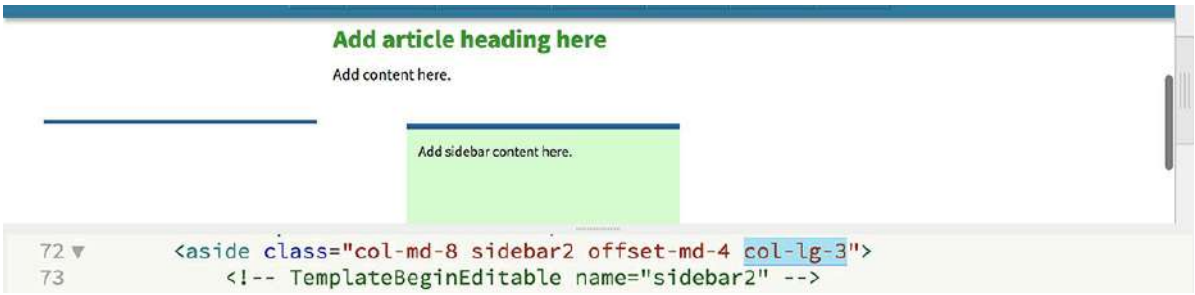
The main content area now returns to its previous size, leaving an open spot for Sidebar 2.

. Locate the `<aside class="col-md-8 sidebar2 offset-md-4">` element.

Add the following classes to Sidebar 2:

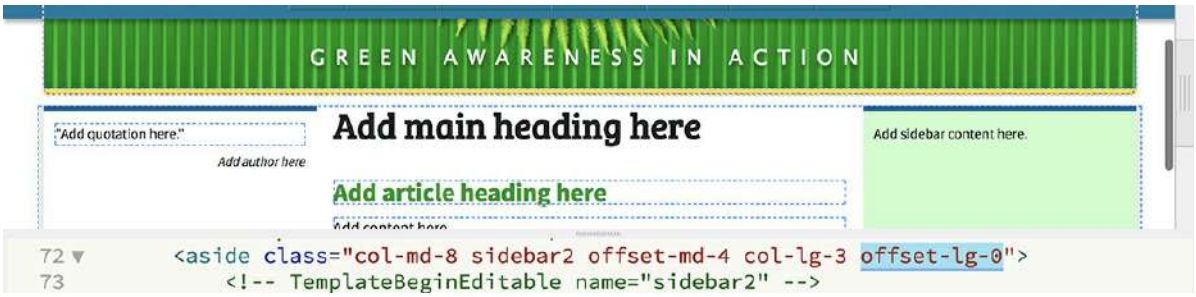
Click here to view code image

```
<aside class="col-md-8 sidebar2 offset-md-4 col-lg-3">
```



The class restores the width of Sidebar 2, but the element doesn't return to its original position. It appears outside the layout on the right side. Since you offset the element to align it to the second column, that class is still being applied. To reset the position of Sidebar 2, you have to apply another class to cancel out that styling.

- Add the following class to the element: **`offset-lg-0`**



Sidebar 2 moves into its expected position in the layout. Once the classes are applied, you should test the styling again.

- Drag the scrubber to 320 pixels.

  Observe the layout as it adapts to the width of the document window.

  The layout converts from three columns to two columns to one as the width narrows.

- Drag the scrubber to open the document window fully.

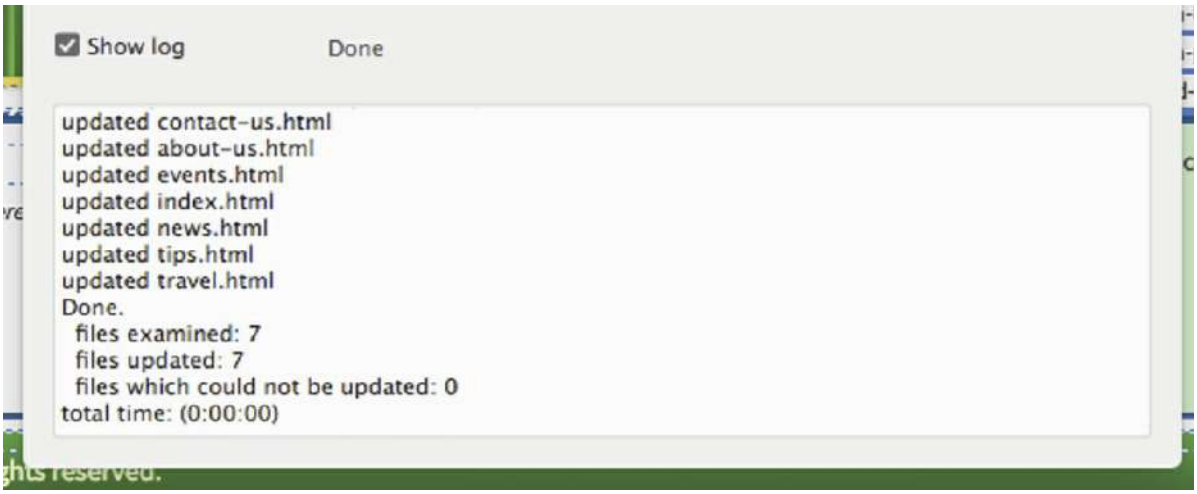
  The layout converts from one to two to three columns as the window opens. Once you are certain the styling is working successfully, you can apply it to all the pages in the site.

- Save the template.

  The Update Template Files dialog appears.

- Click Update to update all child pages.

  All pages in the site are updated with the new classes and design scheme.



. Close the Update dialog. Close the template.

When the template closes, the *Contact Us* page remains open. The asterisk in the document tab indicates that the layout was updated. Whenever you make global changes to the site, you should always review and test every page.

Adapting custom indents to responsive design

The *Contact Us* page now has the new layout scheme applied to it. Let's take a look now at how the new layout styling works with the staff profiles.

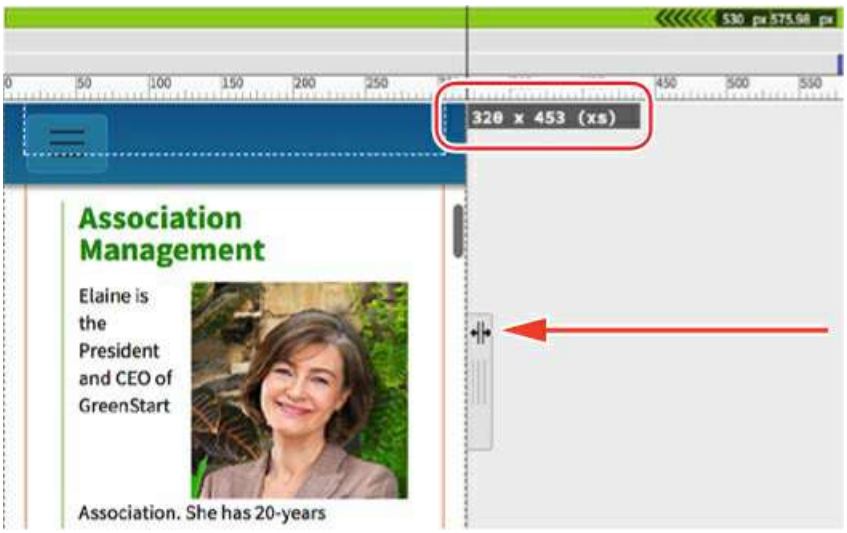
. Drag the scrubber to 900 pixels.

  The layout changes from three to two columns. The content fits nicely in the new layout.

. Drag the scrubber to 500 pixels.

  The layout changes from two to one column. The content looks fine when the document window is at 500 pixels, but how will it handle smaller screen sizes?

. Drag the scrubber to 320 pixels.



At 320 pixels, the layout is too cramped. The borders and indents are using up valuable space and no longer enhancing the content. Let's adjust the layout for the smallest screens.

. In the CSS Designer, click the All button.

Choose **green-styles.css** > `(max-width: 530px).`

Create the following selector: `.profile`

. Create the following properties in the new rule:

Click here to view code image

```
padding: 5px 0
margin: 10px 0
border-top: solid 10px #CADAAF
border-bottom: solid 5px #CADAAF
border-left: none
```

Below 530 pixels, the `.profile` section now expands nearly to the full width of the screen and drops the indents and the left border. There's a bit too much space above the heading in each profile. Luckily, you created a new selector for this element earlier.

- Select **green-styles.css** > `(max-width: 530px)` > `main.row section article h2`.

  Add the following property: `margin: .5em 0`

> **Note**
>
> You may need to click the Refresh button to see the changes in the layout properly.

- Test the new styles by dragging the scrubber left and right.

  The new styling for the profiles works perfectly and looks good too. Remember to test all new components at every screen size and orientation and make changes to the styling as needed.

- Save and close all files.

The next items you have to review are the tables created for the events and class calendars and the one used on the travel page.

Adapting tables to a Bootstrap layout

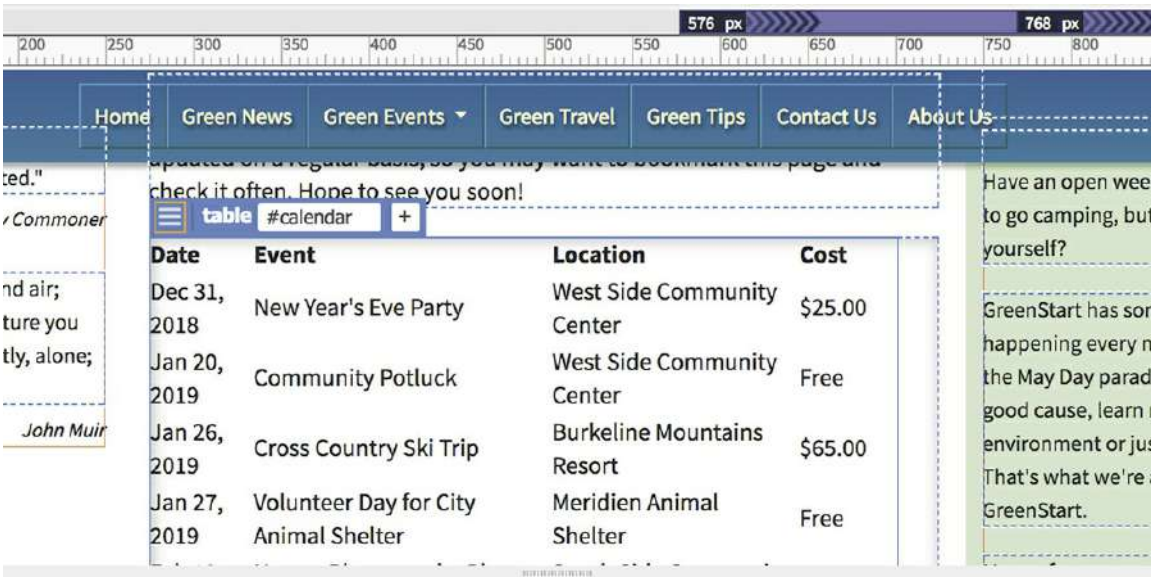
Before you tackle the concept of making the tables responsive, let's review the existing tables and see how they fared moving to the new Bootstrap layout.

Moving CSS rules between style sheets

In this exercise, you will identify the rules needed for styling HTML tables in one style sheet and move them into another.

- Open **events.html** in Live view.

  Make sure the document window is at least 1100 pixels in width.



As soon as the page opens, you can see that the CSS styling was lost in the transition to responsive design. The tables still bear all the content and CSS classes, but the new style sheet has none of the specifications that you created earlier. Luckily, those rules are all still available in **mygreen-styles.css**. Instead of creating all those rules again, let's just copy and paste them into the new style sheet.

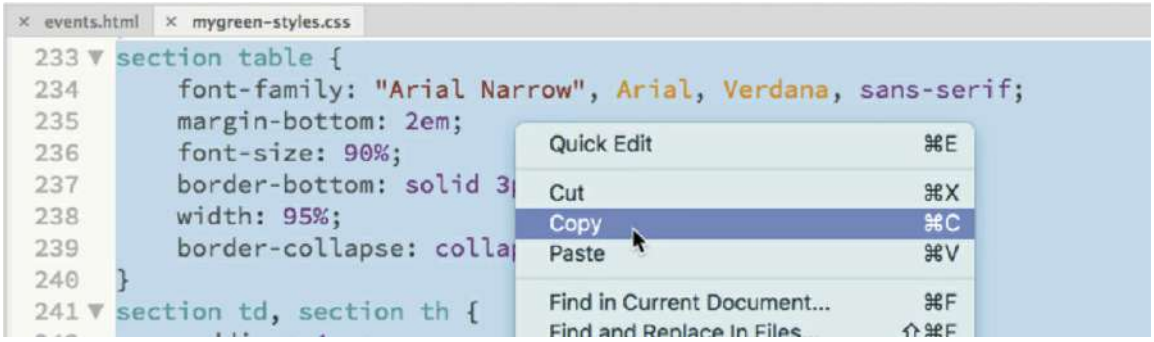
- Open **mygreen-styles.css**.

  The file contains all the styles from the original static GreenStart website. The table styles were created in the same lesson, one after the other. That means they should all appear consecutively in the style sheet.

- Scroll down through the style sheet to locate the first table style.

  Around line 233 you will find the rule `section table`. You will move all the table rules to the new style sheet.

- Select the CSS markup from `section table` (around line 233) down to the rule `table caption` (around line 284). Right-click the selected code and select Copy, or press Ctrl+C/Cmd+C. Be sure you include the properties for the `table caption` rule.



- Switch to **events.html**.

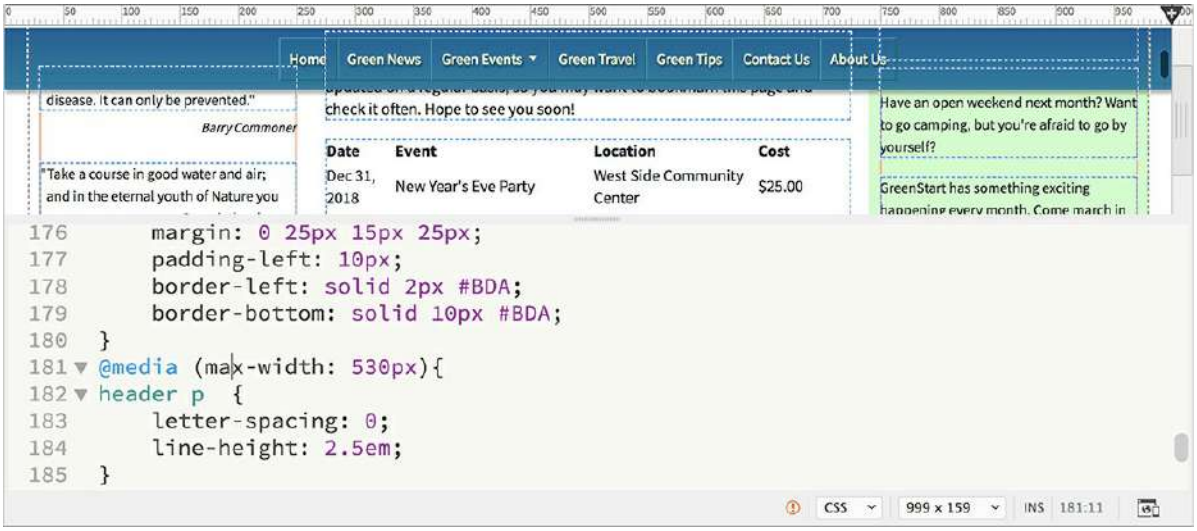
You don't need to open **green-styles.css** if you have one of the webpages that is linked to it already open.

- Select **green-styles.css** in the Related Files interface.

The document window switches to Split view and loads **green-styles.css** in the Code view window. It is very important to insert the styles in the proper spot of the style sheet.

- In the Code view window, scroll down to the entry

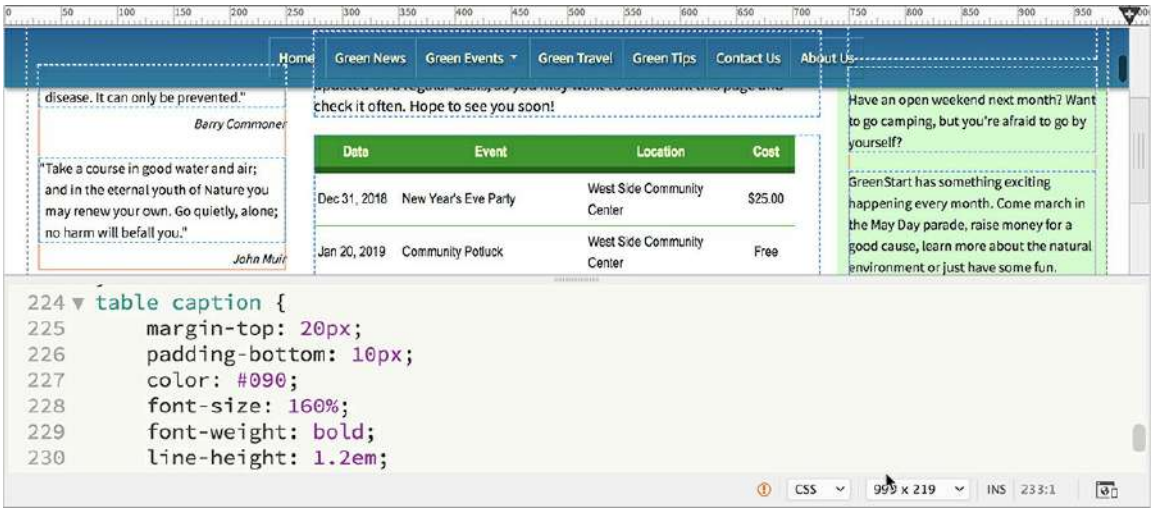
`@media (max-width:530px)` (around line 181).



This is the first custom media query you created earlier. It is essential that you insert the CSS you copied before this media query. You will see an opening curly brace ({) after this entry. All the rules for the media query are contained after this opening brace and before the closing curly brace (}), which appears, in this case, around line 209.

The table styling is considered global styling. It is automatically applied to and inherited by the tables. The styles in the media queries are designed to override the global styles. But that means the global styles have to appear in the style sheet before any media queries. Global rules inserted accidentally after the media queries could actually cancel out the styles in the media queries.

- Insert the cursor before the media query and press Ctrl+V/Cmd+V to paste the CSS markup. Press Enter/Return to move the entry `@media (max-width:530px)` to a new line after the markup you just pasted, if necessary.



. Save all files.

There's no functional need to move the media query entry to a separate line. It just makes the code easier to read and edit. As soon as you paste the CSS you should see that the tables are now formatted as you left them in Lesson 7, "Working with Text, Lists, and Tables." Once the tables are styled again, you can start adapting them to the new layout.

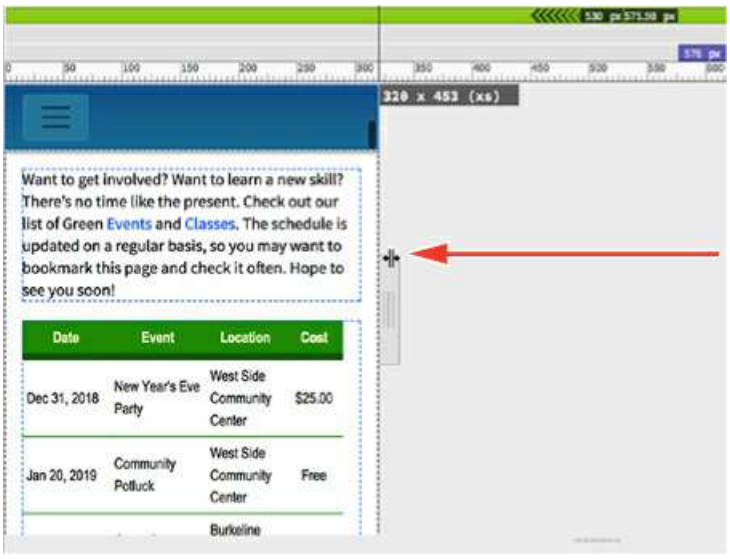
Making tables responsive

Tables are probably the last HTML element you'd think of when you think of responsive design. Tables are notoriously ill suited to smaller screens because they don't naturally adapt to them. But using some new CSS tricks, you'll learn how even tables can be made to be responsive.

. Open **events.html**, if necessary. Switch to Live view.

  Make sure the document window is at least 1100 pixels in width.

. Drag the scrubber to the left. Watch carefully how the tables respond, or don't respond, to the changing window size.



As the screen becomes narrower, the media queries kick in and reformat the page and components to adapt to the smaller screen. Since the table widths are set to 95%, they mostly scale down with the page. But once the screen width drops below 500 pixels, the text within the table is very cramped.

We need to rethink the whole concept of table design and display. You need to change the basic nature of the elements that compose tables so you can display them in a completely different way. During this process, you may sometimes find it easier to work in the CSS Designer; at other times you may want to enter the settings directly in Code view. Feel free to use whichever method feels more comfortable to you.

. Drag the scrubber to 400 pixels.

All the changes will be applied only to the smallest screen sizes.

. Open the CSS Designer, if necessary.

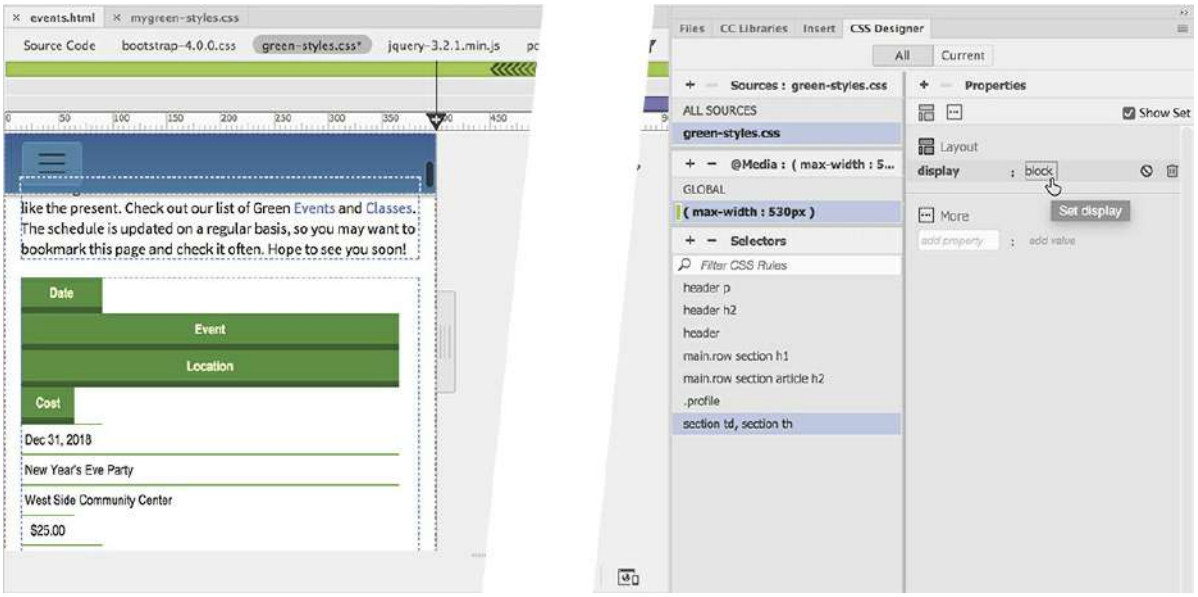
Select **green-styles.css** > `(max-width: 530px)`.

The basic element of the table is the cell, or `td`, element. Cells default to inline block styling. The first step is to alter their basic nature.

. Create the following selector: **`section td, section th`**

Table headers, `<th>`, are basically the same as table cells. You will format both the same way at first.

. Add the **`display: block`** property to the new rule.

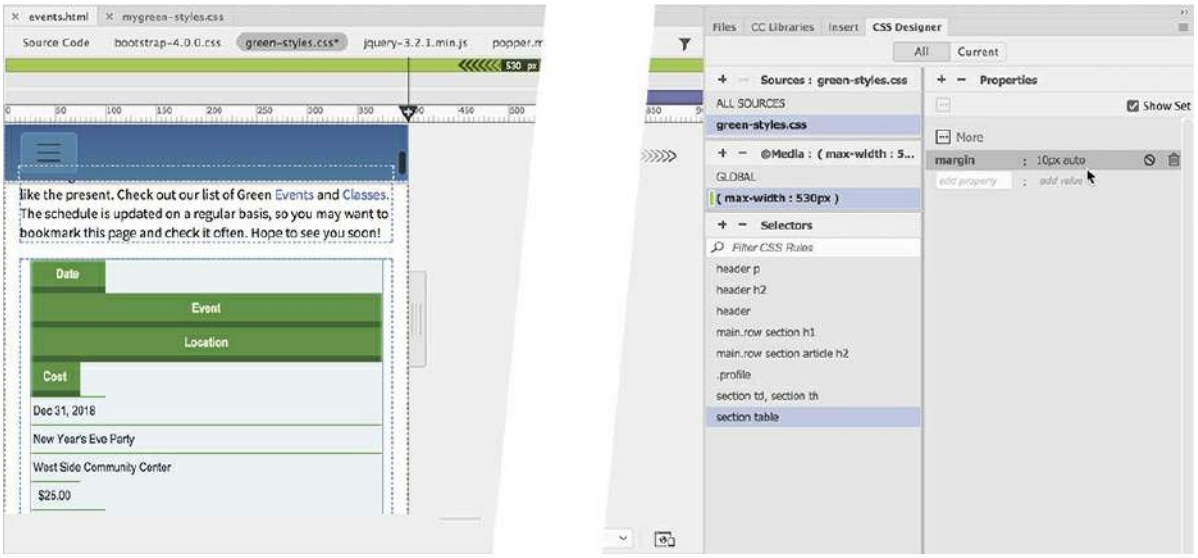


The table cells are now displayed vertically, stacking one atop the other when the width of the document window is narrower than 530 pixels.

This rule resets the default behavior of the table elements so that you can control their appearance on smaller screens. Some cells appear narrower than others because formatting is still being inherited from other parts of the style sheet. You'll have to create additional rules to override these specifications.

. Create the following rule: **`section table`**

Create the following property: **`margin: 10px auto`**



The tables are now centered in the layout.

With the data stacking vertically, it doesn't make much sense now to have a header row. You could set the header row to the `display:none` property to hide it, but that's not recommended for accessibility standards. The next best thing would be to simply format it to

take up no space.

