5 CREATING A PAGE LAYOUT

Lesson Overview

In this lesson, you'll learn:

- The basics of web page design theory and strategy
- How to create design thumbnails and wireframes
- How to insert and format new components into a predefined CSS layout
- How to use Code Navigator to identify CSS formatting
- How to check for browser compatibility

This lesson should take 1 hour and 30 minutes to complete. Before beginning, make sure you have copied the files for Lesson 5 to your hard drive as described in the "Getting Started" section at the beginning of the book. If you are starting from scratch in this lesson, use the method described in the "Jumpstart" section of "Getting Started."

Whether you use thumbnails and wireframes or just a vivid imagination, Dreamweaver can quickly turn design concepts into complete, standards-based CSS layouts.

Web design basics

Before you begin any web design project for yourself, or for a client, there are three important questions you need to answer first:

- What is the purpose of the website?
- Who is the customer?
- 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 and/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 customer?

Are the customers adults, children, seniors, professionals, hobbyists, men, women, everyone? Knowing *who* your market will be is vital to overall design and functionality. 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.

Can two sites be more different than Google and Yahoo? Yet they both perform the same service.

How do they get here?

This sounds like an odd question when speaking of the Internet. But, like a brick-and-mortar business, your online customers can come to you in a variety of different ways. For example, are they accessing your site on a desktop computer, laptop, PDA, or cell phone? Are they using high-speed Internet or dial-up service? What browser do they most like 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 cell phone users may not want to see a lot of graphics or video, while 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. The W3C, for one, keeps track of tons of statistics regarding access and usage, all updated regularly:

- **www.w3schools.com/browsers/browsers_stats.asp**: Provides more information about browser statistics.
- **www.w3schools.com/browsers/browsers_os.asp**: Gives the breakdown on operating systems.
- **www.w3schools.com/browsers/browsers_display.asp**: Lets you find out the latest information on screen resolutions.

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, third-party tools are available, like Google Analytics and Adobe Omniture, which you can incorporate into your code to do the tracking for you for free, or for a small fee.

When you boil down all the statistics, this is what you will find: Windows (80 to 90%) dominates the Internet, with most users divided almost equally between Firefox (46%) and various versions of Internet Explorer (37%). The current average browser resolution is set at 1024 pixels by 768 pixels, or *higher*. If it weren't for cell phones and PDAs, these statistics would be great news for most web designers and developers. But the truth is the scope and penetration of cell phone usage on the Internet is an unknown quantity.

Each day more people are using cell phones to access the Internet. Younger users may use them to access the Internet more frequently than they use desktop computers. This presents a nagging problem to web designers. For one thing, cell phone 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 meager 200 to 300 pixels?

Another problem is that only the high-end smart phones can run web-based Flash content and some other client-based applications available today. Keep this in mind as you go through the process of designing your site.

Many of the concepts of print design are not applicable to the web because you are not in control of the user's experience. A page carefully designed for a typical flat panel is basically useless on a cell phone.

Scenario

For the purposes of this book you have been working on a website for Meridien GreenStart, a fictitious community-based organization dedicated to green investment and action. This website will offer a variety of different products and services and require a broad range of web page types, including dynamic pages using server-based technologies like ASP, ColdFusion, and PHP.

Your customers come from a broad demographic including all ages and education levels. They are people who are concerned about environmental conditions and who are dedicated to conservation, recycling, and reuse of natural and human resources.

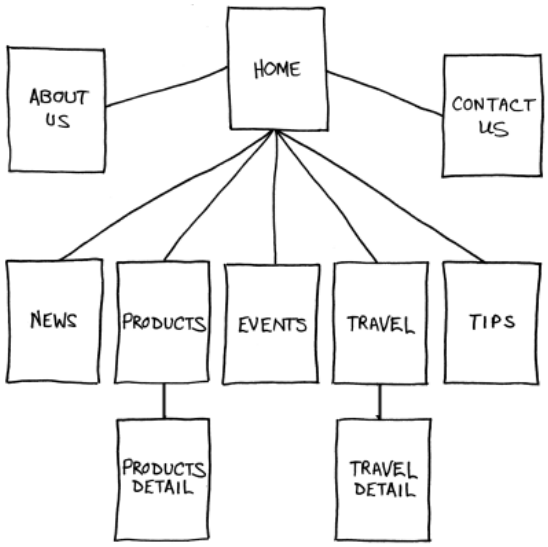
Your marketing research indicates that most of your customers use desktop computers or laptops, connecting via high-speed Internet services, but that you can expect 10 to 20 percent of your visitors via cell phone and other mobile devices.

Working with thumbnails and wireframes

The next step after you have nailed down the answers to the three questions about the website purpose, customer, and access method, is to figure out how many pages you'll need, what those pages will do, and finally, what they will look like.

Creating thumbnails

Many web designers start by drawing thumbnails with pencil and paper. Thumbnails are a graphical shopping list of the pages you need to create for the website. Thumbnails can also help you work out the website navigation. Draw lines between the thumbnails showing how your 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. The first level includes all the pages in your main navigation, the ones a visitor can reach directly from the home page. The second level includes pages you can only reach through specific actions or from specific locations, like 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 on each page, such as headers and footers, navigation, and areas for the main content and sidebars, if any. Put aside any items that won't be needed on every page. What other factors do you need to consider?

1. Header (includes banner and logo)
2. Footer (copyright info)
3. Horizontal navigation (for internal reference, i.e., Home, About Us, Contact Us)
4. Vertical navigation (links to products and services)
5. Main content (one-column with chance of two or more)

Identifying the essential components for each page helps in creating an effective page design that will meet your needs.

Do you have a company logo, business identity, graphic imagery, or color scheme you want to accent? Do you have publications, brochures, or advertising you want to emulate? It helps to gather them all together 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 collage.

Once you've created your checklist of the components you'll need on each page, develop several rough layouts that work for these components. Most designers usually settle on one basic page design that compromises between flexibility and sizzle. Minimizing the number of page designs may sound like a major limitation, but it's key to producing a professional-looking site. It's the reason why some professionals, like doctors and airline pilots, wear uniforms. Using a consistent page design lends a sense of professionalism and confidence to your visitor.

Wireframes allow you to experiment with page designs quickly and easily without wasting time with code.

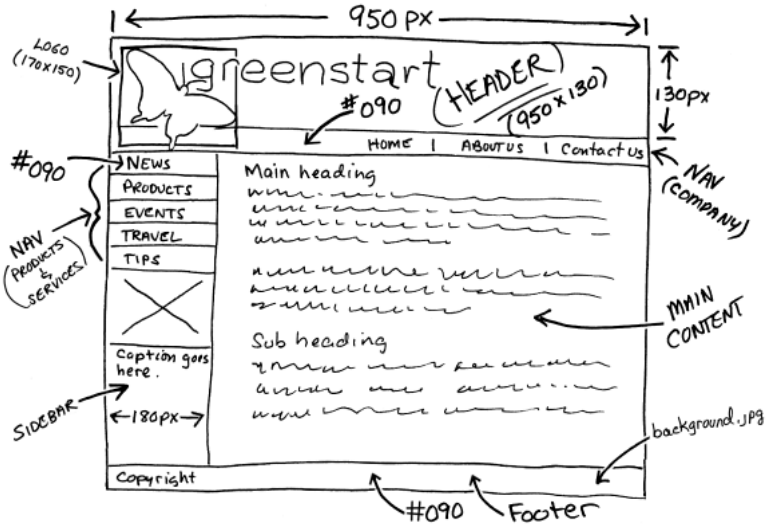
While you figure out what your pages will look like, you'll have to address the size and placement of the basic components. Where you put a component can drastically affect its impact and usefulness. In print, designers know that the upper-left corner of a layout is one of the "power positions," a place 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 work so well because of 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 2-inch cell phone?

That means the only thing you can be certain of is that the user can see the upper-left corner of your page. Do you want to waste this position by slapping the company logo here? Or, make the site more useful by slipping in a navigation menu? This is one of the key dilemmas 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. This step helps to catch or anticipate problems before you discover them when working in the code.



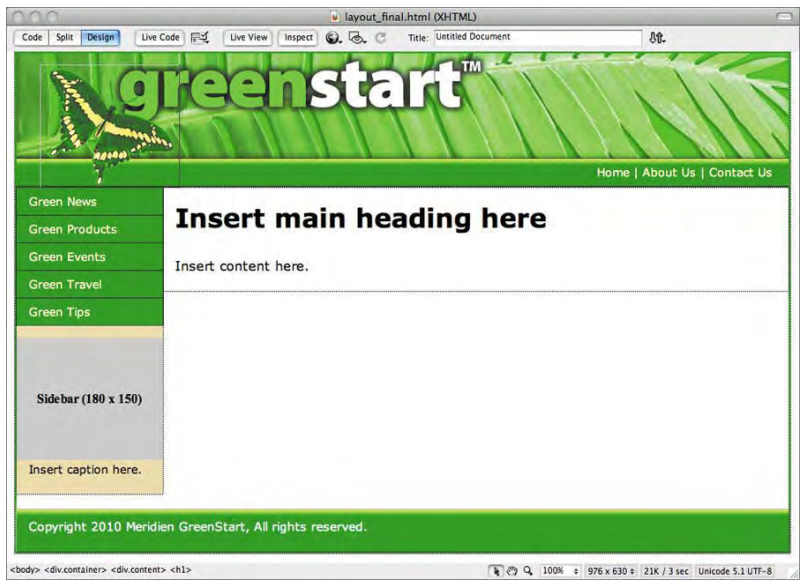
The wireframe for the final design should identify the components and feature markup for content, color, and dimensions.

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 Adobe Fireworks, Photoshop, or even Illustrator. It's a handy thing to know because you'll find that some clients just aren't comfortable giving an approval based only on pencil sketches. The advantage here is that all these programs allow you to export the results to a full-size image (JPEG, GIF, or PNG) that can be viewed in a browser. Such mockups are as good as seeing the real thing but may take only a fraction of the time to produce.

▶ **Tip:** For years designers have started the design process in Fireworks, where they can create a fully functional mockup with menus, links, and hotspots that can then be exported to a CSS-based HTML layout and edited in Dreamweaver.

In some cases, creating a mockup in Photoshop or Fireworks can save hours of tedious coding to receive a needed approval.

Previewing your completed file



This page represents the layout you will create in this lesson. In some ways it is similar to the page you created in Lesson 4, "Getting a Quick Start," but differs in that it has a left sidebar area and two navigation components.

