

14

WORKING WITH A WEB FRAMEWORK

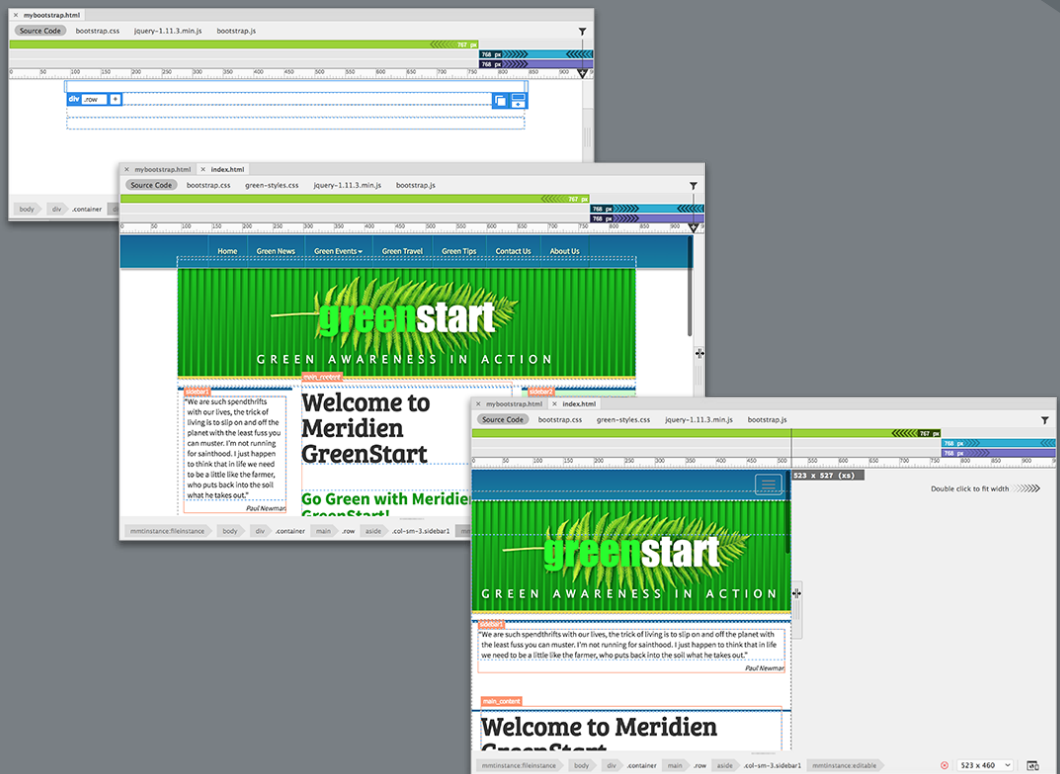
Lesson overview

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.

- 1 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.

- 2 Make sure the document window is displayed at a width of 1100 pixels or greater.

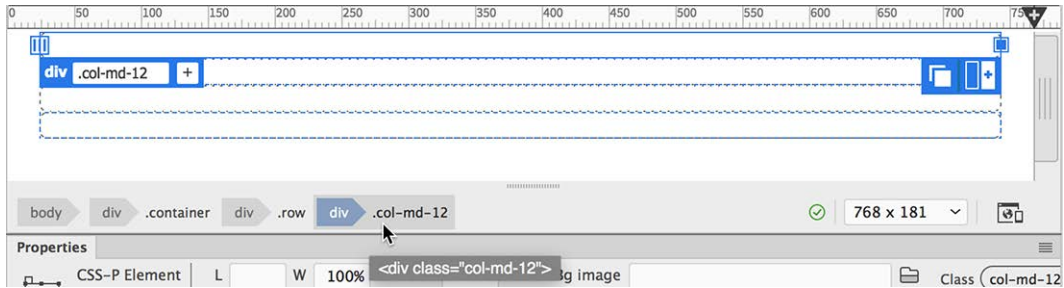
Because the layout and all the components are designed to be responsive, the document window needs to be at least 1100 pixels to display all the elements properly.

► **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.

- 3 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.

- 4 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.

- 5 Press Delete.

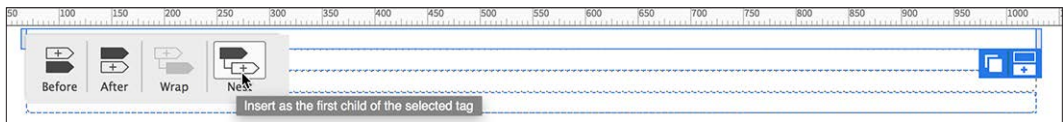
The `<div.col-md-12>` is removed. When an element is deleted, Dreamweaver leaves the cursor at the position of that element. So this is the perfect time to insert the navbar.

- 6 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.

- 7 In the navbar item dropdown menu, click **Basic Navbar**. The position-assist dialog appears.



- 8 Click Nest.



A predefined Bootstrap navbar appears in the first row.

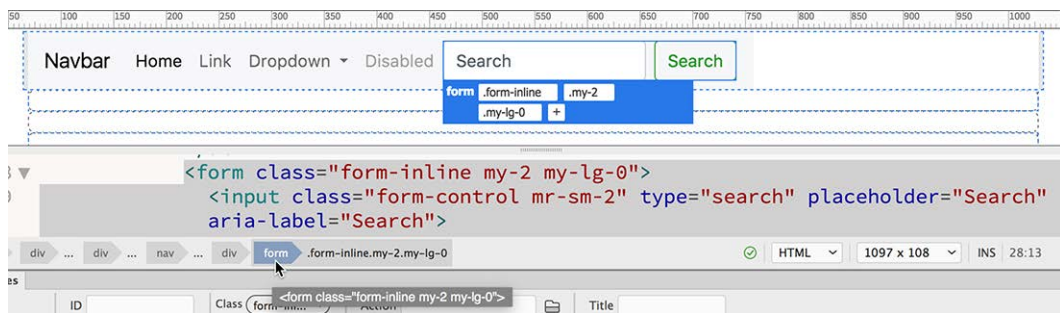
► **Tip:** The DOM panel can also be used for deleting unneeded components.

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.

- 9 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.

- 10 Select `form.form-inline.my-2.my-lg-0` in the tag selector interface.



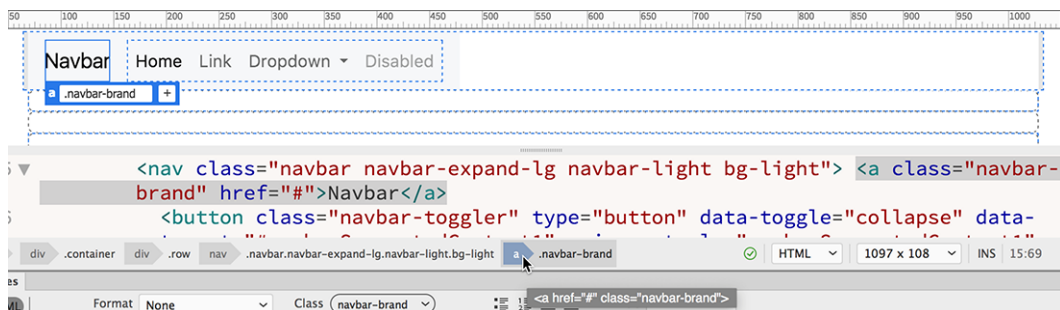
- 11 Press Delete.

The search field and button are removed. The menu now contains four navigation links and one other item.

- 12 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.

- 13 Select the `a.navbar-brand` tag selector.



14 Press Delete.

The Navbar link is removed. All the unneeded components have now been removed.

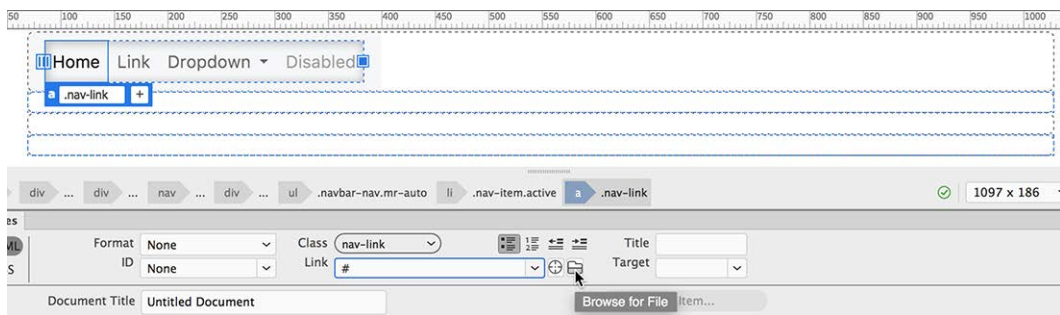
You may have noticed that the navbar features a dropdown menu. Although the current design doesn't have one, we will keep it because it provides an excellent way to target the two tables on the *Green Events* page.

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.

Note: When in editing mode, the text display may be quite distorted. Make sure the `<a>` element is still selected.

15 Click the Home link. Click the a tag selector.

16 In the Property inspector, click the Browse For File icon .



17 Select **index.html** in the site root folder. Click Open.

The first menu item is complete.

18 Double-click the next “Link” placeholder.

The orange text editing box appears.

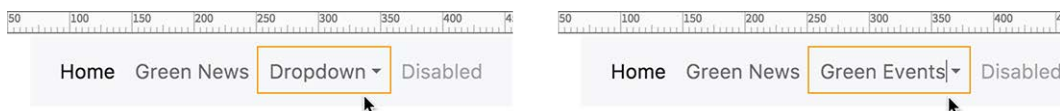
19 Type **Green News** to replace the Link text.



20 Link the item to **news.html**.

21 Double-click the text “Dropdown.”

22 Type **Green Events** to replace the text “Dropdown.”



23 Save the file.

The first two predefined links now match the menu items from the current Green-Start design, but you still need to edit the dropdown menu and create four additional items.

Editing a dropdown menu

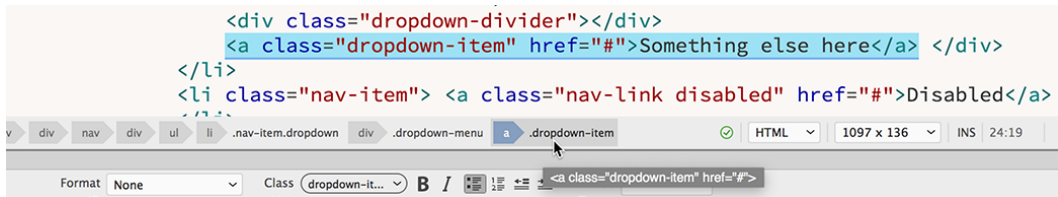
Editing dynamic components, like the dropdown menu, can present some challenges in Live view and Design view. Since the menu is designed to react to the cursor, it may be difficult to select or edit text in the various elements. With elements like this, Code view is often the preferred workspace.

1 Switch to Split view, if necessary.

2 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.

3 Select the `<a>` element containing the text *Something else here* (approximately line 32) in the dropdown menu, and press the Delete key.



```
<div class="dropdown-divider"></div>
<a class="dropdown-item" href="#">Something else here</a> </div>
</li>
<li class="nav-item"> <a class="nav-link disabled" href="#">Disabled</a>
...
<div class="dropdown-menu">
  <a class="dropdown-item" href="#">Something else here</a>
  <a class="dropdown-item" href="#">Another action</a>
  <a class="dropdown-item" href="#">Something else here</a>
  <a class="dropdown-item" href="#">Another action</a>
</div>
```

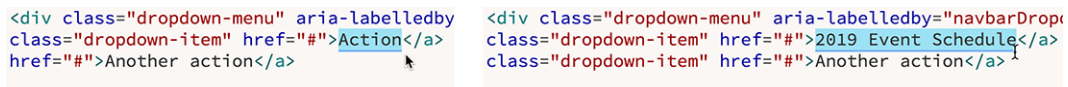
4 Select and delete the element

```
<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.

5 Select the text *Action* in the first sublink.

Type **2019 Event Schedule** to replace it.



```
<div class="dropdown-menu" aria-labelledby="navbarDropdown" style="display: inline-block; vertical-align: middle; margin-left: 10px;">
  <a class="dropdown-item" href="#">Action</a>
  <a class="dropdown-item" href="#">Another action</a>
</div>

