# Lesson 3: Cascading Style Sheets (CSS) and Graphical Elements

## *Objectives*

By the end of this lesson, you will be able to:

- 2.2.1: Incorporate graphical images into HTML pages.
- 2.2.2: Distinguish among and identify the uses and benefits of various graphic file formats, including GIF, GIF89a, JPEG, PNG, TIFF, BMP.
- 2.2.3: Add tiled images and colors to Web page backgrounds.
- 2.2.6: Insert horizontal rules into Web pages.
- 2.3.1: Define the browser-safe color palette.
- 2.3.2: Identify ways that color affects the principles of line, value, shape and form in Web pages.
- 2.3.3: Identify and demonstrate the impact of color combinations to various audiences and cultures.
- 2.3.4: Evaluate Web page design and layout.
- 2.8.1: Explain how to structure Web documents with CSS.
- 2.8.2: Identify ways to apply styles with CSS.
- 2.21.1: Use CSS and HTML5 elements to create document structure.
- 2.21.2: Distinguish between fixed-width and liquid design layouts.

# Pre-Assessment Questions

1. Name the three standard image file formats supported across the Web.

2. Define hexadecimal color values. Why would you use a hexadecimal value instead of the name of a color?

3. Which property determines the image that will be tiled behind the contents of a page?

   a. The background-image property of external CSS
   b. The background attribute of the <body> tag
   c. The background-image property of the <body> tag
   d. The img property of the <background> tag

 

# Cascading Style Sheets (CSS)

You have already learned that a Cascading Style Sheets (CSS) document is an external text file that determines how to display the look and feel of HTML elements in your Web pages. It contains formatting instructions that can define the font, color and phrase elements used on a particular markup page.

If all pages on your site are linked to the same external style sheet, then one simple change to the style sheet will change all elements across the site. If you then want to change those instructions (for example, the style of <h1> headings), you need not change every page manually. You need only change a line in the style sheet file, then all your <h1> headings will change their appearance to conform to the style sheet. This technology can save a great deal of development and maintenance time, as well as make a more consistent, accessible interface.

**CIW Online Resources – Movie Clips**

Visit CIW Online at http://education.Certification-Partners.com/CIW to watch a movie clip about this topic.

*Lesson 3: Cascading Style Sheets (CSS) and Graphical Elements*

## The problem with HTML and styles

CSS was created to solve a problem with HTML. The developers of HTML intended the standard to identify only the content and structure of a document, such as <title> and <body>, not the style, look and feel of the content.

When style elements for HTML4 were introduced without style sheets, the HTML standard grew complicated very quickly. Developers had to check each page of a Web site to ensure it had the correct style tags for each element. Imagine the time and difficultly involved with this task.

To save developers from insanity, the W3C quickly went into action and standardized CSS. It greatly simplifies the application of style to Web sites. Today a background image can be added to all Web pages with one change to an external CSS file.

## Benefits of using CSS

CSS benefits include:

- **Consistency** — CSS easily gives an entire site a consistent look and feel.
- **Easy change management** — You have the ability to change the look and feel of an entire site by simply changing one part of a single line of code, rather than having to change possibly thousands of lines in hundreds of Web pages.

## CSS versions

Currently, three standards exist for style sheets:

- **Cascading Style Sheets (CSS1)** — governs the basic structure of style sheets.
- **Cascading Style Sheets 2 (CSS2)** — adds more capabilities to the CSS1 specification, including the ability to support media types (such as specific printers) and work with tables.

 

**NOTE:** Remember that HTML5, CSS3 and JavaScript are considered the future of Web development as mobile devices continue to proliferate.

- **Cascading Style Sheets 3 (CSS3)** — provides a modularized standard so that when changes need to be made to a specification, only a particular module within CSS3 will need to be updated, rather than the entire standard. This will allow for a more flexible and timely upgrade of the standard as a whole. New functions are being added to CSS3 to enhance its support of borders, backgrounds, colors, text effects and so forth.

**OBJECTIVE** 2.8.2: Applying CSS styles

HTML5 adopts CSS as the preferred way to format a page. The standard has evolved over the years and continues to build upon itself. For instance, HTML 4.01 and XHTML 1.0 used CSS1 and CSS2. HTML5 uses CSS1, CSS2 and CSS3. Any given Web page will include CSS rules from all three versions.

Because CSS3 contains the latest instructions, most non-HTML5-compliant browsers cannot interpret it. Most of the basic Web page formatting uses CSS1 and CSS2. When you validate your CSS pages throughout this introductory course, the CSS code will usually validate as CSS2.