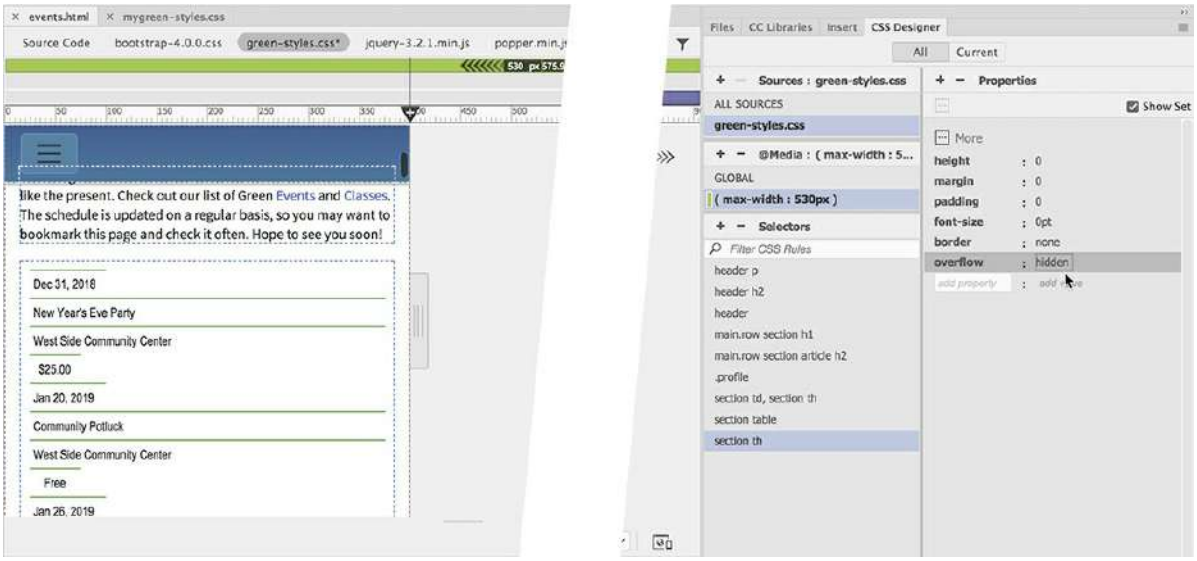
Note

Be sure that all subsequent rules and properties are added only to the custom media query.

. Create the following rule: `section th`

Create the following properties:

```
height: 0
margin: 0
padding: 0
font-size: 0pt
border: none
overflow: hidden
```



The header rows disappear visually but are still accessible to visitors using screen readers or other assistive devices. But now that they are invisible, you have to address the fact that there are no headers describing the data being displayed.

Note

This type of selector is called a pseudo-class and is related to the classes you created for link behaviors.

Note

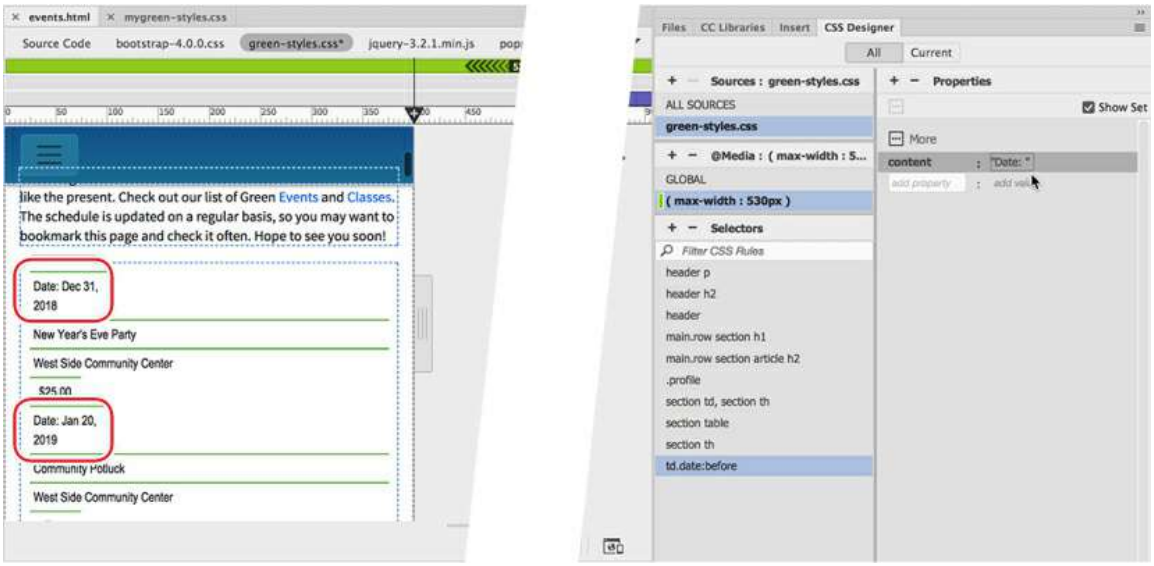
Make sure you add a space after the colon in the label. This will ensure that there is a space between the label and the cell content.

For this purpose, you'll resort to a new CSS3 property that can actually create labels based on the CSS class applied to the cell. Some of the latest CSS3 properties are not directly available in the CSS Designer, but you can enter them manually in the Properties window or in Code view, and Dreamweaver may provide hinting support for them as well.

. Create the following rule: **`td.date:before`**

. Enable the Show Set option.

Enter the following property—value combo: **`content: "Date: "`**



Notice that the label `Date:` appears in all the cells styled by the `date` class. You need to make a similar rule for each of the data elements.

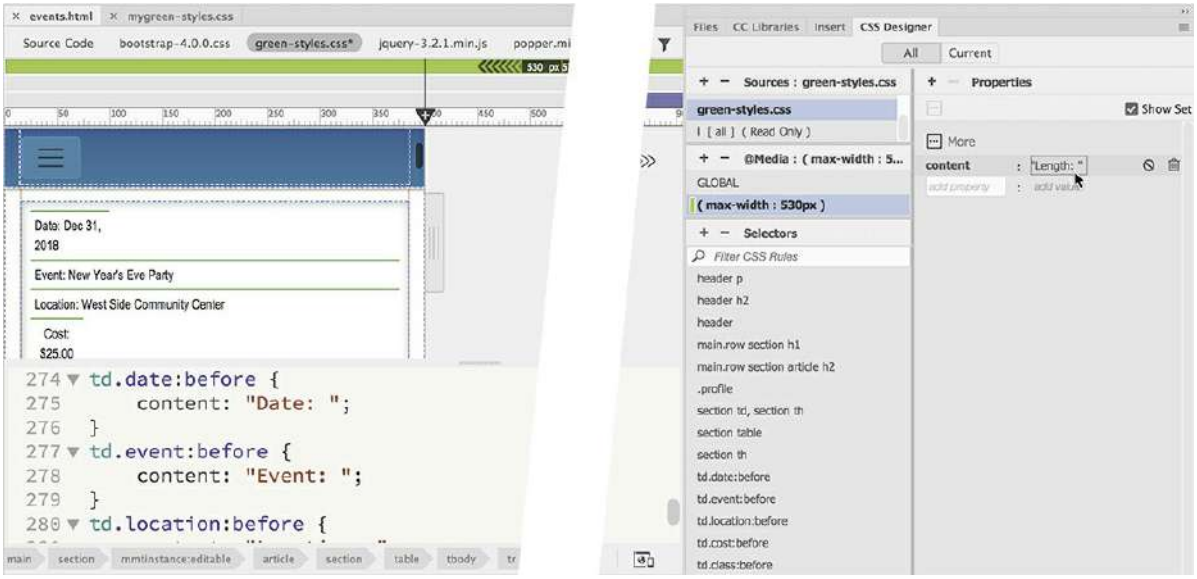
Note

Be sure that all subsequent rules and properties are added only to the custom media query.

. Repeat steps 9 and 10 to create the following rules and properties:

Rule	Property: Value
`td.event:before`	`content: "Event: "`

```
td.location:before      content: "Location: "
td.cost:before          content: "Cost: "
td.class:before         content: "Class: "
td.description:before   content: "Description: "
td.day:before           content: "Day: "
td.length:before        content: "Length: "
```



Note

If you make a single selector, as shown in step 12, do not add a comma on the last selector, which would disable the rule altogether. If you don't make a single selector, be sure to remove the comma from each one.

Tip

Creating long selectors may be easier to do in Code view. You can access the **green-styles.css** file by clicking its name in the Referenced file interface at the top of the document window.

Each data cell now shows the appropriate labels. CSS can also style the data and labels.

. Create the following selector:

```
section .date,
```

```
section .event,

section .location,

section .cost,

section .class,

section .description,

section .length,

section .day
```

I typed this rule on separate lines to make it easier to read, but you should enter it as one long string in the selector name field or in Code view. As long as the styling for all the data cells is identical, you can combine all the selectors into a single rule, separated by commas. Remember to mind the punctuation and spelling carefully. Even a tiny error in the code can cause the formatting to fail. If you want the styling to be different in one or more of the elements, then create eight separate rules.

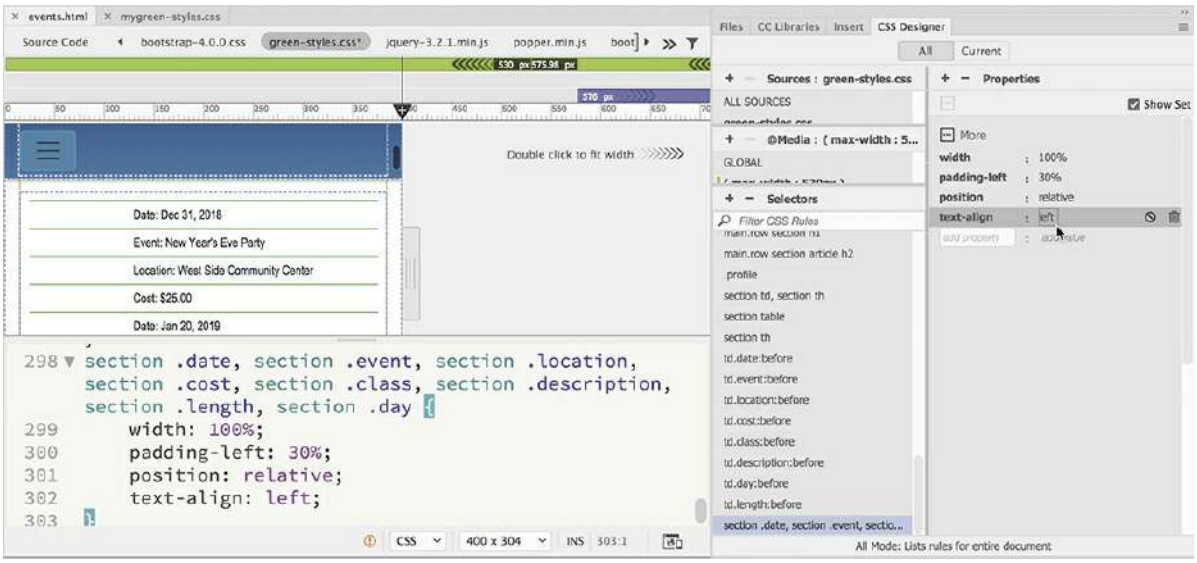
Note

Although the formatting is identical for these classes at this point, you may want to adjust the styling for one or more items later. Making separate rules can add flexibility even though it adds to the amount of code that has to be downloaded.

Next, let's apply some styling to the labels themselves to make them stand out more distinctly.

. Apply the following properties to the new rule or in each of the separate rules:

```
width: 100%

padding-left: 30%

position: relative

text-align: left
```

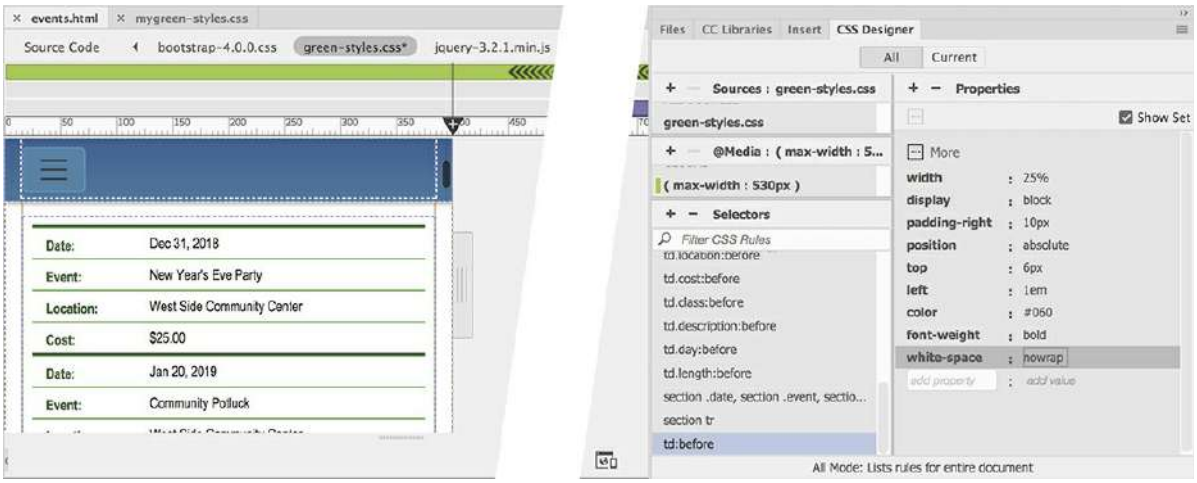


The event and class entries are now indented, and all appear at the same width. Next, you'll add a rule to differentiate the labels from the content of the tables themselves.

. Create the following rule: **`td:before`**

Give the new rule the following properties:

```
width: 25%

display: block

padding-right: 10px

position: absolute

top: 6px

left: 1em

color: #060

font-weight: bold

white-space: nowrap
```

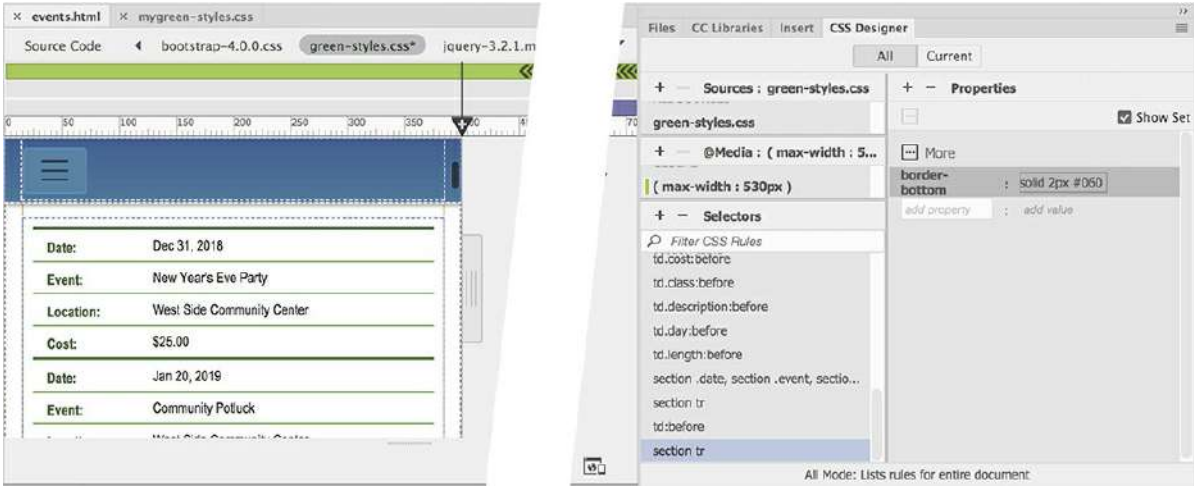


The labels now appear separated from the data and are styled in boldface and dark green. The only thing left to do now is differentiate one record from the next. One way is to simply add a darker border between each table row.

. Create the following rule: **`section tr`**

Give the new rule the following property:

`border-bottom: solid 2px #060`

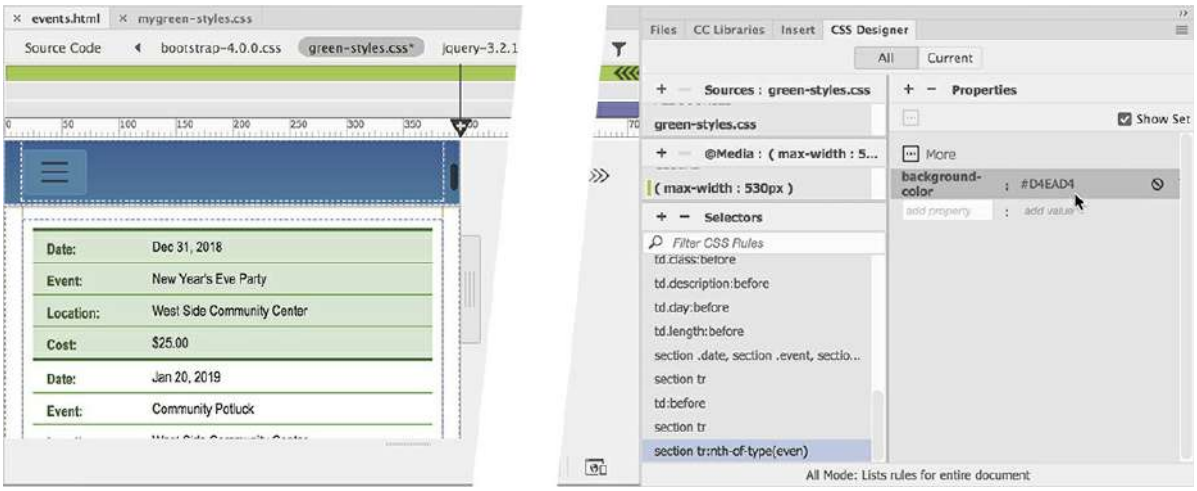


Using a CSS3 selector, you will add a little more pizzazz to the table and make the data easier to read.

. Create the following rule: **`section tr:nth-of-type(even)`**

Give the new rule the following property:

`background-color: #D4EAD4`



Note

Advanced selectors like `nth-of-type(even)` may not be supported by older browsers.

This CSS3-based selector actually applies the background only on even rows of the table.

Both tables are now slickly styled and responsive to any changes in the screen size. Although they look good in Live view, don't get complacent; it's vital to test the design in a variety of browsers and mobile devices too.

. Save all files. Preview the page in the default browser. Test the media queries and the responsive table styling by changing the size of the browser window.

It's likely that everything you tried in this exercise worked perfectly in both Dreamweaver and any browser you tested it in. But you have to remember that CSS3 is still fairly new and has not been fully adopted within the industry.

The good news is that most of the mobile devices you're targeting should support the various settings used in this exercise. And you can be sure Dreamweaver will stay current with the latest updates.

Adapting images to smaller screens

Today, the modern web designer has to contend with a multitude of visitors using different browsers and devices. Depending on the size of the images and how they are inserted, you may need to use several different strategies to get them to work effectively in your page design.

For example, the images used on the *Contact Us* page are small enough that they should be usable all the way down to the size needed on a smartphone. But that's not true for every page.

. Open **news.html** in Live view.

Make sure the document window is at least 1100 pixels in width.

There are two images on the page. The one at the top stretches all the way across the column; the second floats to the right, with the text wrapping around to the left.

. Click the top image to select it.

The Element Display appears on the image focused on the `img` element. The image is too large for the column. When it's selected, you can see that part of it is obscured behind Sidebar 2. Luckily, Dreamweaver has a new built-in way to deal with just this situation.

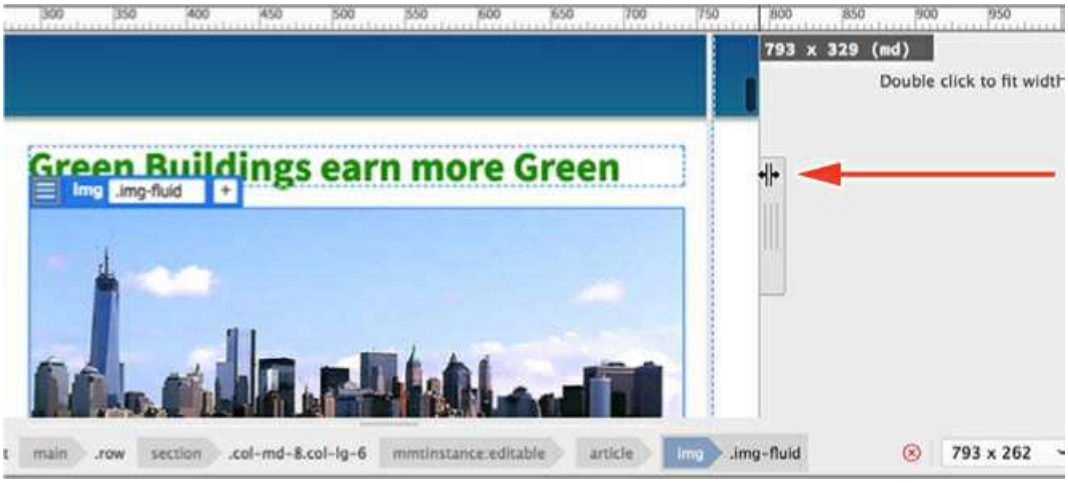
. Click the Edit HTML Attributes icon on the new image.

The Quick Property inspector appears.

. Select the Make Image Responsive checkbox.

The image now conforms to the width of the column, but what happens when the screen gets smaller?

. Drag the scrubber to the left and observe how the image adapts to the changes to the layout.



The image scales automatically as the document window changes sizes. The option in the Quick Property inspector selected in step 4 applies the Bootstrap `.img-fluid` class to the image. This class forces the image to fit within the existing element and to scale as needed.

. Scroll down to observe the second image.

Test the image at various screen widths.

The second image displays acceptably until the width drops below 400 pixels. Then, there's not enough space on the left side of the image for the text to wrap effectively. On the smallest screens, you should turn off the float property and center the image.

. Drag the scrubber to 400 pixels.

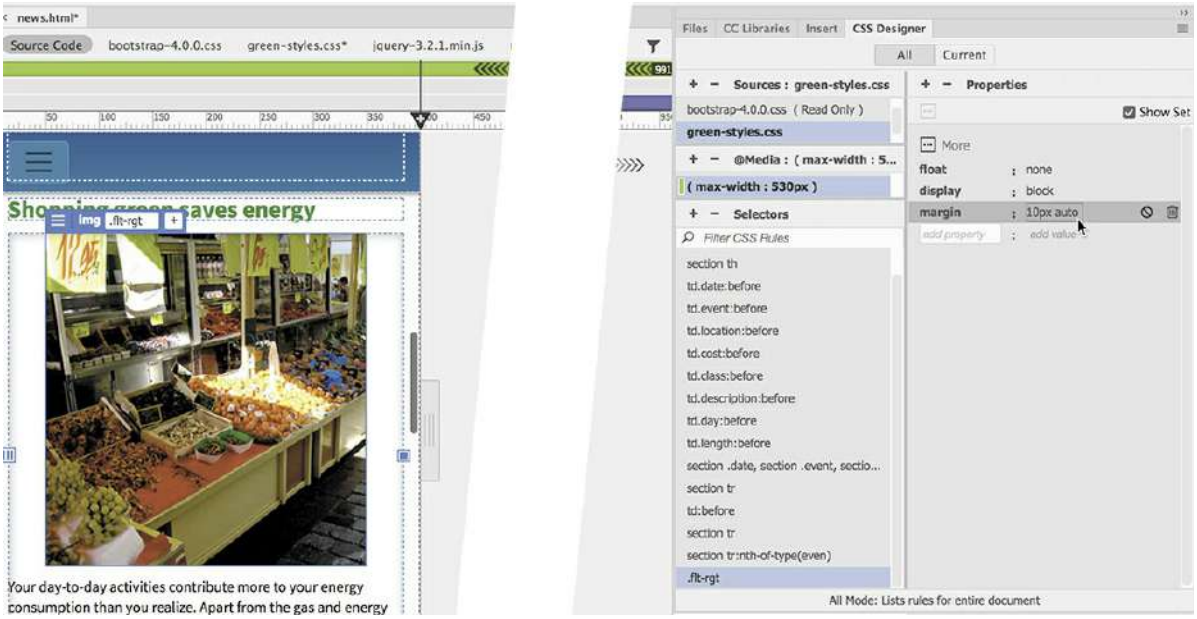
. Choose **green-styles.css** > `(max-width: 530px)`.

Create the following selector: **`.flt-rgt`**

The selector combines two classes to target this one image.

. Create the following properties:

```
float: none

display: block

margin: 10px auto
```



These settings will turn off the float property and center the image in the column. The responsive Bootstrap style will now work properly.

- Save and close all files.

The last items you need to review are the interactive elements added to the *Green Travel* page and the *Green Tips* page.

Hiding components on a responsive layout

Let’s open the *Green Travel* page to see how it works in the new layout.

- Open **travel.html** in Live view.

  Make sure the document window is at least 1100 pixels in width.

  The page contains a table with interactive links that swap the Eco-Tour ad with images from specific Paris tours.

- Drag the scrubber to the left to test how the table responds to the responsive layout.

  The table adapts to the changing screen in a fashion similar to the tables created and styled on the *Green Events* page. Everything seems to display fine, although the Eco-Tour ad does not scale or resize in any way.

  At a width of 530 pixels, the table’s two columns merge and the cells begin to stack one atop the other, including the cell containing the Eco-Tour ad. The image no longer appears beside the text describing the individual tours. Although the rollover effect still functions, the purpose of it is lost completely, as is the need for the ad itself. The simplest plan would be just to hide the ad on screens smaller than 530 pixels.

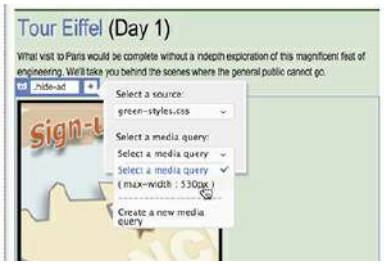
  At this moment, there’s a custom id applied to the Eco-Tour ad but not to the cell containing it.

CSS can hide the image, but that would leave the blank cell behind. Instead, you will create your own custom class to hide the ad.

- Drag the scrubber to 500 pixels.
- Select the Eco-Tour image.

  The Element Display appears focused on the `img` element.
- Press the up arrow once.

  The Element Display now displays the `td` element.
- Click the Add Class/ID icon .
- Type **`.hide-ad`** in the Element Display class field and press Enter/Return to complete the class name.



The CSS Source dialog appears.

Note

If for any reason the CSS Source dialog does not appear, you must create the class reference in **green-styles.css** manually.

- If necessary, select **green-styles.cs**s from the Select A Source menu.

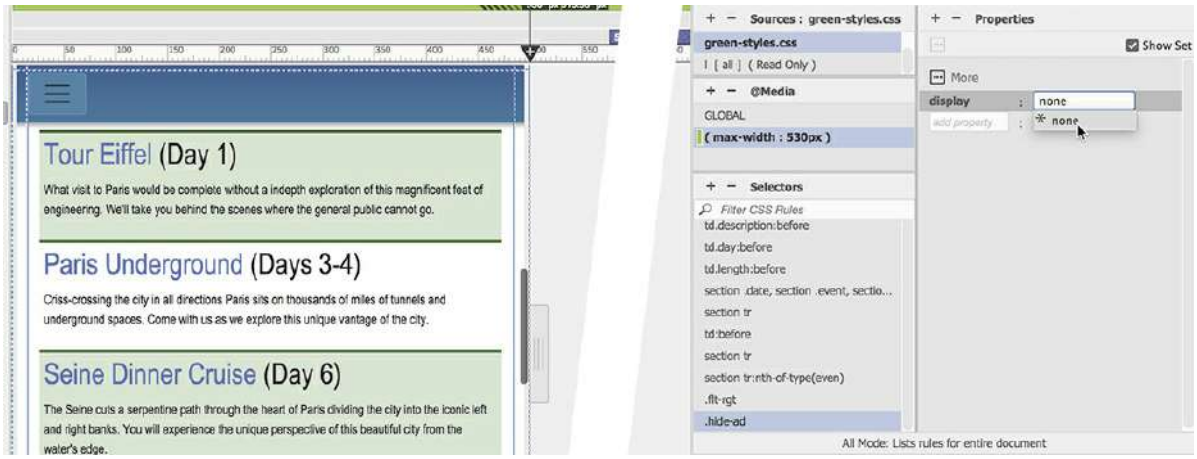
  Select `(max-width:530px)` from the Select A Media Query menu.