<div class="dropdown-menu" aria-labelledby="navbarDropdown" style="display: inline-block; vertical-align: middle; margin-left: 10px;">
  <a class="dropdown-item" href="#">2019 Event Schedule</a>
  <a class="dropdown-item" href="#">Another action</a>
</div>
```

Code view provides an easy way to create links to local site files.

6 Select the hash (#) symbol in the href attribute for the new link.

7 Type **events**.

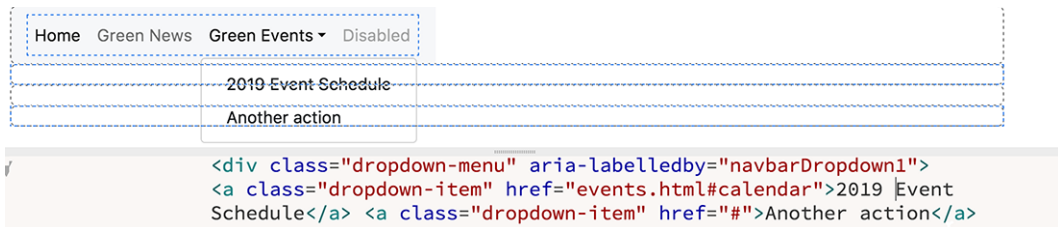
```
<div class="dropdown-menu" aria-labelledby="navbarDrop  
class="dropdown-item" href="events#">2019 Event Schedu  
class="dropdown-item" href="#">Another action</a>
```

As you type, Dreamweaver provides hinting for the local file structure. It will list any filename that matches the text you enter.

8 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.

9 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.



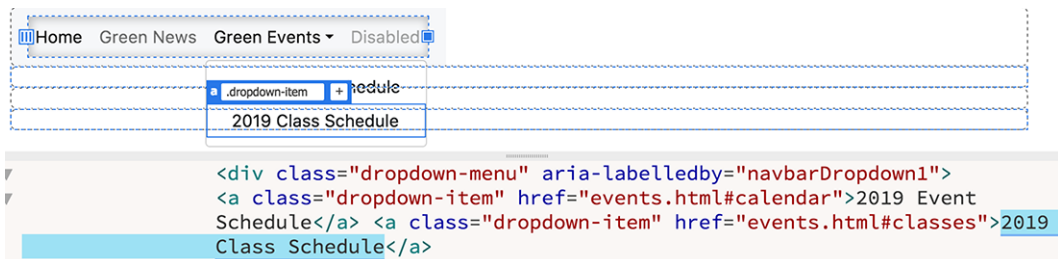
The sublink in the dropdown menu now targets the event schedule directly. You will use the remaining link to target the second table.

10 In Code view, select the text *Another Action*.

Type **2019 Class Schedule** to replace it.

11 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`.

12 Edit the text *Disabled* to say **Green Travel**.

13 Point the link to **travel.html**.

To remove the gray formatting, you need to remove the class `.disabled`.

14 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.

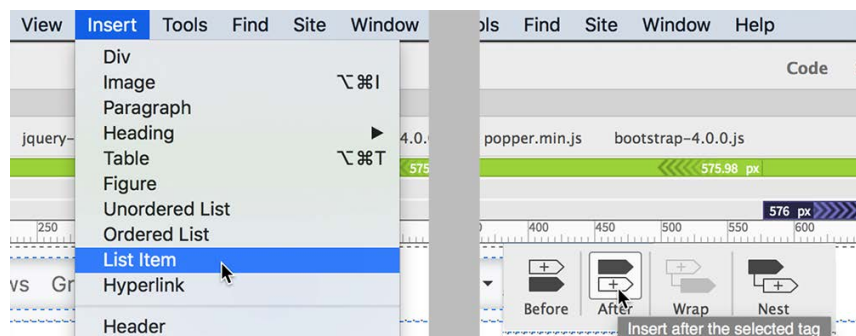
1 In Live view, click the *Green Travel* link.

Select the `li` tag selector.

2 Choose Insert > **List Item**.

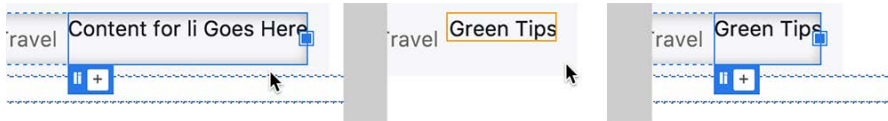
The position-assist dialog appears.

3 Click **After**.

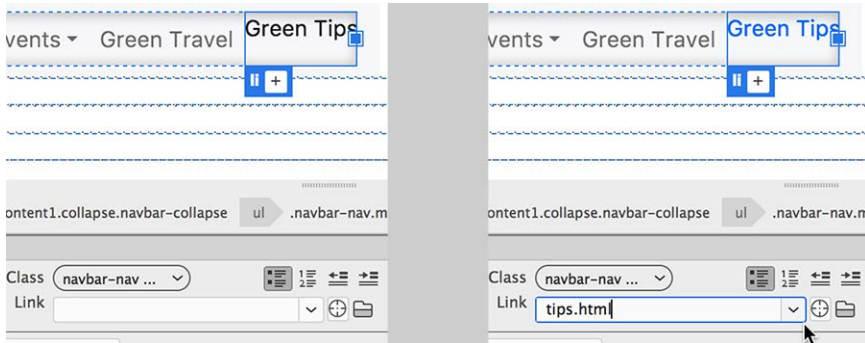


A new list item appears with placeholder text.

- 4 Edit the placeholder text and type **Green Tips** to replace it.

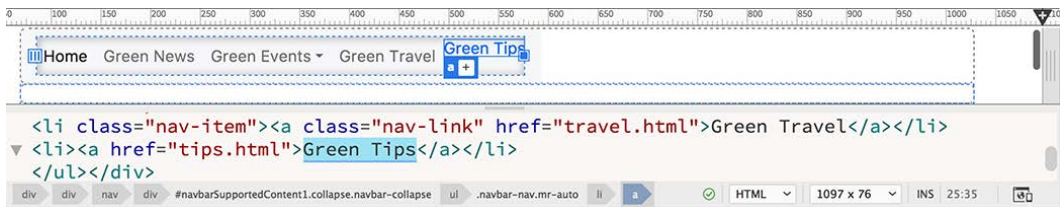


- 5 Link the item to **tips.html**.




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.

- 6 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 `` and `<a>` elements. Let's add the same classes to the *Green Tips* menu item.

- 7 If necessary, select the menu item *Green Tips*.
Select the a tag selector.
The Element Display appears around the `<a>` element.
- 8 Click the Add Class/ID icon .
- 9 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.

10 Select the `li` tag selector for *Green Tips*.

The Element Display appears around the `` element.

11 Click the Add Class/ID icon .

12 Type `.nav-item` and press Enter/Return to apply the class.



The new link is now structured identically to the others.

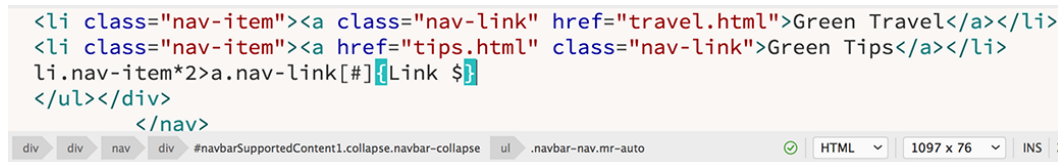
You added *Green Tips* using a menu option. New menu items can also be added by typing the code manually. You could type them one at a time, but don't forget you can use Emmet to supercharge any code-writing chores.

Using Emmet to create new menu items

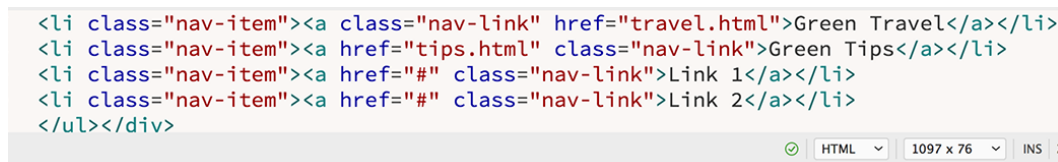
In this exercise, you will use Emmet to create the last two menu items.

1 In Code view, insert the cursor after the closing `` tag for the link *Green Tips*.
Press Enter/Return to create a new line.

2 Type `li.nav-item*2>a.nav-link[#]{Link $}`



3 Press Tab.



Two new list items with hyperlink placeholders are created automatically, along with the custom classes.

- 4 Edit Link 1 to say **Contact Us** and link it to **contact-us.html**.
- 5 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>
```

- 6 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.

- 1 If necessary, open **mybootstrap.html**.
Switch to Split view.
- 2 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`.
- 3 Select and delete the attribute `active` from the *Home* item markup in Code view.

| | |
|--|---|
| <pre><ul class="navbar-nav mr-auto"> <li class="nav-item active"> <a c only">(current) <li class="nav-item"> <a class="n <li class="nav-item dropdown"><a</pre> | <pre><ul class="navbar-nav mr-auto"> <li class="nav-item"> <a class="n (current) <li class="nav-item"> <a class="n <li class="nav-item dropdown"><a</pre> |
|--|---|

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.

- 4 Select and delete the element `(current)` from the *Home* `` 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>

<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.

- 5 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.

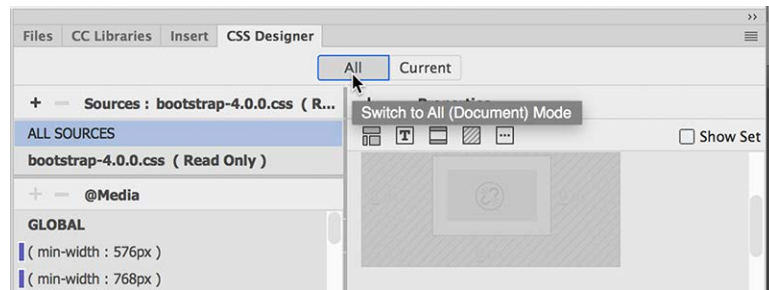
► **Tip:** Not sure how wide your window is? The ruler should be visible at the top of the document window whenever Live view is selected.

► **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.

- 1 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.
- 2 Choose Window > CSS Designer to display it, if necessary.
- 3 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**.

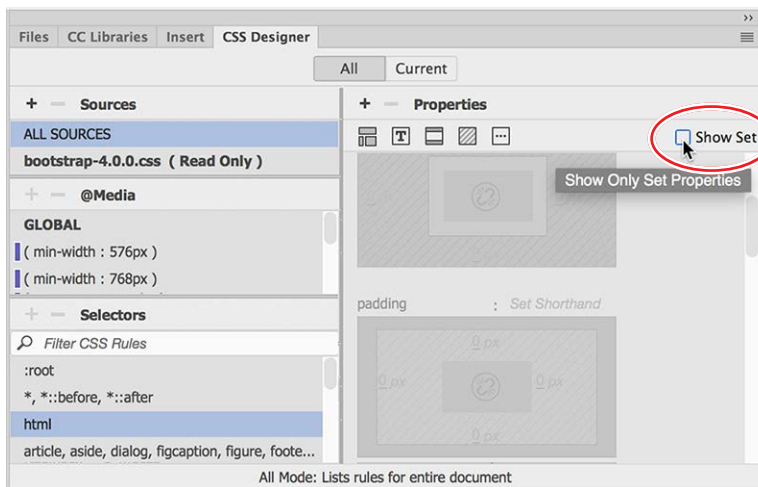
- 4 Select ALL SOURCES in the Sources panel.



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.

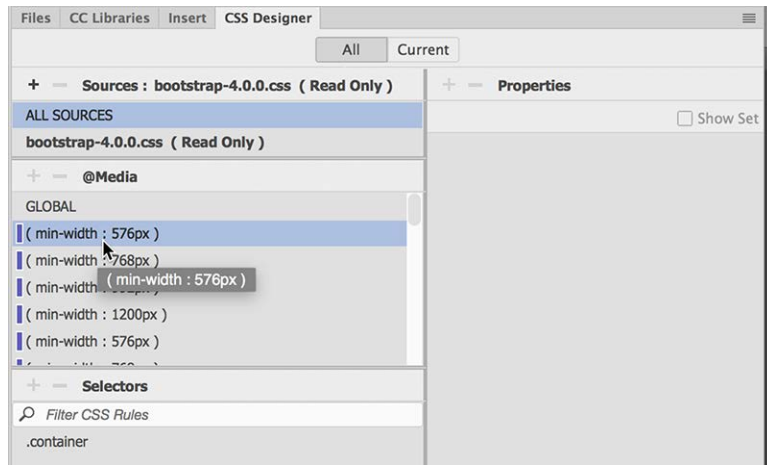
With the exception of the jQuery accordion used in Lesson 10, “Adding Interactivity,” you have had to contend with only global CSS rules: rules that apply universally to the entire page and all screen types. But once you enter the world of Bootstrap and responsive web design, one of the hardest tasks 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.

- 5 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.

- 6 Select the first (min-width : 576px) media query in the @Media window. Observe the changes in the CSS Designer display.



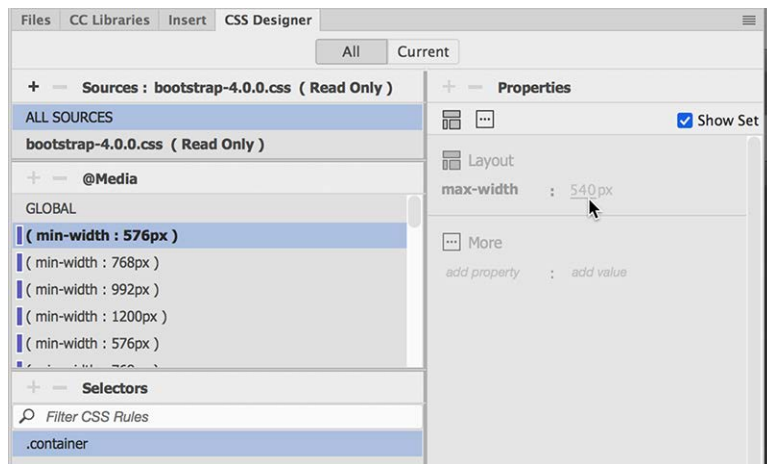
Note that **bootstrap.css** is still selected, but the Selectors panel now shows only one name: `.container`. This indicates that only one rule is defined within the selected media query.

- 7 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.

- 8 If necessary, select the Show Set option in the Properties panel.

► **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.



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.

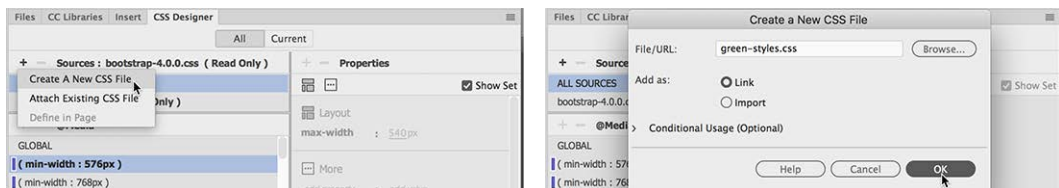
- 9 Click the Add CSS Source icon  in the CSS Designer.

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.

- 10 Choose **Create A New CSS File** from the dropdown menu.

The Create A New CSS File dialog appears.

- 11 Type **green-styles.css** in the Create A New CSS File dialog. Click OK to create the style sheet reference.

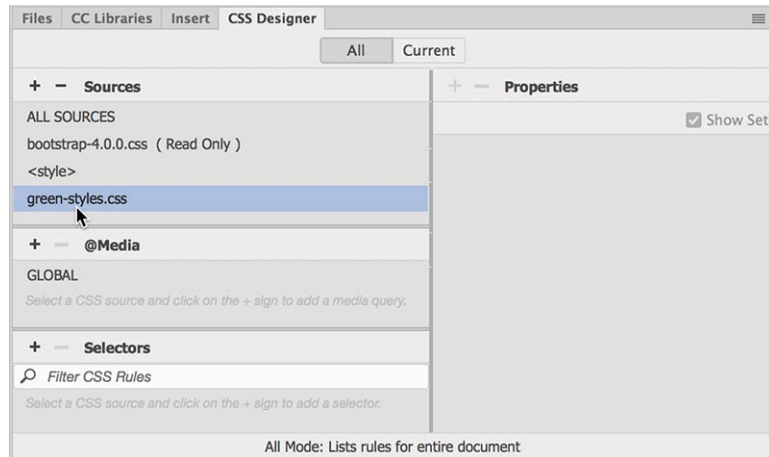


► **Tip:** It is essential to note here that the CSS file has not been created yet, and will not be until you create a CSS rule and save the file. If Dreamweaver crashes before that happens, you will have to create the file in a separate operation.