Modifying an existing CSS layout

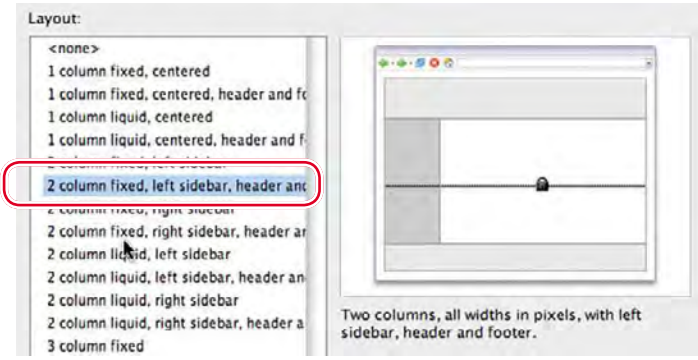
The predefined CSS layouts provided by Dreamweaver are always a good starting point. They are easy to modify and adaptable to most projects. Using a Dreamweaver CSS layout, you will create a proof-of-concept page to match the final wireframe design. This page will then be used to create the main project

template in subsequent lessons. Let's find the layout that best matches the wireframe:

1 Choose File > New.

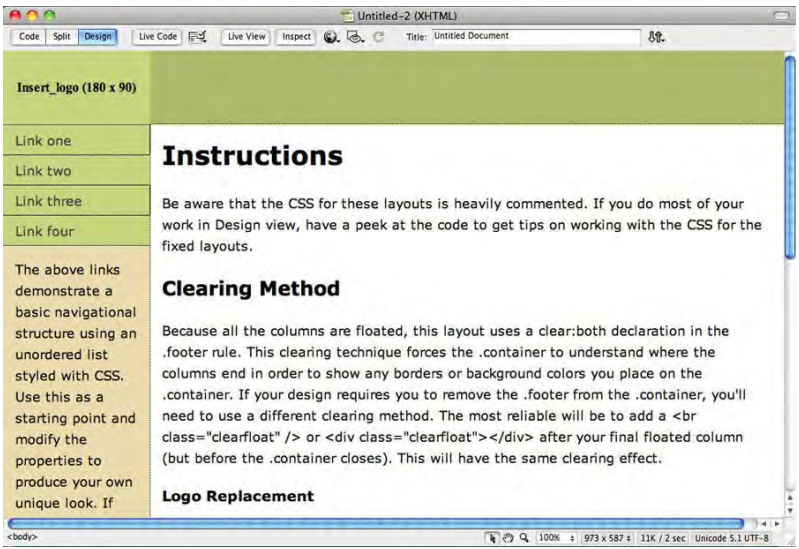
2 Choose Blank Page > HTML in the New Page dialog box. Examine each of the 16 sample CSS layouts to find the one that best fits the needs.

The layout *2 column fixed, left sidebar, header and footer* has the most in common with the target design.



3 Select **2 column fixed, left sidebar, header and footer** from the layout list. Click Open/Create.

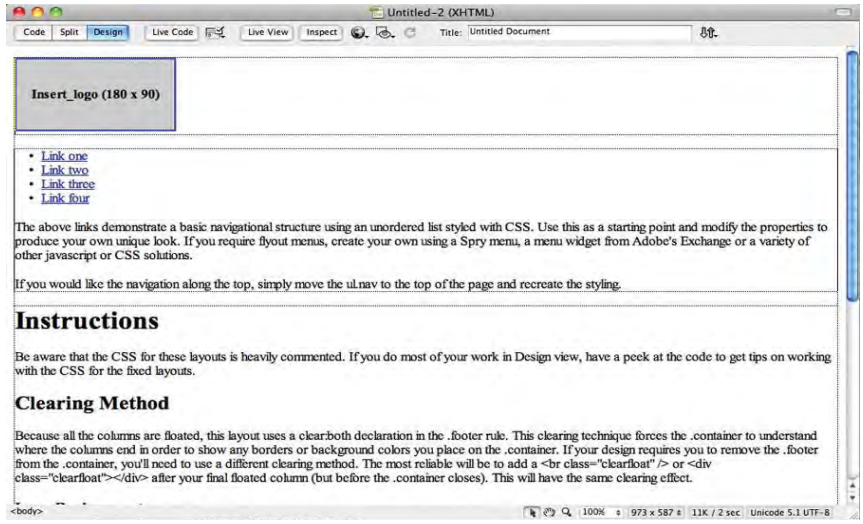
4 Switch to Design view, if necessary. Click in each content area, and using the tag selector display at the bottom of the document window, note the structure of the page components. Don't be afraid of using Code view, if you desire.



Using the skills you have learned in the last few lessons you should be able to deconstruct this CSS layout easily. The page consists of four content `<div>` elements (header, footer, sidebar1, and content) and one `<div>` that wraps around all the others (container). To understand exactly how much this design depends on CSS, it's sometimes a good idea to shut off CSS styling.

5 Choose View > Style Rendering > Display Styles to shut off CSS styling.

Style display is typically on by default (showing a check mark in the menu). By selecting it you'll toggle CSS styling off temporarily.



6 Note the identity and order of each page component.

Without CSS, the HTML skeleton is exposed for all to see. It's instructive to know what the page will look like if somehow the cascading style sheet is disabled or not supported by a particular browser. Now it's easier to identify the page components and their structure.

In fact, the order of elements is an important facet of how CSS styling and positioning works (and sometimes doesn't work). Although it is not strictly required, items that display higher on the page, like the header, usually are inserted before other elements that appear lower; items that align to the left should be inserted before elements aligned to the right. Therefore, the order of the elements in this layout—header, sidebar1, content, and footer—conforms to CSS best practices. The `div.container` holds all the other elements together and helps to make them behave properly in different browsers.

One last important aspect you should notice is the navigation menu. Without the CSS styling, the navigation menu reverted back to a simple bulleted list with hyperlinks. A few years ago this menu would have been built with images

and complex rollover animation. When those menus broke, for whatever reason, they usually became a jumbled unusable mess. Although the hyperlinks continued to work, without the images there were no words to tell the user what they were clicking. In this case, even without styling, these hyperlinks will always be usable.

7 Choose View > Style Rendering > Display Styles to turn on CSS styling again.

It's a good idea to get into the habit of saving files before you modify any settings or add content. Dreamweaver doesn't offer a backup or recovered-file feature; if it crashes before you save, all your work in any open, unsaved file will be lost.

8 Choose File > Save. In the Save As dialog box, navigate to the site root folder, if necessary. Name the file **mylayout.html** and click Save.

Dreamweaver normally saves HTML files to the default folder specified in the site definition, but it's a good idea to double-check the destination to make sure your files end up in the right place. All HTML pages created for the final site will be saved in the site root folder.

Adding a background image to the header

The final design features a banner similar to the one used in the previous lesson. You could insert the banner directly into the header, but adding it as a background image has the advantage of leaving the `<div>` open for other content.

1 Choose Window > CSS Styles to display the CSS Styles panel, if necessary.

2 Select the image placeholder *Insert_logo (180x90)* in the header. Press Delete.

The empty header will collapse to a fraction of its former size after deleting the image placeholder because it has no CSS height specification.

3 Double-click the `.header` rule to edit it.

4 In the Background category of the "CSS Rule Definition for *.header"* dialog box, click the Browse button next to the Background-image field.

5 Select **banner2.jpg** and note the dimensions of the image. Click OK/Choose.

Note that this banner doesn't feature the extra white space at the bottom or the butterfly as did the one used in Lesson 4. The new banner will better accommodate the horizontal navigation menu shown in the wireframe design. Eliminating the white portion also protects the overall page appearance in case something causes the header to expand in height exposing that part of the image. However, you're not totally out of the woods; remember that background images repeat both vertically and horizontally by default. To ensure this setting doesn't cause any undesirable effects, you'll need to change the repeat behavior.

6 Choose **no-repeat** from the Background-repeat menu.

7 Click Apply to view the results.

The header is wide enough but not tall enough to display the entire background image. Since background images aren't truly inserted in an element, they have no effect on the size of a container, good or bad. To ensure the `<div>` is large enough to display the entire image, you need to add a height specification to the `.header` rule.

8 In the Box category, type **130** in the Height field and choose px from the Unit of measurement menu. Click Apply.

There's no point setting the width attribute here. You learned in Lesson 4 the width of the layout in that example is specified in the `.container` rule. Chances are the same is true in this layout, too.

Before clicking OK, let's add some finishing touches to the header.

9 In the Background category, type **#090** in the Background-color field. Click OK.

It's important to start anticipating what would happen if things go wrong, say for example if the background image doesn't display. The green specified is part of the site color scheme and will be used throughout the website.

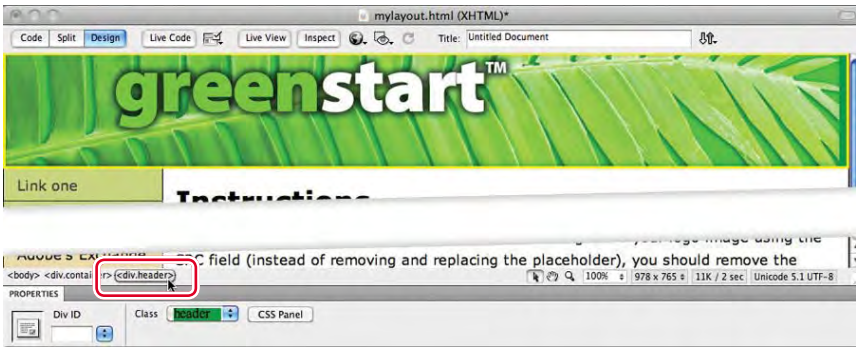
10 Choose File > Save.

Inserting new components

The wireframe design shows two new elements added to the default layout. The first contains the butterfly image, the second the horizontal navigation bar. Did you notice how the butterfly actually overlaps both the header and the horizontal navigation bar? There are several ways to achieve this effect. In this case, an absolutely positioned will work nicely.

1 Insert the cursor into the header, if necessary. Select the <div.header> tag selector. Press the Left Arrow key.

Note: To better understand how this technique works, try this step in Split view.



This procedure should insert the cursor before the opening tag for the header. If you had pressed the Right Arrow key, the cursor would move outside the closing tag for the header instead. Remember this technique—you'll use it frequently in Dreamweaver when you want to insert the cursor in a specific location before or after a code element without resorting to Code view.

2 Choose Insert > Layout Objects > AP Div.

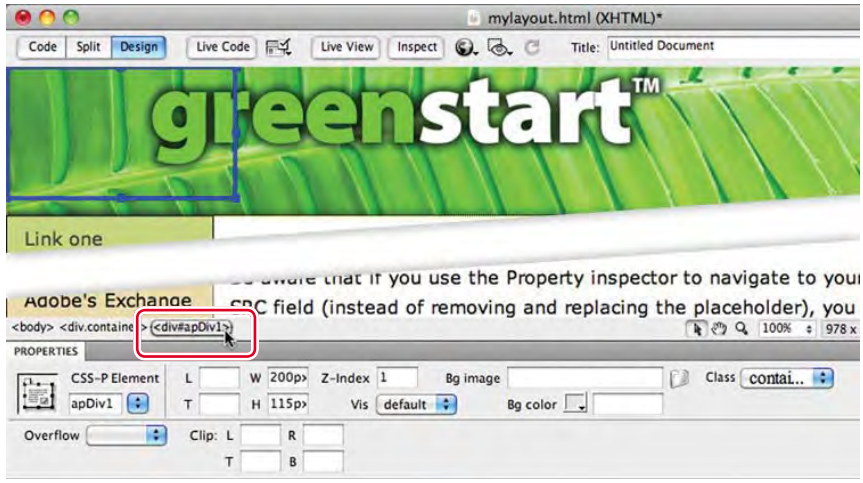
An AP div will appear at the top left of the header. Note the id (#1pDiv1) assigned to the new div in the tag selector. A corresponding rule has been added to the CSS Styles panel.