  Click away from the CSS Source dialog.

  The CSS Source dialog closes. The new class should be added to the media query.
- In the CSS Designer, select **green-styles.css** > `(max-width: 530px)`.

  Create the following property in the rule `.hide-ad`:

  `display: none`



The table cell and all its contents disappear as soon as the property is created. Whenever the screen is 530 pixels or narrower, the Eco-Tour ad will hide.

. Drag the scrubber to 800 pixels.

Observe the changes to the table and its content. Once the screen is 531 pixels or wider, the normal table design returns and the Eco-Tour ad appears again.

. Save all files.

. Close **travel.html**.

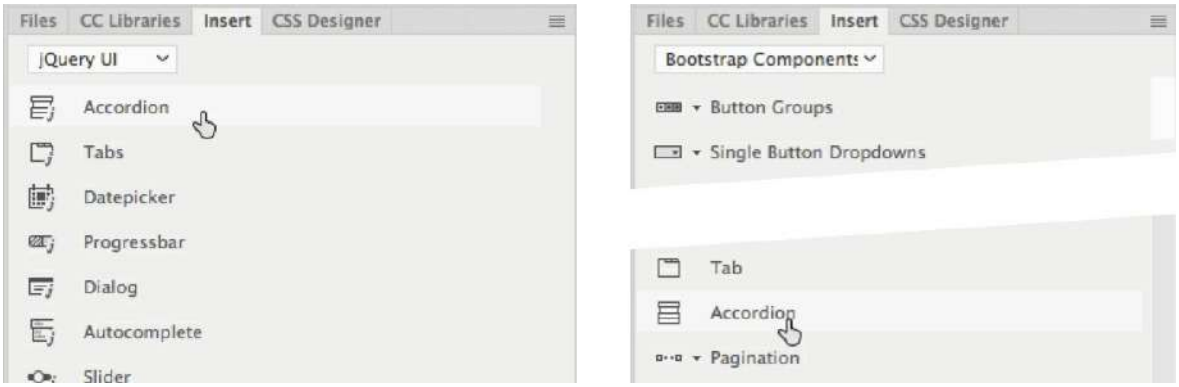
The last item you need to review is the jQuery accordion widget you created in Lesson 10, “Adding Interactivity.”

Trouble-shooting an interactive element

From time to time you will encounter conflicts between different scripts or web frameworks. You may come across a neat widget or interactive element on the Internet that you’d like to deploy on your website only to find that something in the new component is not compatible with your existing system. This is even more likely when using the Bootstrap framework.

Many interactive elements load their own resources to enable their interactivity. Since the new widget has no idea you are already using a web framework, it becomes more likely that the new element will encounter some issues with the existing software.

This is the case with the accordion used on the *Green Tips* page. You used an accordion from the jQuery UI category in Dreamweaver’s Insert panel. Since you were not using Bootstrap at that moment, it was your best option. But if you look at the Bootstrap Components category in the Insert panel, you will see it has its own accordion widget. Both widgets use jQuery libraries, but unfortunately, they are not using the same one. There’s no way of predicting how this will affect the widget, although such conflicts rarely produce acceptable results.



It's impossible to avoid conflicts like this. If you work on the web long enough, you will encounter many similar issues. So it's a good thing that you experience that kind of conflict here, where you can learn how to fix it. Let's take a look at the jQuery accordion widget on the *Green Tips* page.

Styling a jQuery accordion widget in a Bootstrap layout

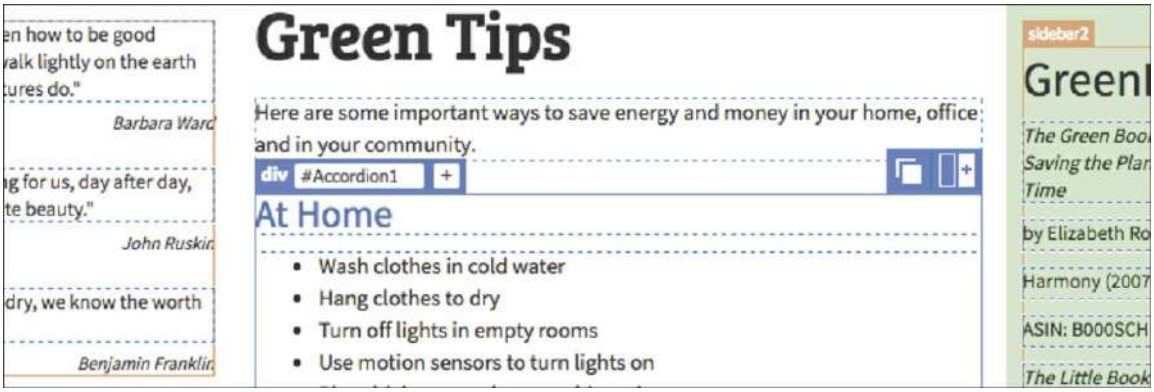
In this exercise, you will inspect the jQuery accordion widget and see how it fared in the transition to the Bootstrap layout.

- Open **tips.html** in Live view.

  Make sure the document window is at least 1100 pixels in width.

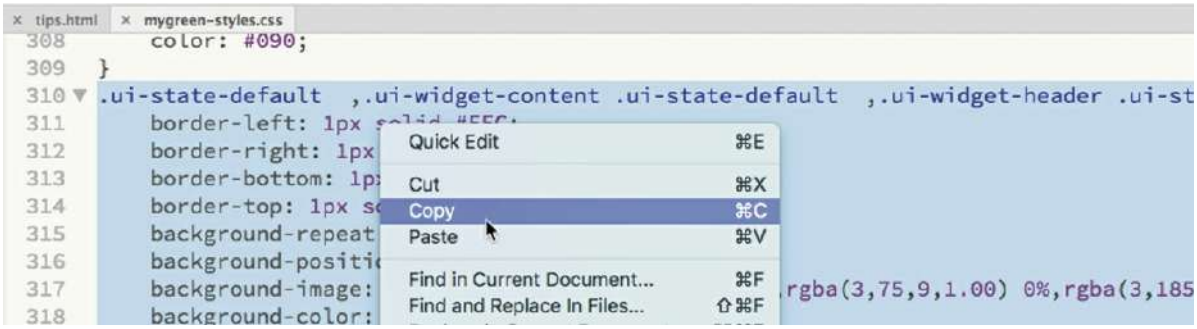
  The first thing you'll notice is that the styling for the accordion did not make it over to the new layout. As with the table styling you encountered earlier in this lesson, it's a simple matter to move the styling for the accordion over from **mygreen-styles.css**.

- Open **mygreen-styles.css**.
- Scroll down through the style sheet to locate the CSS rules formatting the jQuery accordion widget.



Around line 310 you will see styles starting with the class `.ui-state-default`. These are the styles you created to format the accordion in Lesson 10. You will need to move all of these styles over to **green-styles.css**.

- Select all the CSS code formatting the accordion (approximately lines 310 to 341). Copy the code.



- Switch to **tips.html**. Select **green-styles.css** in the Related Files interface.

  The document window switches to Split view and loads **green-styles.css** in the Code view window.

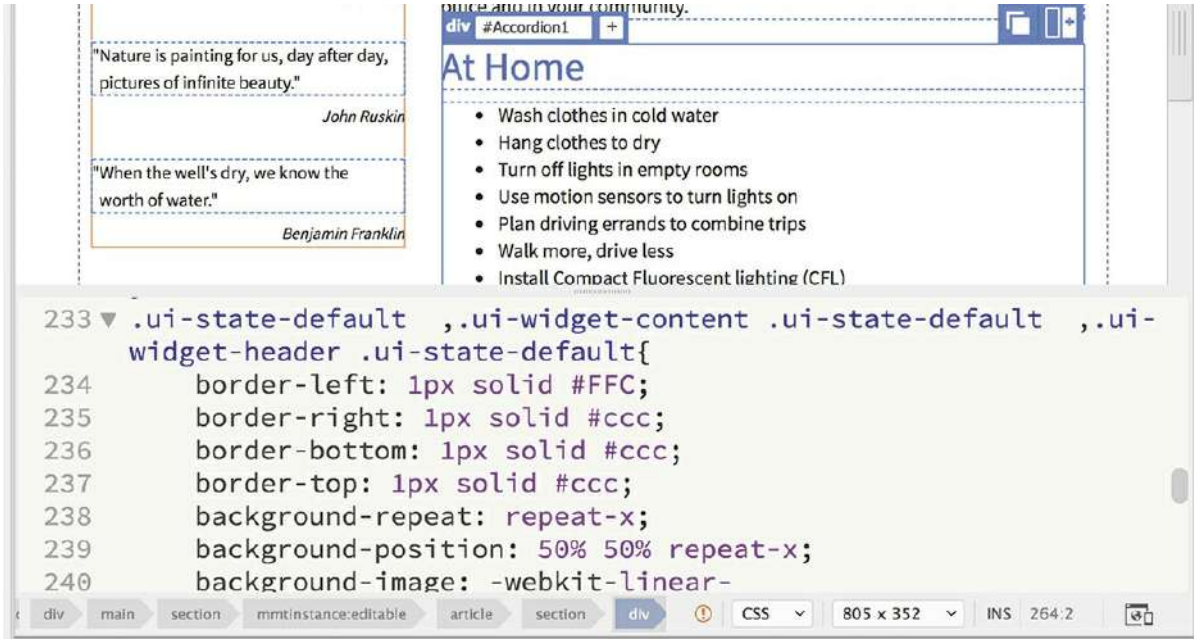
- In the Code view window, scroll down to the entry

  `@media (max-width: 530px)` (around line 233).

  As you did earlier, you'll paste the CSS code before the media query.

- Insert the cursor before the media query. Paste the CSS code.

  Press Enter/Return to move the entry `@media (max-width: 530px)` to a new line, if necessary.



  The CSS is in place, but the accordion still appears unstyled. If you look more closely, you will also notice that the HTML lists are all visible and no accordion or interactivity is visible. What happened?

  There are two problems with the current component. The accordion and the Bootstrap

framework are both based on jQuery, but the two systems implemented in Dreamweaver use different versions. That's the first problem. The second problem is that both versions are being called at the same time. That's not allowed. You can call one library or the other. You can even call one library twice. But you can't load two different libraries at the same time and expect everything to work. Luckily, there's an easy fix.

- Open **mybootstrap_temp.dwt** from the lesson15/Templates folder in Code view.

  The Bootstrap jQuery library (`jquery-3.2.1.min.js`) is being called in the site template to support the responsive layout as well as other Bootstrap components used on the page.

- Scroll down the code. Look for an HTML comment and `<script>` tag loading the Bootstrap library `jquery-3.2.1.min.js` (around line 22).
- Select the related comment and the entire script markup and cut it into memory.

Typically, scripts like this are loaded near the bottom of the code. In this case, it will work fine loading up in the `<head>`, where it needs to be to support and activate the accordion code.

- Scroll up to the closing `</head>` tag (around line 17).

Note

It's not always possible to use a newer library to power every widget or component. Sometimes you may have to scrap an older application and rebuild it from scratch using the newer library.

Notice the editable region just above the closing `</head>` tag. In the template, the editable region contains only the meta description, but this is where the jQuery UI library (`jquery-1.11.1.min.js`) is inserted in **tips.html**. The Bootstrap library is a slightly newer version than this one, so you will use the Bootstrap library and delete the older one.

- Insert the cursor before the closing `</head>` tag (around line 17).
- Paste the code you cut in step 10. Save the template.

The Update Template Pages dialog appears.

. Click Update.

All child pages are updated. You just moved the Bootstrap jQuery library up into the `<head>` section of every page.

. Close the Update Template Files dialog. Close the template. Switch to **tips.html**.

. Switch to Code view. Scroll down to the closing `</head>` tag (around line 31).

You can see the comment and script elements you moved just above the editable region. In the template, and on all the other pages of the site, the editable region was empty. In **tips.html** it features two scripts, one loading the Bootstrap jQuery library and the other loading a specific library supporting the jQuery UI accordion.

. Delete the jQuery UI library script element:

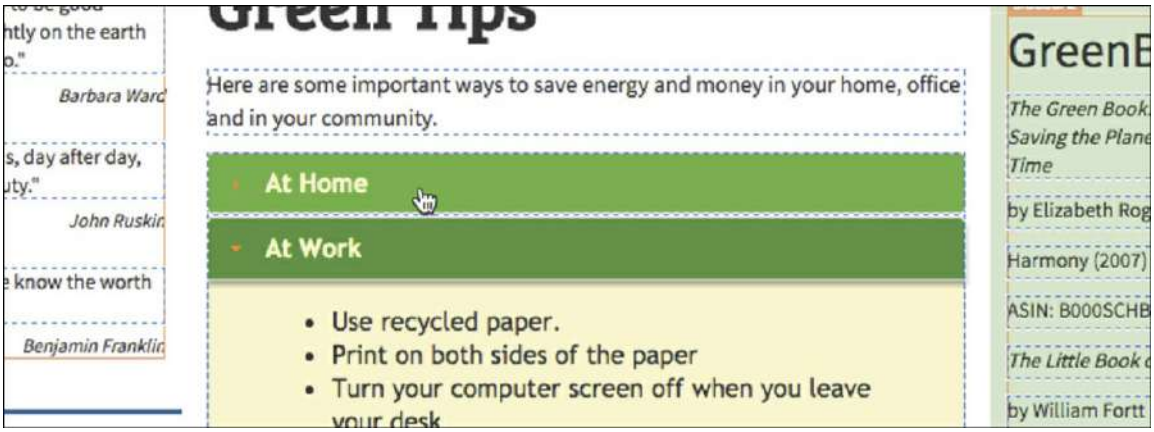
Click here to view code image

```
<script src="jQueryAssets/jquery-1.11.1.min.js"></script>
```

```
17    <!-- InstanceBeginEditable name="head" -->
18  <meta name="description" content="Meridien GreenStart Association - Learn
    the best eco-tips for your home, office and your community">
19  <script src="jQueryAssets/jquery-1.11.1.min.js"></script>
20  <script src="jQueryAssets/jquery.ui-1.10.4.accordion.min.js"></script>
21  <!-- InstanceEndEditable -->
22      <!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
23      <script src="js/jquery-3.2.1.min.js"></script>
24  </head>
```

```
17    <!-- InstanceBeginEditable name="head" -->
18  <meta name="description" content="Meridien GreenStart Association - Learn
    the best eco-tips for your home, office and your community">
19
20  <script src="jQueryAssets/jquery.ui-1.10.4.accordion.min.js"></script>
21  <!-- InstanceEndEditable -->
22      <!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
23      <script src="js/jquery-3.2.1.min.js"></script>
24  </head>
```

. Switch to Live view.



As soon as the conflicting script is deleted, the accordion is fully styled and seems to be back in business. Let's test it.

Note

If your accordion is not formatted properly at this point, make sure you copied and pasted all the rules you created back in Lesson 10 to format it.

. Scroll down so you can see more than one panel in the accordion.

Click one of the closed panels.

The closed panel opens; the open panel closes.

. Click the other closed panel.

The other closed panel opens and the open panel closes.

By deleting the conflicting library and moving the Bootstrap library into the `<head>` section, you corrected the issues in the accordion widget. Now let's see how the accordion looks at various screen sizes.

Adapting an accordion widget to responsive design

The jQuery accordion widget looks and functions properly now in a desktop environment. Let's see how it functions at all screen sizes.

Tip

If the accordion does not operate as it did in Lesson 10, check to see if the jquery script is placed as directed earlier.

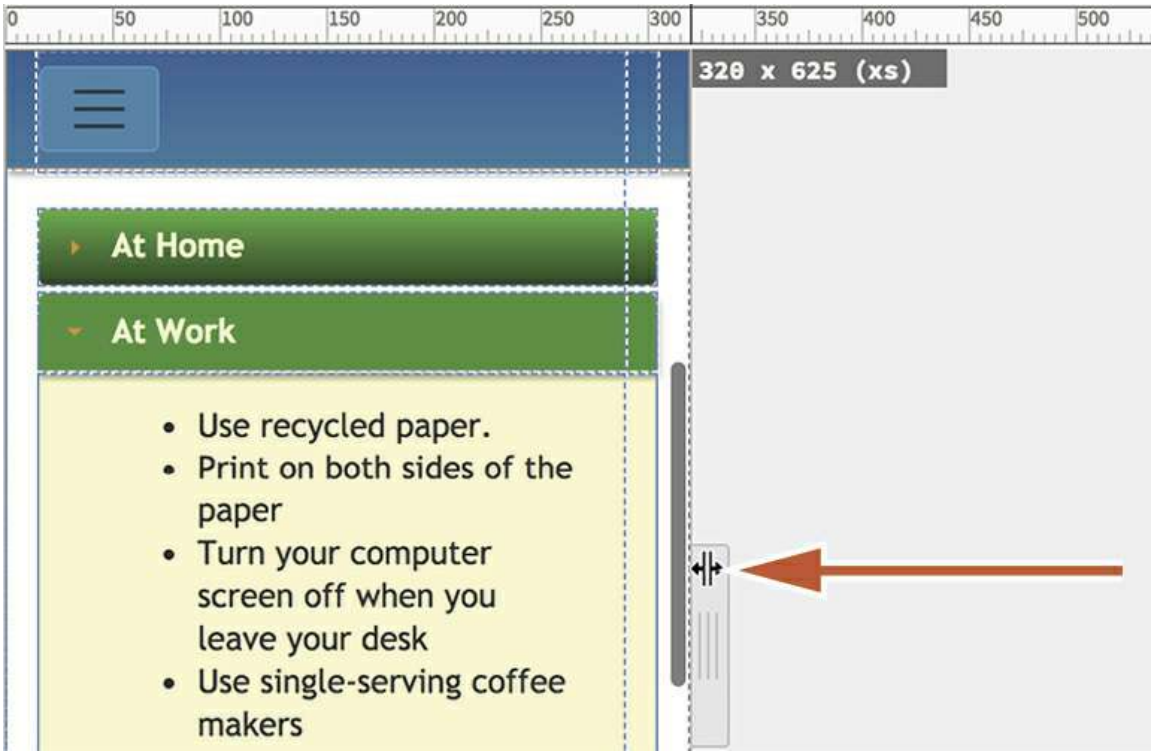
. If necessary, switch to Live view. Drag the scrubber to 800 pixels.

The layout switches to two columns.

. Click a closed tab in the accordion. Test each panel.

The accordion looks correct and functions properly.

. Drag the scrubber to 320 pixels.



The layout switches to one column. At 320 pixels, this is the smallest screen size you should have to support.

. Click a closed tab in the accordion. Test each panel.

The accordion functions properly, but the bulleted lists are not using the space effectively. The lists are indented too far. Let's reduce the indent a bit.

. Click one of the list items.

The Element Display appears focused on the `li` element. Lists can be indented on the `<li>` element, the `<ul>` element, or both.

. Select the `ul` tag selector.

. In the CSS Designer, click the All button.

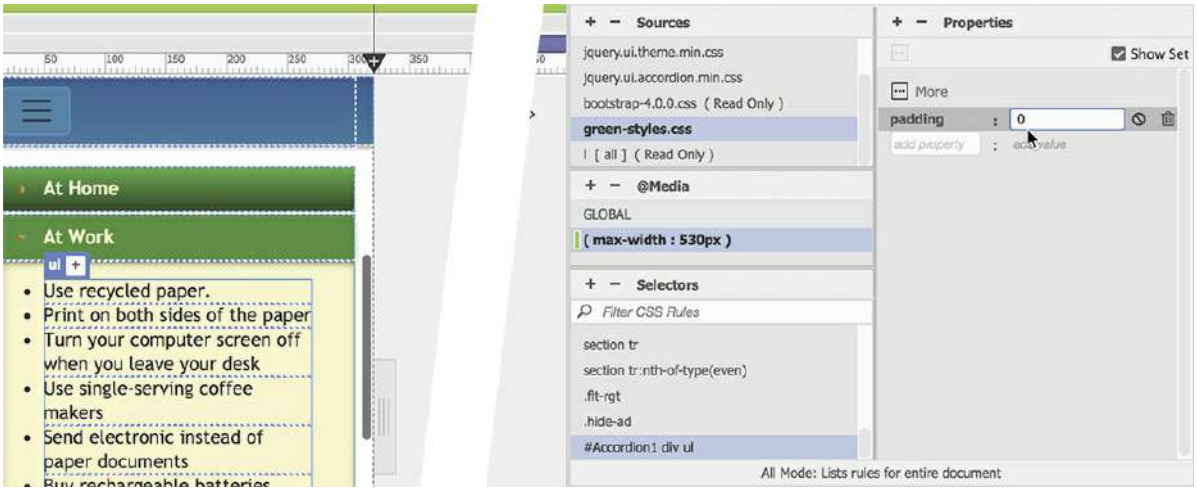
Select **green-styles.css** > `(max-width: 530px)`.

Click the Add Selector icon +.

A new selector, `#Accordion1 div ul`, appears in the Selectors panel. It targets only the lists in this accordion. By adding it to the media query `(max-width: 530px)`, it will kick in only on the smaller screens and devices.

. Press Enter/Return as necessary to create the selector.

Create the following property: **`padding: 0`**



The list items move to the left and occupy most of the screen.

. Drag the scrubber to the right to open the document window fully.

As you drag it open, the accordion adapts to the layout seamlessly, in one, two, or three columns.

. Save and close all files.

Once you complete the review of the site in Dreamweaver, you should review every page, component, and piece of content in a variety of browsers and on any mobile devices you have at hand. Dreamweaver makes this process easy too.

Previewing pages using Real-Time Preview

Since Live view is based on the WebKit web browser engine, it provides a pretty good facsimile of your pages and content. But nothing can take the place of an actual browser, smartphone, or tablet. You cannot be absolutely sure that your pages and design work properly until you have tested them all in the actual environments. Real-Time Preview was added to Dreamweaver to make testing your pages and site a one-step operation.

. Open **index.html** from the lesson15 folder.

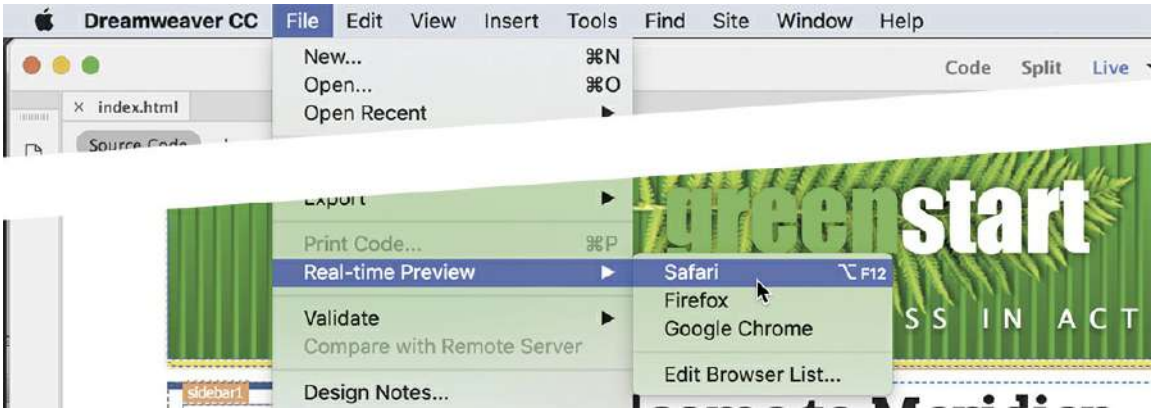
The first part of Real-Time Preview allows you to preview pages in any browser installed on your computer that is registered with Dreamweaver. New ones can be added by choosing File > Real-Time Preview > Edit Browser List.

Note

When using Real-Time Preview, you may receive an error message in your browser. If you do, try clearing your browser cache. Close and re-launch Dreamweaver and try again.

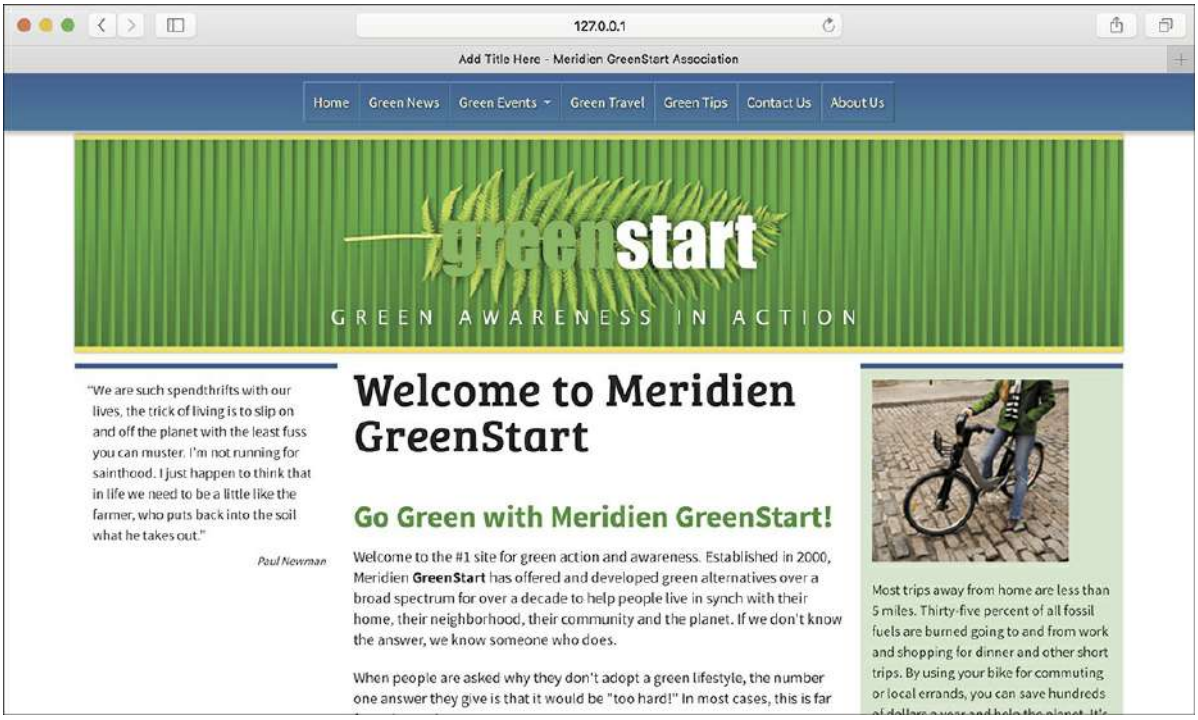
. Choose File > Real-Time Preview > Safari (or whatever is your default browser).

Maximize the browser to fit the entire computer display.



The page opens in the browser, displaying the content in three columns.