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.

- 12 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.

- 13 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.

- 1 If necessary, open **mybootstrap.html** in Live view.
- 2 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

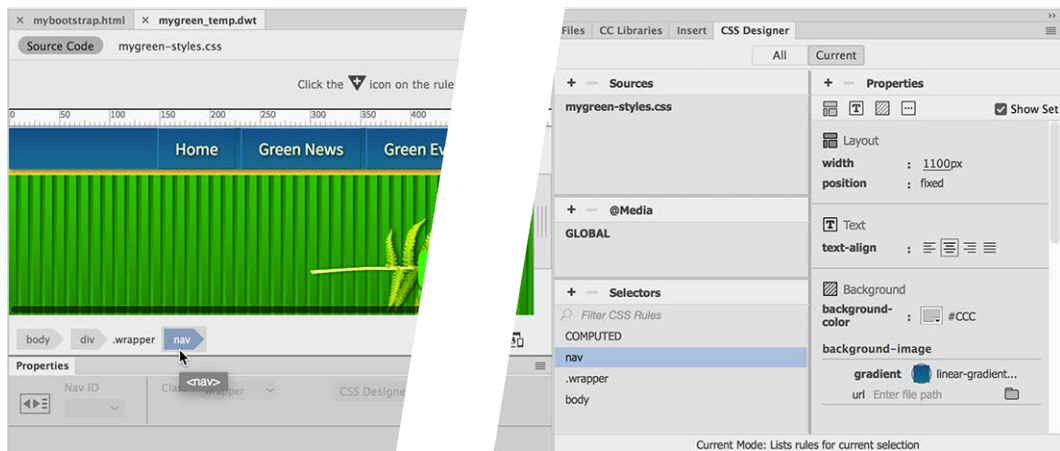
Let's work down from the top of the page. The horizontal navigation menu appears at the top of the template. We won't need the old menu, but we can definitely grab the styling. The menu is composed of several layered elements, each of which contributes to the overall appearance. The first step is to pull the background styles.

- 1 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.

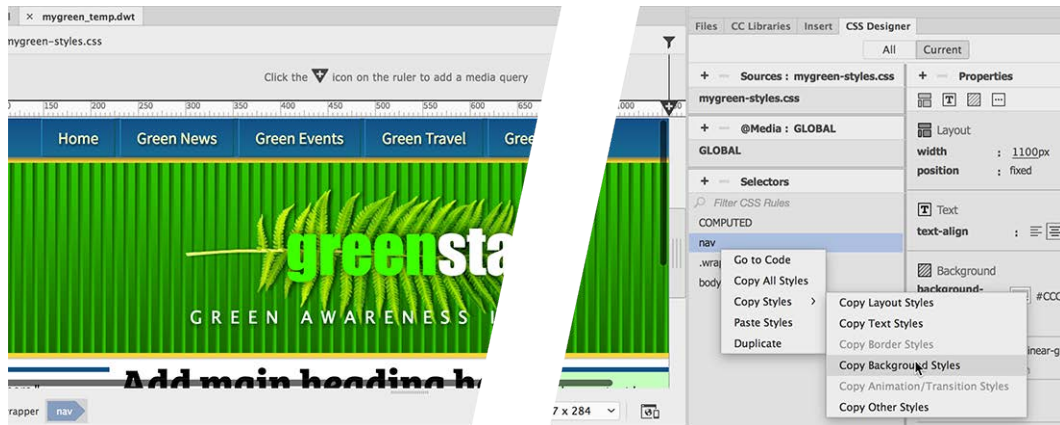
- 2 Select the nav tag selector.

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.

- 3 Right-click the nav selector in the CSS Designer.
Select Copy Styles > Copy Background Styles from the context menu.



- 4 Switch to mybootstrap.html.

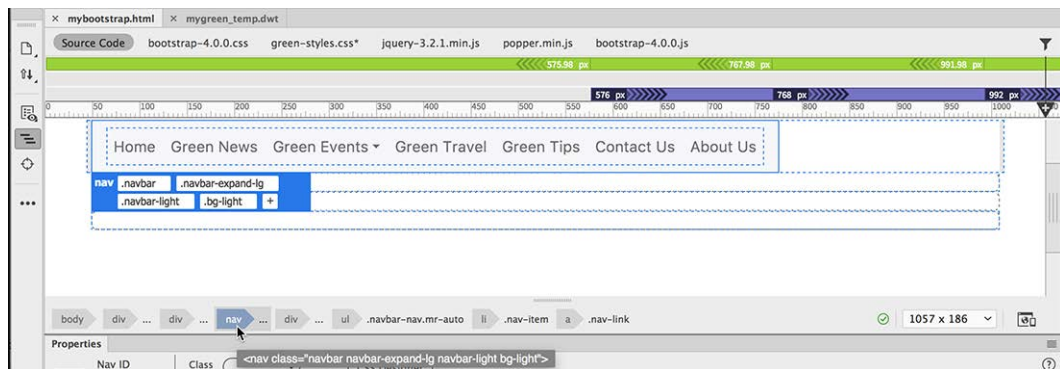
Applying the background styles in the new layout is a two-step process. First, you have to identify any rules currently formatting the same element in the new file. Then, you have to create a new rule in the custom style sheet to apply the specifications.

- 5 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.

- 6 Select the nav tag selector.



The Element Display appears focused on `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.

7 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.

8 Select **green-styles.css** in the Sources panel.

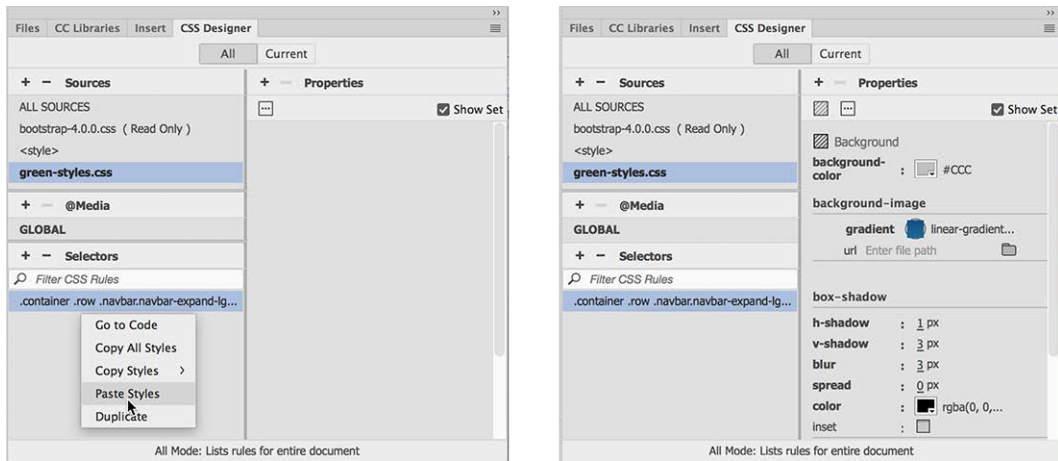
Click the Add Selector icon .

A new selector appears.

9 Press the up or down arrow to create the following selector:

`nav.navbar.navbar-expand-lg.navbar-light.bg-light`.

- 10 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

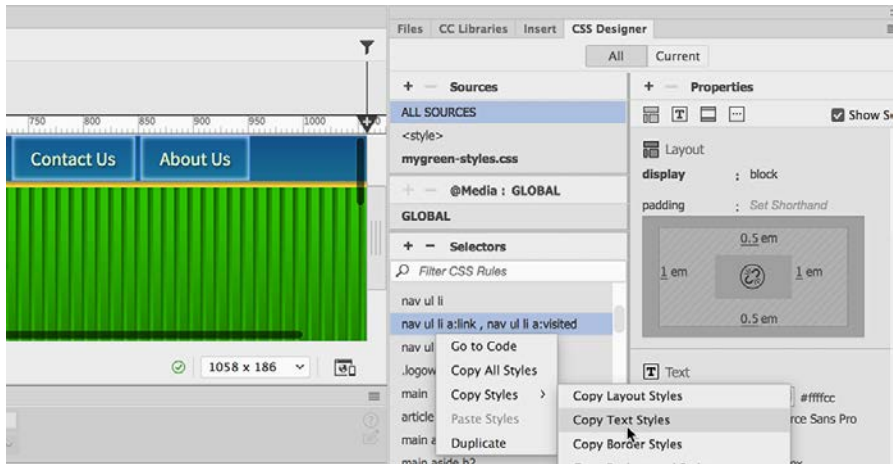
In this exercise, you will bring over the text styling of the navigation menu.

- 1 Switch to **mygreen_temp.dwt**.
- 2 Click the menu item *Green News*.
Click the Current button in the CSS Designer.

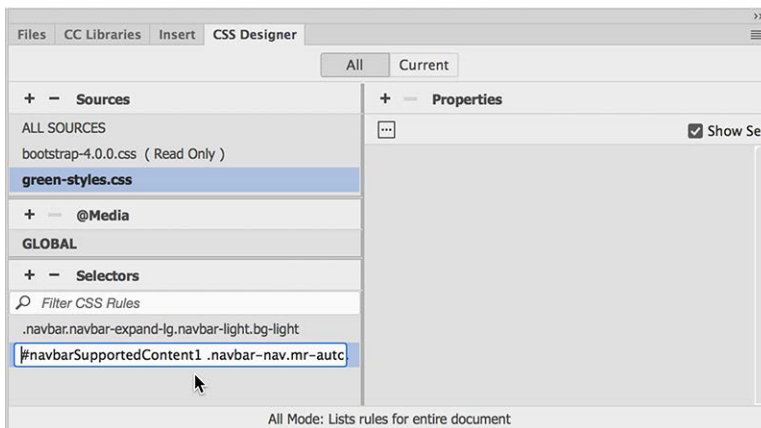
The text styling for the horizontal menu is applied via rules targeting the `a:link`, `a:visited`, and `a:hover` pseudoclasses. You will need to bring over the styling for all these states to complete the basic menu.

- 3 Right-click the following rule:
`nav ul li a:link , nav ul li a:visited`

- 4 Select Copy Styles > Copy Text Styles from the context menu.

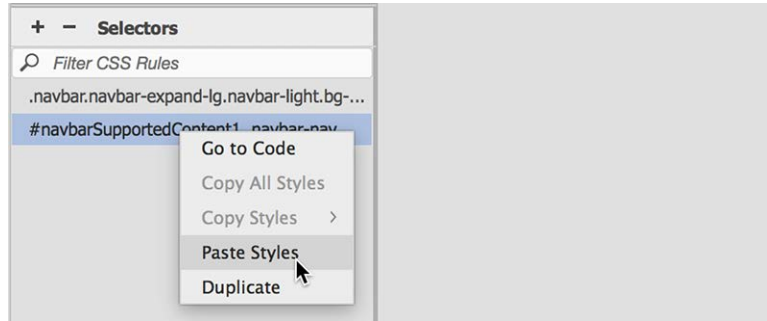


- 5 Switch to **mybootstrap.html**.
- 6 Select Green News in the horizontal menu.
Click the a tag selector, if necessary.
- 7 Click the All button in the CSS Designer.
Select **green-styles.css** in the Sources panel.
- 8 Click the Add Selector icon **+**.
An automatic selector name appears in the Selector pane.
- 9 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.

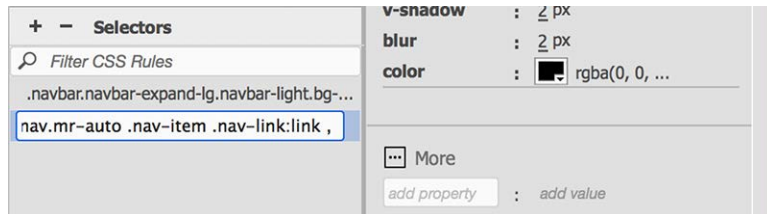
- 10 Press Enter/Return to complete the selector.
- 11 Right-click the new rule and select Paste Styles from the context menu.



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.

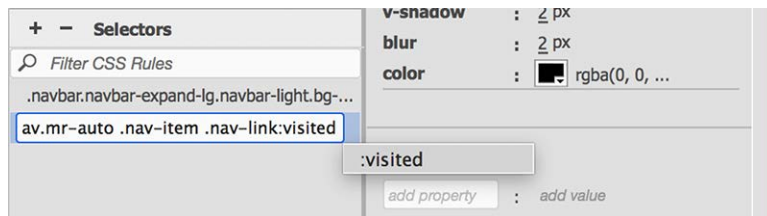
- 12 Double-click the new rule to edit it.
- 13 Select the entire selector name and copy it.
- 14 Insert the cursor at the end of the selector.
Type `:link`, and press Ctrl+V/Cmd+V.

● **Note:** Don't forget the comma (,). The rule will not work without it.



A copy of the selector name appears after the comma.

- 15 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.