*Proper use of style sheets is foundational for creating pages that rank highly in search engines.*

## CSS terminology

**NOTE:** Be sure you do not confuse CSS terms, syntax and declaration with HTML code. Although used together, CSS and HTML are separate technologies.

Before you deploy CSS, you should learn its terminology. The most important terms to understand are selector, property, value, declaration and rule. The following code from a CSS file illustrates the anatomy of these style sheet elements as they are found in linked, or external, style sheets. Inline CSS styles are declared differently, as you learned in the previous lesson.

```
body {background-color:lightblue;}
```

The syntax terms of this CSS code are listed in Table 3-1.

*Table 3-1: CSS syntax terms*

| CSS Syntax | Term |
|---|---|
| *body* | Selector |
| *{* | Opening curly brace |
| *background-color:* | Property |
| *lightblue;* | Value |
| *}* | Closing curly brace |

**selector** In a style sheet, any element to which designated styles are applied.

This CSS file entry creates a light blue background color for the Web site. To make such a change, you must identify the element you want to apply the style in. Any element you want to affect is called a **selector**.

 

**NOTE:**
Selectors are most easily used with standard elements. They can also be applied based on basic patterns. For example, you can declare a selector that will apply to paragraphs <p> that fall within tables. Or you can group selectors to apply the same styling information to multiple elements with a single declaration.

After you have chosen a selector, you can customize it by selecting a property and setting a value. By selecting a property, you will change the way the selector renders in the browser. For example, you could alter the selector's color, size, background, font family, font size and so forth. These changes will apply to all subsequent instances of the element you define as a selector.


*In CSS, values do not require quotation marks. Also, CSS syntax should be written in lowercase.*

A property must then have a value. For example, if you want to change a selector's size property, you must set a value to specify that size. Or you could decide to change the background color of your pages to teal, and so forth.

In the following example, you will see a CSS declaration, which consists of a property and a value. A declaration must always end with a semicolon.

```
background-color:teal;
```

**rule**
In a style sheet, a format instruction that consists of a specified selector and the properties and values applied to it.

The name for a selector, property and value all grouped together is a **rule**. The following rule will change the color of the body background to teal, then set the font color to white:

```
body {background-color:teal;color:white;}
```

CSS declarations and rules are reviewed in Table 3-2.

*Table 3-2: CSS declarations and terms*

| CSS Syntax | Term |
|---|---|
| `background-color:teal;` | Declaration |
| `body {background-color:teal;color:white;}` | Rule |

To define multiple declarations for one selector within a rule, as in the previous example, you must separate each declaration with a semicolon. You must also place a semicolon between the last value of the declaration and the end bracket.

Notice from these examples that the properties and the values must be placed within curly braces. This practice is standard for all style sheet rules, except when declaring inline styles. You will learn more about different ways to define styles shortly.


## CIW Online Resources – Online Exercise

Visit CIW Online at *http://education.Certification-Partners.com/CIW* to complete an interactive exercise that will reinforce what you have learned about this topic.

*Exercise 3-1: Introduction to CSS*

 

## Proper CSS structure

Following is the accepted structure of rules within a style sheet:

```
body
{
font-family:arial, verdana, helvetica;
color:gray;
font-size:14px;
}

h1
{
font-family:arial, sans-serif;
color:black;
font-size:36px;
}
```

Notice that the opening and closing curly braces and each declaration are placed on separate lines. This spacing is considered good coding practice because it makes the CSS file easier to read.

If you want a particular rule in a style sheet to be ignored, you can "comment out" the entry by placing it in between the /* and */ characters. You can also add comments to explain the rules to others. In the following example, the words "STYLE SHEET FOR SYB.HTML" or "A comment is added here" will not be read:

```
/* STYLE SHEET FOR SYB.HTML */
/* A comment is added here */
```

## Inheritance

The concept of inheritance is essential to CSS. In fact, the word "cascading" refers to inheritance. The style you define will flow, or cascade, throughout the documents, unless another style defined inside of a page specifically overrides it. Many styles can be used together to create a completely formatted document. For example, a style sheet rule will override the default <body> font color, which is black. All these characteristics, whether they are defined in a style sheet or exist by default, are inherited throughout the rest of the document.


*As you learned in the previous lesson, styles specified using the inline CSS style attribute within an HTML page will override external CSS entries.*

## Adding CSS to HTML

**NOTE:** Although CSS3 is the most current recommendation from the W3C, the support provided by most browsers is still inconsistent.

Now that you understand basic CSS terminology and syntax, you will learn how to implement it in an HTML document. You can apply CSS styles to HTML documents in several ways. You can:

- Declare an inline CSS style attribute.
- Link to an external style sheet.
- Create an internal style sheet.