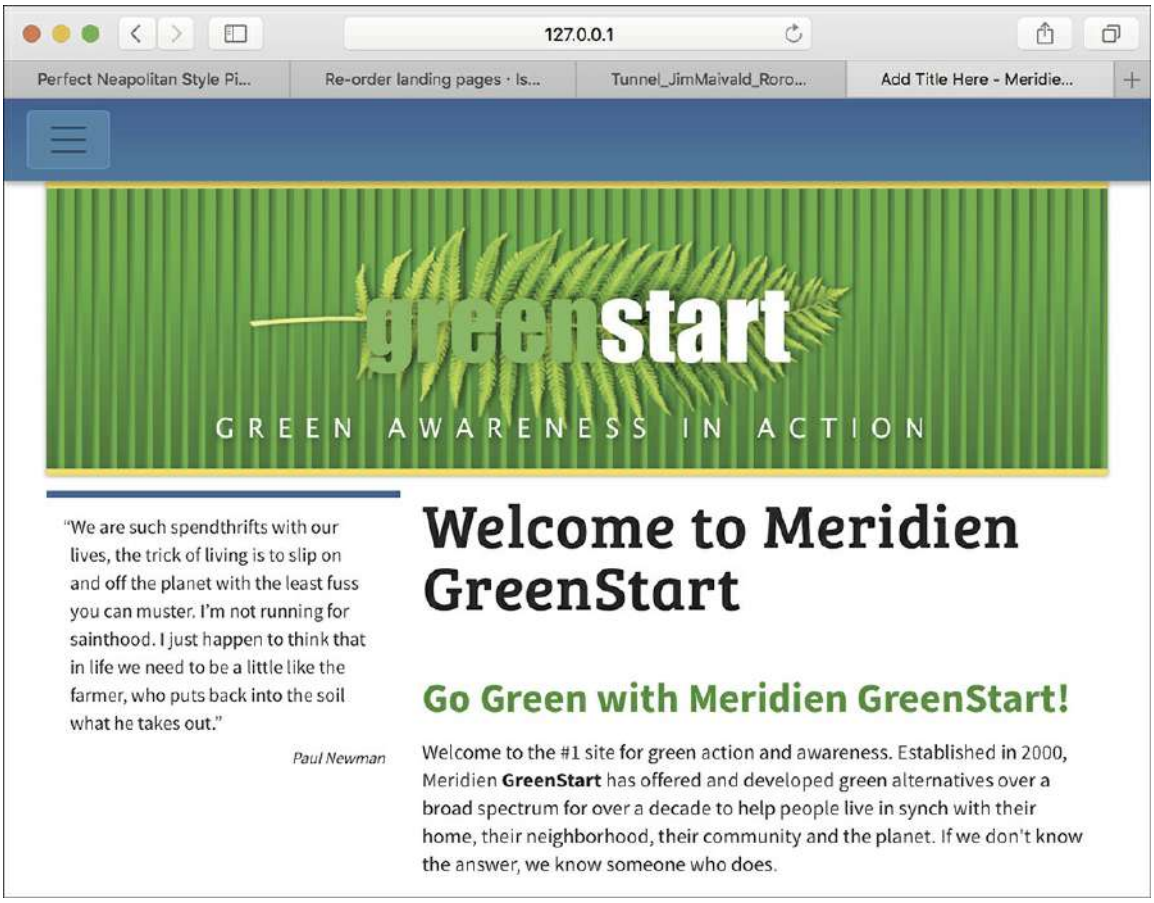
. Observe the entire page and all the content.



You should also test each page at various screen widths.

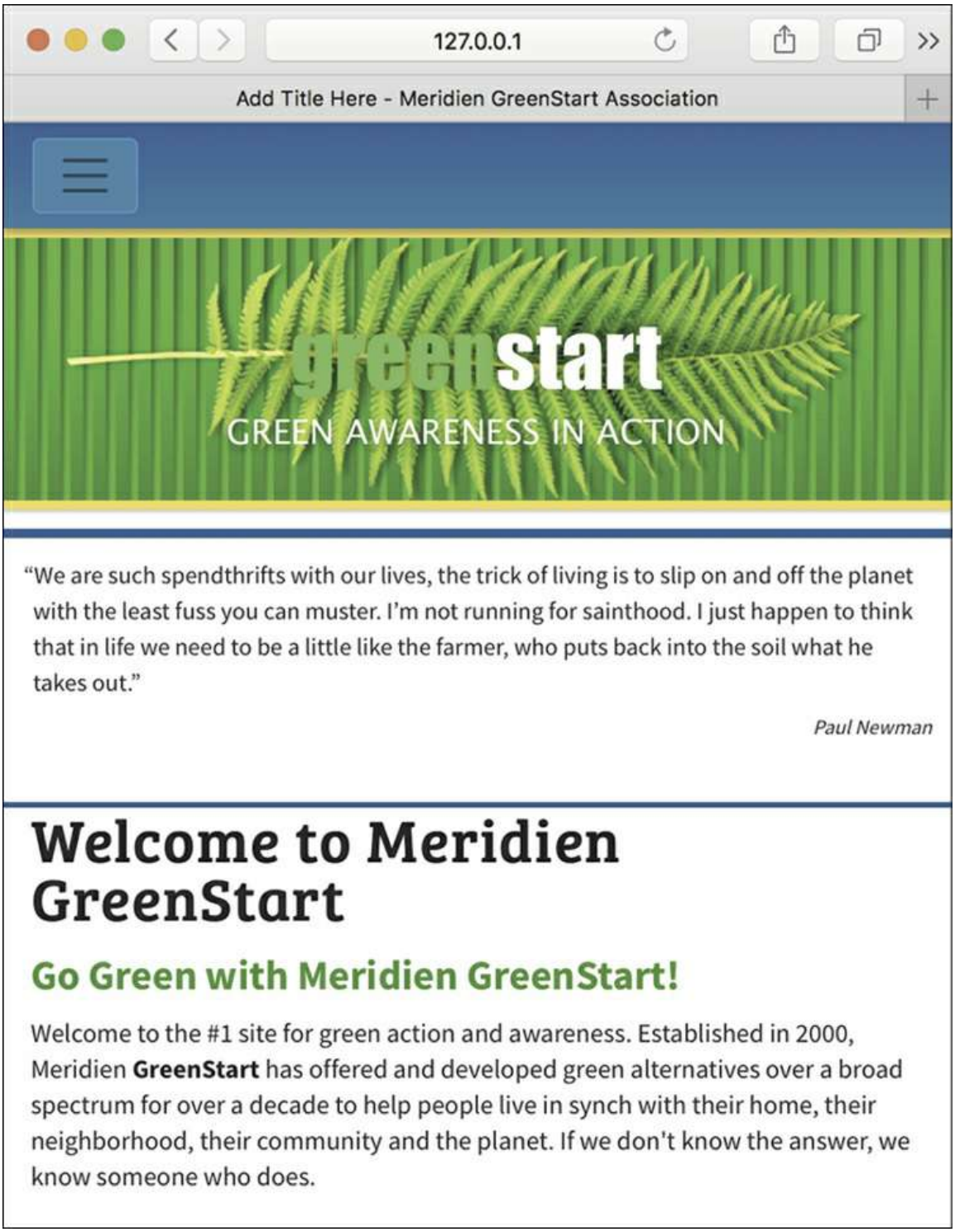
. Drag the right edge of the browser window to reduce the width.

Stop dragging the edge when the layout switches to two columns.



Review the content again to look for anything that doesn't work or doesn't look acceptable.

. Reduce the width again until the layout switches to one column.



Review the content.

. Click the sandwich icon on the main menu.

The hyperlinks should all work.

. Click the *Green News* link.

The *Green News* page loads in the browser, replacing the home page.

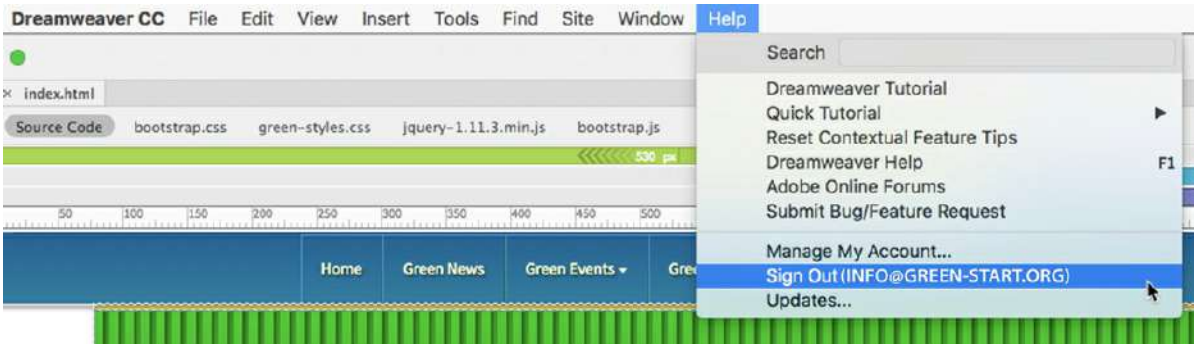
- Test the *Green News* page at various widths.

When you finish testing the page, click the next menu item and start the process again. Once you are done testing the site in one browser, select the next browser on your system and begin the process all over from scratch.

Once you finish testing the website on your desktop browsers and fix any issues that you find, you should test everything again on a range of smartphones and other mobile devices. But you might ask, "How do you test pages without a web server where you can upload all your pages?"

There's no need to have your own web server when you are using Dreamweaver CC (2019 release). That's because Real-Time Preview can upload your site to a preview area of Creative Cloud. All you need is a live connection to the Internet and an active Creative Cloud account. Typically, you are always logged in when you are using Dreamweaver, but you can see your status by checking the Help menu.

- In Dreamweaver, choose Help.



The menu should say "Sign Out" and your login name. If it says "Sign In," you will have to log in to your account before you can use Real-Time Preview.

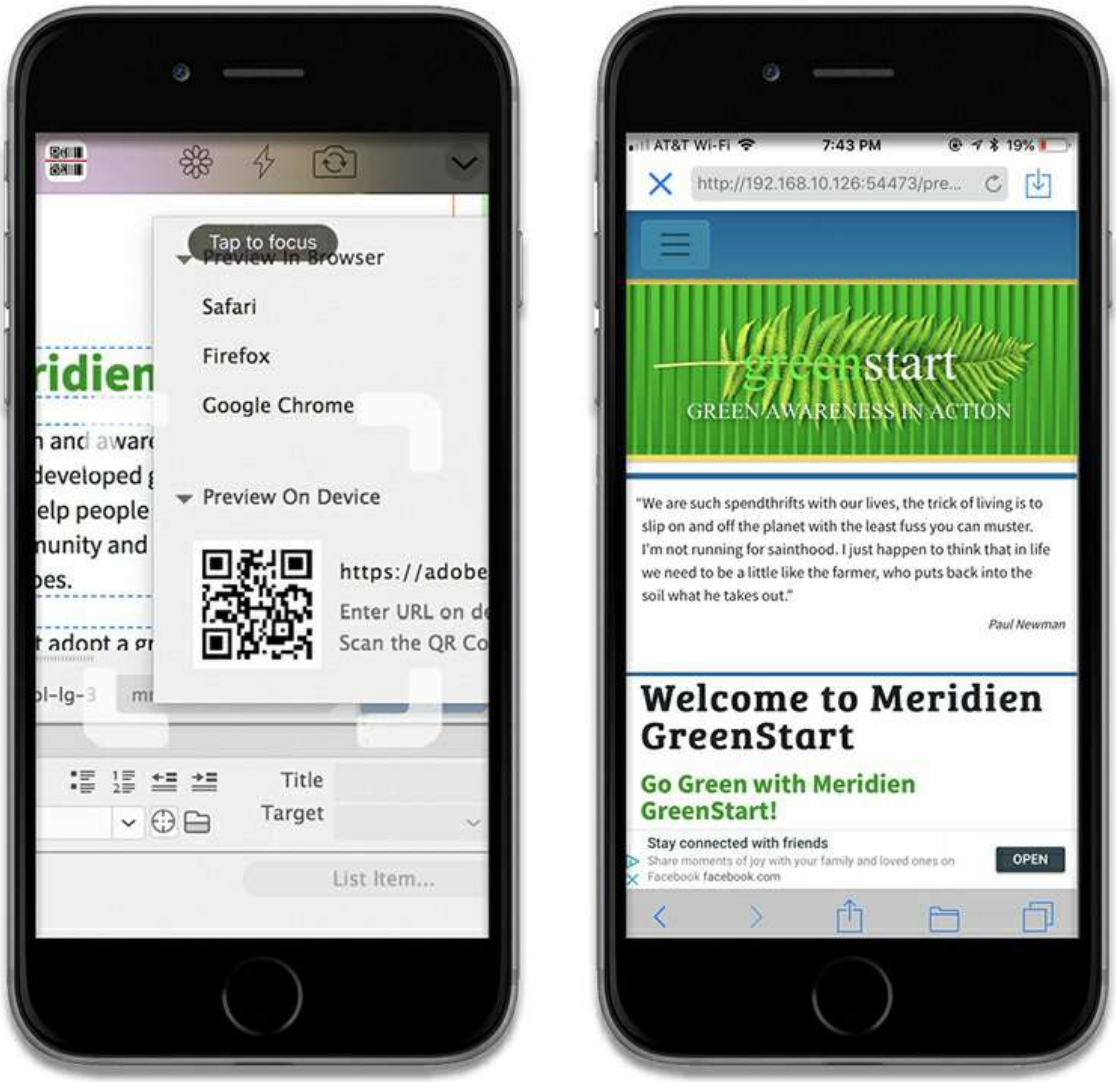
- If necessary, log in to your Creative Cloud account in Dreamweaver; otherwise, skip to step 11.
- If necessary, open **index.html**.
- Click the Preview icon in the lower-right corner of the document window.

Tip

To obtain a QR app for your phone or tablet, just go to the app store for your device and search for QR.

A pop-up window should appear showing a Quick Response (QR) code and a custom Adobe URL. If you have a QR app on your phone or tablet, you simply have to scan the code and you will be redirected to a preview copy of your entire site uploaded to Creative Cloud. You can even share the URL with your coworkers or clients.

- Scan the QR code with your phone or tablet, or enter the URL into a browser on your device.



The home page should load in your device browser. The screenshot shows the preview on an iPhone 6. As you can see, there are problems with the fonts on the page. The logo and motto are not showing the correct fonts. You are seeing firsthand the reason it's vital to test all your pages on many different browsers and devices before taking a site live. Luckily, this is one of the easier issues you can have with a new site.

Fixing issues for mobile design and desktop

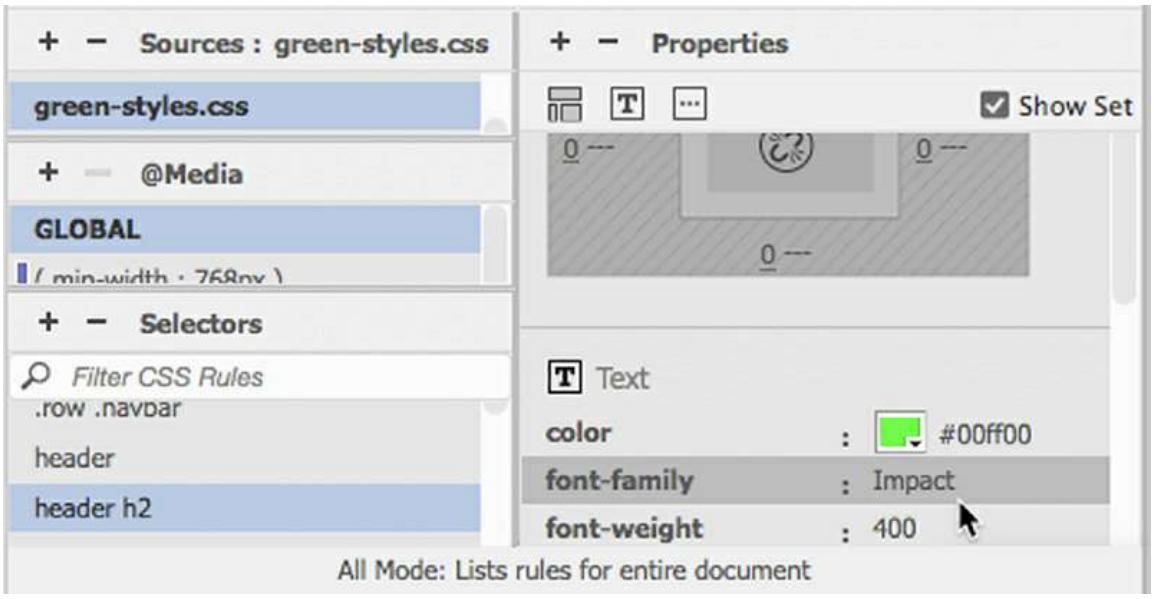
Have you tested all your pages? Did you test them on desktop and mobile devices? In multiple browsers? Did you identify all the issues on your pages and with the content?

In this exercise, you will fix the font problem you saw in the previous exercise, as well as address several others.

. In Dreamweaver click the All button.

Select **green-styles.css** > `GLOBAL` in the CSS Designer.

Select the rule `header h2`.



The rule calls only the font Impact. Since that font is not supported by the iPhone, or any other iOS device, the logo falls back to the default font of the iPhone browser. The same will happen in Android phones too, because that OS has a limited set of fonts installed. If you want the logo to look right, you have to create a font stack that will support a wide range of browsers and devices.

. Deselect the Show Set option in the CSS Designer.

It's easier to create a font stack when this option is turned off.

. Click the `font-family` property.

The font stack pop-up list appears.

. Click Manage Fonts.

Create a custom font stack with the following fonts:

```
Impact
HelveticaNeue-CondensedBlack
Roboto Black
Arial Black
Arial Bold
sans-serif
```

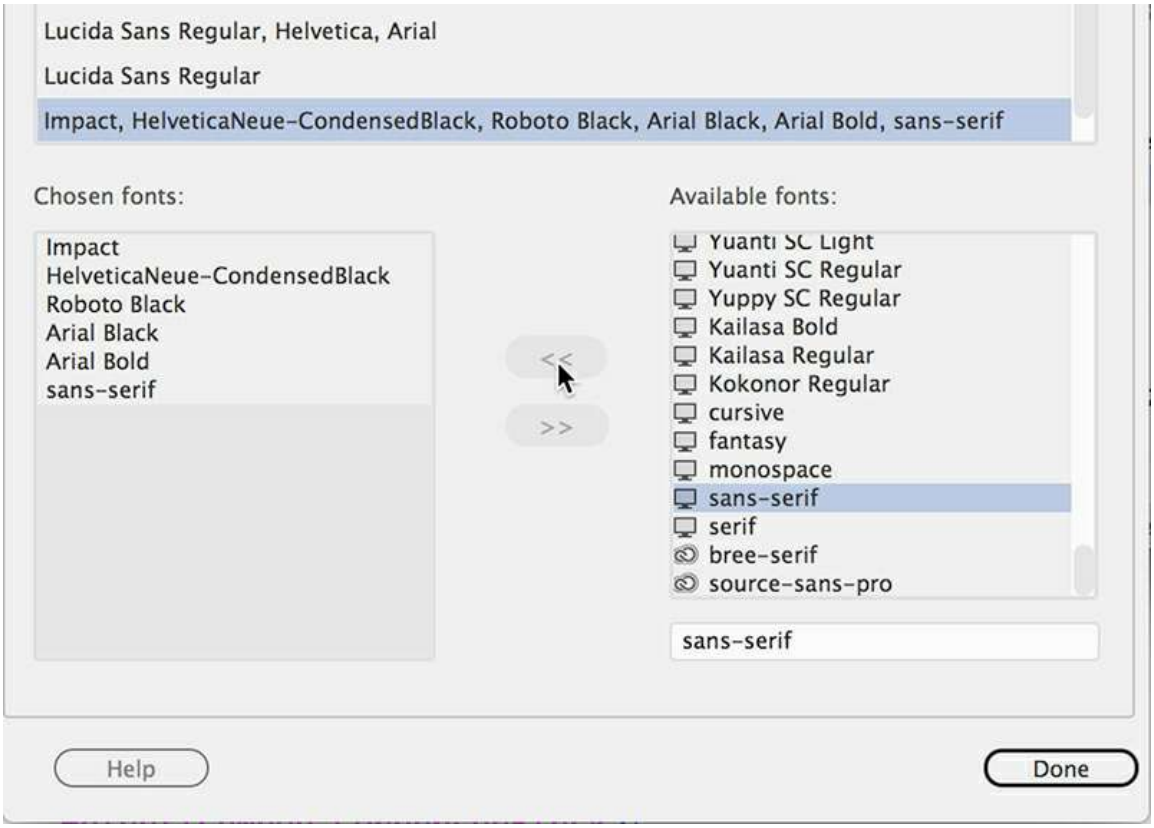
Note

Remember that you can enter any font name manually even if it's not installed on

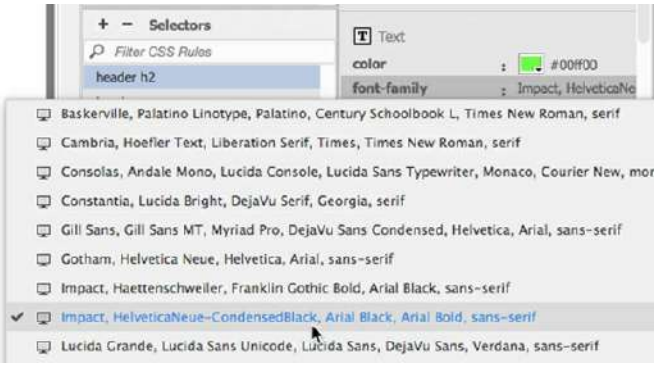
your computer.

Note

You must enter the name for HelveticaNeue-CondensedBlack exactly as shown or it will not load properly on an iOS device.



. Select the new font stack for the rule `header h2`.

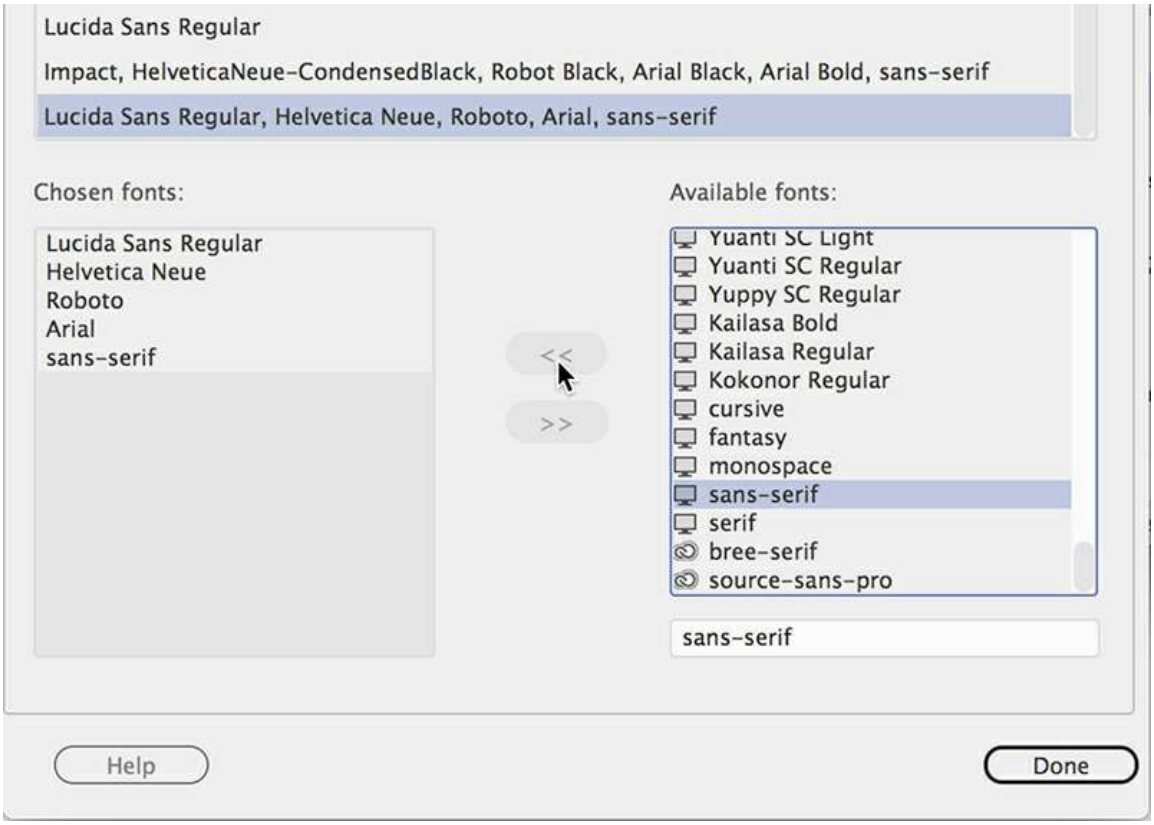


Now let’s fix the motto.

. Repeat steps 3 and 4 to create a new stack for the rule `header p`.

Add the following fonts to the custom font stack:

```
Lucida Sans Regular
HelveticaNeue
Roboto
Arial
sans-serif
```



. Select the new font stack for the rule.

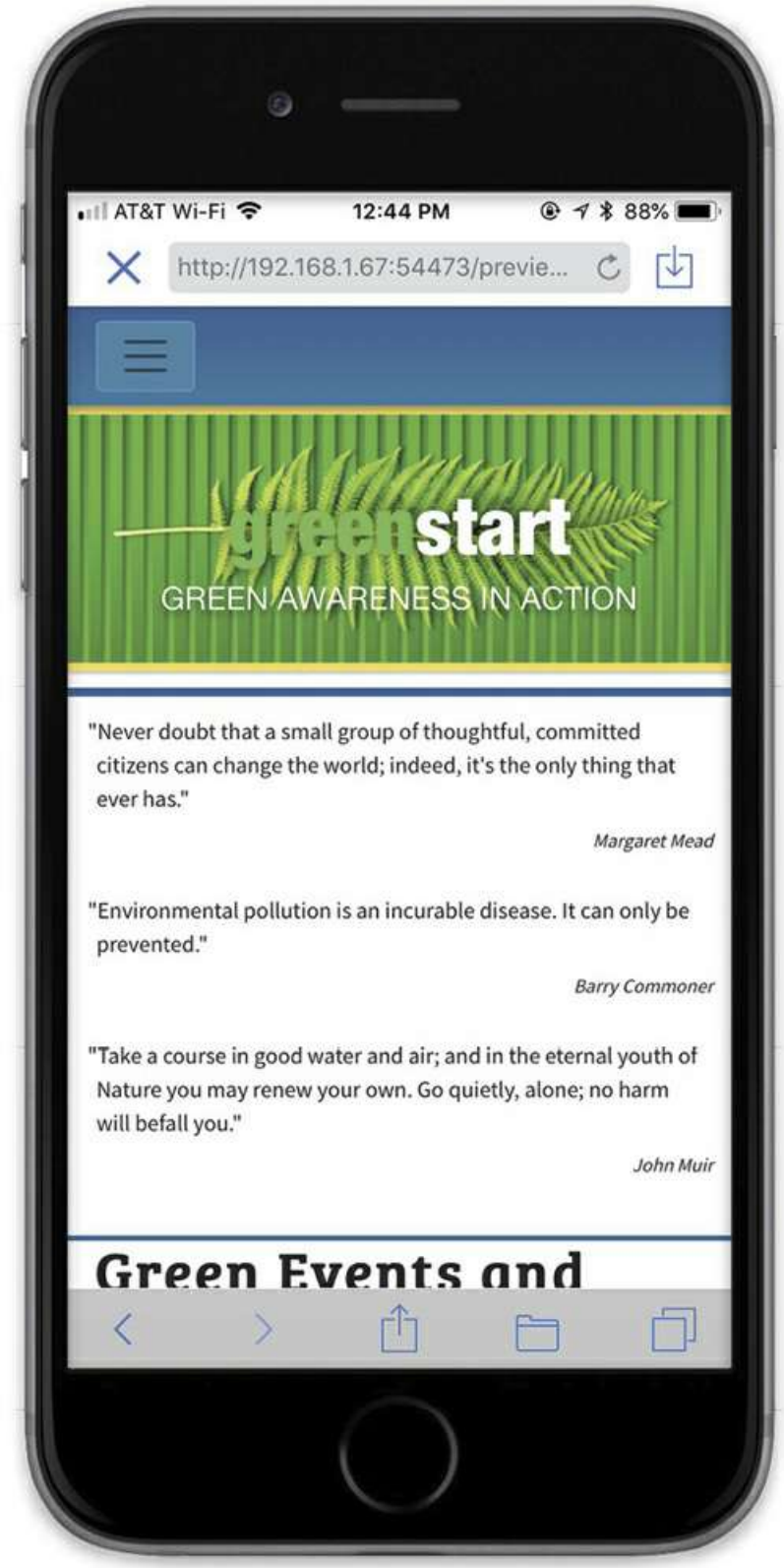
. Save all files.