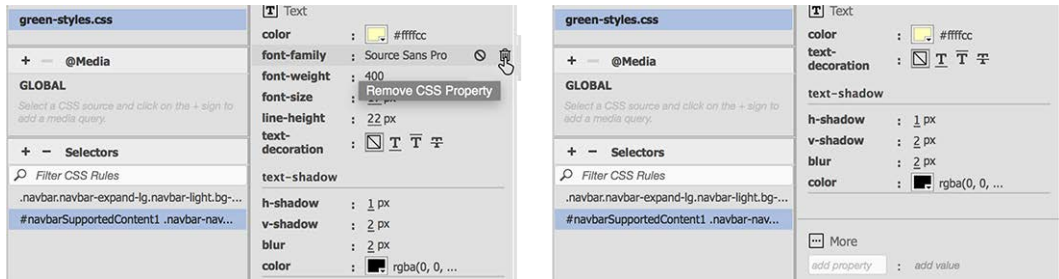
16 Delete the following properties from the rule:

~~font-family: Source Sans Pro~~

~~font-weight: 400~~

~~font-size: 17px~~

~~line-height: 22px~~

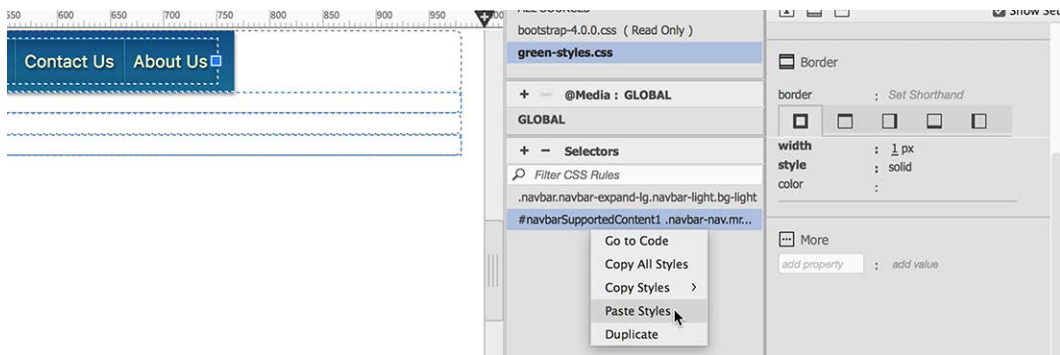


Later in this lesson, you will learn how to properly build site-wide font styling. But first, let's continue working on the menu items by grabbing the border styling.

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.

- 1 Switch to **mygreen_temp.dwt**.
The borders are defined in the same rule that styles the text.
- 2 In the CSS Designer, right-click the following rule:
`nav ul li a:link , nav ul li a:visited.`
- 3 Select Copy Styles > Copy Border Styles from the context menu.
- 4 Switch to **mybootstrap.html**.
- 5 Right-click the rule you created in the previous exercise.
- 6 Select Paste Styles from the context menu.



7 Save all files.

Styling interactive Bootstrap menu effects

1 In CSS Designer, select the following rule:

2 Click the Add Selector icon .

3 Create the following selector:

4 Create the following property in the new rule:

5 Position the cursor over any menu item.

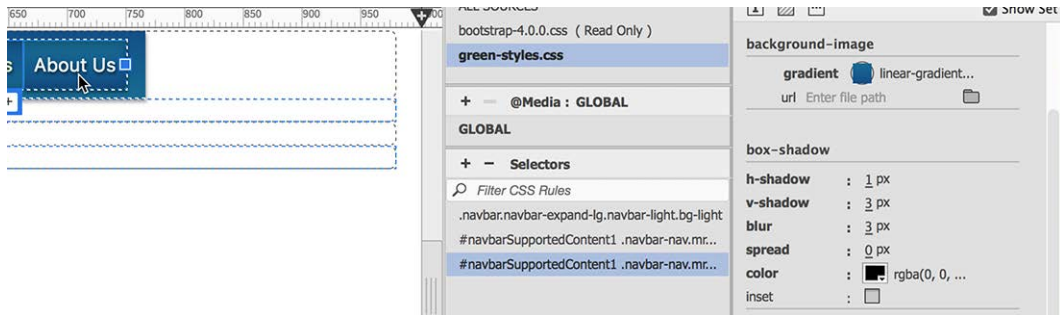
6 Switch to **mygreen_temp.dwt**.

7 Copy the background styles from the following rule:

8 Switch to **mybootstrap.html**.

9 Paste the styles on the `:hover` rule created in step 3.

10 Position the cursor over any menu item.

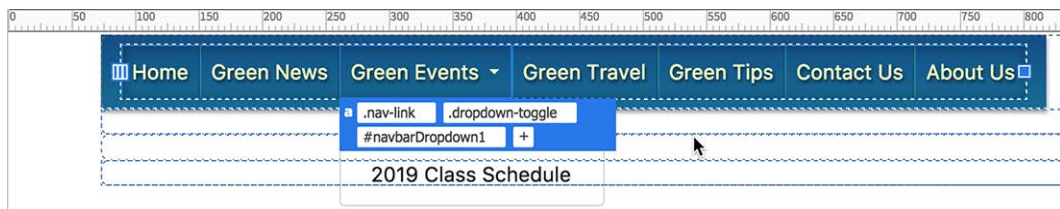


The background of the menu item reverses as the text turns white. The styling is complete in the regular part of the menu, but all is not well. There is a problem lurking in the dropdown component.

11 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.

12 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.

13 Save all files.

In the next exercise, you will identify the pertinent styles for the dropdown menu and override them.

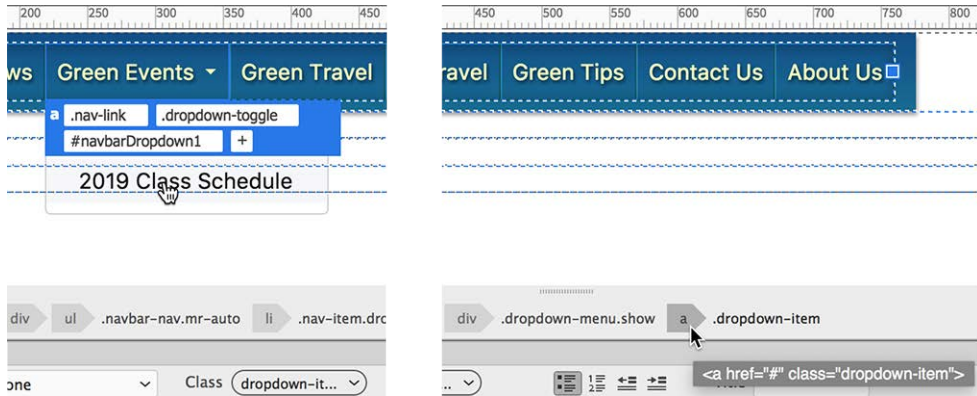
Styling a Bootstrap dropdown menu

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.

1 Open **mybootstrap.html** in Live view, if necessary.

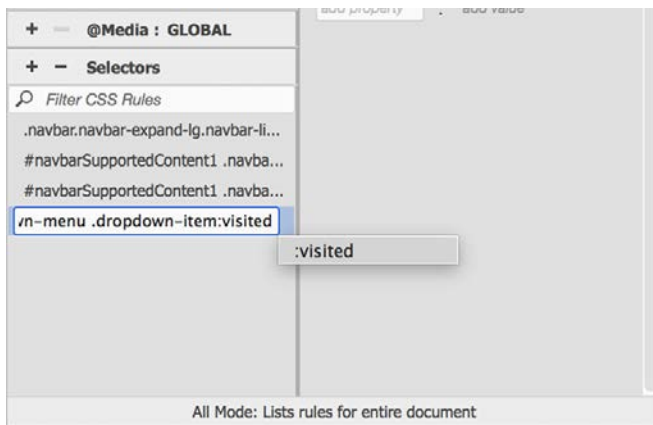
2 Click the Current button in the CSS Designer.

- 3 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.

- 4 In the CSS Designer, select the All button.
Select **green-styles.css**.
Select the last rule displayed.
- 5 Click the Add Selector icon **+**.
Press the up arrow to create the following selector:
`.dropdown-menu .dropdown-item`
- 6 Press Ctrl+A/Cmd+A to select the entire selector.
- 7 Press Ctrl+C/Cmd+C to copy the selector.
- 8 Modify the selector as highlighted:
`.dropdown-menu .dropdown-item:link ,`
This selector will style the default link state.
- 9 Press Ctrl+V/Cmd+V to paste the selector again.
- 10 Modify the selector as highlighted:
`.dropdown-menu .dropdown-item:link ,`
`.dropdown-menu .dropdown-item:visited`



The changes will now style the default and visited states of the link.

- 11 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.

- 12 Click the Add Selector icon .

Press Ctrl+A/Cmd+A, and then paste to insert the new selector name.

- 13 Modify the selector as highlighted:

`.dropdown-menu .dropdown-item: hover`

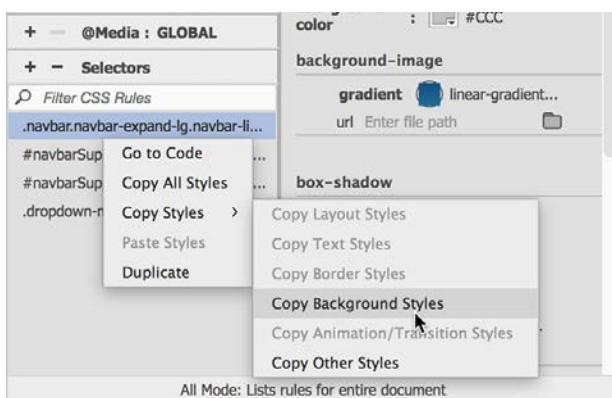
Once the selectors are created, you can copy and paste the appropriate styles in CSS Designer.

- 14 Right-click on the rule

`.navbar.navbar-expand-lg.navbar-light.bg-light`.

This rule applies the background styles for the main menu.

- 15 Select Copy Styles > Copy Background Styles from the context menu.

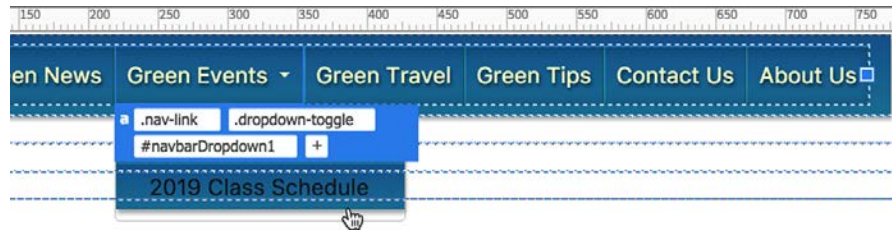


- 16** Right click the rule
- ```
.dropdown-menu .dropdown-item:link ,
.dropdown-menu .dropdown-item:visited.
```

- 17** Select Paste Styles from the context menu.

The dropdown menu items should be formatted now.

- 18** Click the Green Events dropdown menu.

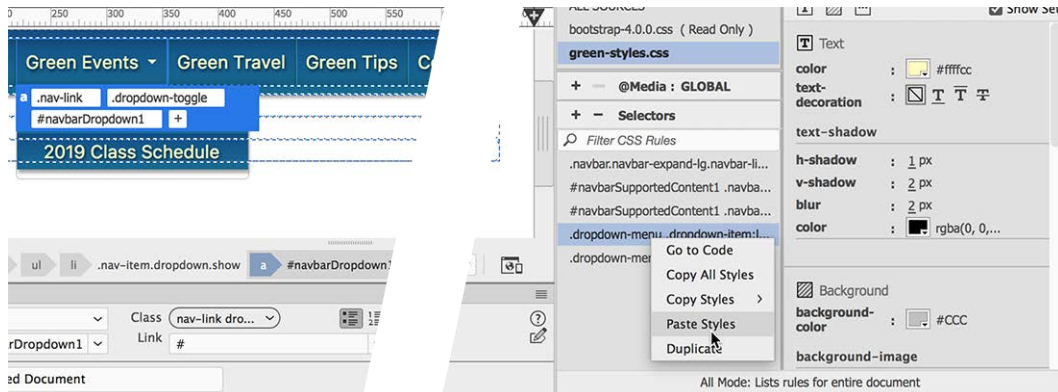


The background of the menu is now formatted, but the text is not. That styling came from a different rule.

- 19** Right-click the rule
- ```
#navbarSupportedContent1 .navbar-nav.mr-auto .nav-item  
.nav-link:link , #navbarSupportedContent1 .navbar-nav.mr-auto  
.nav-item .nav-link:visited.
```

- 20** Copy all styles.

- 21** Paste the styles on
- ```
.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.

**22** Copy all styles from the rule

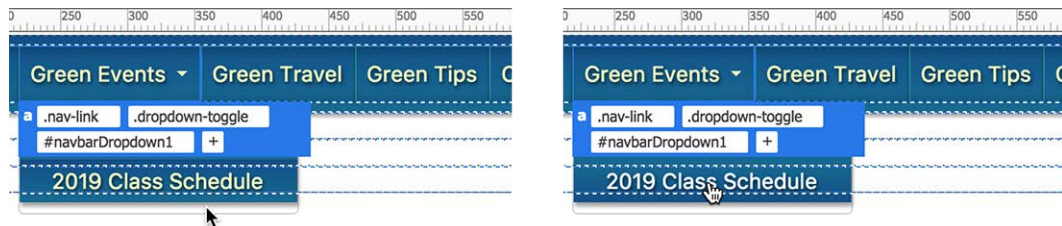
```
#navbarSupportedContent1 .navbar-nav.mr-auto .nav-item
.nav-link:hover.
```

**23** Paste the styles on the rule