In previous versions of HTML, an AP div would have been assigned its size and position using inline HTML attributes. In a concession to new CSS-based web standards, these specifications are applied by CSS via a unique id created by Dreamweaver at the moment you insert the element.

Note: In previous versions of Dreamweaver, values entered in the Property inspector were added as inline HTML attributes. In CS5, most of these values will be added as custom CSS markup.

3 Click the <div#apDiv1> tag selector.

The Property inspector displays the properties for <div#apDiv1>. Note the width and height specifications displayed in the Property inspector. These values are actually stored in the #apDiv1 rule generated by Dreamweaver.



4 Insert the cursor into `<div#apDiv1>`.

5 Choose Insert > Image. Select **butterfly-ovr.gif** from the images folder. Observe the dimensions of the image: 170 pixels by 150 pixels.

6 Click OK/Choose.

The Image Tag Accessibility Attributes dialog box appears.

Note: In most cases, an image that's larger than its container will distort the container. Using images that are too large can destroy a carefully designed CSS layout. Pay close attention to the dimensions of images you use in your site.

7 Type **GreenStart Logo** in the Alternate text field in the Image Tag Accessibility Attributes dialog box. Click OK.

You'll notice that the AP div is slightly wider than the butterfly image. Although the extra space shouldn't cause any trouble, it's a good idea to match the dimensions of the container to the image.

8 Double-click the `#apDiv1` rule in the CSS Styles panel.

The "CSS Rule definition for *.container #apDiv1*" dialog box appears.

9 In the Box category, type **170** in the Width field. Type **150** in the Height field.

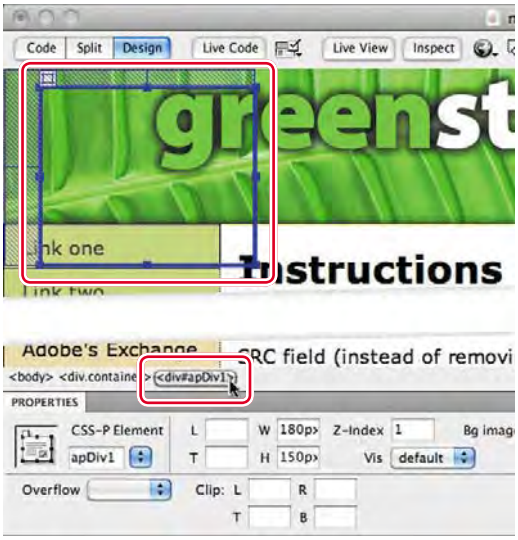
The `<div>` dimensions now match the height and width of the image.

10 Deselect the Same For All checkboxes for Margins.

11 Type **15 px** in the Top margin field. Type **30 px** in the Left margin field. Click OK.

When the rule definition dialog box vanishes, `<div#apDiv1>` appears floating over the header, 15 pixels from the top and 30 pixels from the left.

An AP div acts like a free agent. It ignores the other page components and can even be positioned above or below other `<div>` elements and content.



The <div#apDiv1> is complete. Let's create the for the horizontal navigation bar.

12 Insert the cursor back into the header. Click the <div.header> tag selector. Press the Right Arrow key.

The cursor should now appear after the ending tag of the header.

13 Choose Window > Insert, if the Insert panel is not visible. Choose Layout from the Category menu. Click Insert Div Tag.

The Insert Div Tag dialog box appears. Note that the Insert menu displays the selection "At insertion point". The new will be inserted into the code at that position and not wrap around any other elements.

14 Type **h-navbar** in the ID field. Click the New CSS Rule button.

The New CSS Rule dialog box appears with the ID #h-navbar automatically entered in the Selector Name field. Click OK.

15 In the Type category, type **90** in the Font-size field and select the percentage sign (%) from the pop-up list. Type **#FFC** in the Color field.

16 In the Background category, type **#090** in the Background-color field. Click the Browse button next to the Background-image field.

17 Select **background.jpg** from the images folder. Click OK/Choose.

18 Choose repeat-x from the Background-repeat menu.

19 In the Block category, select Right from the Text-align field.

20 In the Box category, deselect the Same For All check box for Padding.
Type **5 px** in the Top padding field.
Type **20 px** in the Right padding field.
Type **5 px** in the Bottom padding field.

Note: To enter values in the bottom field, remember to deselect the Same For All checkboxes in each section first.

21 In the Border category, enter the following values only in the Bottom border fields: **solid**, **2 px**, **#060**. Click OK in the CSS Rule Definition dialog box. Click OK in the Insert Div Tag dialog box.

The `<div#h-navbar>` appears below the header fully formatted and filled with placeholder text.

22 Delete the placeholder text. Type **Home | About Us | Contact Us** in `<div#h-navbar>` as placeholders for the organizational navigation links. You will convert these to actual hyperlinks in Lesson 10, "Working with Navigation."

23 Press Ctrl-S/Cmd-S to save the file.

Modifying the page width and background color

Before you convert this file into the project template, let's tighten up the formatting and the placeholder content. As you did in the last lesson, the overall width has to be modified to match the banner image.

1 Double-click the `.container` rule in the CSS Styles panel.

2 In the Box category, change the width to **950 px**. Click OK.

The `<div.container>` element now matches the width of the banner image, but you probably noticed there was an unintended consequence when you changed the overall width. The main content area shifted down below the sidebar. To understand what happened, you'll have to do a quick investigation.

3 In the CSS Styles panel, click the `.content` rule and check its properties. Note its width: 780 pixels.

4 Click the `.sidebar1` rule and check its width: 180 pixels.

Combined, the two `<div>` elements total 960 pixels. The elements are too wide to sit side by side in the main container. This type of error is common in web design and easily fixed.

5 Click the `.content` rule in the CSS Styles panel. In the Properties section of the panel, change the width to **770 px**.

The `<div.content>` element moves back to its intended position. This was a good reminder that the size, placement, and specifications of page elements have important interactions that can affect the final design and display.

Let's remove the page background color.

6 Double-click the `body` rule. In the Background category, change the Background-color to **#FFF**. Click OK.

Notice that the absence of the background color gives the impression that the page's content area drifts off into the wide expanse. You could give `<div.container>` its own background color or simply add a border to give the page a definitive edge.

7 Double-click the `.container` rule. In the Border category, select the Same For All option and enter the following values for all border fields: **solid**, **2px**, and **#060**. Click OK.

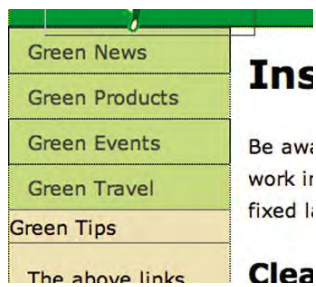
8 Save the file.

Modifying existing content and formatting

As you can see, the CSS layout is already equipped with a vertical navigation menu. The generic hyperlinks are simply placeholders, waiting for your final content. Let's change the placeholder text in the menu to match the pages outlined in the thumbnails and modify the colors to match the site color scheme.

1 Select the placeholder text *Link one* in the first menu button.
Type **Green News**.
Change *Link two* to read **Green Products**.
Change *Link three* to read **Green Events**.
Change *Link four* to read **Green Travel**.

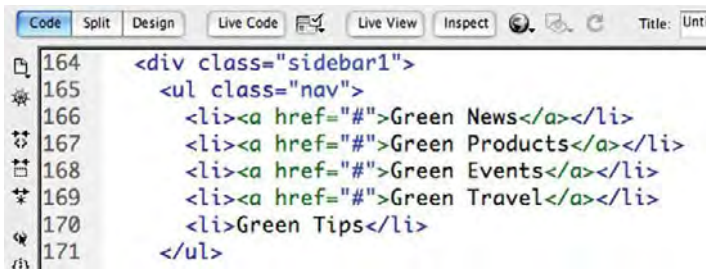
One of the advantages of using bulleted lists as navigational menus is that it's easy to insert new links.



2 With the cursor still at the end of the words *Green Travel*, press Enter/Return. Type **Green Tips**.

The new text appears in what looks like a button structure, but the background color doesn't match and the text isn't aligned with other menu items. You could probably figure out what's wrong in Design view, but in this case, the problem may be identified faster in Code view.

3 With the cursor still inserted in the menu, select Code view. Observe the menu items and compare the first four with the last. Can you see the difference?



The difference is obvious in Code view. The last item is formatted with the `<li>` element like the others—as part of the bulleted list—but it doesn't feature the `<a href="#">` code that's used as a hyperlink placeholder. For *Green Tips* to look like the other menu items, you have to add a hyperlink, or at least a similar placeholder.

4 Select the text *Green Tips*. In the Link field of the HTML Property inspector, type # and press Enter/Return.

The code in all the items is identical now.

5 Select Design view.

All the menu items are identically formatted now. You'll learn more about how to format text with CSS to create a menu in Lesson 6, "Working with Cascading Style Sheets."

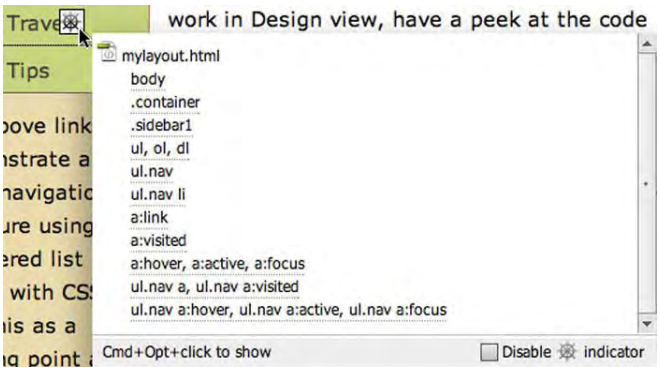
The current menu color doesn't match the site color scheme. To change the color, you'll have to do some investigative work to find which CSS rule controls this formatting. This is a good time to learn how to use the Code Navigator.

6 Insert the cursor into any of the menu items.

After a few seconds the Code Navigator () icon will appear.

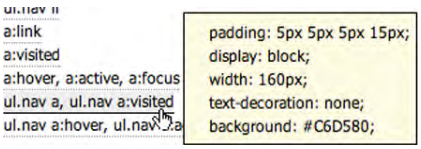
7 Click the Code Navigator icon.

A small window opens displaying a list of 11 CSS rules that affect the targeted element. The list is in order of specificity, with the most powerful rule at the bottom. In some cases, the rules listed may only affect the element in a roundabout way, as in the `body` rule, which affects *all* HTML elements on the page.



8 Move the cursor over the first CSS rule in the list. Observe the formats displayed.

Another window appears listing all the formats specified in the selected rule. You're looking for a background color that's applied to the menu items. Be careful. There's probably more than one background-color in the listed rules, so if you find one, it's important to determine whether it actually affects the menu or something else.



9 Examine each rule until you find the pertinent one.

The `.sidebar1` rule specifies a background-color, but it affects the `<div>` and not the menu. The one you're actually looking for is applied by the rule: `ul.nav a, ul.nav a:visited`. This rule specifies a background-color for a `<ul>` (unordered list) element with a class attribute of `nav` that contains an `<a>` (hyperlink) element. Sound right?

10 Locate the rule `ul.nav a, ul.nav a:visited` in the CSS Styles panel and select it. In the Properties section of the panel, change the existing background-color to **#090**.