Once the new font stacks are created and applied to the logo and motto, you can preview them again using Real-Time Preview.

. Click the Preview icon and scan the QR code with your mobile device to preview the changes on your smartphone or tablet.

As you can see from the new iPhone screenshot, the changes to the styling look much better.

. Close all files.

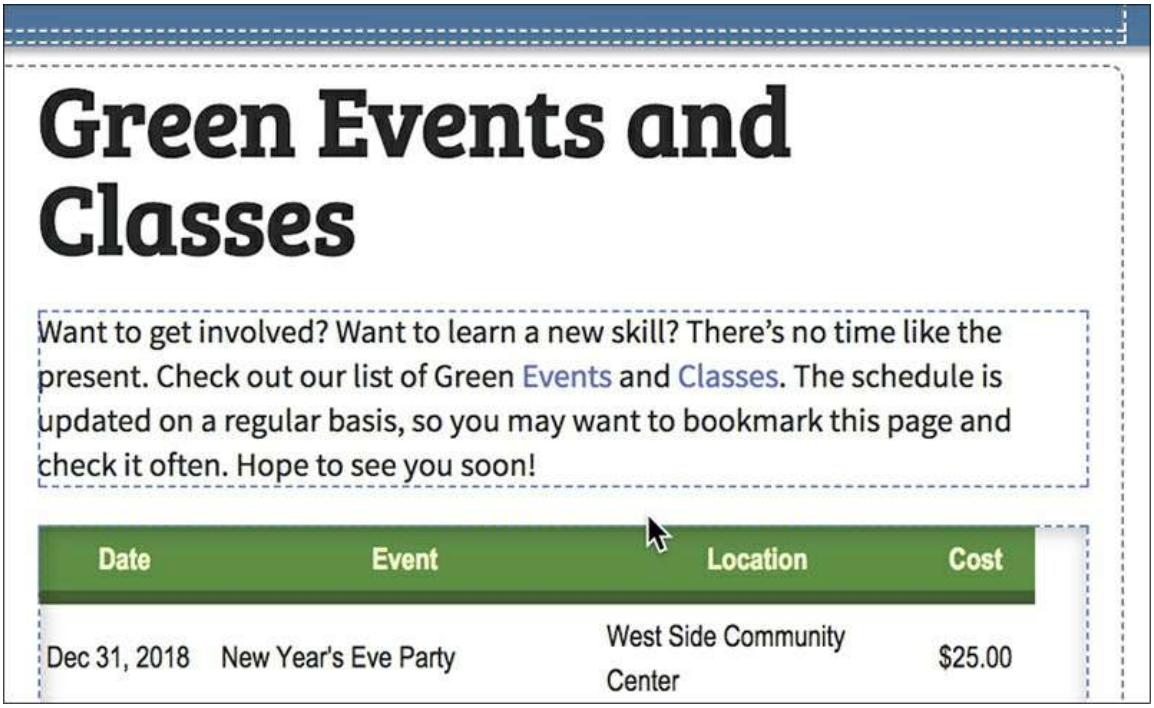


You've come a long way, but you are not finished. Next, we'll tackle an issue that is caused by the interaction of the Bootstrap layout and the table captions on the events page.

Addressing HTML5 styling

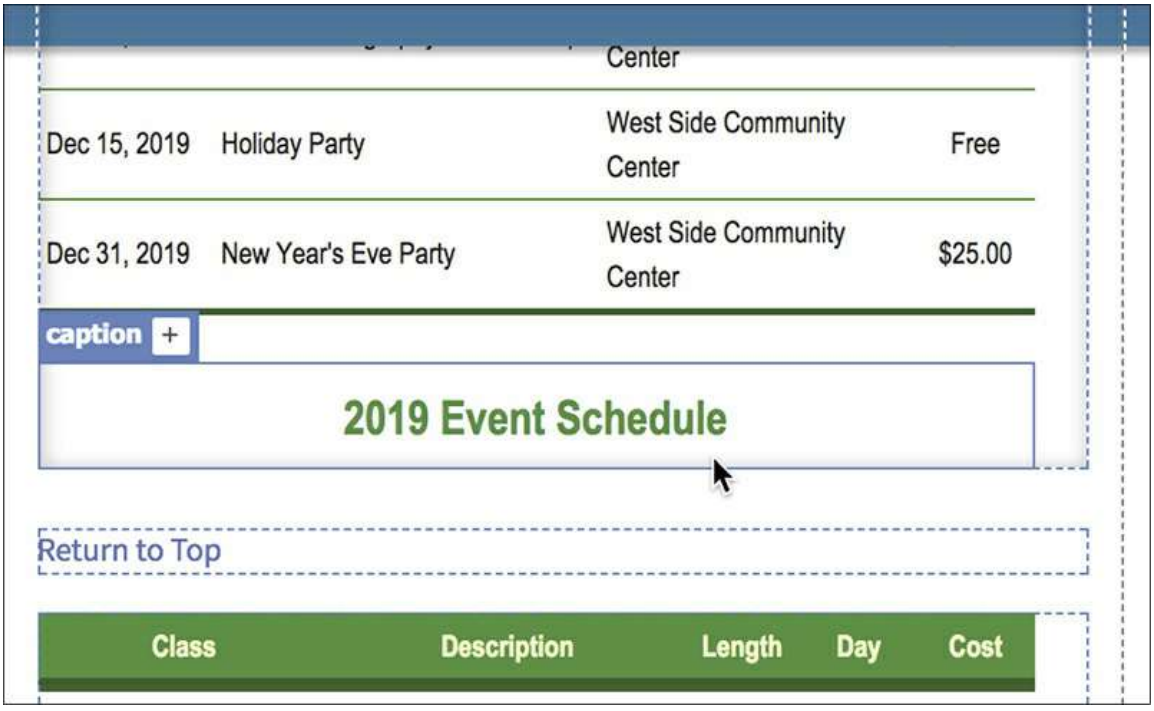
As we have seen throughout the book, HTML5 has brought many amazing things to web designers. But in this case, the new features have thrown us an unintended design issue with the table captions.

- Open **events.html** in Live view from the lesson15 folder and scroll down to examine the calendar table.



You may not notice the issue if you only look at the top of the table.

- Scroll down to the bottom of the calendar table.



The captions for both tables are no longer displaying at the *top* of the tables, as they did in the original non-responsive HTML layout. For some reason, the captions are now appearing at the bottom of the tables in the Bootstrap-based design. Luckily, Dreamweaver provides an easy way to identify the cause.

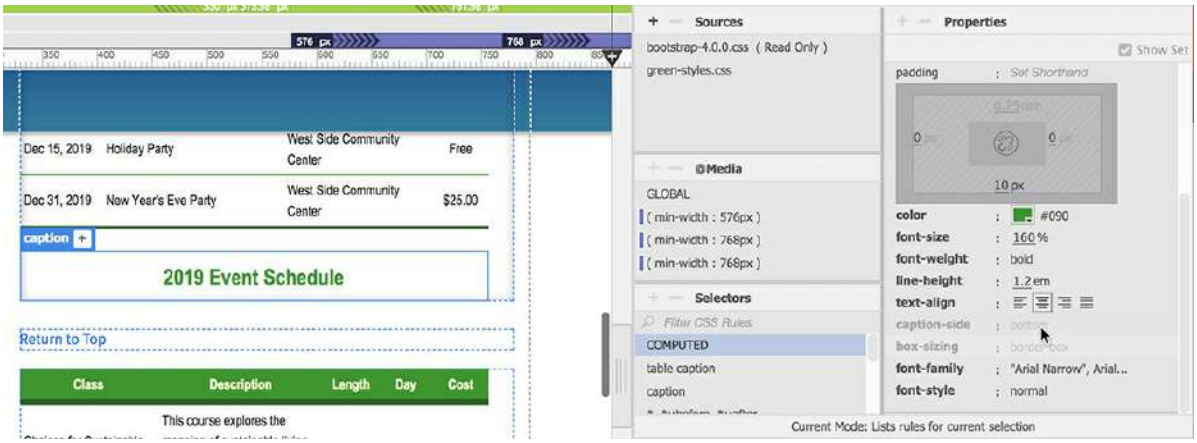
. Select the caption *2019 Event Schedule*.

The Element Display appears around the caption.

. In CSS Designer, click the Current button, if necessary.

. In the Selectors pane, click the `COMPUTED` option.

Examine the properties assigned to the `<caption>` element.



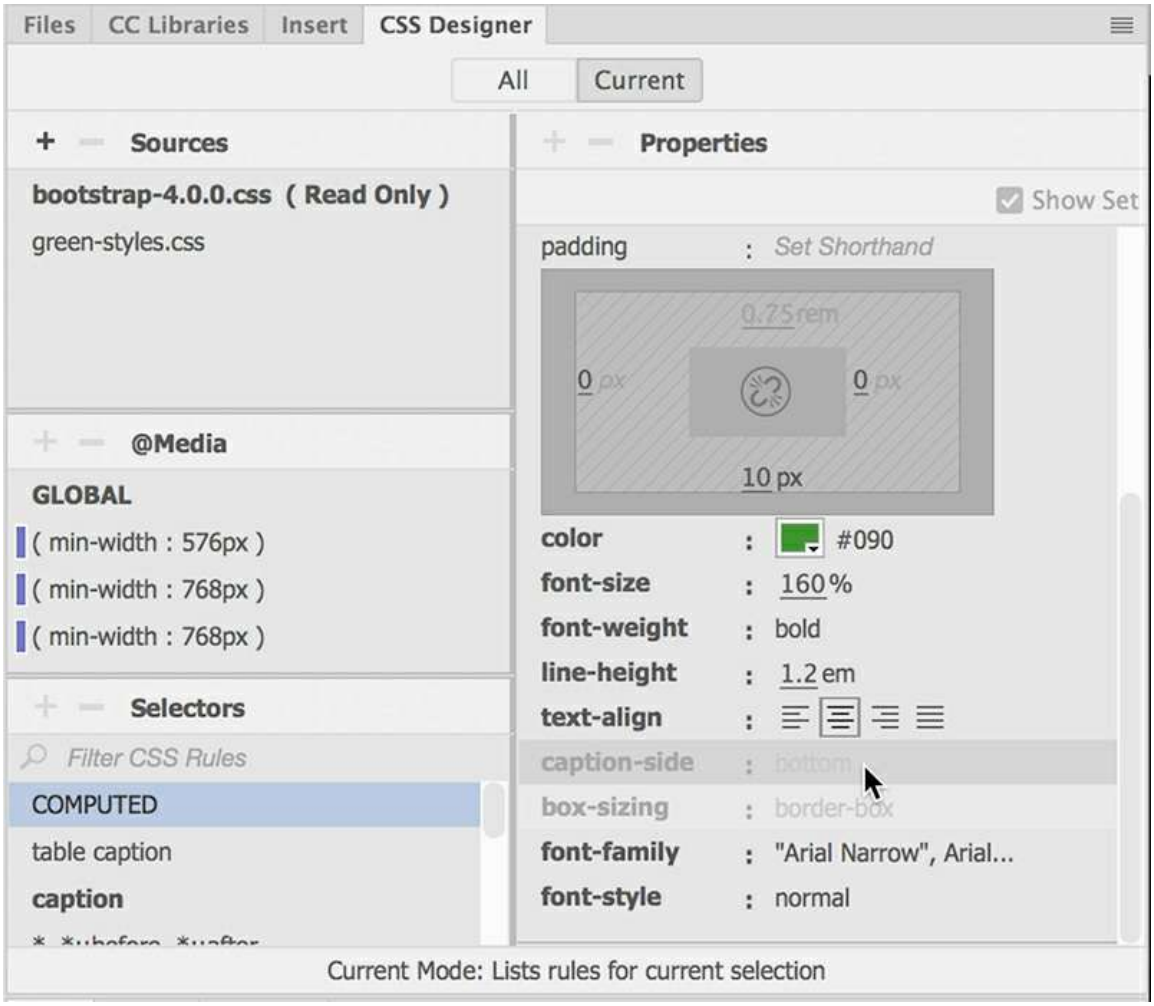
The Properties pane displays the styles compiled from all rules that are now formatting this element. The culprit affecting the `<caption>` element is easy to find.

Note

Some users report that the Bootstrap style sheet is not marked “Read Only.”

When the shift was made away from HTML attributes, CSS properties were created to replace them. Several of the properties are specific to the `<caption>` element. There are some recent additions in CSS3, but the one affecting the elements in your tables is older. By using the `COMPUTED` feature, you can see that the offending property is obviously `caption-side`. The item is grayed out in the CSS Designer, which indicates that it resides in a read-only style sheet.

. Click the property `caption-side` in the Properties pane.



When you click it, CSS Designer highlights in bold the source and the media query, if any, the property is hosted in. In this case, the property is a `GLOBAL` attribute in `bootstrap-4.0.0.css`. Obviously, the creators of Bootstrap wanted captions to display at the bottom of the parent element.

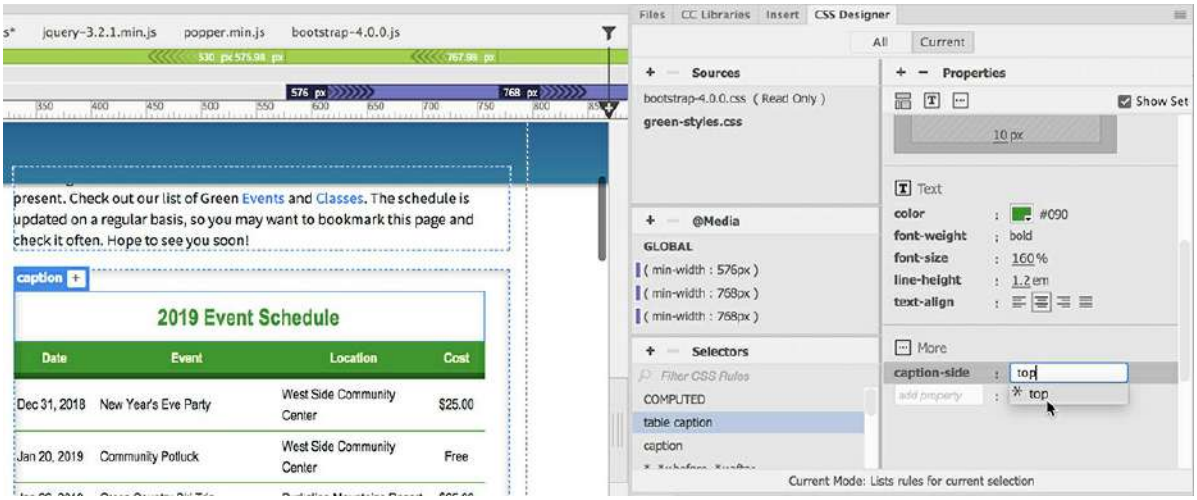
Since that style sheet is read-only, you can override the property in **green-styles.css**. Notice that the first rule listed in the Selectors pane is

`table caption`.

. In CSS Designer, select the rule `table caption`.

This rule was created in Lesson 7 to format the table captions so that they would display more like headings. Since it is a global style, you can use that same rule to override the Bootstrap styling.

. Create the property **`caption-side: top`** in the rule `table caption`.



As soon as the new property is complete, the captions return to the top of the tables. When you update frameworks like Bootstrap—or apply them to an existing site, like we did—you will often find many instances where elements are behaving badly. Don't take any styling for granted.

In fact, did you notice the other glaring issue in the site's styling? It affects every page and is so subtle you may have not recognized it. Give up? Take a look at the bottom border of the left and right sidebars. After we applied the Bootstrap framework, the borders started to display at the bottom of the page content, the full height of the tallest column. What's causing this?

Unfortunately, this problem won't be solved by a trip to the CSS Designer. It took several minutes of troubleshooting in a browser before I could track down the culprit in this case. This new bug was introduced by Bootstrap 4, which was recently added to Dreamweaver. Version 3 used floats to move items around to create the columns we're using. But that method has lots of issues and requires some tricks and workarounds to make it function properly in every browser and device. A technique dubbed Flexbox was developed in 2009 and added to CSS3 to address the problems designers had using floats.

The techniques used by Flexbox are too complex to discuss here, but a feature of this new layout mode is causing our issue. In Lesson 3: "CSS Basics Bonus," we observed one of the problems with floating multiple columns of content over a colored or graphical background: If one column contains less content, it will be shorter than the others. In that bonus lesson, we used a simple solution; we just applied the same background color to all the columns and the

containing element.

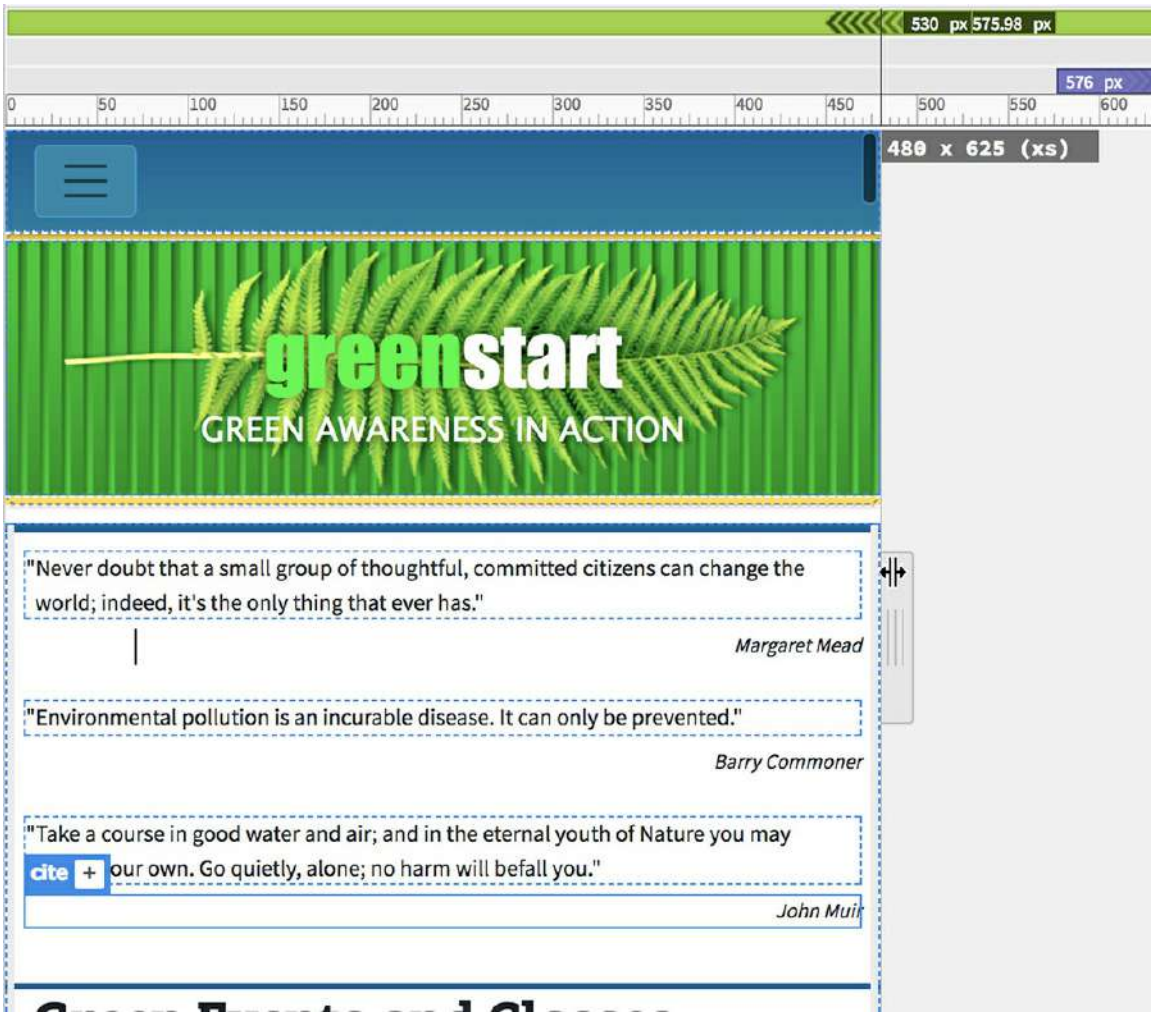
Designers who create multicolumn designs often want all columns to be the same height, even if they do not contain the same amount of content. They also don’t always want to apply the same background color. Enter Flexbox. The basic concept of Flexbox was to take all the guesswork and frustration out of creating HTML layouts. In this case, the current layout bug is actually the framework trying to *help* us.

Unfortunately, the property that is creating this issue is a default setting of Bootstrap 4, which means you won’t be able to use any troubleshooting tool to track it down. Instead, we have to create a new media query and declare a new property to fix it. The first step is to identify the screen widths at which the height bug is relevant.

. Drag the scrubber to the left.

Note the width where the page switches to one column.

Examine Sidebar 1 and Sidebar 2.

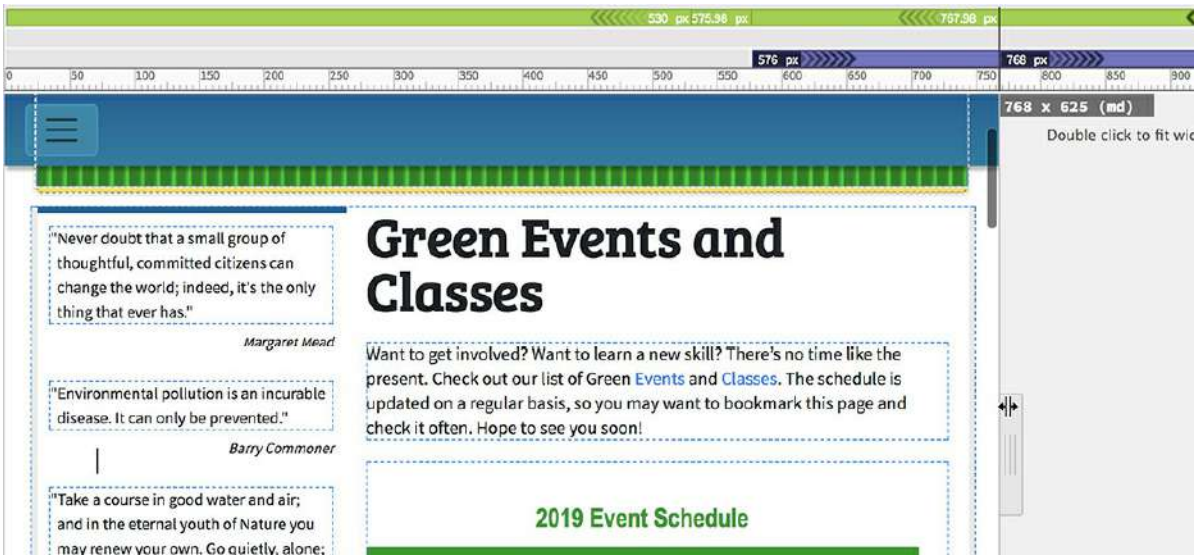


The layout converts to one column at 767 pixels. Note how the sidebars now collapse to the

height of their content. The height bug is not affecting them in the one-column layout.

- Drag the scrubber to the right until the layout appears in two columns.

Examine Sidebar 1 and Sidebar 2.



At 768 pixels, the layout converts to two columns. In a two-column layout, Sidebar 1 matches the height of the main content. Sidebar 2 remains collapsed to the height of its content. So the height issue only affects the columns when the layout forms two or more columns. The solution is to create a rule that styles the column layout at hand. The two classes that need to be targeted are `.col-md-4` and `.col-lg-3`, but only at screen widths 768 pixels and larger.

- If necessary, drag the scrubber to 768 pixels.
- Click the Add Media Query icon at the top of the scrubber.

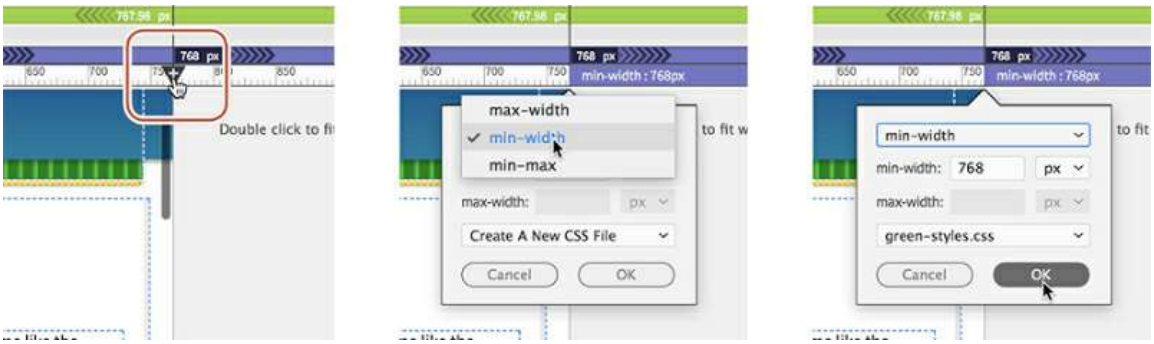
The Media Query Definition dialog appears. The `max-width` field is populated with the current scrubber position (768 px). The `min-width` field is grayed out. The new rule will need to apply to the elements 768 pixels and above, so we need to change the default settings in the dialog.

- In the Media Type dropdown, select `min-width`.

As soon as you select the `min-width` option, the dialog reflects the selection, moving 768 pixels into the `min-width` field.

- If necessary, select **green-styles.css** from the Source dropdown.

Click OK to create the new media query.



The new media query will apply its styles only when the screen is 768 pixels and wider.