```
.dropdown-menu .dropdown-item:hover.
```

The styling of the dropdown menu should be complete.

**24** 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.

**1** In the CSS Designer, select **green-styles.css**.

Create the following rule:

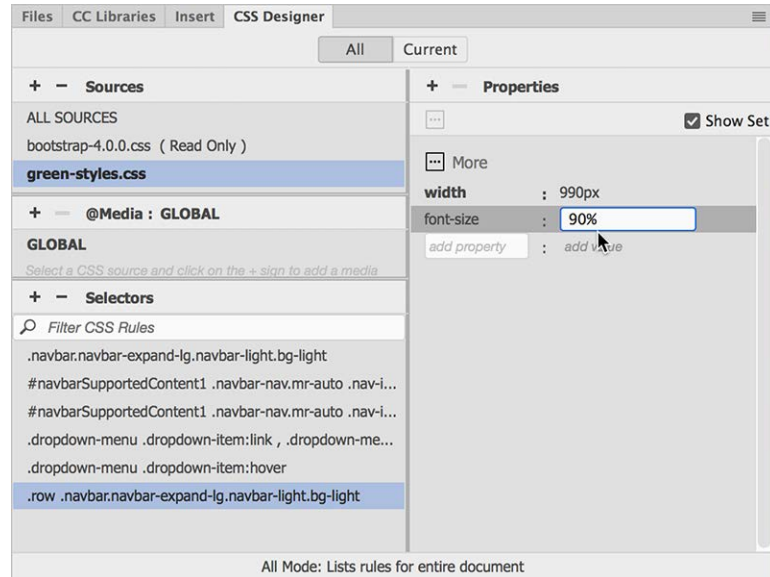
```
.row .navbar.navbar-expand-lg.navbar-light.bg-light
```

● **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.

- 2 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.

- 3 Click any menu item.

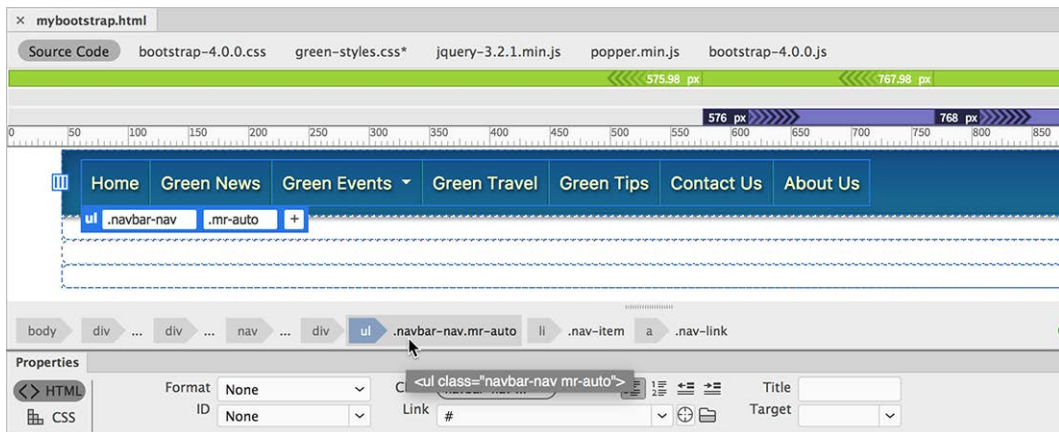
Examine the menu structure in the tag selector interface.

The menu is composed of a `<ul>` element with `<li>` and `<a>` elements as children.

- 4 Select the `ul.navbar-nav.mr-auto` tag selector.

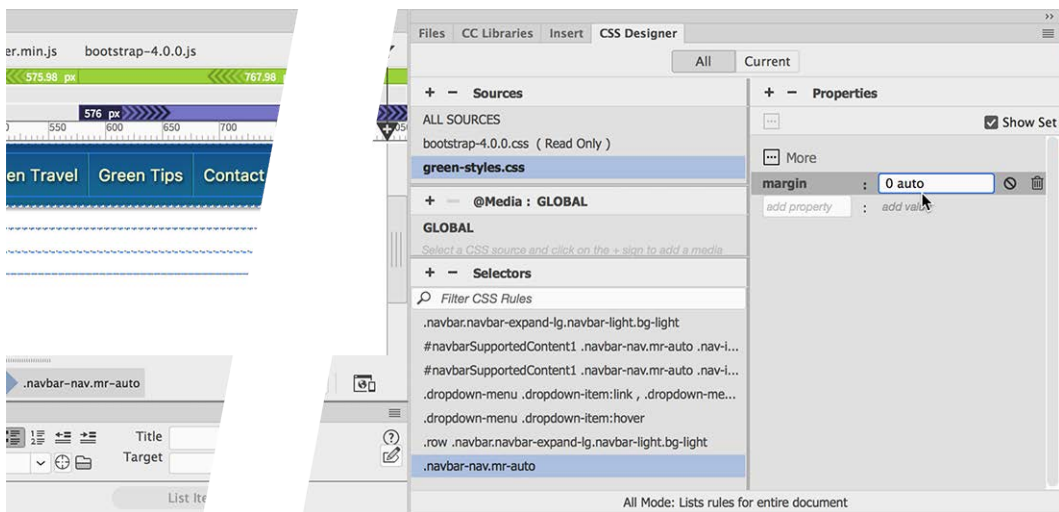
5 Create the following selector:

**.navbar-nav.mr-auto**



6 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.

7 Choose File > Save All.

The next task is to fill in the content area with the template placeholders.

# Transferring content to a responsive template

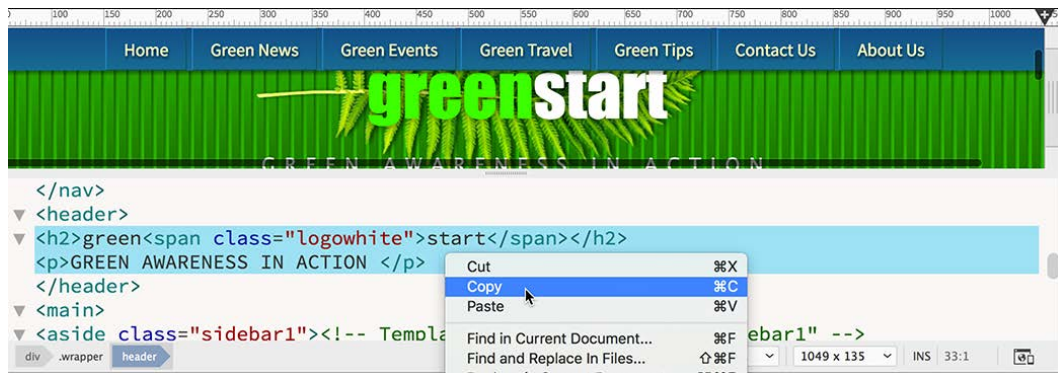
Once the responsive menu is complete and styled properly, you will be able to start populating the rest of the layout. In most cases, you will be able to use the content and styling from the old template without major modifications.

● **Note:** In the following exercise, you will be copying and pasting HTML code and CSS styling using Split view and the Related File interface.

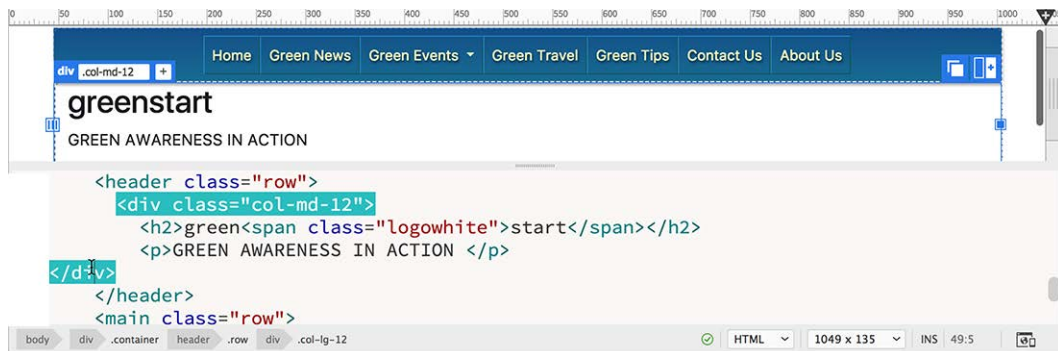
## Creating a responsive header

Working from the top down, the header element is the next task on your list.

- 1 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.
- 2 In **mygreen\_temp.dwt**, select and copy the `<h2>` and `<p>` elements in the `<header>` element in the Code view window.

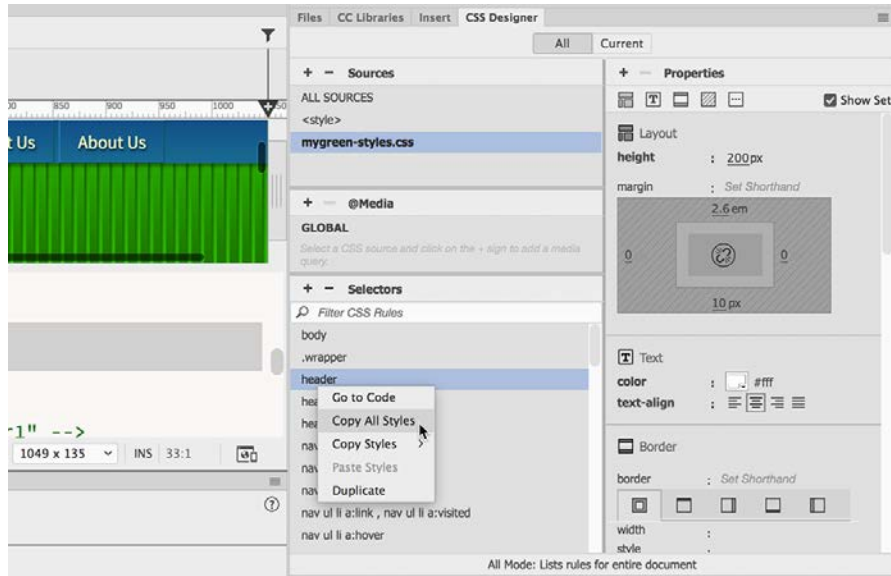


- 3 Switch to **mybootstrap.html**. In the Code view window, scroll down and insert the cursor in `<div class="col-md-12">` in the `<header>` element.
- 4 Paste the `<h2>` and `<p>` elements.

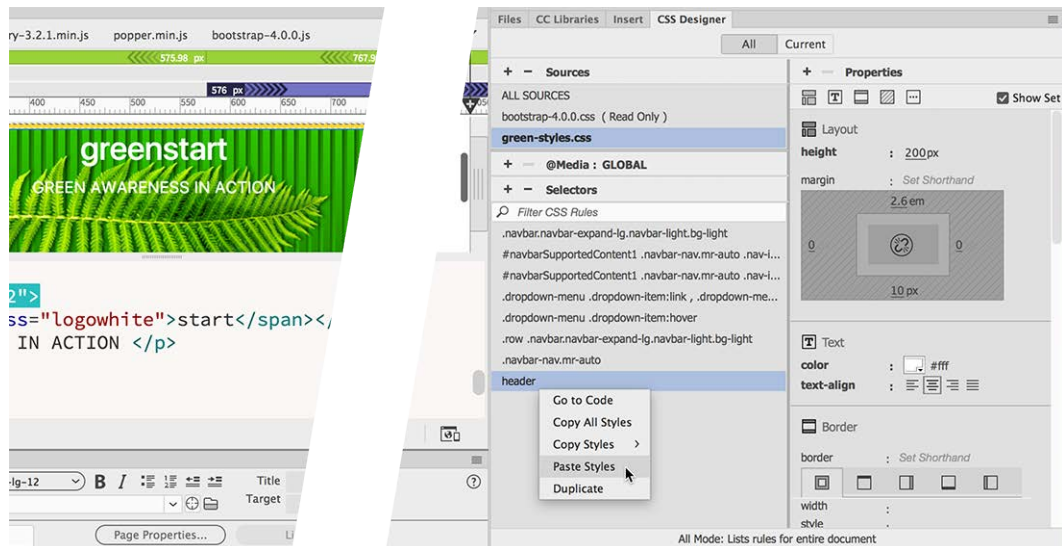


The GreenStart name and motto appear within the second row of the Bootstrap layout. At the moment, the text and header are unstyled.

- 5 Switch to **mygreen\_temp.dwt**.  
In the CSS Designer, right-click the header rule.
- 6 Select Copy All Styles from the context menu.



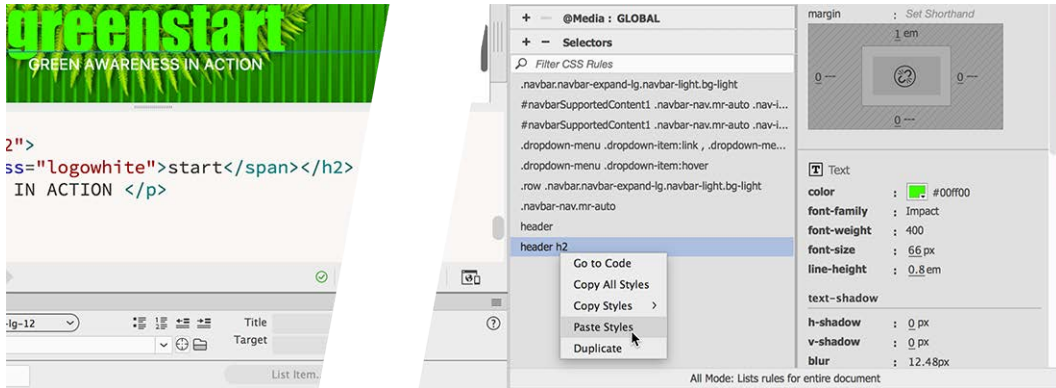
- 7 In **mybootstrap.html**, create the rule **header** in **green-styles.css**.
- 8 Right-click the new rule and select Paste Styles from the context menu.



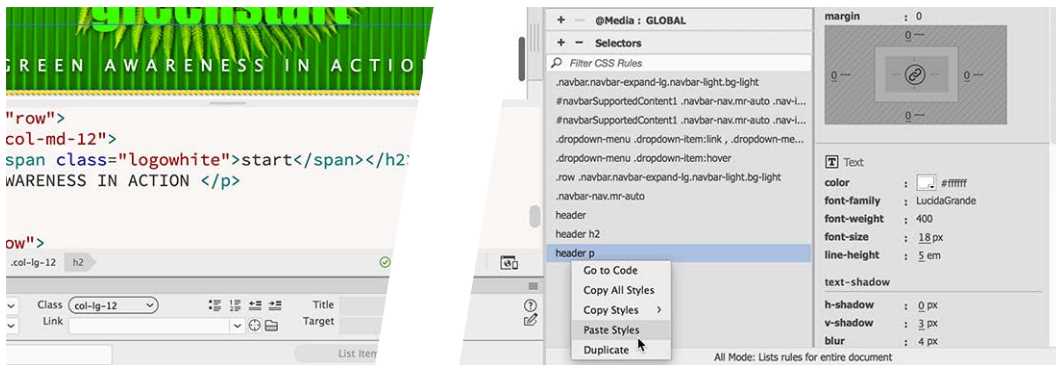
The background of the header displays the vertical stripes and fern images.  
Space was also added above the header.

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.

- 9 In **mygreen\_temp.dwt**, copy all styles from the rule header h2.
- 10 In **mybootstrap.html**, create the rule **header h2** in **green-styles.css** and paste the styles.

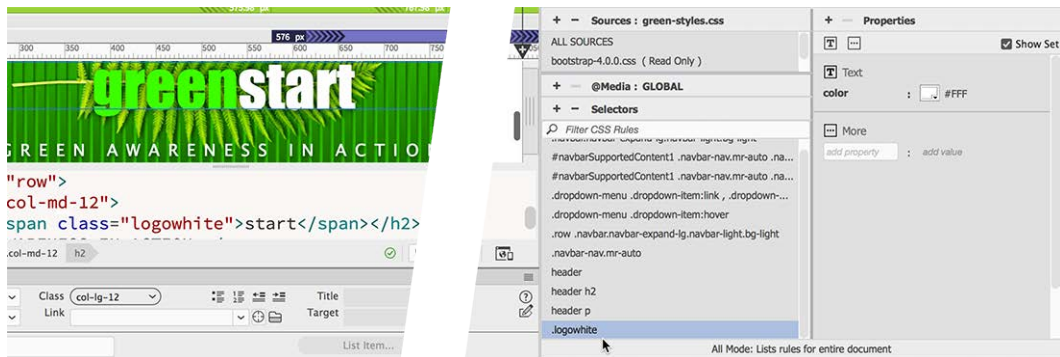


- 11 In **mygreen\_temp.dwt**, copy all styles from the rule header p.
- 12 In **mybootstrap.html**, create the rule **header p** in **green-styles.css** and paste the styles.



- 13 In **mygreen\_temp.dwt**, copy all styles from the rule .logowhite.

- 14** In **mybootstrap.html**, create the rule **.logowhite** in **green-styles.css** and paste the styles.



- 15** Save all files.

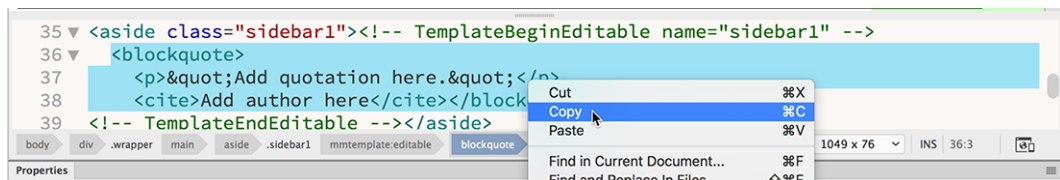
The styling for the responsive header is now complete and matches the GreenStart template.

## Creating responsive placeholders

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.

- 1** 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.
- 2** In **mygreen\_temp.dwt**, select and copy the blockquote element in sidebar 1.

**Note:** You can perform most tasks in any document view, but remember to copy and paste in the same view.



The blockquote element contains the quotation and citation placeholders.

- 3 Paste the blockquote placeholder in `aside.col-md-4` in **mybootstrap.html**.