In the previous lesson, you worked with inline CSS style attributes. In this lesson, you will learn how to link to and modify an external style sheet.

 


*This lesson will not cover importing style sheets. Certain older browsers may crash when rendering pages with an import statement. If you want to use an external style sheet, use a linked style sheet rather than an imported style sheet.*

**OBJECTIVE**
2.8.2: Applying CSS styles

In the following lab, you will use an external style sheet. Suppose your project manager reviews some of your pages and requests that you eliminate the background color from all pages. You could perform this task quickly and simply by editing the style sheet.

## Lab 3-1: Using an external style sheet with HTML

**NOTE:**
In this lab, you will first add the style sheet link, then delete the style="*background-color:white*" attribute. Remember the rule that inline CSS overrides external CSS and that is why it must be deleted.

In this lab, you will attach an external style sheet to an HTML document, then edit the style sheet's background color declaration.

1. **Windows Explorer:** Navigate to the **Habitat\CCYP** folder on your Desktop and create a subfolder named ***syb***. If you have not created a Habitat\CCYP\ directory, please do so now. The new folder (Habitat\CCYP\syb\) will contain the style sheet as well as all images and other files for the syb.html page.

2. **Windows Explorer:** Navigate to **C:\CIW\Site_Dev\Lab Files\Lesson03\Lab_3-1**. Copy the file **syb.css** to the Habitat\CCYP\syb\ folder on your system. If you just created the Habitat\CCYP\ directory, then copy the lab file **syb.html** to the Habitat\CCYP folder.

   *Note: Remember that you are going to place all files associated with your syb.html document in a specific subfolder. In this case, files will be in the syb subfolder, including the CSS file.*

3. **Editor:** Open the **Habitat\CCYP\syb.html** file, and enter the code shown in **bold** to the <head> element. Also delete the inline CSS style attribute and value shown in ~~**strikethrough**~~ from the <body> element.

```
<!DOCTYPE html>

<html>

<head>
<meta name="keywords" content="CIW, HTML5, Habitat for Humanity"/>
<meta name="description" content="Simple XHTML page for Habitat site"/>
<meta charset="UTF-8"/>
<title>Habitat for Humanity International Summer Youth Blitz Program</title>
<link rel="stylesheet" type="text/css" href="syb/syb.css"/>
</head>

<body ~~style="background-color:white"~~>
```

   *Note: Make sure you delete the inline CSS style attribute (style="background-color:white") within the <body> tag. Do not delete the actual <body> tag or the closing wicket. The attribute must be deleted because inline CSS overrides external CSS. Also, please note that only the beginning of the HTML document is shown in this lab step to save space.*

4. **Editor:** Save your changes.

5. Open **syb.html** in your Web browser. You will not notice any changes because the style entries in the CSS file are currently commented out using the syntax /* */.

 

6. Open the file **syb.css** in Notepad. Remove the comments from the line that begins with *body*. Make sure to remove both the beginning /* and ending */ characters of the comment. When you are finished, save your changes. Keep **syb.css** open.

7. Before you close syb.css, review the entry. Note especially the *background-color:tan* entry.

8. After you have removed the comment from the body entry and reviewed all entries, close **syb.css**, making sure that you have saved your changes.

**serif**
A font style that uses characters with small decorative additions at the outermost points of the characters, called strokes. Includes the Times and Times New Roman fonts.

**sans-serif**
A font style that does not use decorative strokes at the tips of characters. Includes the Arial font family.

9. Refresh your browser's display. Notice that the page's background color has changed to tan. Also, notice that the font type (i.e., the font face) has changed. Instead of the standard **serif** font most Web browsers use, you now see the **sans-serif** Arial font.

10. Open **syb.css** again and change the *body* entry so that it reads ***background-color:white***, then save and close the file.

11. Refresh your browser's display. What color is the background?

12. Visit the W3C CSS Validation Service at **http://jigsaw.w3.org/css-validator** and validate your CSS file.

13. Close any **Notepad** windows.

In this lab, you used an external style sheet.

### CIW Online Resources – Movie Clips

Visit CIW Online at http://education.Certification-Partners.com/CIW to watch a movie clip about this topic.

*Lesson 3: Creating CSS*

# Separating Content in HTML

Several simple graphical elements can be added to a Web page to provide structure and visual interest. One such element is the horizontal rule, or <hr> element, which is simple to add. The <hr> element is used to define a thematic break in a Web page, such as a topic change, or to separate content. It creates a horizontal line that separates one part of the Web page from another.