. In CSS Designer, select **green-styles.css** > `(min-width: 768).`

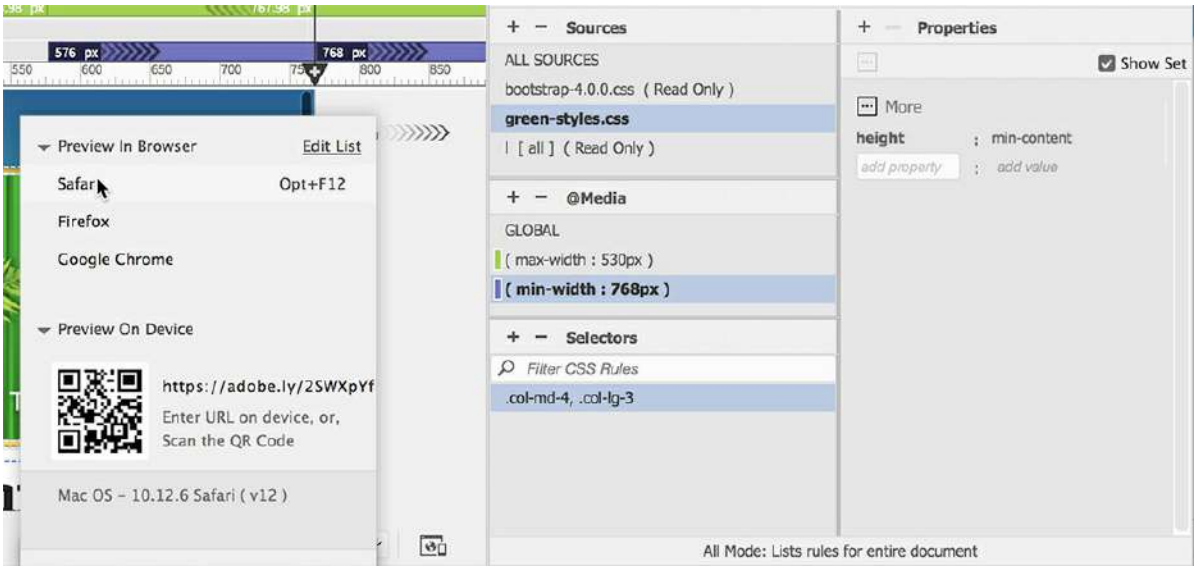
Create the following selector: **`.col-md-4 , .col-lg-3`**

. Create the following property in the new rule: **`height: min-content`**

The new rule may not affect the Live view display. At the time of this writing, the `min-content` value is not supported in Dreamweaver. Since all the Creative Cloud apps are updated from time to time, this property may be supported in the near future. Until that happens, this rule will format the height of the sidebars. To see the effect of the rule, you will have to use a browser.

. Save all files.

. Click the Preview icon in the lower-right corner of the document window. Select your favorite browser from the list.



The page should open in the selected browser. Check out the sidebars and change the width of the browser window to test the new styling at various widths.

. Save and close all files.

The new rule and setting should fix the issue with the sidebar height for all screen widths above

768 pixels. To learn more about Flexbox check out https://spaceninja.com/2015/08/24/what-is-flexbox.

You now know how to test and fix a variety of issues with your new website. Continue testing the pages and tweaking the content for as many different devices as you can lay hands on.

Congratulations! You've adapted your static HTML site for responsive design. By finishing all the exercises in this book up to this point, you have gained experience in all aspects of the design, development, and testing of a standard website compatible with desktop computers and mobile devices. In the final lesson, you will learn how to add web-compatible animation and video to your site.

Review questions

1. How can you target a specific device size or orientation for styling content?
2. True or false? When resetting the styling of a rule you must duplicate all the properties of the rule in the media query.
3. How can you control the number of columns displayed for different screen sizes in a Bootstrap layout?
4. Can tables be made to be responsive?
5. How can you test your webpages for responsive design if you don't have a web server?
6. What do you need to have to enable Real-Time Preview?
7. Can you use one set of fonts for all browsers and devices?

Review answers

1. To target a specific size screen or device, you can create a custom media query in your style sheet.
2. False. You have to create only the properties that need to be reset.
3. In Bootstrap, you can control the number of columns displayed at a specific breakpoint by assigning a predefined class to the structural elements.
4. Yes. Using some CSS3 properties you can format table content to be viewable on any size screen.
5. Dreamweaver provides Real-Time Preview, which enables you to preview your pages on desktop browsers and any available mobile device.
6. You must have a live connection to the Internet and be logged in to an active Creative Cloud account to use Real-Time Preview.
7. At this point, no single font technology will support every computer and mobile device.

The best advice is to create a font stack that will target specific fonts compatible with as many browsers and devices as possible.

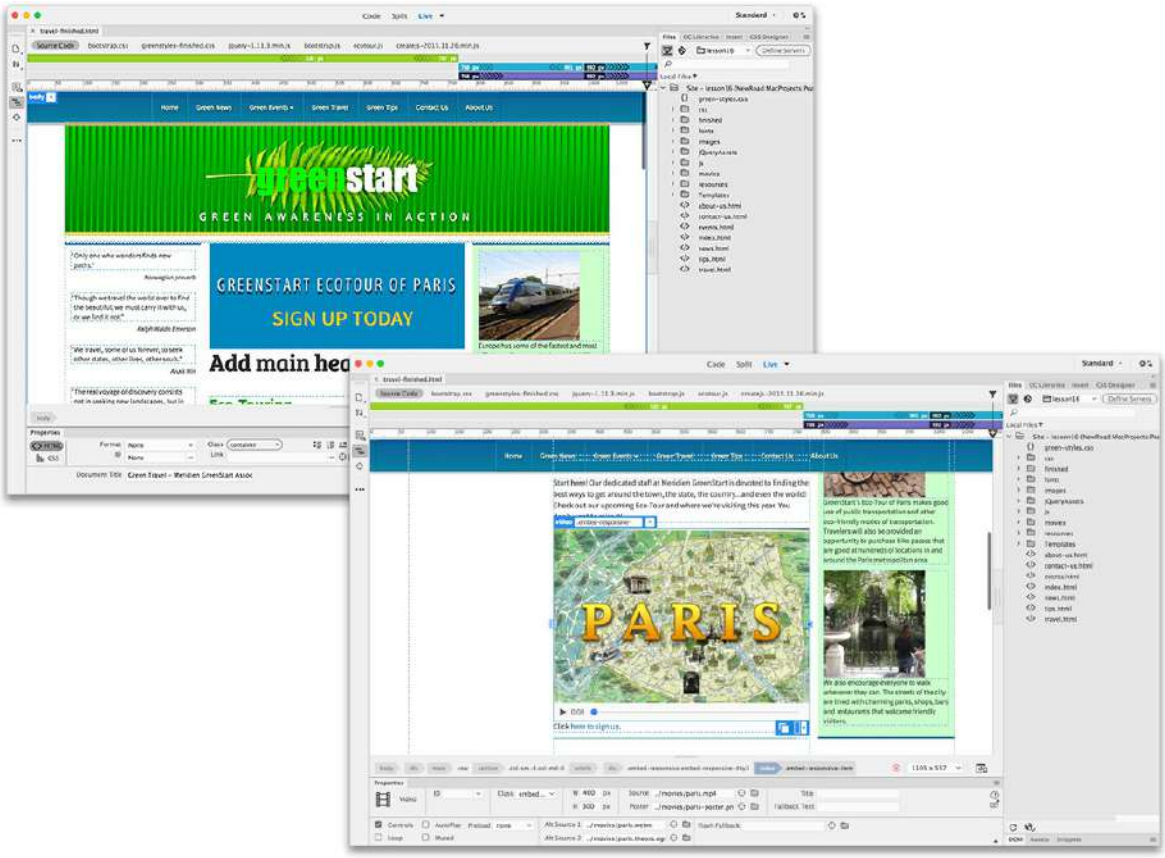
16 Working with Web Animation and Video Bonus

Lesson overview

In this lesson, you'll learn how to incorporate web-compatible animation and video components into your webpage and do the following:

- Insert web-compatible animation
- Insert web-compatible video

This lesson will take about 45 minutes to complete. Download the project files for this lesson from the Lesson & Update Files tab on your Account page at www.peachpit.com, store them on your computer in a convenient location, and define a new site based on the lesson16 folder, as described in the "Getting Started" section at the beginning of this book. Your Account page is also where you'll find any updates to the lessons or to the lesson files. Look on the Lesson & Update Files tab to access the most current content.



Dreamweaver allows you to integrate HTML5-compatible animation and video.

Understanding web animation and video

The web can provide a variety of experiences to the average user. One second you are downloading and reading a best-selling novel. Next, you're listening to your favorite radio station or performing artist. Then, you're watching live television coverage or a feature-length movie. Before Adobe Flash, animation and video were difficult to incorporate onto websites. That's because HTML was invented at a time when even static images were difficult to use on the Internet; video was a dream far off in the future.

For a time, Adobe Flash brought order to this chaos. It provided a single platform for creating both animation and video. However, with the invention and rise in popularity of smartphones and tablet devices over the last decade, Flash has fallen on hard times. For most manufacturers, the power and capability of Flash were too difficult to support on these devices and it was abandoned. Flash is not dead. It's still unmatched for its multimedia power and functionality. But today, all bets are off when it comes to animation and video.

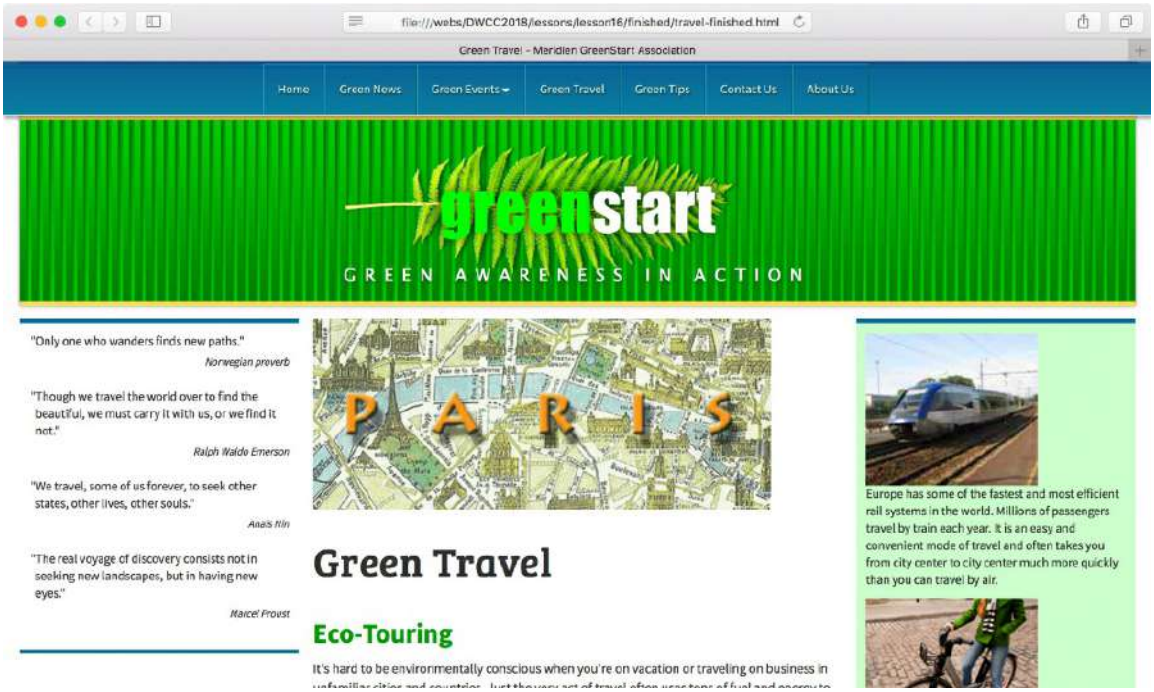
The techniques for creating web-based media are being reinvented. As you may have guessed, this trend away from Flash is ringing in a new era of chaos on the web-media front. Half a dozen or more codecs are competing to become the "be-all, end-all" format for video distribution and playback for the web. The only ray of sunshine in this morass is that HTML5 was developed with built-in support for both animation and video.

Great strides have already been made to replace much of the capability of Flash-based animation using native HTML5 and CSS3 functionality. The status of video is not as clear. So far, a single standard has not yet emerged, which means that to support all the popular desktop and mobile browsers, you'll have to produce several different video files. In this lesson, you'll learn how to incorporate web animation and video into your site.

Previewing the completed file

To see what you'll work on in this lesson, preview the completed page in a browser. The finished page is based on the travel page you created in Lesson 10, "Adding Interactivity."

- Launch your favorite web browser.
- Open **travel-finished.html** from the finished-files folder in lesson16.



The page includes two media elements: the banner animation at the top of the `main_content` region and the video inserted below. Depending on the browser used to view the page, the video playback may be generated from one of three different formats: MP4, WebM, or Ogg.

Note that the banner ad plays when the page loads and loops after it calls you to sign up for the Eco-Tour.

- To view the video, click the Play button. Move the cursor over the video to display the control skin, and click the Play button.

Different browsers support different types of video. Depending on the video format your browser supports, you may notice that the controls fade if you move the cursor away from the video but that they return once you position the cursor over the video again.

. When you're finished previewing the media, close your browser.

Video and animation provide powerful venues for rich web content, and Dreamweaver makes it a simple matter to insert this type of content.

Adobe Animate CC

The animation used in this lesson was built in Animate CC, Adobe's replacement for Flash Professional. Animate works similarly to Flash but creates web animation and interactive content natively using HTML5, CSS3, and JavaScript, among other formats. Animate is available to all Creative Cloud subscribers.

Check out www.adobe.com/products/animate.html for more information about Animate CC.

Adding web animation to a page

Dreamweaver has a built-in and simplified workflow for inserting Animate compositions, making the process a point-and-click operation. Dreamweaver takes advantage of a feature in Animate designed to assist in deploying compositions to other programs and workflows, such as Adobe InDesign, Adobe Dreamweaver, and Apple's iBooks Author. The File > Publish Settings command enables you to export your Animate compositions into a single file or folder. By defining your Publish settings appropriately, you can create a complete set of files that are compatible with these applications.

For the purposes of this exercise, I created an animated banner ad 480 pixels by 200 pixels in

Animate CC and published it to an OAM file, which is an archive file format that contains all the constituent elements needed to support the animation in Dreamweaver and other Adobe applications. In this exercise, you will learn how to insert the animation into a layout and make it responsive.

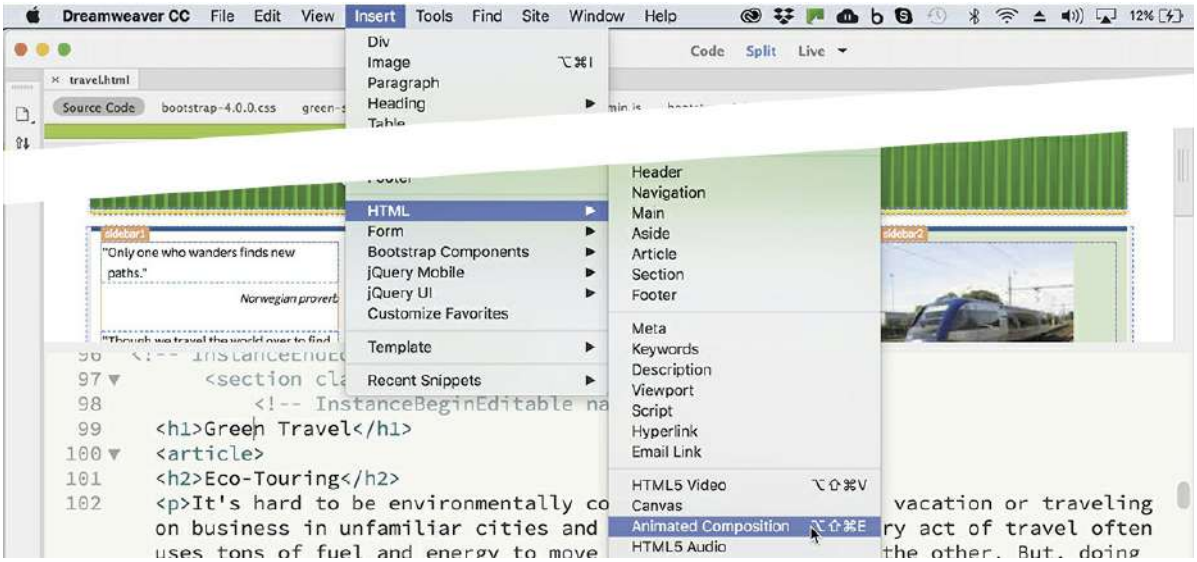
- Open **travel.html** from the lesson16 site root folder in Live view. Ensure that the program is maximized to fit the entire display and that the document window is at least 1100 pixels in width, but less than 1200 pixels.

  The banner will be inserted above the main heading.

- Click the heading *Green Travel*.

  The Element Display appears focused on the `h1` element.

- Choose Insert > HTML > Animated Composition. You can also use the Animated Composition option in the Insert panel's HTML category.

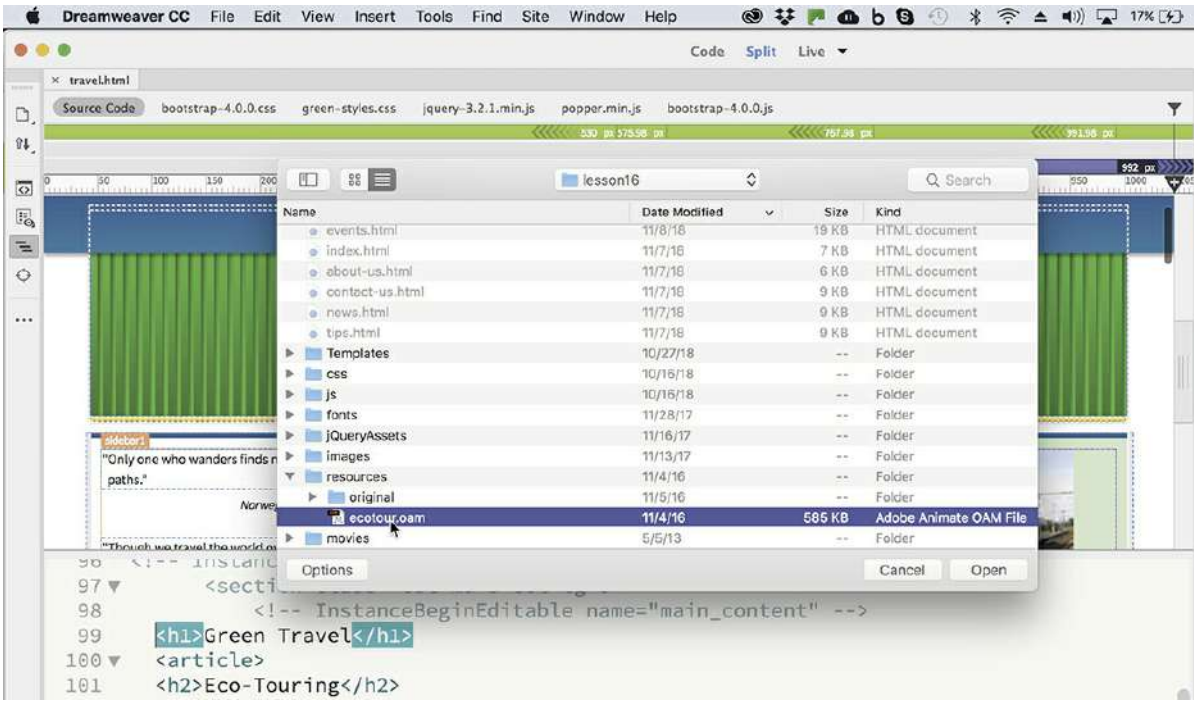


- Click Before.

  A file dialog opens.

- Navigate to the lesson16/resources folder.

  Select **ecotour.oam**.



. Click OK/Open to insert the composition.

The banner should appear and begin to play. It is designed to loop over and over. Although the animation is designed to fit in the center column, there seems to be a small issue with the width of the animation below 1200 pixels. The blue frame of the animation seems to extend underneath Sidebar 2.

. Save the file and switch to Code view.

Examine the HTML code.

The parent structure for the animation should still be selected. This is not the animation itself; instead, Dreamweaver has inserted an `<object>` element that points to the actual animation stored within the site.

The first issue you have to deal with is that Dreamweaver for some reason has increased the width of the animation. The actual banner is supposed to be 480 pixels by 200 pixels, but the `<object>` element displays HTML attributes set for 500 by 220.

. In the Code view window, edit the width to say **480**.

Edit the height to **200**.

```
97 ▼       <section class="col-md-8 col-lg-6">
98             <!-- InstanceBeginEditable name="main_content" -->
99 ▼   <object id="EdgeID" type="text/html" width="480" height="200" data-dw-
       widget="Edge" data="animation_assets/ecotour/Assets/ecotour.html">
100    </object>
101    <h1>Green Travel</h1>
```

. Switch to Live view. Observe the animation.

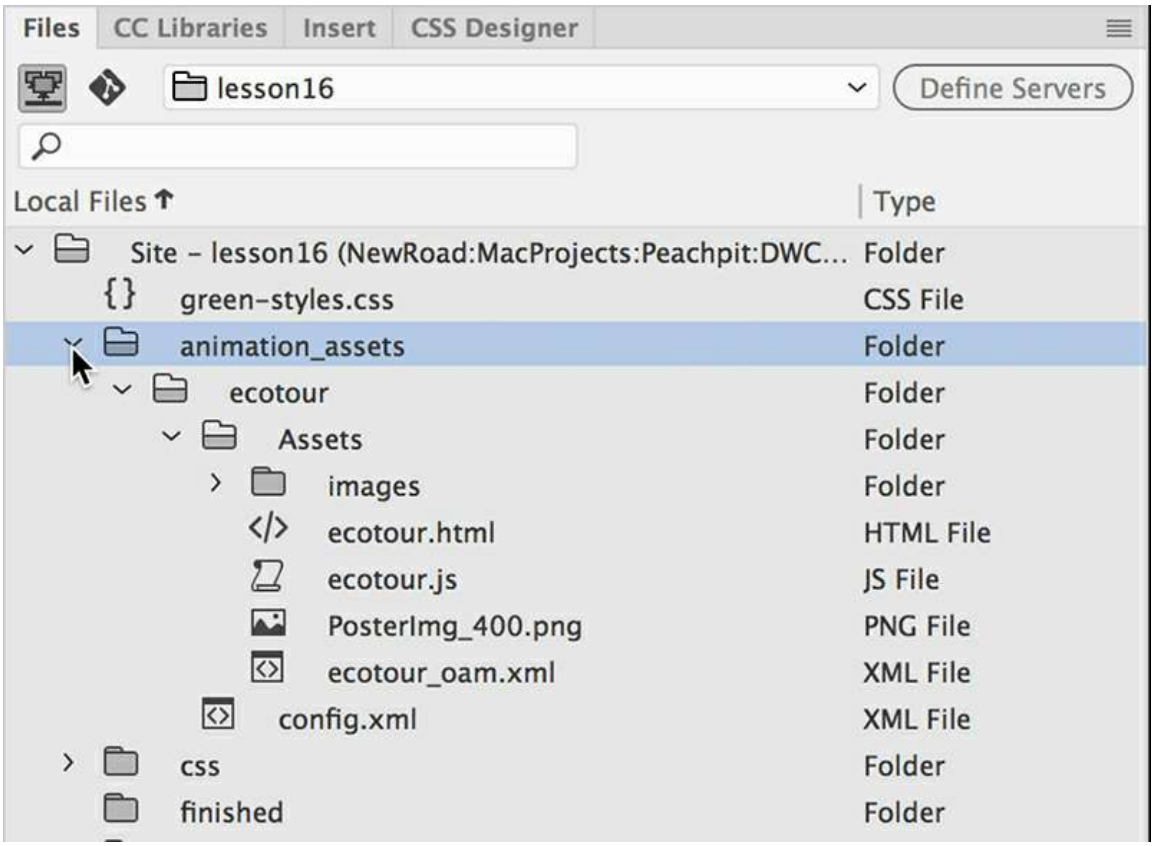
The animation fits within the center column. The banner animation plays automatically in Live view once the code is processed.

It may seem like a miracle that this amazing banner animation was created by one simple file, but what you cannot see is that Dreamweaver just created in the site root directory a new folder that is populated by a half dozen files.

Note

You may need to click the Refresh icon at the bottom of the panel to see all the files in the site folder. The files may appear in a different order than that pictured.

. Open the Files panel and examine the list of folders in the site root.



Note

Dreamweaver will not automatically upload all the support files needed for the Animate composition when you select dependent files. Be sure the entire contents of the animation_assets folder is uploaded when publishing the site to the web.

The folder named animation_assets now appears in the root directory. The folder was generated automatically and contains all the files needed to support the composition. The entire folder must be uploaded to the web host when **travel.html** is posted.

. Save all files.

Congratulations! You've successfully incorporated an HTML5- and CSS3-based animation in your page. But you're not done yet. Whenever you insert a component in your layout, especially animations and video, you need to check to see whether they conform properly to your responsive design.

Making the animation responsive

The heading on this exercise is a bit misleading. The animation you inserted earlier is already responsive. But that doesn't mean it will respond properly in the current layout. Let's check it

out.

- If necessary, open **travel.html** from the lesson16 site root folder in Live view. Ensure that the program is maximized to fit the entire display and that the document window is at least 1100 pixels in width, but no more than 1200 pixels.

  The layout should be displayed in three columns, with the animation playing at the top of the middle column.

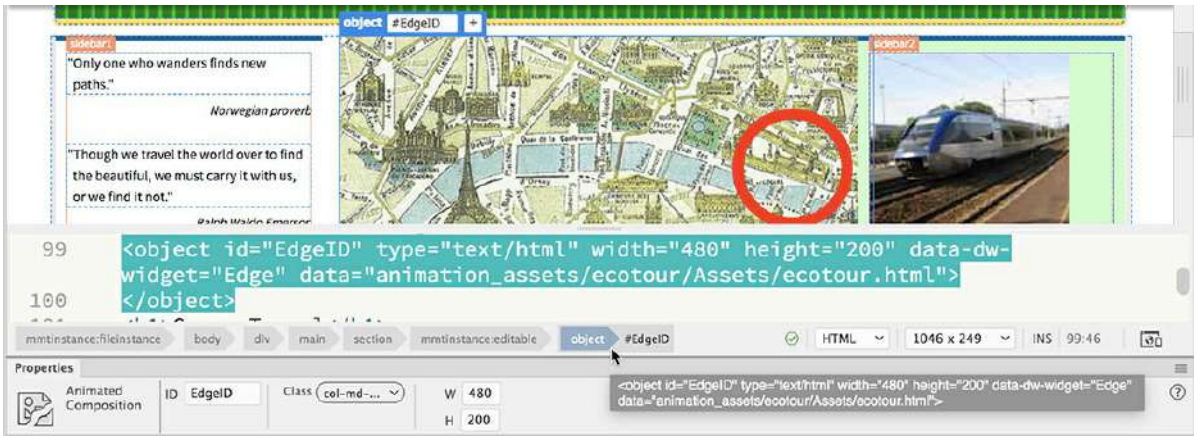
- Drag the scrubber to the left until the document window is narrower than 480 pixels.

  The animation appears fully in all widths until you get below 500 pixels or so. At smaller widths, the animation protrudes from the main content section and protrudes off the screen on the right side.

  Bootstrap provides a method for supporting responsive animation and video, but at the time of this writing, Dreamweaver only has built-in support for embedding video. You'll have to create and modify the code for the animation yourself. The first step is to wrap the animation in a Bootstrap responsive container.