The color of the menu items now matches `<div#h-navbar>`. The black text is difficult to read in the green background color. You can use the Properties section of the CSS Styles panel to add as well as edit element properties.

11 Click the Add Property link in the Properties section of the CSS Styles panel.

A new property field appears.

12 Choose Color from the Property field menu. Enter **#FFC** in the Value field. Press Enter/Return to complete the new rule property.

13 Save the file.

Inserting an image placeholder

The sidebar will feature photos, captions, and short blurbs on environmental topics. Let's insert a placeholder image and caption below the vertical menu.

1 Insert the cursor into the text directly below the vertical menu. Click the <p> tag selector.

The placeholder image should not be inserted within the <p> element. If it were, it would inherit any margins, padding, and other formatting applied to the paragraph, which could cause it to break the layout.

Tip: Use Split view whenever you're unsure where the cursor is inserted.

2 Press the Left Arrow key.

The cursor is moved to the left of the opening <p> tag.

3 Choose Insert > Image Objects > Image Placeholder.

The Image Placeholder dialog box appears.

4 In the Image Placeholder dialog box, type **Sidebar** in the Name field.

Type **180** in the Width field.

Type **150** in the Height field.

Click OK.

An image placeholder appears in `<div.sidebar1>` below the vertical menu.

5 Select all the text below the image placeholder. Type **Insert caption here**.

The caption placeholder replaces the text.

6 Press Ctrl-S/Cmd-S to save.

Inserting placeholder text

Let's simplify the layout by replacing the existing headings and text in the main content area.

1 Double-click to select the heading *Instructions*. Type **Insert main heading here** to replace the text.

2 Select the remaining text in `<div.content>`. Type **Insert content here** to replace the text.

3 Press Ctrl-S/Cmd-S to save.

Modifying the footer

You need to reformat the footer and insert the copyright information.

1 Double-click the `.footer` rule in the CSS Styles panel.

2 In the Type category, enter **90%** in the Type-size field and **#FFC** in the Color field.

3 In the Background category, click the browse icon and insert **images/background.jpg** in the Background-image field.

4 Click OK/Choose.

5 Choose repeat-x from Background-repeat field menu.

6 Type **#090** into the Background-color field. Click OK.

7 Select the placeholder text in the footer. Type **Copyright 2010 Meridien GreenStart, All rights reserved**.

8 Press Ctrl-S/Cmd-S to save.

The basic page layout is complete.

Checking browser compatibility

The CSS layouts included with Dreamweaver have been thoroughly tested to work flawlessly in all major browsers. However, during the lesson you made major modifications to the original layout. These changes could have ramifications in the compatibility of the code in certain browsers. Before you use this page as your project template, you should check its compatibility.

1. If necessary, open **layout.html** in Dreamweaver.
2. Choose File > Check Page > Browser Compatibility.

   When the Report box opens, there should be no issues listed.
3. To close the report, double-click the Browser Compatibility tab in the Report panel or right-click the tab and choose Close Tab Group from the context menu.

Congratulations. You created a workable basic page layout for your project template and learned how to insert additional components, image placeholders, text, and headings; adjust CSS formatting; and check for browser compatibility.

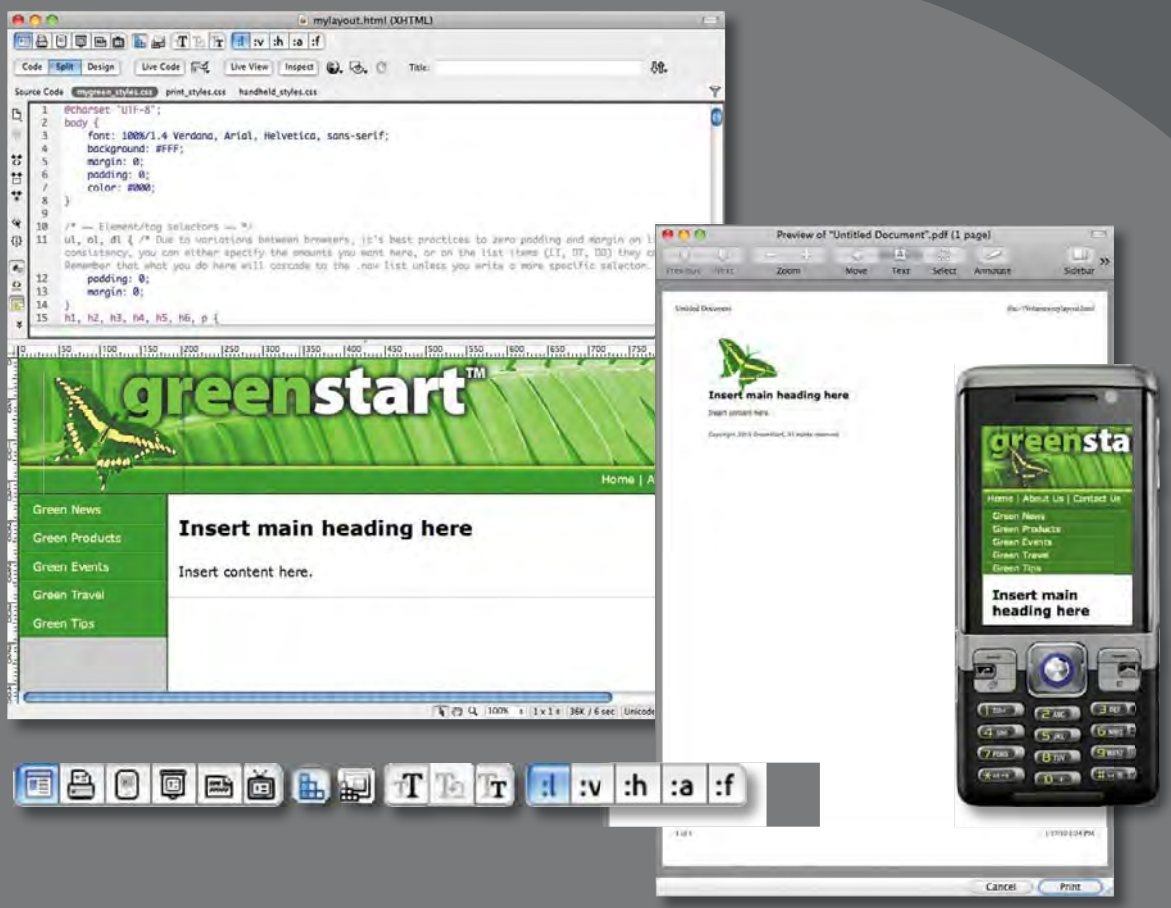
6 WORKING WITH CASCADING STYLE SHEETS

Lesson Overview

In this lesson, you'll work with cascading style sheets (CSS) in Dreamweaver and do the following:

- Manage CSS rules using the CSS Styles panel
- Create new CSS rules
- Create and apply custom CSS classes
- Create descendant selectors
- Create styles for page layout elements
- Move CSS rules to an external style sheet
- Create a style sheet for print applications

This lesson will take about 2 hours to complete. Before beginning, make sure you have copied the files for Lesson 6 to your hard drive as described in the "Getting Started" section at the beginning of the book. If you are starting from scratch in this lesson, use the method described in the "Jumpstart" section of "Getting Started."



Today, pages designed in compliance with web standards separate the content from the formatting. Stored in a cascading style sheet (CSS), the formatting can be quickly changed and substituted for specific applications and devices.

Previewing a completed file

To see the finished page you will create in this lesson, you can preview it in a browser.

1. In the Files panel, expand the lesson06 folder.
2. Select the **layout_finished.html**.
3. Preview the page in your primary browser.

   Note the layout, various colors, and other formats applied to the text and page elements—all created by CSS styles.
4. In Dreamweaver, select File > Close to close **layout_finished.html**, if necessary.

Working with the CSS Styles panel

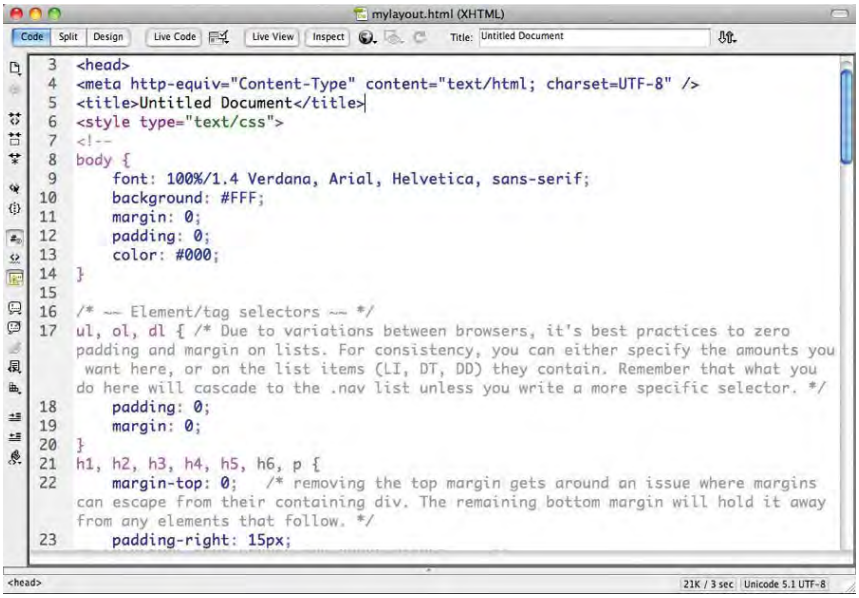
In Lesson 5, "Creating a Page Layout," you used one of the CSS layouts provided by Dreamweaver to start building your project site template page. These layouts come equipped with an underlying structure and a whole set of predefined CSS rules that establish the basic design and formatting of the page components and content. In the upcoming exercises in this lesson, you'll modify these rules and add new ones to complete the site design. But before you proceed, it's a vital aspect of your role as designer to understand the existing structure and formatting before you can effectively complete your tasks. It's important to take a few minutes at this time to examine the rules and understand what role they perform in the current document.

1. Open **mylayout.html** from the site root folder, if necessary. Or, if you are starting from scratch in this exercise, see the "Jumpstart" instructions in the "Getting Started" section at the beginning of the book.
2. Display the CSS Styles panel, if it isn't visible.

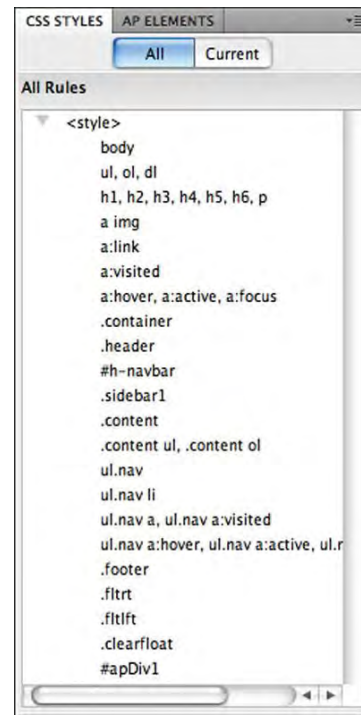
   The CSS Styles panel displays the `<style>` tag indicating that the style sheet is embedded in the `<head>` section of the document.

Tip: If you don't see line numbers along the side of your Code view window, choose View > Code View Options > Line Numbers to turn on this feature.

3 Select Code view and locate the <head> section (starting around line 3). Locate the element <style type="text/css"> and examine the subsequent code entries.