```
52 <aside class="col-md-4">
53 <blockquote><p>"Add quotation here."</p>
54 <cite>Add author here</cite></blockquote></aside>
55 <section class="col-md-4"></section>
56 <aside class="col-md-4"></aside>
57 </main>
```

- 4 Create the following rules in **green-styles.css**:

```
.sidebar1 blockquote
.sidebar1 blockquote p
.sidebar1 blockquote cite
```

- 5 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?

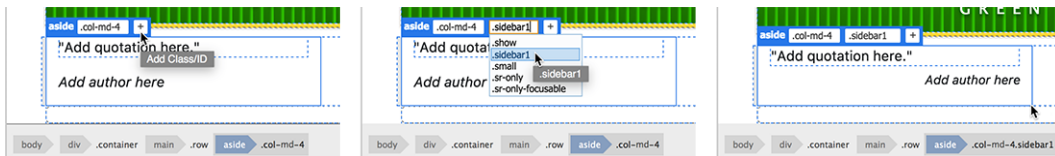
All the rules are based on the class `sidebar1`. That class does not exist in the Bootstrap layout. You could delete the class from each selector name, but then the rules would apply to all blockquotes. Instead, you will simply add the class to the appropriate element.

- 6 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.

- 7 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.

- 8 Switch to **mybootstrap.html**, if necessary.

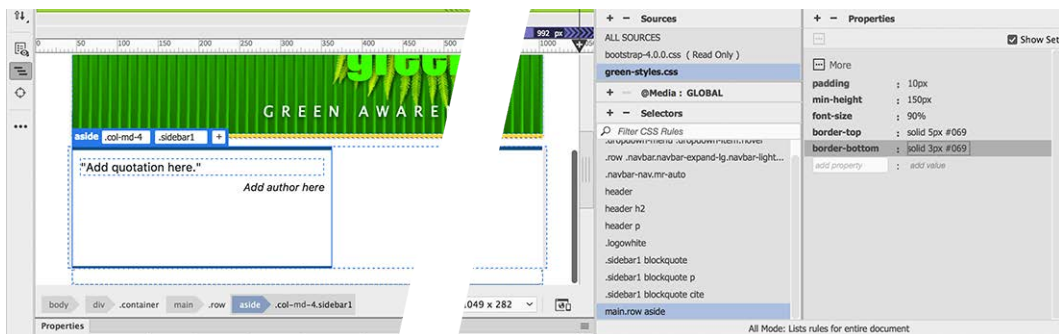
Create the following rule in **green-styles.css**:

```
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.

- 9 Create the following properties:

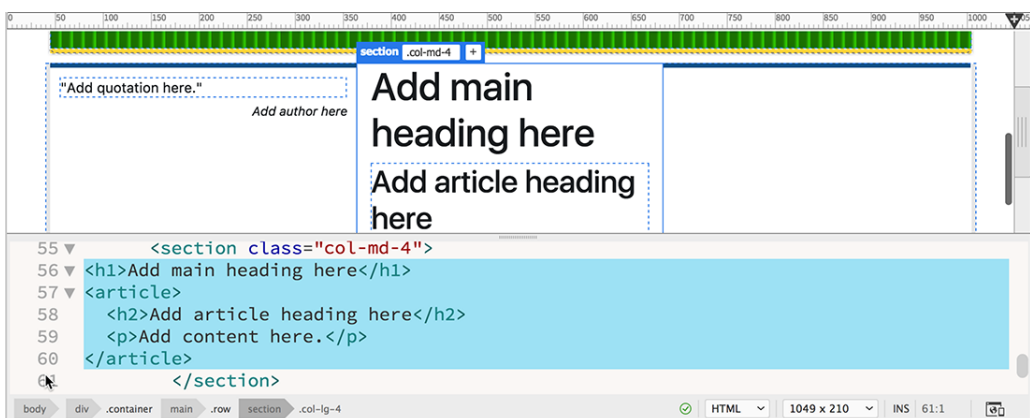
**padding: 10px**  
**min-height: 150px**  
**font-size: 90%**  
**border-top: solid 5px #069**  
**border-bottom: solid 3px #069**



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.

- 10 In **mygreen\_temp.dwt**, select and copy the `<h1>` and `<article>` elements.
- 11 In **mybootstrap.html**, paste the elements in `section.col-md-4`.



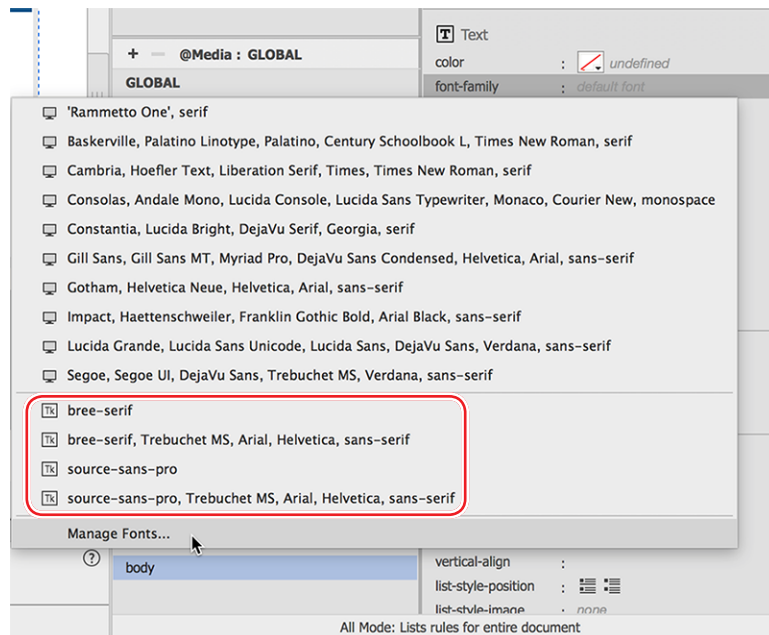
Once you have the placeholders in position, you will create the rules needed to style them. But in this case you will not bring over the old specifications. That's because the GreenStart design uses Adobe web-hosted fonts, and simply copying and pasting the specifications will not support the use of 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.

- 1 Switch to **mybootstrap.html** in Live view, if necessary.  
Typically, the base font for a site is defined in the body rule.
- 2 In the CSS Designer, select **green-styles.css**.  
Create a new selector named **body**
- 3 In the Properties panel, deselect the Show Set option, if necessary.  
The panel now displays all CSS specifications.
- 4 Click the Text category icon **T**.  
The panel display focuses on CSS Properties for text.
- 5 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.

- 6 If Source Sans Pro appears in the custom font stack, skip to step 14. Otherwise, click Manage Fonts.

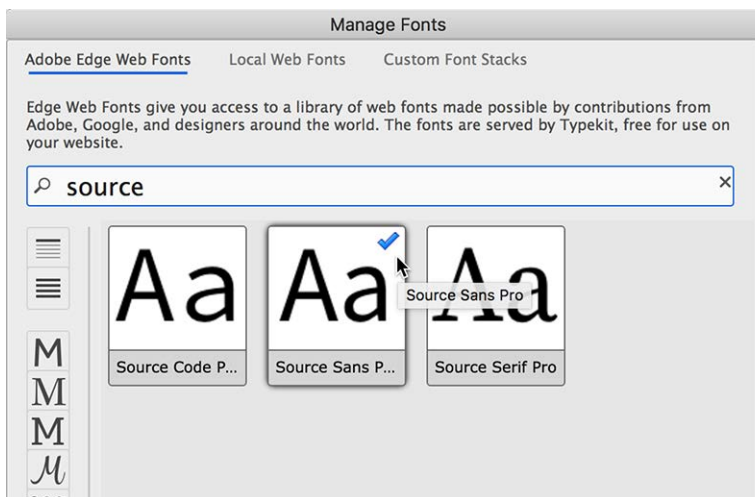
The Manage Fonts dialog appears. The panel has three tabs, for Adobe Edge Web Fonts, Local Web Fonts, and Custom Font Stacks. The dialog defaults to Adobe Edge Web Fonts. In the GreenStart design, the base font was Source Sans Pro.

- 7 In the search field, type **source**.

The window filters the list of fonts showing items that contain the word “source.”

- 8 If necessary, select Source Sans Pro from the items displayed.

**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.

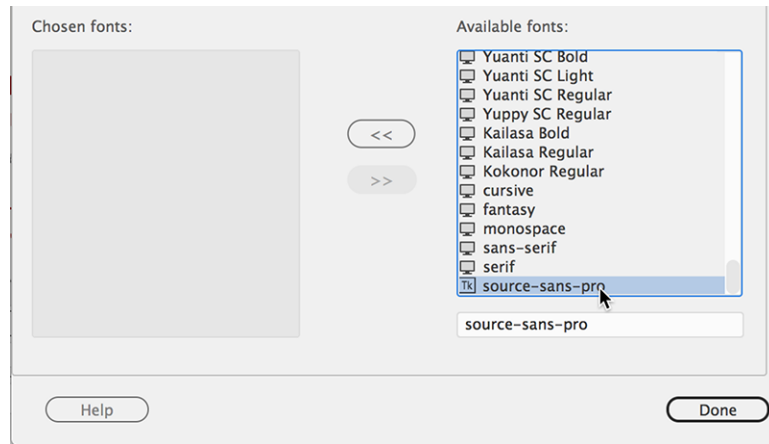


A blue checkmark appears on the selected font. You could use this font immediately, but it's a good idea to build a complete custom stack. That way, if for some reason Source Sans Pro doesn't load, you can specify what font or fonts will be displayed instead.

- 9 Click the tab Custom Font Stacks.

Source Sans Pro is now displayed at the bottom of the Available Fonts window in the dialog.

- 10 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.

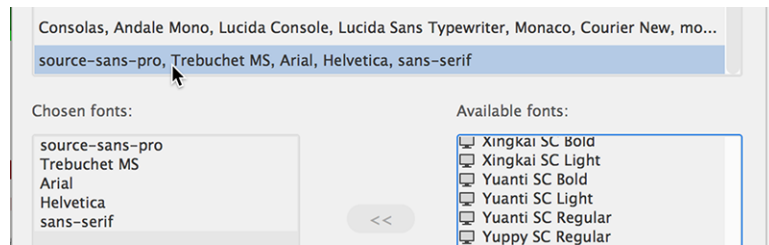
- 11 Add the following fonts:

**Trebuchet MS**

**Arial**

**Helvetica**

**sans-serif**



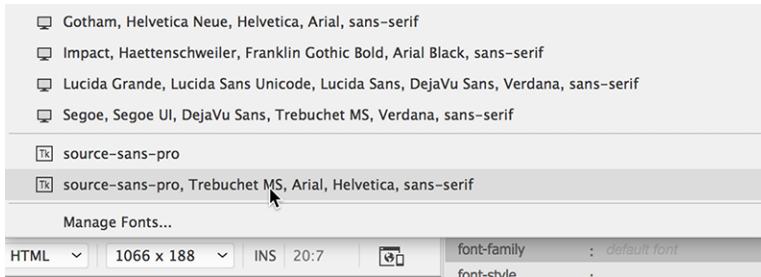
A new custom font stack with your select fonts appears in the dialog.

- 12 Click Done to close the panel.