- Drag the scrubber back to the right side.
- Switch to Split view. In Code view, locate the `<object>` element containing the animation (around line 99).
- Insert the cursor anywhere in the `<object>` element.

  Select the `object#EdgeID` tag selector.



The entire element is highlighted in Code view.

Note

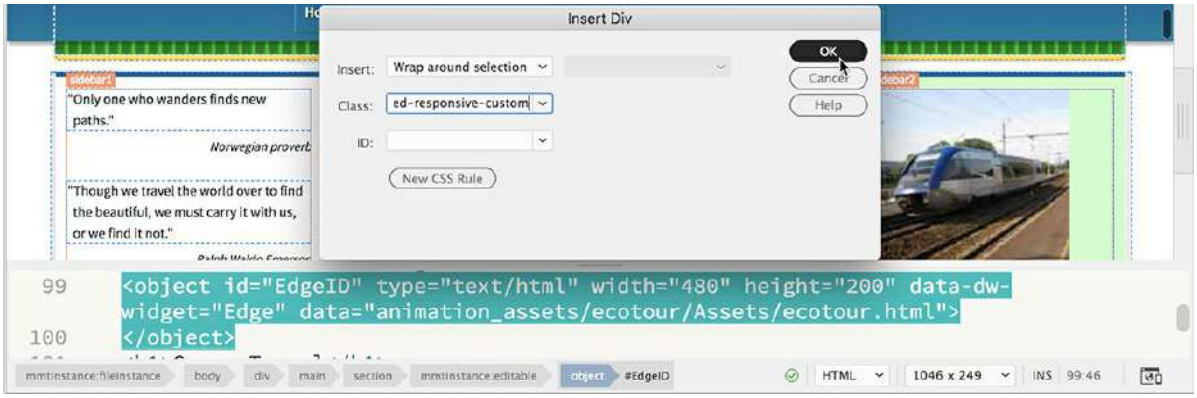
Make sure the Insert menu shows the option Wrap Around Selection.

. Select Insert > Div.

The Insert Div dialog appears. You'll need to enter two class names.

. In the Class field, enter the following classes:

```
embed-responsive embed-responsive-custom
```



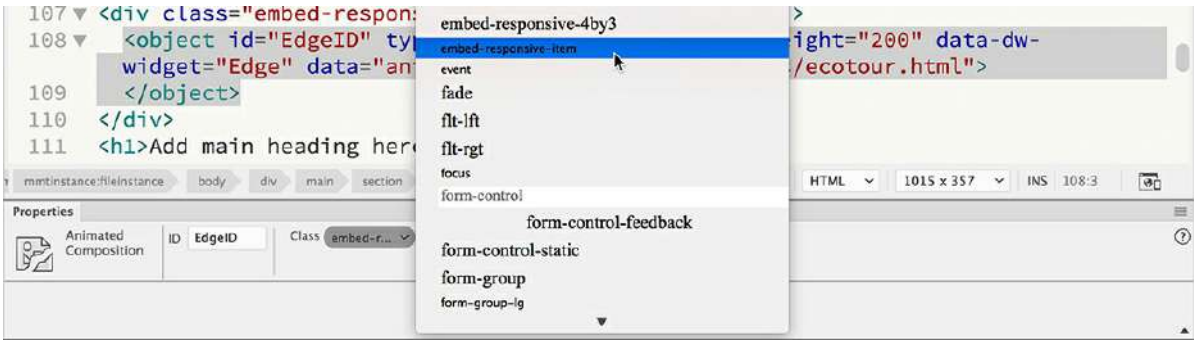
As you type, the first class may appear in the field. Feel free to select it using the mouse or keyboard. The second class name will have to be entered fully. Be sure to insert a space between the two names.

. Click OK.

The responsive `<div>` now wraps the `<object>` element. You'll need to add a responsive class to it as well. The animation disappears from the layout. The class you assigned to the `<object>` element caused it to collapse visually. It's still there, as you can see from the code, but you'll have to add a new rule to make it appear properly and work within the new structure.

. Select the `object#EdgeID` tag selector again.

In the Property inspector, click the Class field and select this class: **`embed-responsive-item`**



Note

Remember that the All button must be enabled before you try to select the CSS source.

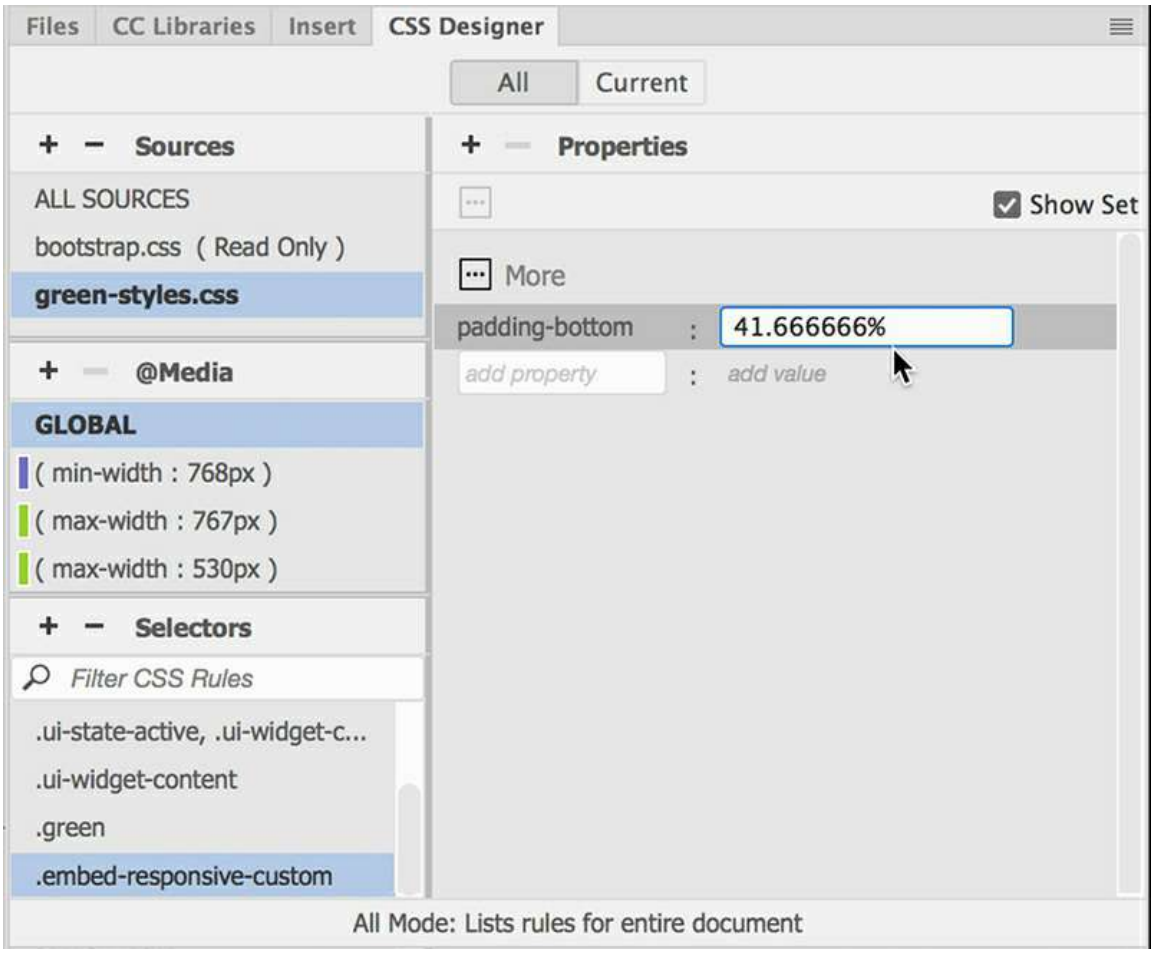
. Choose **green-styles.css** > `GLOBAL` in the CSS Designer.

. Create the following selector:

`.embed-responsive-custom`

. Add the following property:

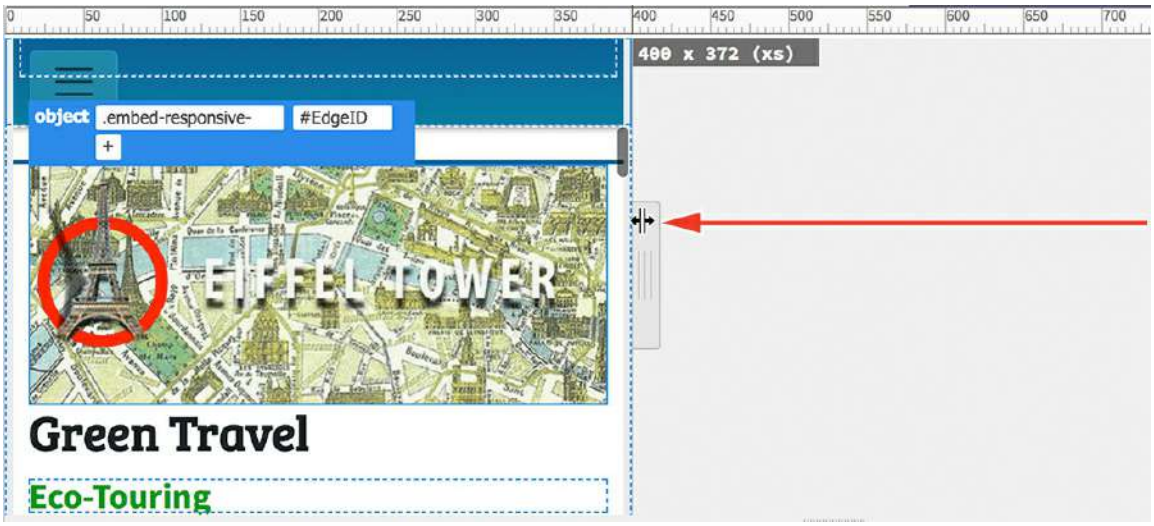
`padding-bottom: 41.666666%`



As soon as the property is created, the animation reappears in the layout.

The Bootstrap framework currently supports four basic embed ratios. In most cases, videos will conform to 4:3 or 16:9, which match the most popular formats. If the animation matched one of these ratios, you would select an existing Bootstrap class that supports it. But it doesn't, so we had to use a custom padding property derived from a factor of its height and width. Check out http://tinyurl.com/fluid-width-animation to learn more about working with Bootstrap components.

. Drag the Scrubber to the left until the document window is narrower than 400 pixels.



The animation scales down to match the width of the column.

. Drag the Scrubber all the way to the right.

Save all files.

You've now completed an HTML5-compatible animation. Now, let's see how easy it is to work with video.

Adding web video to a page

Implementing HTML5-compatible video in your site is a bit more involved than it was when you had to insert only a single Flash-based file. Unfortunately, there still is no single video format that is supported by all browsers in use today. To make sure your video content plays everywhere, you'll have to supply several different formats. Dreamweaver provides a built-in technique to add multiple video files so you won't have to do all the coding yourself. In this exercise, you will learn how to insert HTML5-compatible video on a page in your site.

. If necessary, open **travel.html** in Split view. Ensure that the program is maximized to fit the entire display and that the document window is at least 1100 pixels in width.

You will insert the video in the `main_content` section of the page.

Note

Open the Property inspector and dock it to the bottom of the document window, if necessary.

. Click the paragraph *Click here to sign up*.

The Element Display appears focused on the `p` tag. As mentioned earlier, the Insert panel

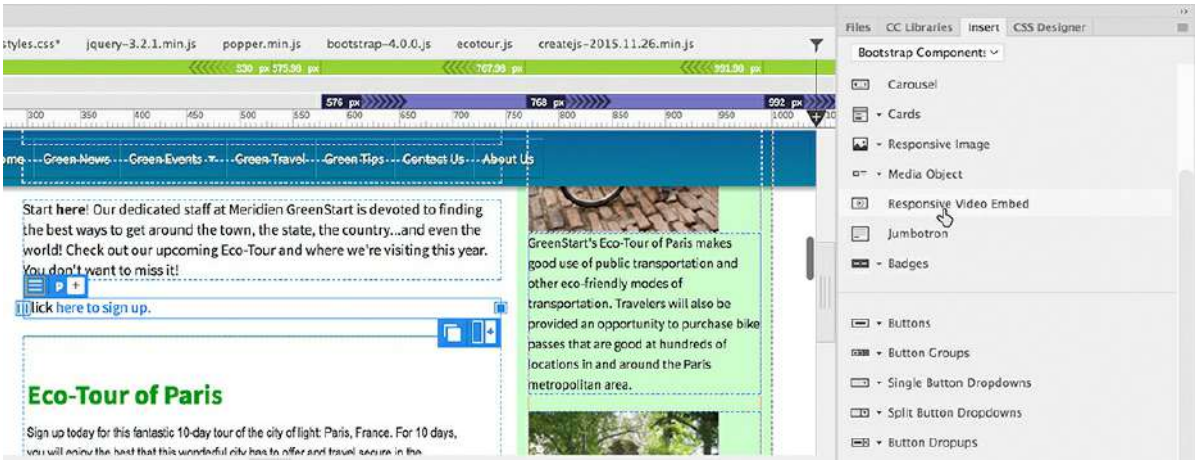
provides a Bootstrap option to insert responsive video.

. Choose Window > Insert, if necessary.

Select the **Bootstrap Components** category.

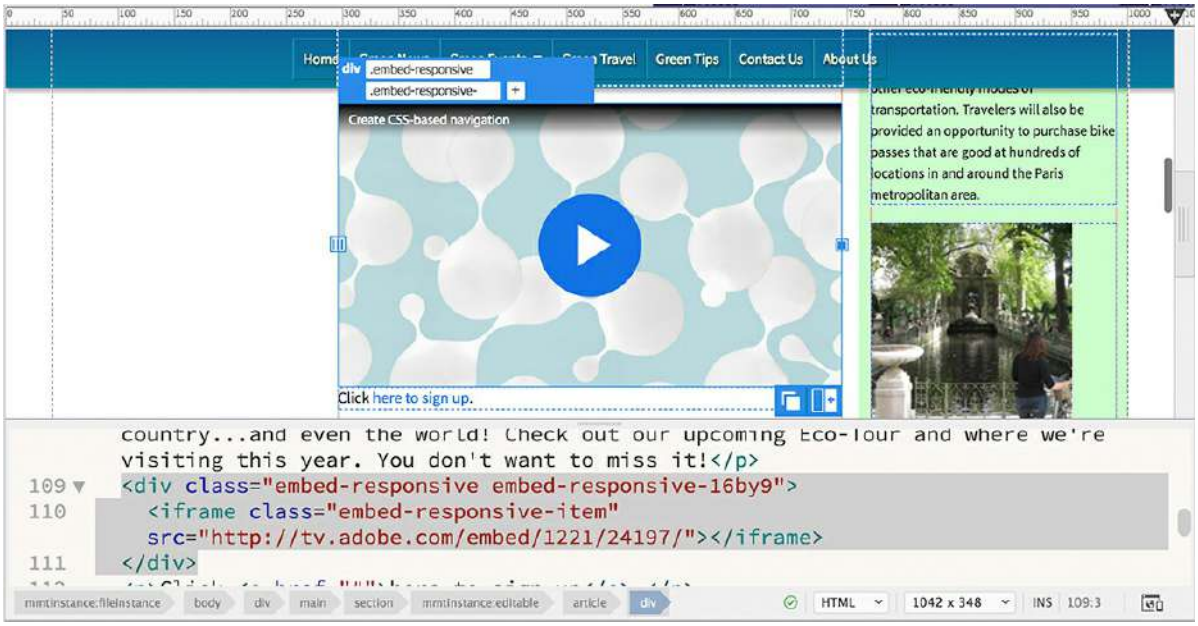
Click **Responsive Video Embed**.

The Position Assist dialog appears.



. Click Before.

A Bootstrap-compatible video placeholder appears on the page. It contains a sample video hosted on the web. If you have a live Internet connection, you can actually play the sample video.



This placeholder uses an `<iframe>` element to host the video. An `<iframe>` element is usually employed to host web content from a third-party website, such as YouTube or Vimeo. Since you are going to host your own video, there's no need for the `<iframe>`. You're going to replace the `<iframe>` with an HTML5-compatible `<video>` element.

. In the Code view window, insert the cursor in the `<iframe>` element.

. Select the `iframe.embed-responsive-item` tag selector. Note the class name.

The `<iframe>` is now selected. Let's replace it with the HTML5 `<video>` element.

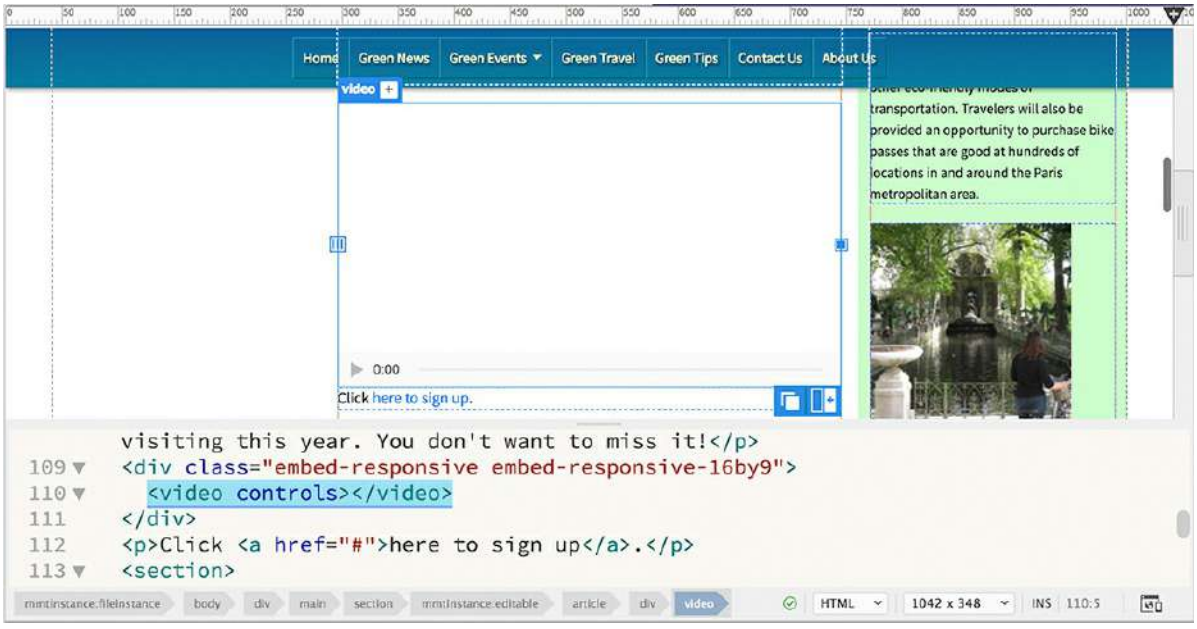
. Delete the entire `iframe` element.

Note

The video workflow requires the use of the Property inspector. If the Property inspector is not visible, select Window > Properties to display it.

The cursor is still in the location of the video element.

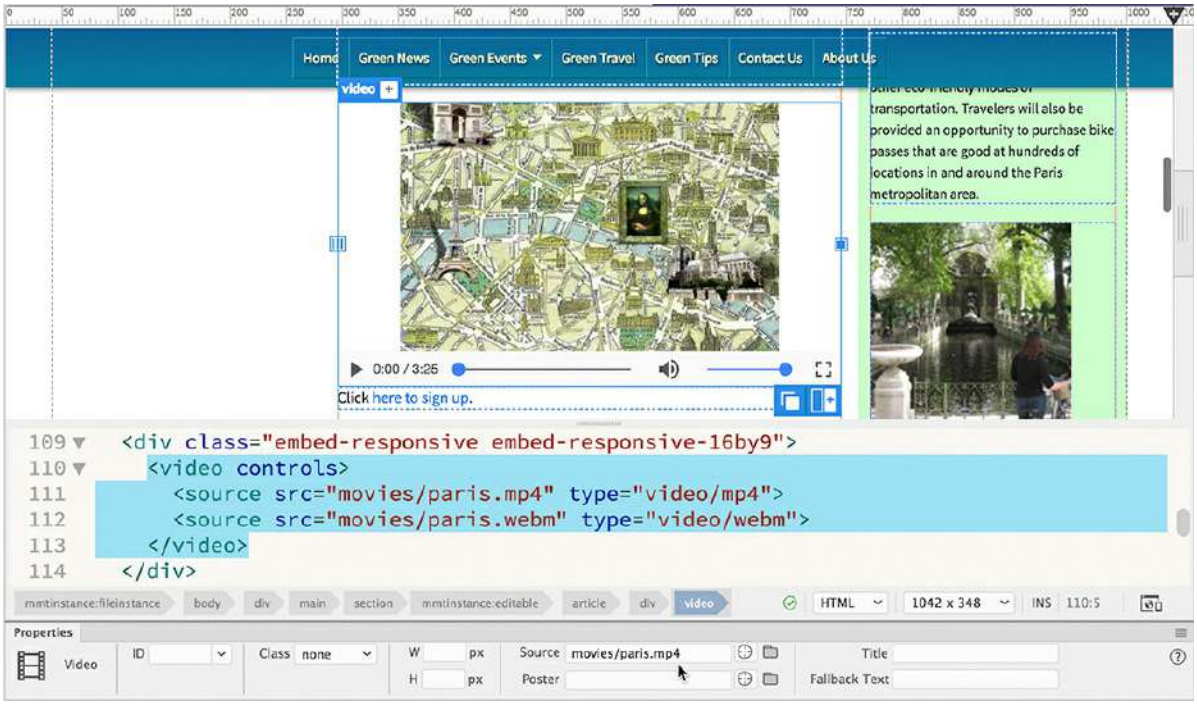
. Choose Insert > HTML > HTML5 Video.



The HTML5 `<video>` element appears. The Property inspector displays options for targeting

the video source files. Note that this interface enables you to specify up to three video source files and one Flash fallback file. The first step is to select your primary video source.

- In the Property inspector, click the Browse icon in the Source field. Navigate to the **movies** folder and select the **paris.mp4** file. Click OK/Open.

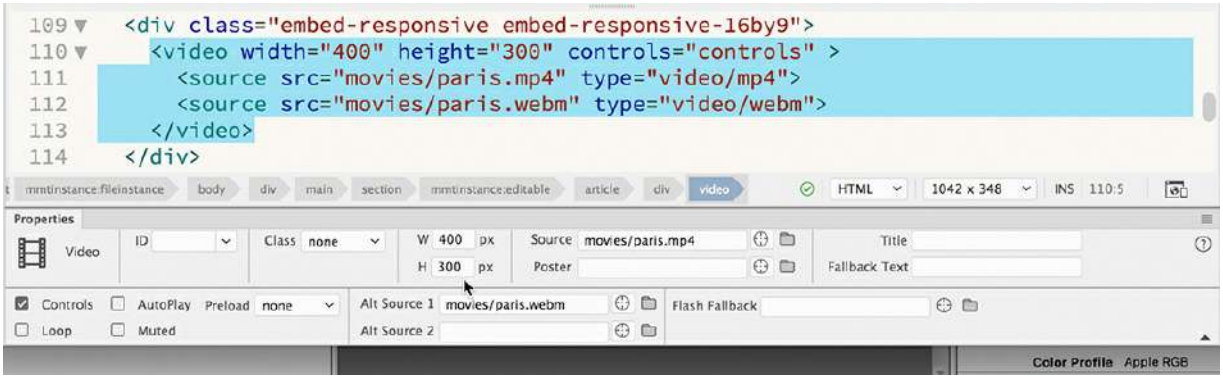


The MP4 video displays, replacing the original placeholder.

Now let's enter the width and height of the video. Although you will use CSS to control the height and width of the video, entering width and height attributes is recommended to help render your page faster in the browser.

- In the Property inspector's Width field, enter **400**.

In the Height field, enter **300**.



The attributes entered don't seem to affect the size of the video; it still looks a bit small. You'll address this issue later when we make it responsive, but first you need to complete the `<video>` element. Best practices for HTML5 suggest you provide alternate source files.

Adding alternate video sources

The MP4 file format will be the primary video format loaded. MP4, also known as MPEG-4, is a video format based on Apple's QuickTime standard. It is supported natively by iOS-compatible devices, such as the iPhone and iPad, as well as by Apple's Safari browser. Many experts advise loading MP4 files first; otherwise, iOS devices may ignore the `video` element altogether. MP4 is now supported by Chrome, Firefox, and Opera.

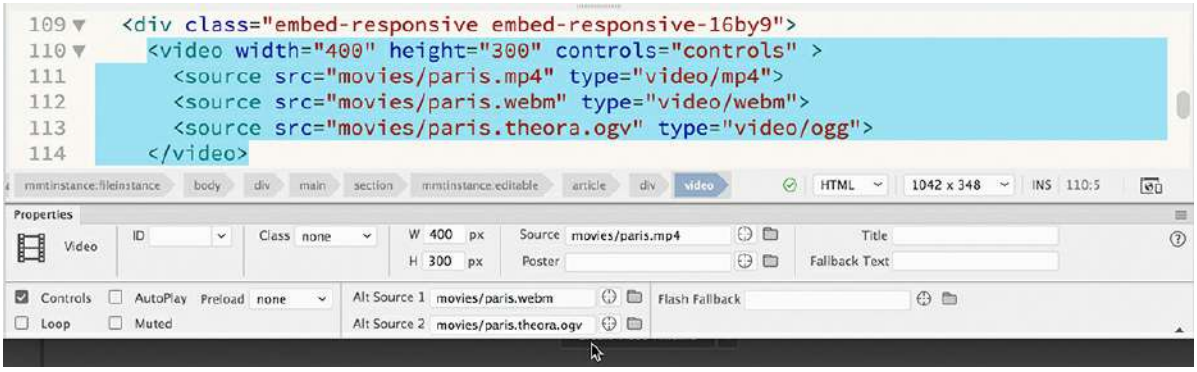
You may have noticed that Dreamweaver automatically inserted a WebM version of the movie as Alt Source 1. WebM is an open source, royalty-free video format sponsored by Google. It is compatible with the latest versions of Firefox, Chrome, Microsoft Edge, and Internet Explorer 9. Older versions of these browsers may not support WebM, but the latest ones do.

To round out our HTML5 video selections, the next format you'll load is a lossy, open source multimedia format: Ogg. It is designed for the distribution of multimedia content that is free of copyright and other media restrictions and is supported by older browsers.

Note

Ogg is a container format. When the container contains a video, it uses the extension .ogv.

- Click the Browse icon for the Alt Source 2 field. Select the file **paris.theora.ogv** from the **movies** folder and click OK/Open.