All the CSS rules displayed in the list are contained within the <style> element.



4 Note the names and order of the selectors within the CSS code.

5 Select Design view. In the CSS Styles panel, expand and examine the list of rules.

The list shows the same selector names in the same order you saw in Code view. There is a one-to-one relationship between the CSS code and the CSS Styles panel. When you create new rules or edit existing ones, Dreamweaver makes all the changes in the code for you, saving you time and reducing the possibility of code-entry errors. The CSS Styles panel is just one of the many productivity enhancements that you'll use and master in this book.

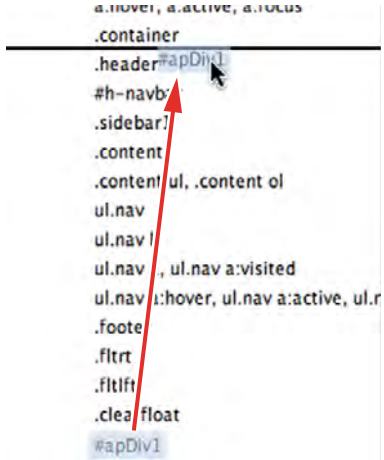
You should have 22 rules at this time—20 that came with the CSS layout, and 2 you made yourself in the previous lesson. The order of your rules may vary from this figure. The CSS Styles panel makes it easy to reorder the list.

In the last lesson, you created <div#apDiv1> and inserted it into the layout. The #apDiv1 rule formats the holding the butterfly logo and appears in the code between <div.container>

and <div.header>. But as you can see in the pictured CSS Styles panel, it appears at the bottom of all the rules. Moving this rule within the style sheet will not affect how it formats the element, but it will make it easier to find if you need to edit later.

Note: The names and order of styles in your panel may vary from those pictured.

6 Select the #apDiv1 rule and drag it directly underneath the .container rule.



Dreamweaver has moved the rule within the list, but that's not all. It has also rewritten the code in the embedded style sheet, moving the rule to its new position. Arranging related rules together can save time later when you need to format specific elements or components. But be watchful for unintended consequences. Moving rules in the list can upset the established cascade or inheritance relationships you have already created. Review Lesson 3, "CSS Basics," if you need a refresher on these theories.

Before you move any other rules, you should first understand what function each one performs.

7 Select the body rule in the CSS Styles panel. Observe the properties and values that appear in the Property section of the panel.

Most of these settings came with the layout, although you changed the background color in the last lesson. Note how the margins and padding are set to zero.

8 Select the next ul, ol, dl rule and observe the values that appear.

As in the body rule, this rule sets all margin and padding values to zero. Do you know why? An experienced web designer could select each rule in turn and probably figure out the reasons for each of the formats and settings. But you don't need to resort to hiring a consultant, when Dreamweaver provides much of the information you need already.

9 Right-click the `ul, ol, dl` rule and choose Go to Code from the context menu.

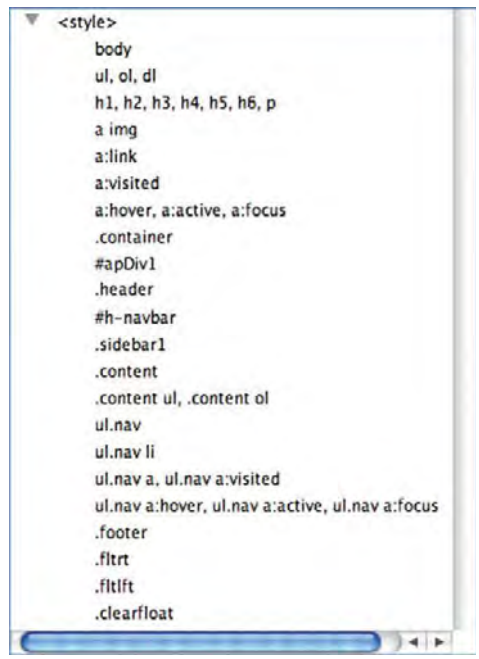
Dreamweaver displays the document in Split view and focuses on the section of code that contains the `ul, ol, dl` rule. Observe the text between the opening `/*` and closing `*/` markers. This markup is called a *comment* and contains text that usually provides behind-the-scenes information that will not be displayed within the browser or affect any elements. Comments are a good way to leave handy reminders within the body of the web page or notes to yourself, or others, explaining why you wrote the code in a particular way. You'll notice that some of the comments are used to introduce a set of rules, and others are embedded in the rule itself.

10 Scroll down through the style sheet and study the comments, paying close attention to the embedded ones.

The more you understand what these predefined rules are doing, the better results you can achieve for your final site. Here's what you'll find: the rules `body`, `.header`, `.container`, `.sidebar1`, `.content`, and `.footer` define the basic structural elements of the page. The rules `a:img`, `a:link`, `a:visited`, `a:hover`, `a:active`, and `a:focus` set up the appearance and performance of the default hyperlink behavior; `ul.nav`, `ul.nav li`, `ul.nav a`, `ul.nav a:visited`, `ul.nav a:hover`, `ul.nav a:active`, and `ul.nav a:focus` define the look and behavior of the vertical menu. The remaining rules are intended to reset default formatting or add some desired styling as outlined in the embedded comments.

For the most part, there's nothing unacceptable, or fatal, in the current order of the rules, but keeping related rules together will pay productivity dividends later when the style sheet gets more complicated.

11 Using the CSS Styles panel, reorder the rules in the list, as shown in the figure on the right.



Note: When moving rules using the CSS Styles panel, the position of comments that are not embedded may not be preserved.

Now that you are more aware of the rules and rule order, taking special care with rule order from this point forward when you create new styles is a good practice.

12 Save the file.

Creating new CSS rules

In most of the previous exercises, you merely edited the rules that were predefined in the CSS layout. In the next exercise, you will learn how to create your own custom rules for HTML elements, classes, and ids.

Creating descendant selectors

The predefined style sheet declares a rule for multiple elements that will affect all `h1`, `h2`, `h3`, `h4`, `h5`, `h6`, and `p` tags no matter where they appear on the page. But, if you want to target a style at a specific tag within a specific `<div>`, it requires a *descendant* selector. Dreamweaver makes it easy to create such rules.

1 Insert the cursor in the main heading in the main content area. Note the names and order of the tag selectors at the bottom of the document window.

The heading is an `<h1>` element in `<div.content>` in `<div.container>` in the `body` element. As described in the last exercise, when creating new rules take care about where they appear in the style sheet. Rules at the top of the sheet pass formatting to rules appearing later. Inserting a rule in the wrong place could cause the browsers to ignore it altogether.

2 Select the `.content` rule in the CSS Styles panel.

By selecting the `.content` rule first, Dreamweaver will insert the new rule immediately following it in the style sheet.

3 In the CSS Styles panel, click the New CSS Rule (+) icon. If the Compound selector type is not displayed, choose it from the Selector Type menu.

The New CSS Rule dialog box opens. Typically, when the cursor is inserted into page content, the dialog box defaults to the Compound selector type and displays a descendant selector based on the location of the cursor, in this case `.container .content h1`.

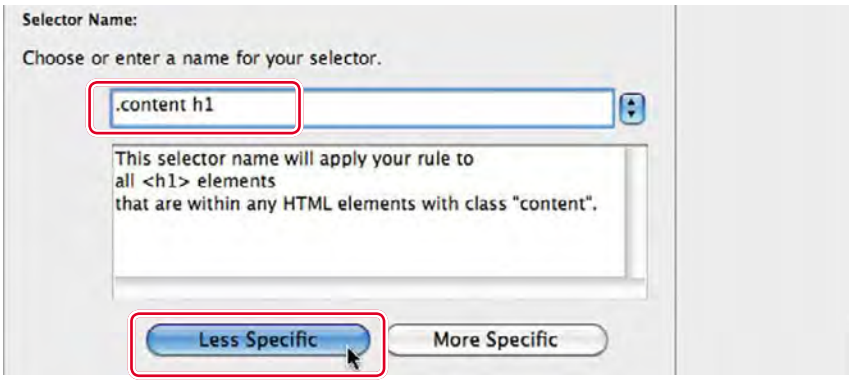
Note: When the cursor is inserted into the page content, Dreamweaver will always create the compound selector for you, even if the Compound option is not displayed when the dialog box first appears.

If you remember the CSS syntax you learned in Lesson 3, you know that this new rule will affect <h1> elements, but only when they appear within an element formatted with a class of .content, when both appear within an element formatted with a class of .container. All other <h1> elements will remain unaffected.

Since there will only be one <div.content> element in this page design, there's no need for such specificity in the rule. Whenever possible, rules should be simplified to reduce the total amount of code that needs to be downloaded. Although in this case it's only the word .container that isn't needed, unnecessary code adds up across the entire site (and Internet) overall.

4 In the New CSS Rule dialog box, click the Less Specific button. Click OK.

The word .container is removed from the Selector Name field.



5 In the Type category of the "CSS Rule Definition For *.content h1"* dialog box, enter **160%** in the Font-size field.

6 In the Box category, deselect Same For All and enter **10px** in the top margin field only. Enter **5px** in the Bottom margin field. Click OK.

The main heading appears smaller in size and 10 pixels lower on the page. Note that the new rule was inserted directly after the .content rule in the CSS Styles panel.

7 Save the file.

Creating custom classes

CSS class attributes allow you to apply custom formatting to a specific element, or portion of a specific element. Let's create a class that will allow you to apply the GreenStart logo color as you did in Lesson 4, "Getting a Quick Start."

1 In the CSS Styles panel, click the New CSS Rule icon.

2 Choose Class from the Selector Type menu. Type **green** in the Selector Name field. Click OK.

▶ **Tip:** When the desired styling doesn't appear as expected, use the Code Navigator to suss out the conflict.

3 In the "CSS Rule Definition For *.green*" dialog box, type **#090** in the Color field of the Type category. Click OK.

The `.green` rule is added to the style sheet. In most instances, a class attribute will override any default or applied styling, so it should not matter where in the style sheet it appears. However, differing, even contradictory, effects can occur when classes are combined with elements and/or ids to form compound selectors.

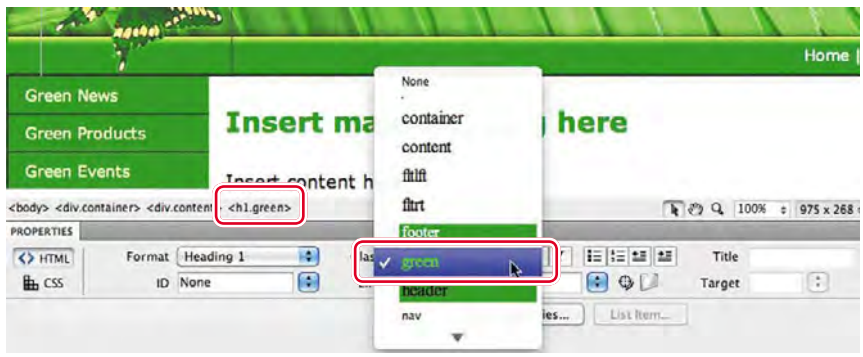
Dreamweaver makes it easy to apply classes. Let's apply a class to an entire element.

4 Insert the cursor anywhere in the `<h1>` element in `<div.content>`. Make sure the cursor is flashing in the element and no text is selected.