**13** 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.

**14** 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.

**15** 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.

**16** Switch to Live view.

**17** In CSS Designer, select **400** in the font-weight property.

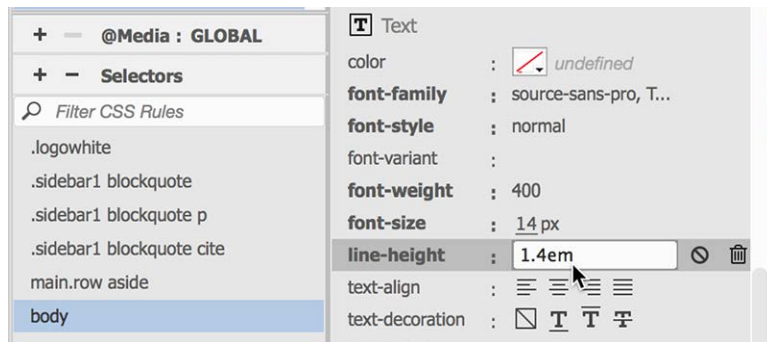
**18** Set the font-size property to **14px**.

## 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.

**19** 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.

**20** Create the following rule in **green-styles.css**:

**main.row section h1**

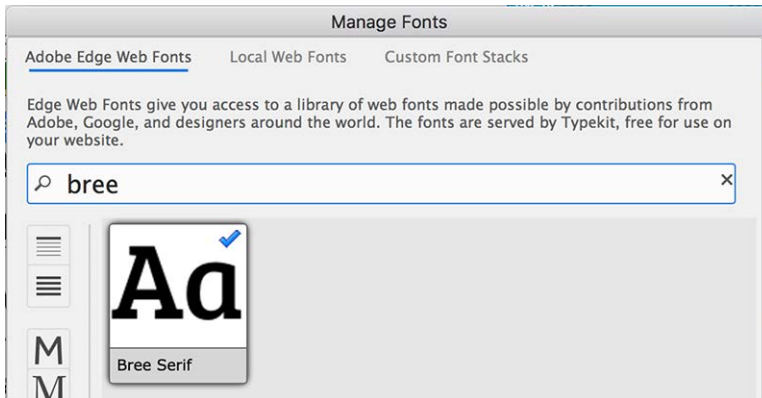
Since this heading uses web-hosted fonts too, you will define this style manually.

**21** Click to open the font-family property.

If Bree Serif appears in the custom font stacks, skip to step 24; otherwise, click Manage Fonts.

**22** In the search field type **Bree**.

**23** Select Bree Serif in the dialog.



A blue checkmark appears on the selected font.

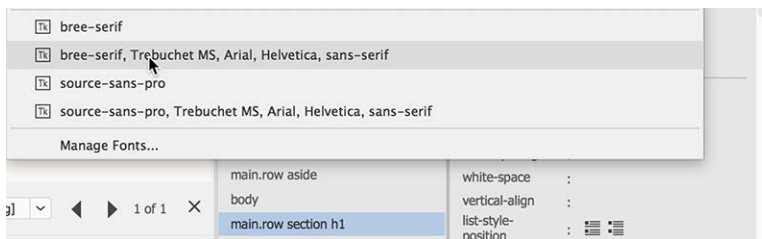
**24** Click the tab Custom Font Stacks.

**25** Create a custom font stack with the following fonts:

**bree-serif**  
**Trebuchet MS**  
**Arial**  
**Helvetica**  
**sans-serif**

**26** Click Done.

**27** In the rule **main.row section h1**, select the new custom font stack.

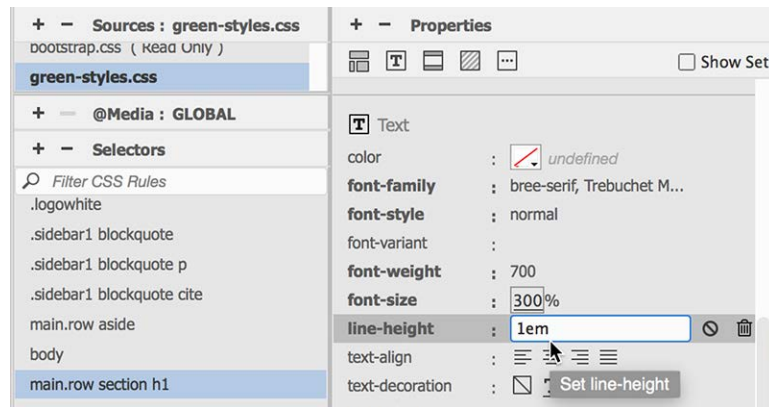


**Note:** When web-hosted fonts are selected, you may notice some properties are applied automatically.

A reference to Bree Serif should now be added to the script loading the web-hosted fonts in your `<head>` section.