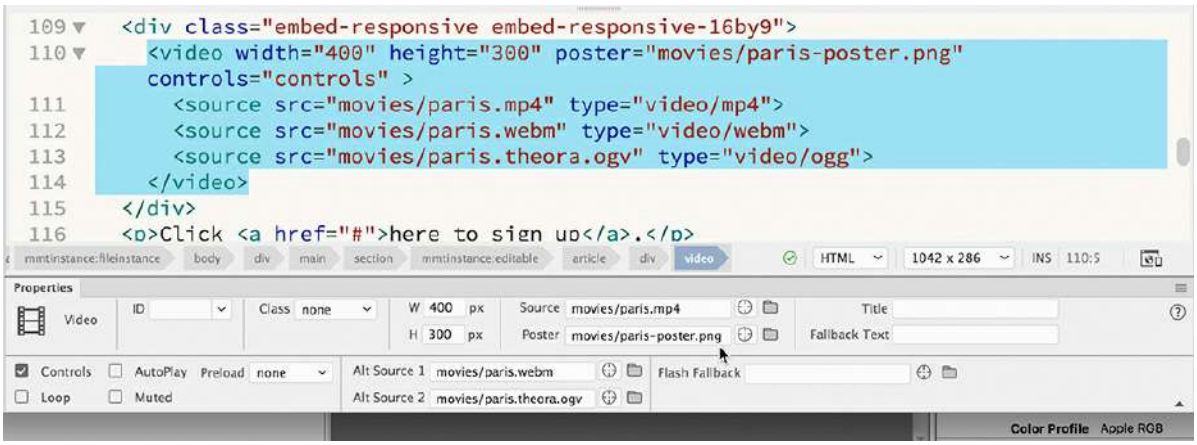


These three formats support all the modern desktop and mobile browsers.

- Save the file.
- If necessary, switch to Live view.

In some browsers, the `<video>` element won't generate a preview of the video content. You should add an image placeholder by using the Poster field in the Property inspector.

- If necessary, select the `<video>` tag selector. In the Property inspector, click Browse in the Poster field.
- In the **movies** folder, select **paris-poster.png** and click OK/Open.



A preview image has been applied to the `<video>` element. It won't be visible in Dreamweaver if a visitor uses a browser that doesn't support HTML5 video; they will see the poster image instead.

By supplying multiple video sources and a poster, you are ensuring that something will always appear in the browser whenever this file loads.

. Save all files.

Different video controls appear below the element, depending on what video format is displayed. In the next exercise, you will learn how to configure these controls and how the video will respond to the user.

Setting HTML5 video options

One of the final steps for configuring the video is to decide what other HTML5-supported options to specify. The options are displayed within the Property inspector whenever the `<video>` element is chosen. The options are selectable in all views.

. If necessary, open **travel.html** in Split view.

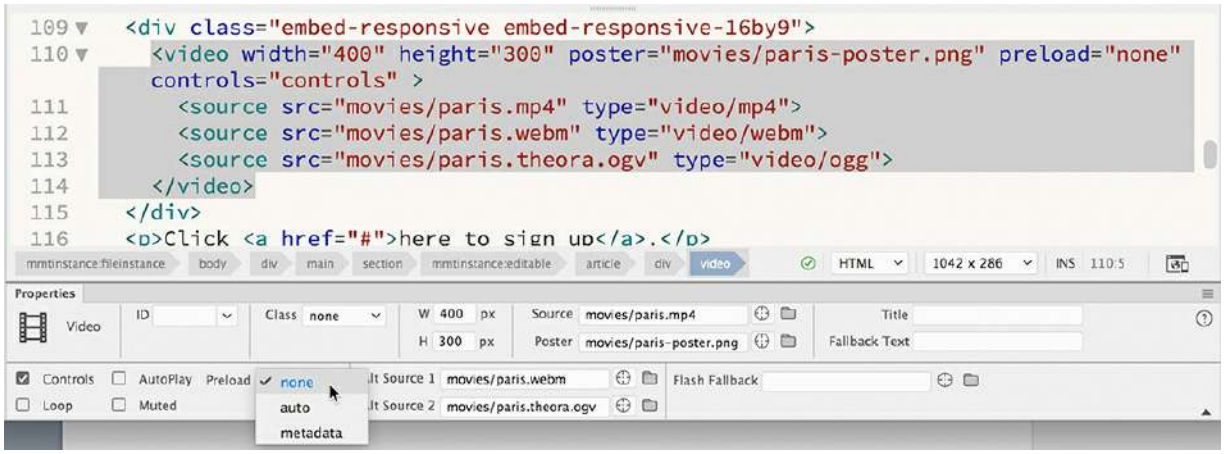
Select the `<video>` tag selector.

Observe the left side of the Property inspector.

- **Controls** displays visible video controls.
- **AutoPlay** starts the video automatically after the webpage loads.
- **Loop** causes the video to replay from the beginning automatically once it finishes.
- **Muted** silences the audio.
- **Preload** specifies the method in which the video loads.

. If necessary, select the Controls option and deselect the AutoPlay, Loop, and Muted options.

Set Preload to **none**.



Note

The Preload option has to be selected manually. It appears to already be selected, but the option is not added to the element until you select it.

Video is very memory- and bandwidth-intensive. This is especially true for phones and tablets. Setting Preload to None prevents any video resources from downloading until the user actually clicks the video. It may require a few extra seconds for the video to download when launched, but your visitors will appreciate that you are respectful of their minutes and data plan.

Although these settings complete the structure of the `<video>` element, you still need to make sure the video works properly on all screens and devices.

Making the video responsive

The `<video>` element is complete, but you may have noticed that the preview doesn't seem to match the dimensions you entered earlier. That's because the CSS controlling the Bootstrap structure of the element has already taken over from the HTML attributes. Unfortunately, the default settings for the embedded video are expecting a video with an aspect ratio of 16:9. The one you're using is actually 4:3. Luckily, this is an easy fix.

. If necessary, open **travel.html** in Split view.

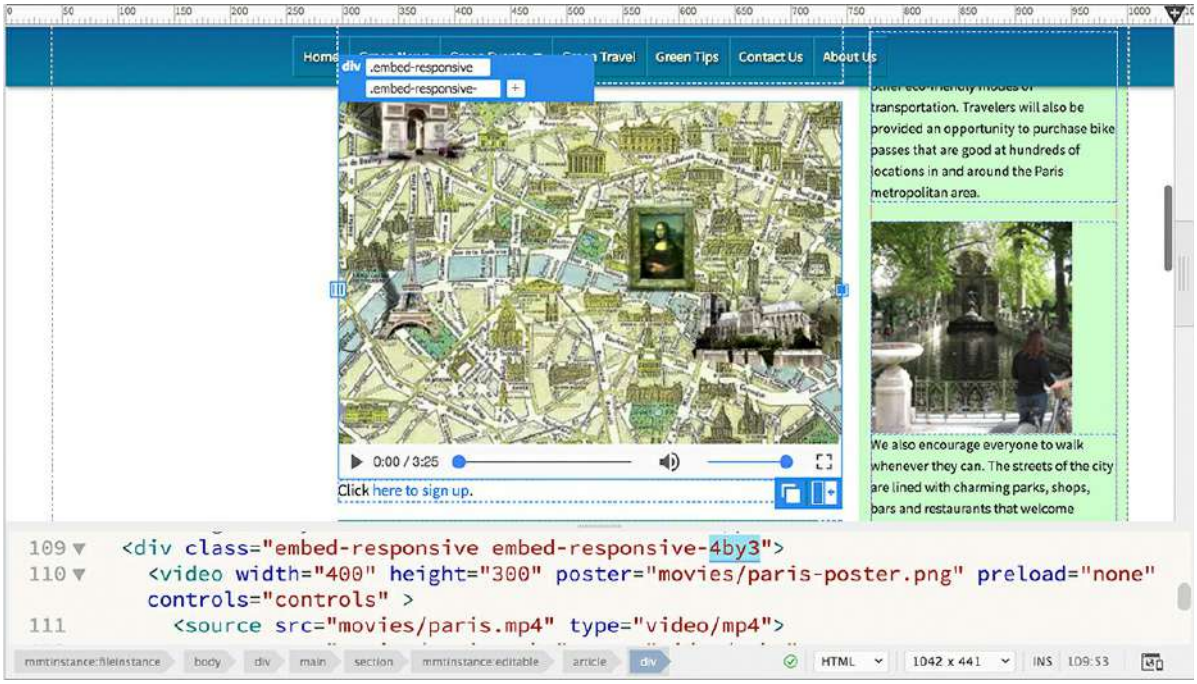
  Make sure that the program fills the entire display and that the document is at least 1100 pixels wide.

. In the Code view window, locate the `<video>` element (around line 110).

Note the parent `<div>` element.

This is the responsive Bootstrap embed element. It controls the size of the video on all screens and devices. Note the classes assigned to it. The second class, `embed-responsive-16by9`, designates this structure for an aspect ratio of 16:9.

- Edit the class as highlighted: `embed-responsive-`**`4by3`**



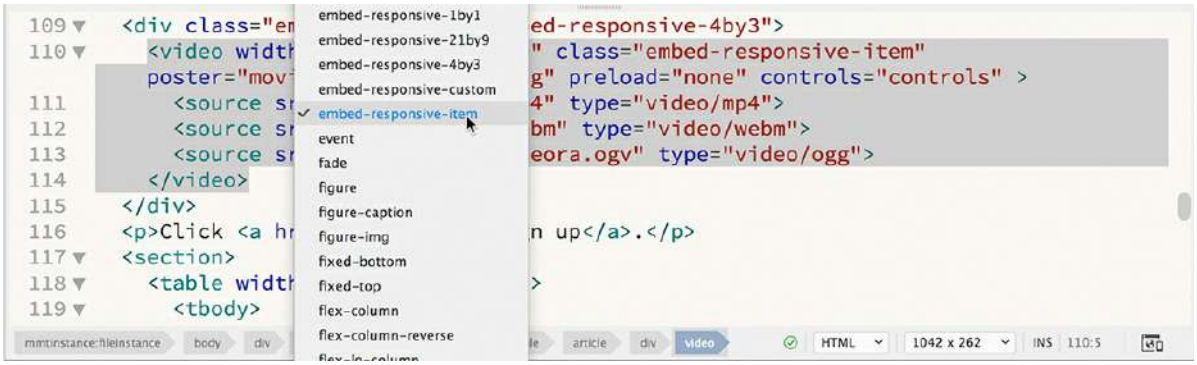
As soon as you complete the change, the video preview expands to fill the entire column. There's only one modification left to do. The original `<iframe>` had a responsive Bootstrap class assigned to it. To complete the responsive styling, you need to apply the same class to the new HTML5 `<video>` element.

Tip

You may need to click the Refresh button in the Property inspector to see the tag selector.

- Select the `video` tag selector.

- In the Property inspector's Class menu, select `embed-responsive-item`.



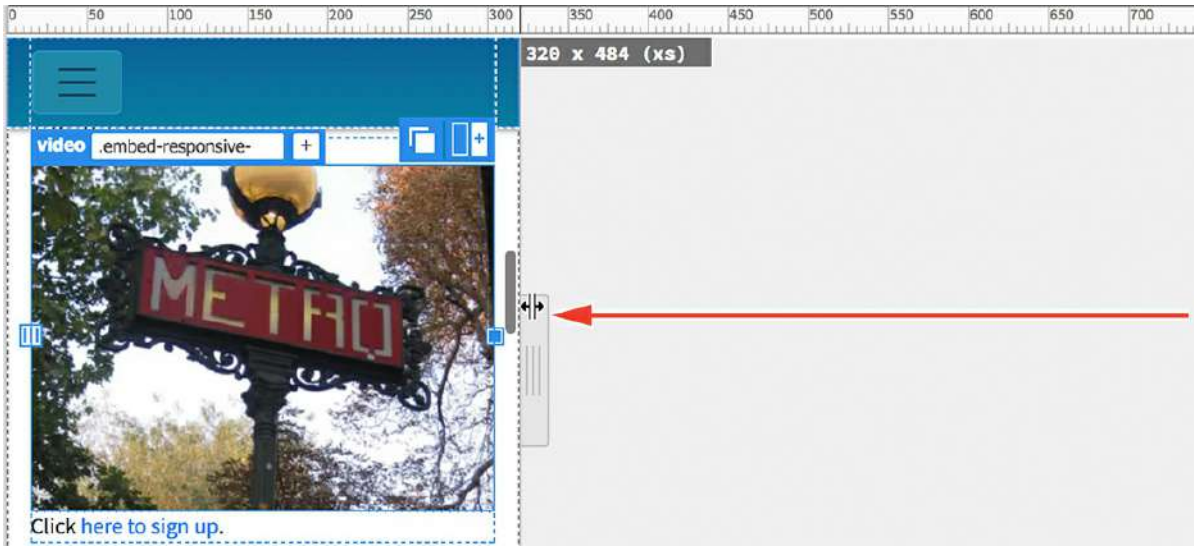
- Save all files.
- Preview the page in Live view or in a browser. If the video controls are not visible, move your cursor over the still image to display them. Click the Play button to view the movie.

Note

The animation may not preview properly using real-time preview. You may need to open the file directly in the browser.

Depending on where you preview the page, you will see one of the four video formats or the static poster image. For example, in Live view you will see the MP4-based video. The controls will also look different depending on what format is displayed. This movie has no sound, but the controls will often include a speaker button to adjust the volume or mute the audio.

- In Dreamweaver, drag the Scrubber to the left to check the display of the video at various screen widths.



The video resizes as needed to fit the available space.

You've embedded three HTML5-compatible videos, which gives you support for most browsers and devices that can access the Internet, and a static poster for the rest. But you've learned only one possible technique for supporting this evolving standard. To learn more about HTML5 video and how to implement it, check out the following links:

http://tinyurl.com/video-HTML5-1

http://tinyurl.com/video-HTML5-2

http://tinyurl.com/video-HTML5-3

To learn more about implementing video for mobile devices, check out these links:

http://tinyurl.com/fluid-video

http://tinyurl.com/fluid-video-1

Don't host your own videos

After showing you how to insert and host your own web-compatible videos, it's a good time to tell you that many experts think that hosting your own videos is not a good policy. For one thing, video formats have not been standardized across all browsers and devices. You can't simply upload one video format and support all your potential visitors. Millions of individuals and even large corporations have given up hosting their own video and contracted with video-hosting services like YouTube or Vimeo.

One advantage of using a hosting service is that you only have to upload a single video format. They will handle the conversion of your video to any other formats needed. And if the standards change again, the service will usually convert your videos for you as needed. At least that's what happened at YouTube when Flash was dumped a few years ago.

For noncommercial users, you can usually host many gigabytes of video for free or for little cost.

Some pay to remove advertising or the logos of the hosting services. You may even partner with these services to make money for yourself or your company.

Check out http://tinyurl.com/do-not-host-video to learn why some people think that hosting your own video is not the best plan. Check out http://tinyurl.com/video-hosting-overview for an overview of several hosting services.

Review questions

1. What advantage does HTML5 have over HTML 4 regarding web-based media?
2. What programming language(s) created the HTML5-compatible animation used in this lesson?
3. True or false: To support all web browsers, you can select a single video format.
4. In browsers or devices that do not support video, what can you do to provide some form of content to these users?
5. What advantage do video hosting services offer over doing it yourself?

Review answers

1. HTML5 has built-in support for web animation, video, and audio.
2. The animation used in this lesson was created by Adobe Animate natively using HTML5, CSS3, and JavaScript.
3. False. A single format supported by every browser has not emerged. Developers recommend incorporating four video formats to support the majority of browsers: MP4, WebM, Ogg, and FLV.
4. You can add a static poster image (GIF, JPEG, or PNG) via an option in the Property inspector to provide a preview of the video content in incompatible browsers and devices.
5. Most video hosting services only require you to upload a single video format. They will convert it as needed to support all the various browsers and devices.

The fastest, easiest, most comprehensive way to learn

Adobe Creative Cloud

Classroom in a Book®, the best-selling series of hands-on software training books, helps you learn the features of Adobe software quickly and easily.

The **Classroom in a Book** series offers what no other book or training program does—an official training series from Adobe Systems, developed with the support of Adobe product experts.

To see a complete list of our Classroom in a Book titles covering the 2019 release of Adobe Creative Cloud go to:
www.adobepress.com/cc2019

Adobe Photoshop CC Classroom in a Book (2019 release)
ISBN: 9780135261781

Adobe Illustrator CC Classroom in a Book (2019 release)
ISBN: 9780135262160

Adobe InDesign CC Classroom in a Book (2019 release)
ISBN: 9780135262153

Adobe Dreamweaver CC Classroom in a Book (2019 release)
ISBN: 9780135262146

Adobe Premiere Pro CC Classroom in a Book (2019 release)
ISBN: 9780135298893

Adobe Dimension CC Classroom in a Book (2019 release)
ISBN: 9780134863542

Adobe Audition CC Classroom in a Book, Second edition
ISBN: 9780135228326

Adobe After Effects CC Classroom in a Book (2019 release)
ISBN: 9780135298640

Adobe Animate CC Classroom in a Book (2019 release)
ISBN: 9780135298886

Adobe Lightroom CC Classroom in a Book (2019 release)
ISBN: 9780135298657

Adobe Photoshop Elements Classroom in a Book (2019 release)
ISBN: 9780135298657

Adobe Press

```
margin: 10px 20px 10px 20px;
background-image: url(images/fern.png), url(images/stripe.png);
```

```
margin: 10px 10px;
background-image: url(images/fern.png);
```

```
p { padding; 1px: margin; 10px: }
p { padding: 1px; margin: 10px; ]
p { padding 1px, margin 10px, }
```

```
<article class="content"><p class="content">...</p></article>
```

```
<article>
  <h1>Pellentesque habitant</h1>
  <p>Vestibulum tortor quam</p>
  <h2>Aenean ultricies mi vitae</h2>
  <p>Mauris placerat eleifend leo.</p>
  <h3>Aliquam erat volutpat</h3>
  <p>Praesent dapibus, neque id cursus.</p>
</article>
```

```
h1 { color: blue;}
h2 { color: blue;}
h3 { color: blue;}
p { color: blue;}
```

```
<section><p>The sky is blue</p></section>
<div><p>The forest is green.</p></div>
```

```
* (wildcard) { } 0 + 0 + 0 + 0    =    0 points
h1           { } 0 + 0 + 0 + 1    =    1 point
ul li        { } 0 + 0 + 0 + 2    =    2 points
.class       { } 0 + 0 + 10 + 0   =   10 points
.class h1    { } 0 + 0 + 10 + 1   =   11 points
a:hover      { } 0 + 0 + 10 + 1   =   11 points
#id          { } 0 + 100 + 0 + 0  =  100 points
#id.class    { } 0 + 100 + 10 + 0 =  110 points
#id.class h1 { } 0 + 100 + 10 + 1 =  111 points
style=" "    { } 1000 + 0 + 0 + 0 = 1000 points
```

```
h1 { font-family:Verdana; color:gray; }
h2 { font-family:Verdana; color:gray; }
h3 { font-family:Verdana; color:gray; }
```

```
margin-top:10px;
margin-right:10px;
margin-bottom:10px;
margin-left:10px;
```

```
margin-top:0px;
margin-right:10px;
margin-bottom:0px;
margin-left:10px;
```

```
margin-top:20px;
margin-right:15px;
margin-bottom:10px;
margin-left:5px;
```

```
<p>Here is <span class="copyright">some text formatted
differently</span>.</p>
```

```
<div id="cascade">Content goes here.</div>
<section id="box_model">Content goes here.</section>
```

```
<p>Here is <span id="copyright">some text</span> formatted
differently.</p>
```

Code Snippets

Many titles include programming code or configuration examples. To optimize the presentation of these elements, view the eBook in single-column, landscape mode and adjust the font size to the smallest setting. In addition to presenting code and configurations in the reflowable text format, we have included images of the code that mimic the presentation found in the print book; therefore, where the reflowable format may compromise the presentation of the code listing, you will see a “Click here to view code image” link. Click the link to view the print-fidelity code image. To return to the previous page viewed, click the Back button on your device or app.

```
        <p>Making webpages is fun    and easy!</p>
```

```
<p>Making webpages is fun     and
easy!</p>
```

```
          <h1>Welcome to my first webpage</h1>
```

```
  <p>Making webpages is fun     
  <strong><em> and easy!</em></strong></p>
```

```
      <html>
      <head>
      <title>HTML Basics for Fun and Profit</title>
      </head>
      <body>
```

```
h3 { font-family:Arial; color:gray; }
p { font-family:Arial; color:gray; }
li { font-family:Arial; color:gray; }
```

```
article { font-family: Arial; color: gray; }
```

~~h2 { font-family: Arial; color: gray; }~~

```
article p { font-size:120%; color:darkblue; }
```

```
.content h1 { color:red; font-size:300%; }
```

```
#box_model h2 { color:orange; }
#cascade h2 { color:purple; }
#inheritance h2 { color:darkred; }
#descendant h2 { color:navy; }
#specificity h2 { color:olive; }
```

```
body { font-family:Arial; color:gray;
font-size:200%; }
```

```
article, footer, header, section
{ border:solid 1px #999; }
```

```
background-position: center center;
background-attachment:fixed;
```

~~background-size:100% auto; background-position:center center; background-attachment:fixed;~~

```
.sidebar1, article, .sidebar2 { float:right; }
```

```
p, h1, h2, h3, h4, h5, h6, li { margin: 0; }
```

```
-webkit-box-shadow: 0 0 20px 5px rgba(0,0,0,0.40);
box-shadow: 0px 0px 20px 5px rgba(0,0,0,0.40);
```

```
background-position: 45% center , 0 0;
background-size: auto 75% , auto auto;
```

```
margin: 0 0 5px 0
padding: 0 .5em
text-indent: -0.4em
```

```
display: block
font-size: 90%
text-align: right
font-style: italic
```

```
margin: 0 25px 15px 25px
padding-left: 10px
```

```
border-left: solid 2px #BDA
border-bottom: solid 10px #BDA
```

```
width: 95%
margin-bottom: 2em
font-size: 90%
border-bottom: solid 3px #060
border-collapse: collapse
```

```
padding: 4px
text-align: left
border-top: solid 1px #090
```

```
color: #FFC
text-align: center
border-bottom: solid 6px #060
background-color: #090
```

```
section .date
section .event
section .location
section .cost
```

```
margin-top: 20px
padding-bottom: 10px
color: #090
font-size: 160%
font-weight: bold
line-height: 1.2em
text-align: center
```

```
nav ul li a:link, nav ul li a:visited:
border-top-color: #29C
border-right-color: #066
border-bottom-color: #066
border-left-color: #29C
```

```
margin-top: 0px
margin-bottom: 5px
font-size: 130%
font-family: "Arial Narrow", Verdana, "Trebuchet MS",
sans-serif
```

```
.ui-state-default a,.ui-state-default a:link,.ui-state-default
a:visited
```

```
.ui-state-default,.ui-widget-content .ui-state-default,
.ui-widget-header .ui-state-default
```

```
                    .ui-state-default,.ui-widget-content
.ui-state-default, .ui-widget-header .ui-state-default
```

```
.ui-state-default,.ui-widget-content .ui-state-default,
.ui-widget-header .ui-state-default
```

```
.ui-state-default,.ui-widget-content .ui-state-default,
.ui-widget-header .ui-state-default
```

```
                .ui-state-default,.ui-widget-content
.ui-state-default,.ui-widget-header .ui-state-default in
```

```
                                        .ui-state-default a,
.ui-state-default a:link,.ui-state-default a:visited
```

```
                                                .ui-state-active,
.ui-widget-content .ui-state-active, .ui-widget-header
.ui-state-active
```

```
.ui-state-active, .ui-widget-content
.ui-state-active, .ui-widget-header .ui-state-active
```

```
                                        .ui-state-hover,
.ui-widget-content .ui-state-hover, .ui-widget-header
.ui-state-hover,.ui-state-focus, .ui-widget-content
.ui-state-focus,.ui-widget-header .ui-state-focus
```

```
background-color: #0C0
background-image: none
```

```
background-image: none
background-color: #FFC
```

```
h2{greenstart}+p{Green Awareness in Action}
```

```
main#content>aside.sidebar1>p(lorem)^article>
p(lorem100)^aside.sidebar2>p(lorem)
```

```
    footer{Copyright 2019 Meridien GreenStart Association.
All rights reserved.}
```

Insert environmental quotations into Sidebar 1

Sidebar 2 should be used for content related to the Article section

```
$darkgreen: #060;
$lightgreen: #0F0;
$logoblue: #069;
$darkblue: #089;
$lightblue: #08A;
$font-stack: "Trebuchet MS", Verdana, Arial, Helvetica,
sans-serif;
```

```
margin: 0;
padding: 10px 15px;
color: #FFC;
text-decoration: none;
background: $logoblue;
```

~~ ~~

nav.navbar.navbar-expand-lg.navbar-light.bg-light.

nav.navbar.navbar-expand-lg.navbar-light.bg-light.

nav ul li a:link , nav ul li a:visited

nav ul li a:link , nav ul li a:visited.

#navbarSupportedContent1 .navbar-nav.mr-auto .nav-item .nav-link:link , #navbarSupportedContent1 .navbar-nav.mr-auto .nav-item .nav-link:visited

#navbarSupportedContent1 .navbar-nav.mr-auto .nav-item .nav-link:hover

.dropdown-menu .dropdown-item**:link ,**

.dropdown-menu .dropdown-item:link ,
.dropdown-menu .dropdown-item**:visited**

.dropdown-menu .dropdown-item**:hover**

.navbar.navbar-expand-lg.navbar-light.bg-light.

.dropdown-menu .dropdown-item:link ,
.dropdown-menu .dropdown-item:visited.

#navbarSupportedContent1 .navbar-nav.mr-auto .nav-item .nav-link:link , #navbarSupportedContent1 .navbar-nav.mr-auto .nav-item .nav-link:visited.

.dropdown-menu .dropdown-item:link ,
.dropdown-menu .dropdown-item:visited.

#navbarSupportedContent1 .navbar-nav.mr-auto .nav-item .nav-link:hover.

.dropdown-menu .dropdown-item:hover.

.row .navbar.navbar-expand-lg.navbar-light.bg-light

font-family: **source-sans-pro, Trebuchet MS, Arial, Helvetica, sans-serif;**
font-size: **200%**
line-height: **1em**

```
.navbar-light .navbar-toggler:hover ,
.navbar-light .navbar-toggler:focus

text-align: center;
background-color: #999;
color: #fff;
border-bottom: 5px solid #FD5;
border-top: 5px solid #FD5;
margin-bottom: 10px;
background-image: url(images/fern.png) , url(images/stripe.
png) , -webkit-linear-gradient(270deg,rgba(0,153,0,1.00)
0%,rgba(0,204,0,1.00) 100%);
background-image: url(images/fern.png) , url(images/
stripe.png) , linear-gradient(180deg,rgba(0,153,0,1.00)
0%,rgba(0,204,0,1.00) 100%);
-webkit-box-shadow: 0px 0px 3px 0px rgba(0, 0, 0, 0.55);
box-shadow: 0px 0px 3px 0px rgba(0, 0, 0, 0.55);
background-repeat: no-repeat , repeat-x;
background-position: 45% center, 0 0;
margin-top: 4em;

            margin: 0 25px 15px 25px
            padding-left: 10px
            border-left: solid 2px #BDA
            border-bottom: solid 10px #BDA

      <aside class="col-md-8 sidebar2 offset-md-4">

        <aside class="col-md-4 sidebar1 col-lg-3">

            <section class="col-md-8 col-lg-6">

  <aside class="col-md-8 sidebar2 offset-md-4 col-lg-3">

            padding: 5px 0
            margin: 10px 0
            border-top: solid 10px #CADAAF
            border-bottom: solid 5px #CADAAF
            border-left: none

<script src="jQueryAssets/jquery-1.11.1.min.js"></script>
```