● **Note:** You may need to refresh the page display to see the updated tag selector.

5 In the Property inspector, choose **green** from the Class menu.

All the text in the `<h1>` element is now formatted in green. At the bottom of the document window, `<h1.green>` now displays in the tag selectors.



6 Switch to Code view. Examine the opening tag of `<h1.green>`.

```
<h1 class="green">Insert main heading here</h1>
```

▶ **Tip:** You can also apply the class by clicking the tag selector before selecting the class from the Property inspector.

The rule is applied as an attribute to the tag as `<h1 class="green">`. When the cursor is inserted in an existing element, Dreamweaver assumes you want to apply the class to the entire element.

Now let's remove a class from an element.

7 Insert the cursor anywhere in the formatted element.

The tag selector displays `<h1.green>` and *green* appears in the Class menu of the Property inspector.

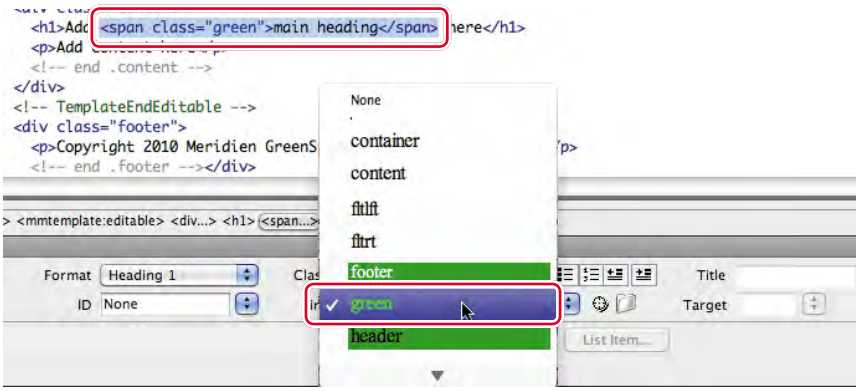
8 Choose None from the Class menu of the Property inspector.

The class attribute is removed from the code. The tag selector is now <h1> and *None* appears in the Class menu.

Next let's apply a class to a range of text.

9 Select the words *main heading* in the <h1> element. Choose green from the Class menu in the Property inspector.

The class is applied to the selected text using .



Now remove the class.

10 Switch to Design view. Insert the cursor anywhere in the formatted text. Choose None from the Class menu.

The text returns to the default formatting. When the cursor is inserted in a class-formatted element, Dreamweaver assumes you want to remove the formatting from the entire range of text.

▶ **Tip:** Classes can be applied and removed in either Design or Code view.

11 Save the file.

Creating custom ids

The CSS id attribute is given more specific weight in CSS styling because it is used to identify unique content on a web page and should trump all other styling. The AP div containing the butterfly logo is a good example of a unique element. The <div#apDiv1> is positioned in a specific location on the page, and you can be pretty certain you'll have only one such per page. Let's modify the existing rule for this element to reflect its use in the layout.

1 Select the #apDiv1 rule in the CSS Styles panel. Right-click the selector name and choose Edit Selector from the context menu.

The selector name becomes editable in the panel list.

2 Change the name to **#logo**. Press Enter/Return to complete the editing process.

The rule name changes, but it no longer formats <div#apDiv1>. The layout reflects the default behavior of the unformatted element; without height and width and other key attributes, it expands to the full width of <div.container> and pushes <div.header> down below the height of the butterfly image.

To restore the layout, you have to assign the #logo rule to <div#apDiv1>.

3 Insert the cursor in <div#apDiv1> or click the butterfly to select it. Click the <div#apDiv1> tag selector at the bottom of the document window.

The Property inspector displays the properties of <div#apDiv1>. Note the id displayed in the Property inspector.

4 Choose logo from the ID field pop-up menu.

Reformatted, <div#logo> resumes its former size and positioning.

Classes can be used as many times as you want, but an id is supposed to be used only once per page. Although you could conceivably type one id multiple times manually yourself, Dreamweaver won't offer any help in your attempt at rule breaking. You can demonstrate this functionality with a simple test.

5 Examine the CSS Styles panel and note the available class and id selectors.

There are 10 classes and 2 ids.

6 Insert the cursor in <div#logo>. Click the <div#logo> tag selector.

The Property inspector reflects the formats of <div#logo>.

Note: On a Mac, you may not be able to access the ID pop-up menu if no unassigned ids are available.

7 Open the ID field pop-up menu and examine the available ids.

The only id available is logo. As each id stored in the style sheet is used in your layout, Dreamweaver interactively removes it from the menu.

A apart

An AP div is treated differently from normal elements. You see this difference as soon as you insert an AP div into a document; Dreamweaver creates a rule for it automatically and assigns it attributes for width, height, position, and z-index. This doesn't happen with a normal div. In fact, the special treatment persists even after the rule is created. If you change the id of the AP div in the Property inspector, Dreamweaver will update the name of the rule in the CSS Styles panel at the same time.

However, the reverse is not true. If you change the rule name (as you did in this lesson), Dreamweaver does not change the id on the element itself. The program leaves this chore up to you.

8 Choose green from the Class field menu.

The tag selector displays <div.green#logo>.

9 Insert the cursor in the horizontal menu <div#h-navbar>. Click the <div#h-navbar> tag selector.

10 Open the ID menu and examine the available ids.

The only id available is h-navbar.

11 Open the Class field menu. Examine the available class attributes.

Note that the class green is still available. The Dreamweaver interface allows you to apply the same class to both elements but prevents you from applying either id more than once.

12 Choose None from the Class field menu.

13 In the Property inspector, apply None to <div#logo> using the Class menu.

14 Save the file.

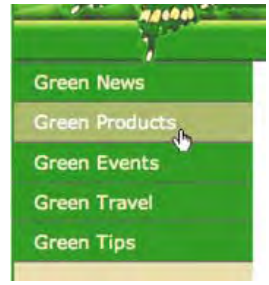
Creating an interactive menu

By combining descendant selectors, classes, and ids together, you can produce amazing behaviors from seemingly static elements.

1 If necessary, switch to Design view and click the Live View button.

The document window will preview the layout as it would appear in a browser. Videos, Flash animation, and JavaScript behaviors will all perform as they would on the Internet.

2 Position the cursor over the vertical navigation menu in the sidebar. Observe the behavior and appearance of the menu items.



As the mouse moves over each button, the cursor icon changes to the hand pointer, indicating that the menu items are formatted as hyperlinks. The buttons also change color momentarily as the mouse passes over each, producing a dramatic graphical experience, too. These effects are all enabled by HTML hyperlink behaviors, formatted by CSS.

Hyperlink pseudoclasses

In all, there are four *states,* or distinct behaviors, available to the <a> element that can be modified by CSS using what are called *pseudoclasses*:

- The `a:link` pseudoclass creates the default display and behavior of the hyperlink. The `a:link` pseudoclass is interchangeable with the "`a`" selector in CSS rules.
- The `a:visited` pseudoclass applies formatting after the link has been visited.
- The `a:hover` pseudoclass applies formatting while the cursor passes over the link.
- The `a:active` pseudoclass formats the link when the mouse is clicked on it.

When used, the pseudoclasses must be declared in this order to be effective. Remember, whether declared in the style sheet or not, each state has a set of default formats and behaviors.

3 Position the mouse cursor over the items in the horizontal navigation menu in `<div#h-navbar>`. Observe the behavior and appearance of the menu items, if any.

The pointer and background color do not change. The items are not formatted as hyperlinks yet.

Note: In most cases, you can't create new CSS rules or format elements in the document while Live view is active.

4 Click the Live View button to return to the normal document display.

5 Select the word *Home* in `<div#h-navbar>`. Do not select the spaces on either side of the word or the vertical bars, or *pipes*, that separate the words.

Correct

Incorrect

6 Type # in the Link field of the Property inspector. Press Enter/Return.

Adding a hash mark # in the Link field creates a hyperlink placeholder and will allow you to create and test the necessary formatting for the horizontal navigation menu without having to create a complete link. Note how the text now displays the formatting of a typical text hyperlink.

7 Add hyperlink placeholders to the items *About Us* and *Contact Us*.

Be sure to select both words in each item before applying the placeholder. If you don't, each word will be treated as separate links instead of as one.

The first step in making the horizontal menu look more like the vertical one is to remove the default hyperlink underscore.

8 Insert the cursor in any of the hyperlinks in `<div#h-navbar>`. Select the `#h-navbar` rule in the CSS Styles panel. Click the New CSS Rule icon.

9 Choose Compound from the Selector Type menu, if necessary.

The Selector Name field displays `.container #h-navbar a.`

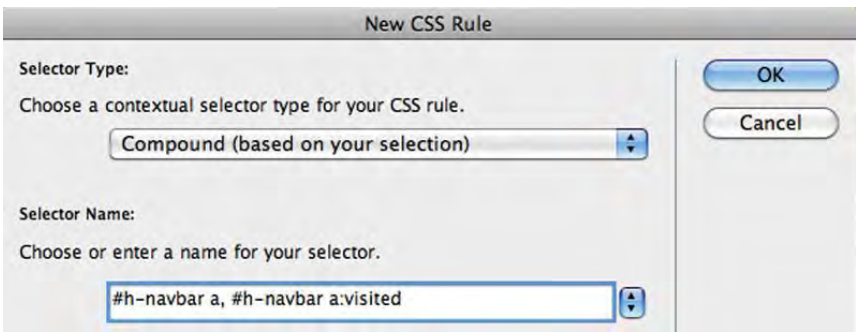
10 In the New CSS Rule dialog box, click the Less Specific button.

The `.container` class is removed from the Selector Name field.

11 Insert the cursor in the Selector Name field. Press Ctrl-A/Cmd-A to select the entire selector name. Press Ctrl-C/Cmd-C to copy the selector.

12 Press the Right Arrow key to move the cursor to the end of the selector text. Type a comma (,) and press the spacebar to insert a space. Press Ctrl-V/Cmd-V to paste the selector from the clipboard. Type **`:visited`** at the end of the pasted selector.

The selector should now appear as `#h-navbar a, #h-navbar a:visited` in the Selector Name field.



13 Click OK.

The new `#h-navbar a, #h-navbar a:visited` rule appears in the CSS Styles panel below the `#h-navbar` rule.

Tip: If you click Apply two times in a row, the CSS Rule definition dialog box will close. If you accidentally close the dialog box before you complete the formatting, simply double-click the rule in the CSS Styles panel to reopen it.

14 In the Type category Text Decoration option, select none. Click Apply.

The underscore is removed from the hyperlinks. Let's make the text easier to read against the green background.

15 Type **#FFC** in the Color field. Click OK.

The color #FFC is easier to see on the green background color. Let's test the hyperlink properties of the items in the horizontal menu.

16 Click the Live View button. Position the cursor over the hyperlink placeholders in the horizontal menu.

The mouse icon changes to the hand pointer, indicating that the text is formatted as a hyperlink, but it has none of the flair of the vertical menu, with its changing background color. That interactive behavior is controlled by the pseudoclass `a:hover`. Let's use this selector to create a similar behavior.

17 Click the Live View button to return to the normal document display. Save all files.

Modifying hyperlink behavior

In this exercise, you will modify the default hyperlink behavior and add interactivity.