- 28** 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
```



- 29** Save all files.

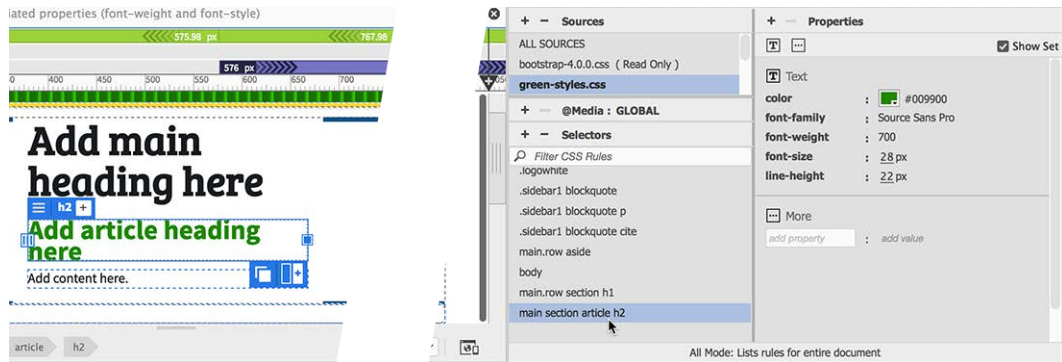
The web-hosted fonts are now defined properly. Keep an eye out for any references to these two fonts in any rules you copy over from the old style sheet. The old settings may use fixed sizes or a different spelling of the font names. If they do, be sure to replace them with the proper spelling and size 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.

- 1** 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.
- 2** 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.
- 3** Switch to `mybootstrap.html`. Create the following rule in `green-styles.css`:  
**`main.row section article h2`**

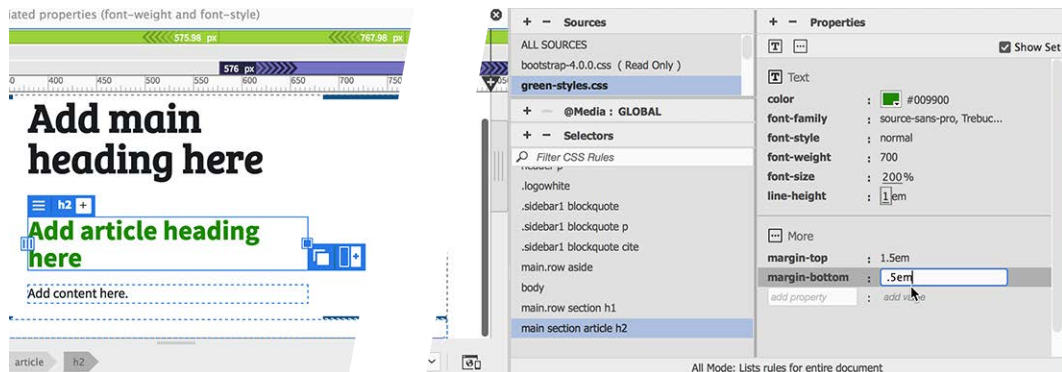
- 4 Paste the styles on the new rule.



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.

**Note:** Fonts on your screen may appear different from the screenshots.


- 5 Select the Show Set option.
- 6 In the rule `main.row section article h2` edit the following properties:  
font-family: **source-sans-pro, Trebuchet MS, Arial, Helvetica, sans-serif;**  
font-size: **200%**  
line-height: **1em**
- 7 In the rule `main.row section article h2` add the following properties:  
**margin-top: 1.5em**  
**margin-bottom: .5em**

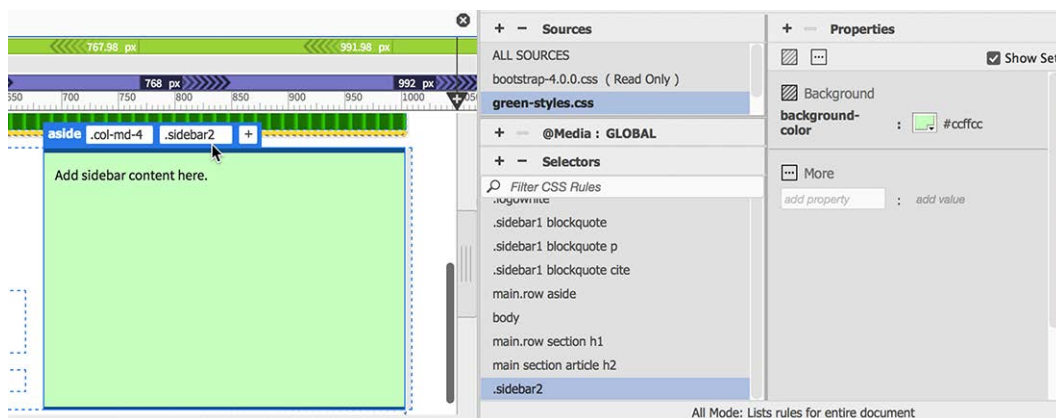


The article heading is now complete. Since the paragraph text inherits the styling from the body rule, no further styling needs to be done for the `<p>` element. You can move on to Sidebar 2.

- 8 In `mygreen_temp.dwt`, select and copy the `<p>` element in Sidebar 2. Notice that Sidebar 2 has a light-green background color.

● **Note:** You can use Code view or Live view to copy the placeholders. Be sure to use the same view when pasting.

- 9 In **mybootstrap.html**, paste the element in `aside.col-md-4` in the third column.
- 10 Switch to **mygreen\_temp.dwt**.  
Copy the background styles for the `.sidebar2` rule.
- 11 Switch to **mybootstrap.html**.  
Create the following rule in **green-styles.css**:  
**.sidebar2**
- 12 Paste the styles on the new rule.  
As with Sidebar 1, you have to add the new class to the `<aside>` element.
- 13 Select the `aside.col-md-4` tag selector for Sidebar 2.
- 14 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.

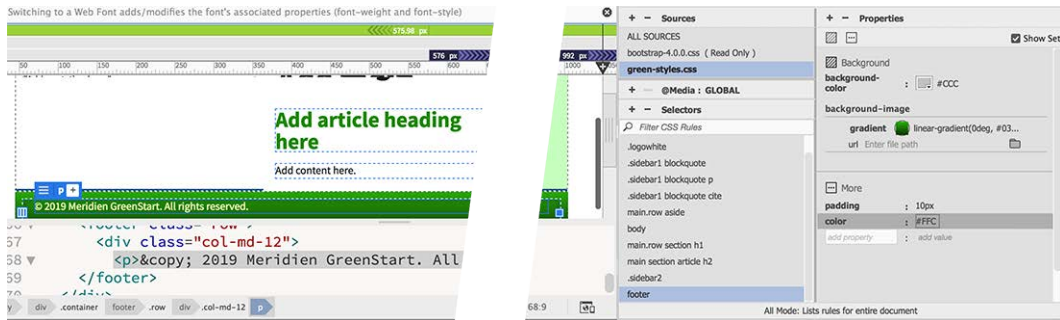
- 15 In **mygreen\_temp.dwt**, select and copy the `<p>` element in the `<footer>` element.  
Notice that the footer has a gradient background color.
- 16 In **mybootstrap.html**, paste the `<p>` element in `div.col-md-12` in the footer.
- 17 Switch to **mygreen\_temp.dwt**.  
Copy the background styles for the footer rule.
- 18 Switch to **mybootstrap.html**.  
Create the following rule in **green-styles.css**:  
**footer**

**19** Paste the styles on the new rule.

The footer displays the gradient background. The text is still black and seems a bit crowded.

**20** 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.

**21** 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.

- 1 Open **mybootstrap.html** in Live view, if necessary.

At the moment, all three columns have the same class name and are equal in width.

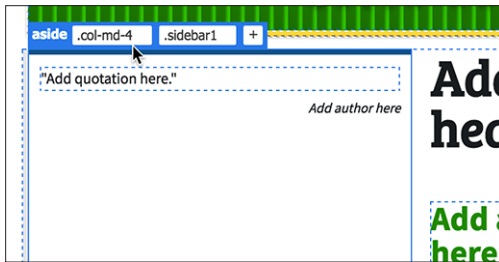
- 2 Click Sidebar 1, in Live view.

The Element Display appears focused on one of the elements in Sidebar 1.

- 3 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.

- 4 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.

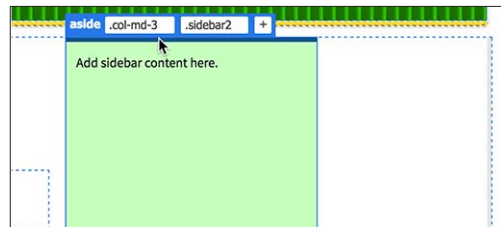
The width of the first column narrows. The other columns shift to the left, leaving open space on the right side of the third row. The Element Display is still focused on the first column.

- 5 Click Sidebar 2.

The Element Display focuses on one of the elements in `aside.col-md-4.sidebar2`.

- 6 If necessary, select the tag selector for `aside.col-md-4.sidebar2`.

- 7 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.

- 8 Click the heading *Add main heading here*.

The Element Display focuses on the h1 element.

- 9 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.


- 10 Save all files.

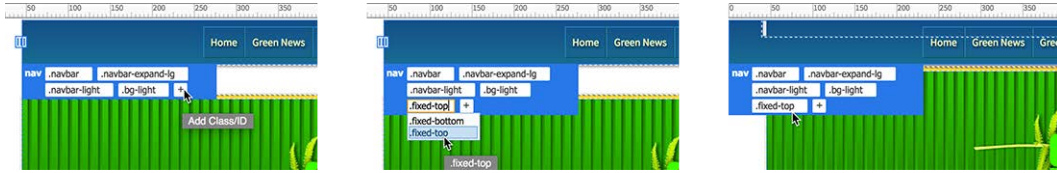
You have now completed the layout and have populated it with the boilerplate text and content placeholders. But there's still one thing you need to do before you convert the layout into the new site template. The main navigation menu on the current GreenStart site is styled to remain visible at the top 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.

- 1 Open **mybootstrap.html** in Live view, if necessary.
- 2 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.

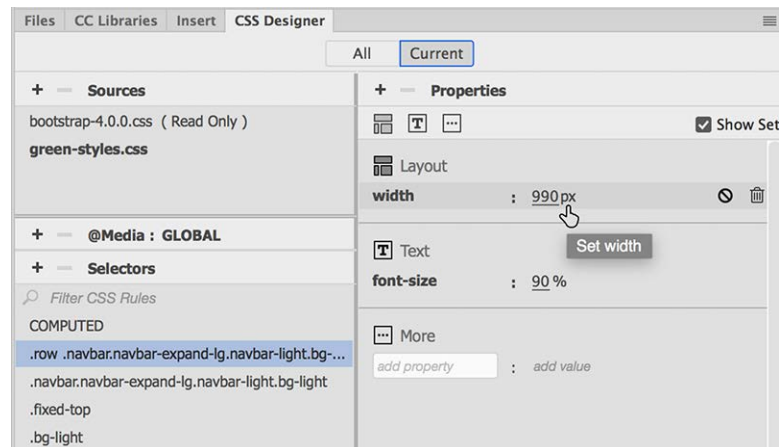


As you type, you will see the hinting list display any matching classes. Feel free to select the class when it appears. This is a predefined class in the Bootstrap style sheet, which formats these kinds of menus.

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.

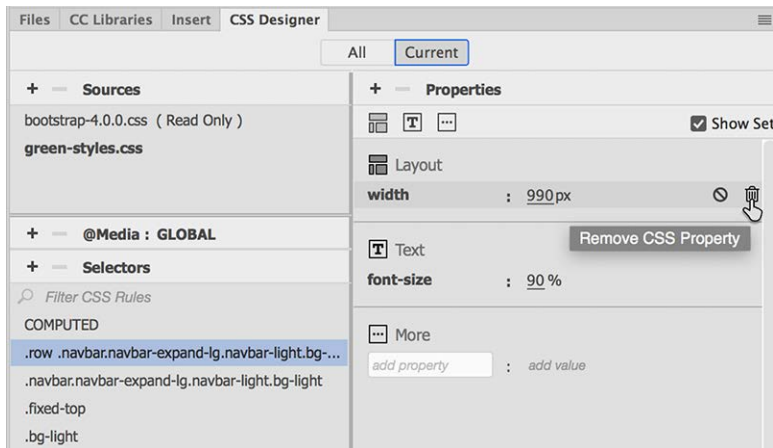
Because the header styling you brought over already anticipated a fixed menu, the previous gap between the navbar and header is now gone and everything snugs up nicely together. The only thing left to tweak is the width of the menu. For some reason, it touches the left side but doesn't stretch all the 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.

- 6 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.

- 7 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.

- 1 Open **mybootstrap.html** in Live view, if necessary.

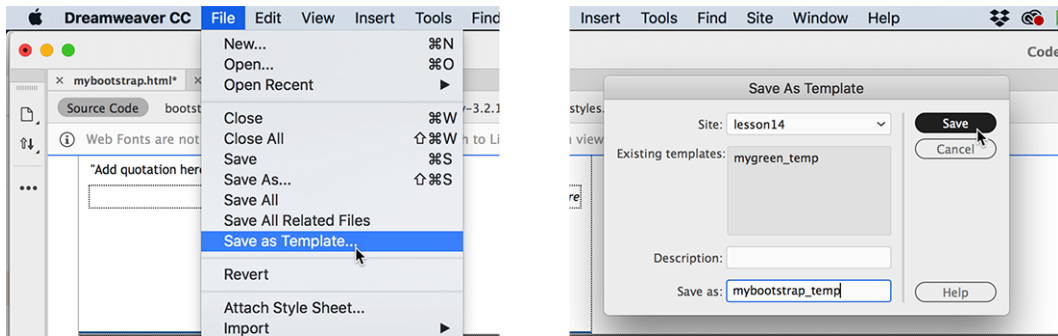
As of this writing, the template workflow functions only in Design view and Code view.

- 2 Switch to Design view.

Much of the styling and responsive nature of the layout is not supported in Design view.

- 3 Select File > Save As Template.

- 4 In the Save As Template dialog, name the template **mybootstrap\_temp** and click Save.



- 5 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.

- 6 Click the quotation placeholder.

Select the `blockquote` tag selector.

The `<blockquote>` element is highlighted in the layout, showing that it is selected.

- 7 Select Insert > Template > Editable Region.

The New Editable Region dialog appears.

- 8 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.

- 9 Click the text *Add main heading here*.

Select the h1 selector.

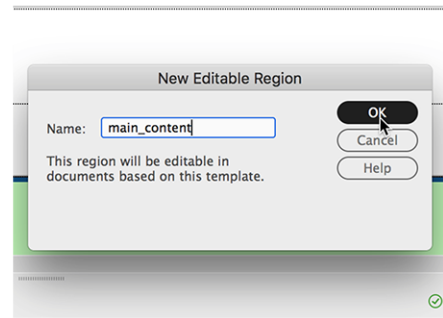
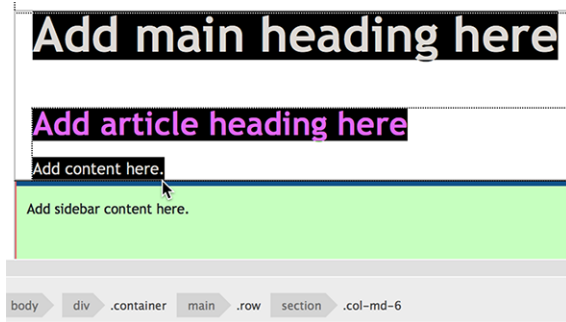
The <h1> element is highlighted in the layout, showing that it is selected.

- 10 Hold the Shift key and click at the end of the text “Add content here.”

The content placeholder is now completely selected.

- 11 Select Insert > Template > Editable Region.

- 12 Enter **main\_content** in the Name field and click OK.

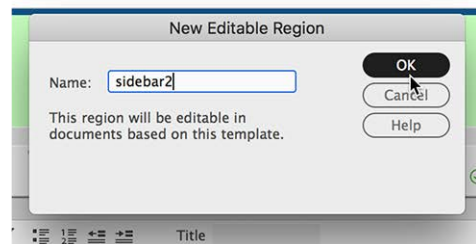
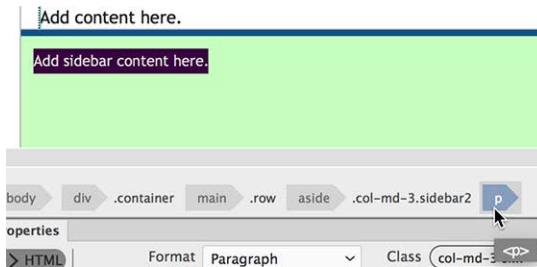


- 13 Click the text placeholder in Sidebar 2.

Select the p tag selector.

- 14 Select Insert > Template > Editable Region.

- 15 Enter **sidebar2** in the Name field and click OK.



- 16 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
16 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.

- 1 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.

- 2 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.

- 3 Drag the scrollbar to the right until the window is fully open.

To apply a new template, you have to switch to Code view or Design view.

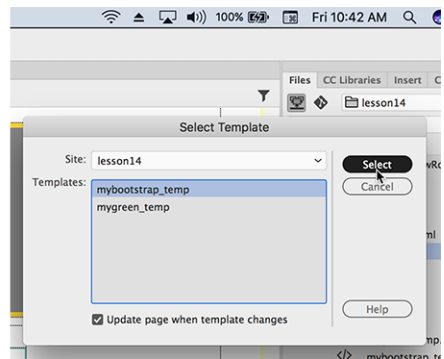
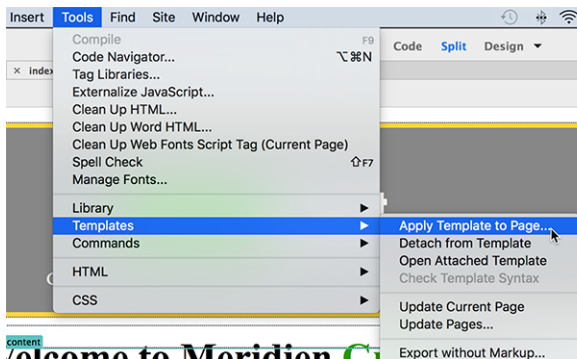
- 4 Switch to Design view.

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.

- 5 Select Tools > Templates > Apply Template To Page.

The Select Template dialog appears. The dialog lists both templates saved in the site.

- 6 Select **mybootstrap\_temp** and click the Select button.



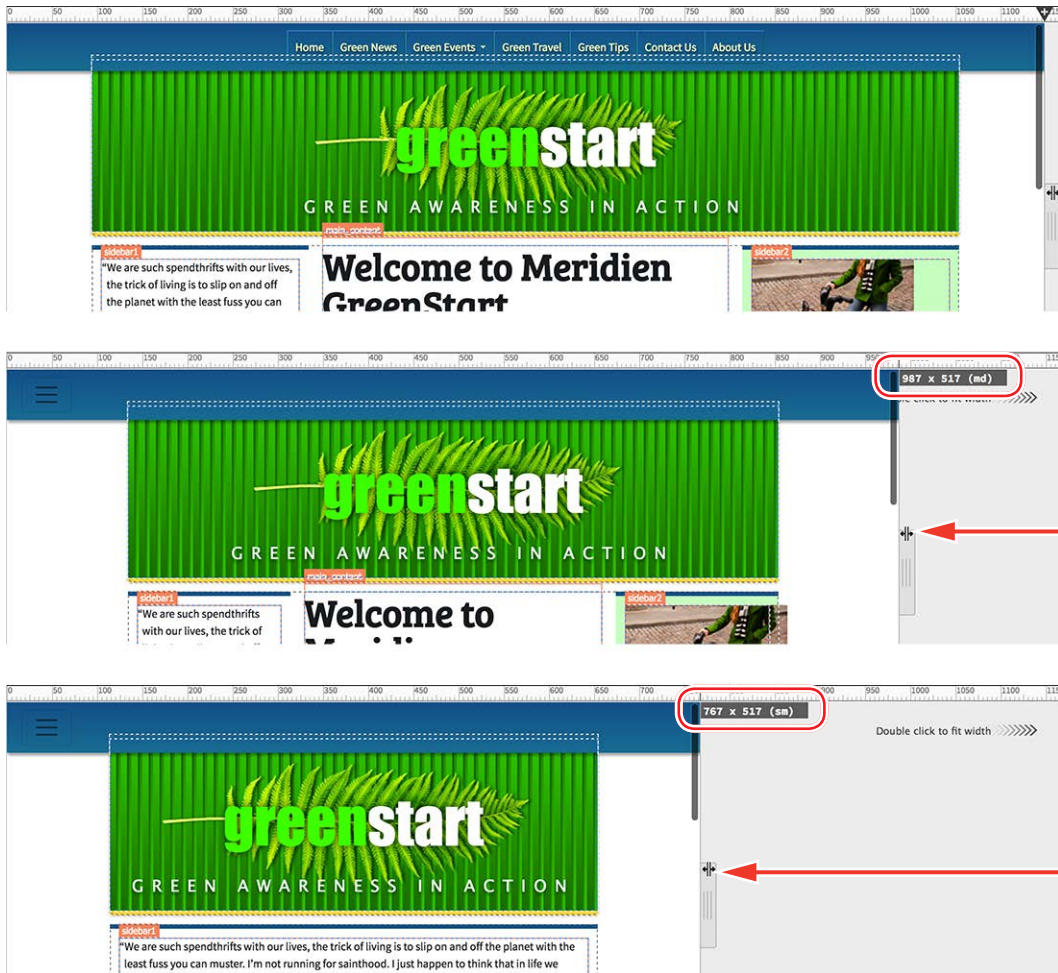
The new template is applied to the page instantly. In the top-right corner of the page, you will see the name of the new template displayed in a yellow box. The page is now based on the Bootstrap layout.

**7** Save the file.

The page is now ready to preview.

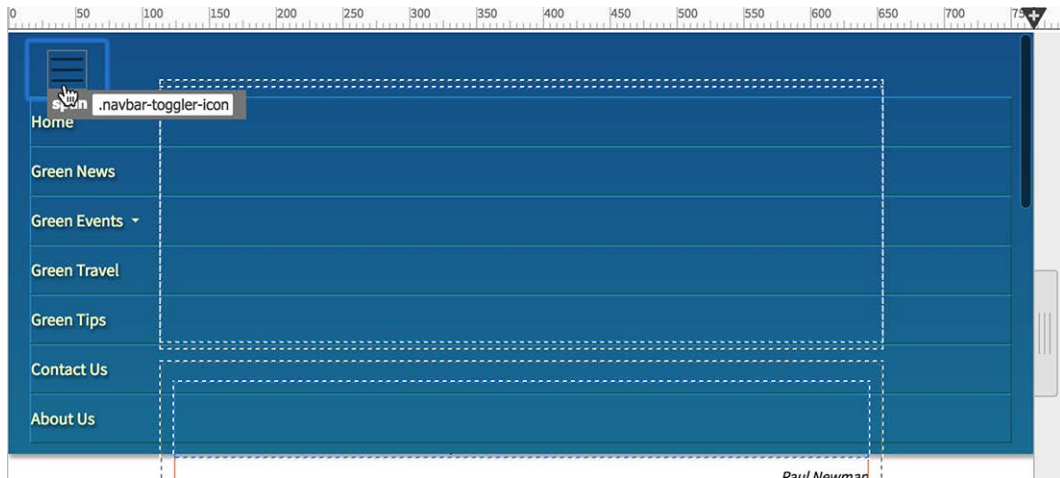
**8** Switch to Live view.

**9** 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.

- 10** Click the sandwich icon in the navigation bar.



The blue bar expands to show the menu. The seven menu items are stacked vertically.

- 11** Click the sandwich icon again.

The menu closes, showing the blue bar again.

- 12** 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.

- 13** Open **news.html**, **tips.html**, **events.html**, **travel.html**, **about-us.html**, and **contact-us.html** in Design view.

- 14** Apply **mybootstrap\_temp.dwt** to all pages.

- 15** 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.