Note: The `a:hover` state inherits much of its formatting from a or `a:link`. In most cases, you only need to declare values for formatting that will change when this state is activated.

1 Insert the cursor in any one of the hyperlinks in the horizontal menu. You don't need to select any characters in the link. Select the `#h-navbar a, #h-navbar a:visited` rule in the CSS Styles panel. Click the New CSS Rule icon.

The New CSS Rule dialog box appears with the Compound selector type displayed and the text `.container #h-navbar a` entered in the Selector Name field.

2 Select the Compound selector type, if necessary. Edit the Selector Name to say `#h-navbar a:hover`. Click OK.

The new `#h-navbar a:hover` rule appears in the CSS Styles panel. The "CSS Rule Definition For *#h-navbar a:hover*" dialog box appears.

3 In the Type category, type **#FFC** in the Color field.

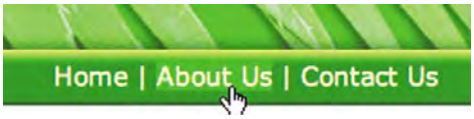
In the Background category Background-color field, type **#0C0**. Click OK.

Note: Adding space to the margins won't work, because margins add space outside the background color.

4 Activate Live view and test the hyperlink behavior in the horizontal menu.

The background behind the hyperlink text changes to light green as the mouse passes over it. This is a good start, but you may notice that the color change doesn't extend to the edges of the `<div>` or even to the pipes dividing one link

from another. You can create a more interesting effect by adding a little padding to the element.



5 Deactivate Live view. Double-click the `#h-navbar a:hover` rule to edit it.

6 In the Box category, enter **5px** in the Padding field, with the Same For All option selected. Click OK.

7 Activate Live view and test the hyperlink behavior in the horizontal menu.

The background color of each link now extends five pixels all around the hyperlink. But there's an unintended consequence: The added padding causes the text on either side of the link to shift whenever the `a:hover` state is activated. The solution to this problem is actually quite simple.

8 In the CSS Styles panel Properties section, select and delete the Padding property for the rule: `#h-navbar a:hover`.

9 Double-click the `#h-navbar a, #h-navbar a:visited` rule to edit it. In the Box category, enter **5px** in the Padding field, with the Same For All option selected. Click OK.

10 Activate Live view and test the hyperlink behavior in the horizontal menu.

When the mouse moves over the links, the background color extends five pixels around the link without shifting. By adding padding to the default state of the hyperlink, the `hover` state inherits the extra padding and allows the background color to work as desired.

11 Save the file.

Congratulations. You've created your own version of the interactive navigation menu in `<div#h-navbar>`. But you may have noticed that the predefined background color selection for the `a:hover` state in the vertical menu doesn't match the site color scheme. To be consistent, the colors used in the site should adhere to the overall theme.

Modifying existing hyperlink behavior

As you gain more experience in web design and working with CSS, identifying design inconsistencies and knowing how to correct them becomes easier. Since you know that the *hover* state is responsible for creating the interactive link behavior, it should be a simple matter to change the background color in the vertical menu. The first step is to assess what rules pertain specifically to the vertical menu itself.

1 Insert the cursor into one of the vertical menu items. Observe the names and the order of elements in the tag selector display.

The vertical menu is using a `.nav` class applied to the `<ul>` (unordered list) element.

2 Locate any rules in the CSS Styles panel that format the `.nav` class. Is there an `a:hover` pseudoclass associated with it?

The CSS Styles panel displays the `ul.nav a:hover, ul.nav a:active, ul.nav a:focus` rule that formats the hyperlink behavior you're looking for.

3 Double-click the `ul.nav a:hover, ul.nav a:active, ul.nav a:focus` rule to edit it. Change the background color to **#0C0**. Click OK.

4 Using Live view, test the behavior of the vertical menu.

The background color of the vertical menu matches the horizontal menu and the site color scheme.

5 Save the file.

Adding visual appeal to menus

Another popular CSS trick that can give menus a bit more visual interest is to vary the border colors. By applying different colors to each border, you can give the buttons a 3-D appearance. As in the previous exercise, you first need to locate the rules formatting the elements.

1 Insert the cursor in one of the menu items and examine the tag selector display.

The menu buttons are built using `<ul.nav>`, `<li>`, and `<a>` elements. Since you know that the `<ul>` element creates the entire list, and not the individual items, you can eliminate it as a suspect.

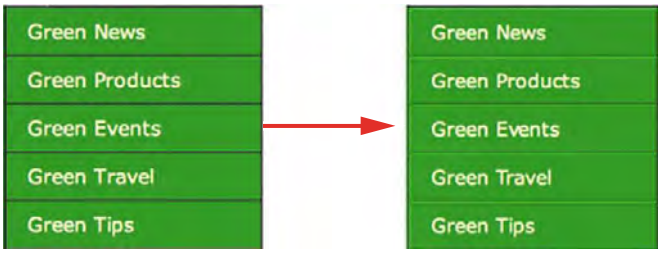
2 Select the `ul.nav li` rule in the CSS Styles panel. Observe the attributes displayed in the Properties section of the panel.

The `ul.nav li` rule formats the basic structure of the menu button.

3 Double-click the `ul.nav li` rule.

4 Select the Border category in "CSS Rule Definition For *ul.nav li.*"
Enter **solid, 1px, #0C0** in the Top border fields.
Enter **solid, 1px, #060** in the Right border fields.
Enter **solid, 1px, #060** in the Bottom border fields.
Enter **solid, 1px, #0C0** in the Left border fields.
Click OK.

By adding lighter colors to the top and left and darker colors to the right and bottom, you have created a subtle but effective three-dimensional effect.



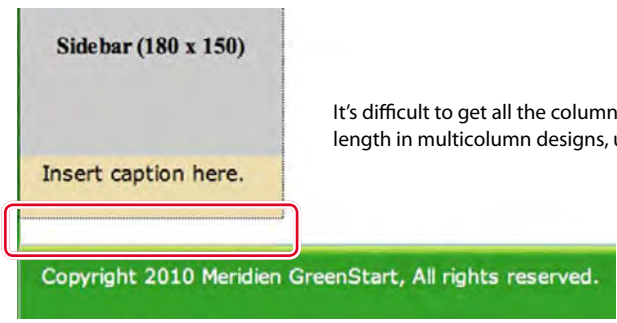
Before After

5 Save the file.

Creating faux columns

Although multicolumn designs are very popular on the web, HTML and CSS have no built-in commands to produce true column structures in a web page. Instead, columnar designs—like the one used in the Dreamweaver CSS layouts—are simulated by using `<div>` elements and various formatting techniques, usually involving margins and the float attribute.

Unfortunately, these methods have their limitations and downsides. For example, one of the problems with the current layout in this lesson is getting both columns to display at the same height. Either one column or the other will almost always be shorter. Since the sidebar has a background color, there will be a visible gap at the bottom as content is added to the main page. There are methods, using JavaScript and other tricks, to force columns to display at equal height, but these are not fully supported by all browsers and could cause your page to break unexpectedly. Many designers sidestep the issue altogether by simply refusing to use background colors.



It's difficult to get all the columns to appear to be the same length in multicolumn designs, using only CSS formatting.

Instead, you will create the effect of a full-height sidebar column by using a background graphic and the repeat function. This technique works well with fixed-width website designs, like this one.

1 Insert the cursor in <div.sidebar1> below the vertical menu. Examine the tag selector display.

The <div.sidebar1> is contained within <div.container> and then, together, in the body element.

2 Select the .sidebar1 rule in the CSS Styles panel. Examine its properties.

The .sidebar1 rule applies a background color to the sidebar. Since the background color assigned to the appears already to be failing to extend to the bottom of the document, the .sidebar1 rule is not the solution to this problem. It's a good idea to remove the background color if it's not producing the desired results.

Note: The Trash Can icon in the CSS Styles panel is context sensitive. It can be used to delete a rule property or the entire rule, depending on how it is invoked.

3 Select the background color reference in the Properties section of the CSS Styles panel. Click the Delete CSS Property ( ) icon at the bottom of the panel.

Now, you'll modify the .container rule to produce a background effect for the sidebar.

4 Double-click the .container rule. In the Background category, click the Browse button. Select **divider.gif** from the default images folder. Click OK/Choose.

5 Choose repeat-y in the Background-repeat field menu. Click OK.

A graphic 180 pixels wide appears at the left edge of <div.container> and extends from the top to the bottom. Since the other elements are contained entirely within <div.container>, the background appears behind them and is visible only where appropriate.

Before After

Let's make one last tweak to the sidebar by removing the extra space that appears between the menu and image placeholder. The first step is to identify the rules that may be creating this styling.

6 Insert the cursor in the last button of the vertical menu. Right-click the button and choose Code Navigator from the context menu.

The Code Navigator appears, displaying a list of 11 CSS rules affecting this item. Chances are a margin setting is producing the spacing effect.

7 Scan the rules for a bottom-margin setting.

The `ul.nav` rule features a bottom-margin of 15 pixels.

8 Select the `ul.nav` rule in the CSS Styles panel. Select the bottom-margin setting in the Properties section. Click the Delete CSS Property ( ) icon.

The gap between the menu and the image placeholder closes up.

9 Save the file.

Moving rules to an external style sheet

When prototyping a web page design, it's more practical to keep the CSS embedded. It makes the process of testing and uploading quick and simple. But, an internal style sheet is limited to one page. An external style sheet can be linked to any number of pages and, for most web applications, is the normal and preferred workflow. Before this page is put into production as a template, it's a good idea to move the CSS styles from the `<head>` section of the document to an external CSS style sheet. Dreamweaver provides the means to handle that task quickly and easily.

1 In the CSS Styles panel, select the first defined style, `body`. Hold the Shift key and select the last style.

2 In the CSS Styles panel, choose Move CSS Rules from the Options menu in the upper-right corner of the panel.

You could also right-click the selected area to access the Move CSS Rules option from the context menu.

Note: The last style in your style sheet may differ from the figure. Remember to select the last one.

3 When the Move To External Style Sheet dialog box appears, choose A New Style Sheet in the Move Rules To options. Click OK.

The Save Style Sheet File As dialog box appears.

4 Navigate to the site root folder, if necessary. Type **mygreen_styles** in the File Name field. Click Save.

Dreamweaver adds the .css extension to the filename, moves the selected styles from the <head> area to the newly defined style sheet, and simultaneously inserts a link to the style sheet. Note at the top of the document window that Dreamweaver now displays the name of the external style sheet in the Related Files interface.

The last chore is to remove the no-longer-needed <style> tag.

5 In the CSS Styles panel, click the <style> entry and press Delete, or click the Trash Can icon.

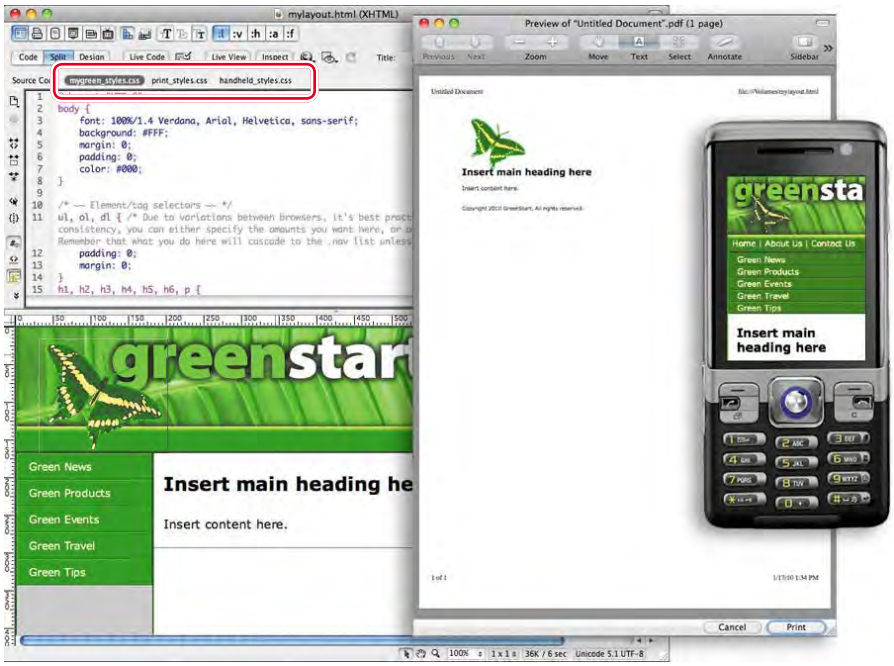
If the reference does not disappear, you can right-click it and choose Delete from the context menu.

6 Choose File > Save All. Or, press Ctrl-Alt-Shift-S/Cmd-Ctrl-S to access the keyboard shortcut for the Save All command you created in Lesson 1, "Customizing Your Workspace."

▶ **Tip:** Once you move the CSS to an external file, remember to use the Save All command moving forward. Pressing Ctrl-S/Cmd-S saves only the top document in the Dreamweaver interface. Other files that are opened and referenced that have been changed are not automatically saved.

Creating style sheets for other media types

Current best practices call for the separation of presentation (CSS) from content (the HTML tags, text, and other page elements). The reason is simple: by separating the formatting, which may only be relevant for one type of medium, one HTML document can be formatted instantly for multiple purposes. More than one style sheet can be linked to a page. By creating and attaching style sheets optimized for other media, the specific browsing application can select the appropriate style sheet and formatting for its own needs. For example, the style sheet created and applied in the previous exercises was designed for a typical computer screen display.



In this exercise, you'll convert a CSS screen-media file to one optimized for print. This is a popular technique for many websites today. Designers frequently include a "Print" link on pages heavy with text so that users can send the information to the printer more effectively. Print style sheets often adjust colors to work better in print, hide unneeded page elements, or adjust sizes and layouts to be more suitable for printing.

When the print queue is activated, the printing application checks for a print-media style sheet. If one is present, the relevant CSS rules are taken into account. If not, the printer defers to the rules in the screen or all-media style sheets or the CSS defaults for print.

Displaying the Style Rendering toolbar

Dreamweaver's default display medium is screen media. However, Dreamweaver has the capacity to switch how the media is rendered in Design view using the Style Rendering toolbar. With Design view's rendering of the rules in a print style sheet, you can see the effect your style rules will have on a printed page.

1 If necessary, open the **mylayout.html** file by double-clicking its filename in the Files panel.

2 Choose View > Toolbars > Style Rendering.

The Style Rendering toolbar appears above the document window. Leave it visible for the next exercise.

A Render screen media type
B Render print media type
C Render handheld media type
D Render projection media type
E Render TTY media type
F Toggle displaying of CSS styles
G Render TV media type
H Design-time style sheets
I Increase text size
J Reset text size
K Decrease text size
L Show styles for :link pseudoclass
M Show styles for :visited pseudoclass
N Show styles for :hover pseudoclass
O Show styles for :active pseudoclass
P Show styles for :focus pseudoclass

Converting an existing style sheet for print

Although you can develop a print style sheet from scratch, it's usually much faster to convert an existing screen-media style sheet. The first step is to save the existing external style sheet under a new name.

1 In the Files panel, double-click **mygreen_styles.css** to open it.

2 Choose File > Save As.

3 When the Save As dialog box opens, type **print_styles.css** in the File Name/Save As field. Make sure the site root folder is targeted. Click Save.

4 If necessary, open **mylayout.html** from the site root folder. In the CSS Styles panel, click the Attach Style Sheet ( ) icon.

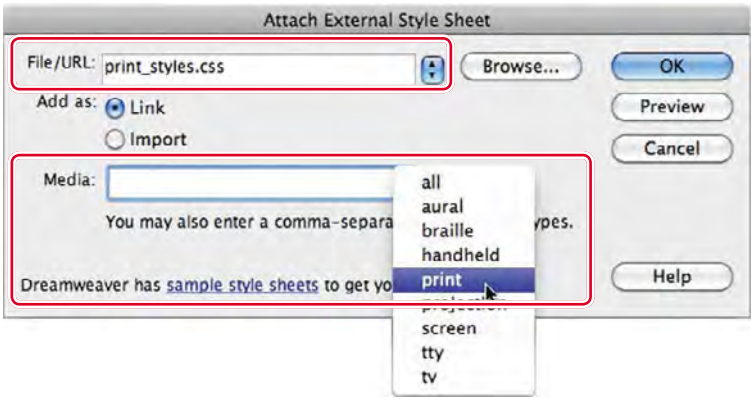
The Attach External Style Sheet dialog box opens

5 Click Browse.

The Select Style Sheet File dialog box appears.

6 Select **print_styles.css** from the site root folder. Click OK/Choose.

7 In the Attach External Style Sheet dialog box, select the Link option for the Add As value. From the Media field menu, choose print. Click OK.



8 In the CSS Styles panel, click the All button, if necessary.

A new entry—**print_styles.css**—has been added. At the moment, both style sheets are identical. You will modify the print style sheet in the next exercise.

9 Close **print_styles.css** and **mygreen_styles.css**.

10 Save **mylayout.html**.

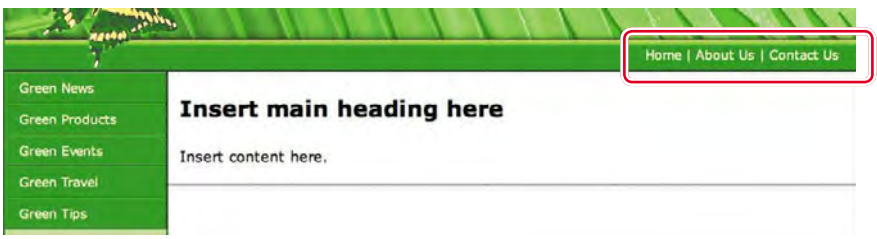
Hiding unwanted page areas

Using the Style Rendering toolbar, you can see your document rendered according to the print-media style rules.

1 In the Style Rendering toolbar, click the Render Print Media Type ( ) icon.

One of the main differences between screen and print media is that interactive items on a web page are often meaningless in print. This would include all navigation elements in the horizontal and vertical menus. Using the print-media style sheet, you can hide unwanted portions of a page. For example, the horizontal menu isn't needed for printing.

Note: Remember to click the Render Screen Media Type icon when you're ready to work on screen-media formatting again.



2 In the CSS Styles panel, double-click `#h-navbar` in the **print_styles.css** rules.

3 In the Block category, choose none from the Display field menu. Click OK.

The entire `<div#h-navbar>` disappears from the document window. The `<div>` has not been deleted; Dreamweaver has simply stopped rendering it temporarily as long as the Print Media Type icon is selected. Let's turn off `<div.sidebar1>`, too.

4 In the **print_styles.css** rules, double-click `.sidebar1`.

5 In the Block category, choose none from the Display field menu. Click OK.

The sidebar vanishes, and the main content expands to the full width of the `<div.container>`. The background image is visible under the content and may make it harder to read the text.

6 In the **print_styles.css** rules, double-click `.container`.

7 In the Background category, delete the **divider.gif** image reference in the Background-image field. Delete repeat-y from the Background-repeat field. Click Apply.

The background image continues to display in `<div.container>`. Deleting the image reference is not enough. Although the print application defers to the print-media style sheet, formatting is still inherited from all referenced CSS style sheets. Even though you deleted the background image reference in the print style sheet, it's still applied in the screen styles. It won't disappear until you reset the rule by choosing none. This goes for other such rules, too.

8 Choose none from the Background-image field menu. Click Apply.

The background image vanishes. Let's check the page in Live view.

9 Click OK to complete the change. Click the Live View button.

Dreamweaver ignores the print-media styles and renders the image as if for the screen. Live view is intended for browser preview and can't render print-based styling. To properly test the page, you have to use the print preview function in an actual browser.

10 Save all files. Choose File > Preview In Browser and select your preferred browser.

11 In the browser, choose File > Print Preview.

As you can see, the print application converted the text to black and automatically dropped all the background images and colors, but it still prints the butterfly logo and the borders. Let's eliminate the border.

12 In the **print_styles.css** rules, double-click `.container`. In the Border category, select the Same For All option in the Style section. Choose none from the Top Style field. Click OK.

13 Save all files.

14 Choose File > Preview in Browser and select your preferred browser.

15 In the browser, activate print preview.

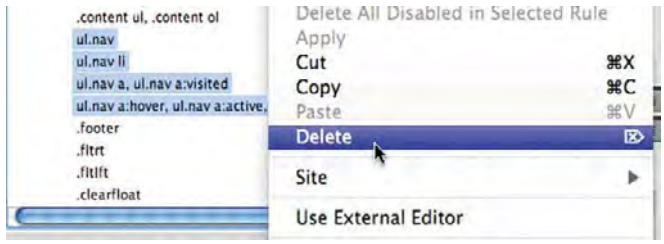
The border has been removed successfully. You should know enough now to keep the butterfly logo from printing, too. Take a few minutes here and see if you can do it.

Note: Some browsers have a different technique to access the print preview mode. You may need to choose File > Print first and access the preview function from within the print dialog box.

Removing unneeded styles

Even with the changes in the previous exercises, many rules in the two style sheets are exactly the same. To reduce file size, it's a good idea to remove rules from the print-media sheet that haven't changed or ones that don't pertain anymore. You can delete unneeded styles using the CSS Styles panel. But be careful—even though a rule hasn't changed doesn't mean it's not needed for print rendering.

1 Select all rules that format the `ul.nav` menu in **print_styles.css**. Click the Delete CSS Rule icon or right-click the selected rules and choose Delete from the context menu.



Since the vertical menu isn't displayed, there's no need for those rules. In fact, you can remove all rules that format hyperlink behavior, too.

2 Select all hyperlink rules in **print_styles.css** and delete them.

The hyperlinks in the horizontal and vertical menus are not printing at all, and the other rules are still identical to the ones in the screen styles and will be inherited, if supported by the print application. After deleting any rules, always test the page in the browser and in the print application.

3 Save all files.

4 Click the Render Screen Media Type ( ) icon. Observe the screen display in Design view.

Dreamweaver renders the document for the web.

5 Click the Render For Print Media Type ( ) icon.

Dreamweaver renders the screen using the print style sheet. You have adapted a screen-media style sheet to make a web page render more useful in print.

You have completed the basic design of the page that will be used as the project template and adapted it to print media. In the next lesson, you will learn how to convert this layout into a dynamic web template.