To add a horizontal rule to your page, insert an <hr> tag at the position where you want the line to appear. In HTML, <hr> is an empty tag that should be written as <hr/>. This tag format is called a "stand-alone non-empty tag." This tag is written similarly to the break, or <br/> tag, that you learned about in the previous lesson.

**OBJECTIVE**
2.2.3: Background images and colors

2.2.6: Horizontal rules

Consider the following code:

```
<h1>Horizontal Rules</h1>
<hr/>
Horizontal rules: Lines used to make visual divisions in your document.
```

This code will render in a browser as shown in Figure 3-1.

 

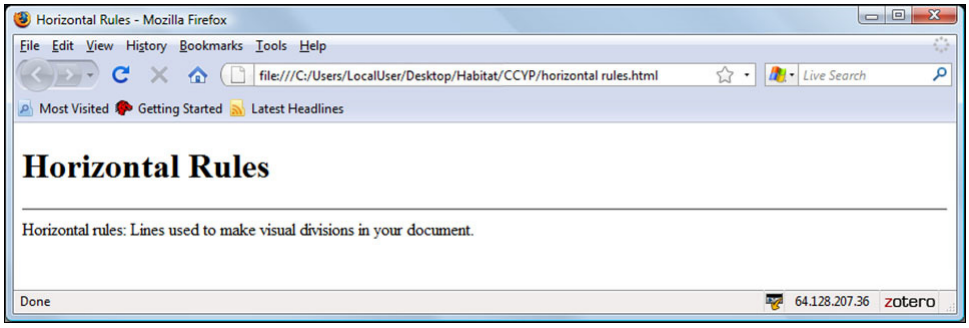


*Figure 3-1: Page displaying horizontal rule*

By default, these lines include a 3-D shading effect, which can be removed. In addition, the lines can be set to various sizes and widths. Because the line is added by a single tag, <hr>, how is this other information passed to the browser?

HTML5 does not support layout attributes for the <hr> tag. Any styles must be applied using CSS. Common styles added to the <hr> element are alignment, size and width.

## Using CSS to stylize horizontal rules

**NOTE:**
The *width* attribute or property value can also be defined in pixels. Pixels allow the width to remain the same, regardless of the size of the visitor's browser window.

In the previous lesson, you learned about the three components of a tag: an element, attributes and values. You also learned about the inline CSS *style* attribute's properties and values. The *style* attribute is a universal attribute that can be added to any HTML element.


*Adding inline CSS style attributes to HTML elements is time-consuming. It is much more practical to style HTML elements using an external CSS file. You are learning inline CSS style attributes to introduce you to CSS. As you progress through the course, you will work with external CSS files more often.*

The *style* attribute has a *width* property that can control how far the line extends across the screen. By default, the value of the *width* value is 100 percent. Thus the line in the preceding figure extends from the left margin across to the right margin in the browser window, or the entire width of the window. If you want the line to extend across only 50 percent of the window, you would write the tag as follows:

```
<hr style="width=50%;"/>
```

Note that the property name (in this case, *width*) precedes an equal sign (=). Following the equal sign is the desired value for this property, in this case "50%". The property and value must always be enclosed in quotation marks.

You can use the *height* attribute with the style attribute as well. For example, to create a line 10 pixels high spanning halfway across the page, you could use either of the following syntax options:

```
<hr style="width=50%; height=10px;"/>
<hr style="height=10px; width=50%;"/>
```

The order in which the properties appear is not dictated. However, you cannot reverse the properties and value; the property must always precede the equal sign, and the properties and values must always be enclosed in quotes and separated by semicolons.

In the following lab, you will add horizontal rules to a Web page. Suppose your project manager suggests that the syb.html page could use a visual separation or graphic of some sort between the page content and the contact information at the bottom. You can add a horizontal rule to serve this purpose.

 

## Lab 3-2: Assigning inline CSS attribute values to the <hr> tag in HTML

In this lab, you will learn how to assign and change the values of inline CSS attributes in the <hr> tag.

**NOTE:**
You should understand how graphical elements can occupy a smaller percentage of their default value, which is 100 percent.

1. **Editor:** Open the version of **syb.html** you edited in the previous lab.
2. **Editor:** Enter the following code, just above the "*For more information . . .* " line near the bottom of the page:

   ```
   <hr/>
   ```

3. **Editor:** Save your changes.
4. **Browser:** Load **syb.html**. You will see a horizontal line appear in the document.
5. Edit the <hr> tag you have inserted so that the line spans 80 percent of the Web page's width and is 5 pixels in height. The line is centered by default. Enter:

   ```
   <hr style="width:80%; height:5px;"/>
   ```

   When you are finished, a horizontal line will appear, as shown in Figure 3-2.

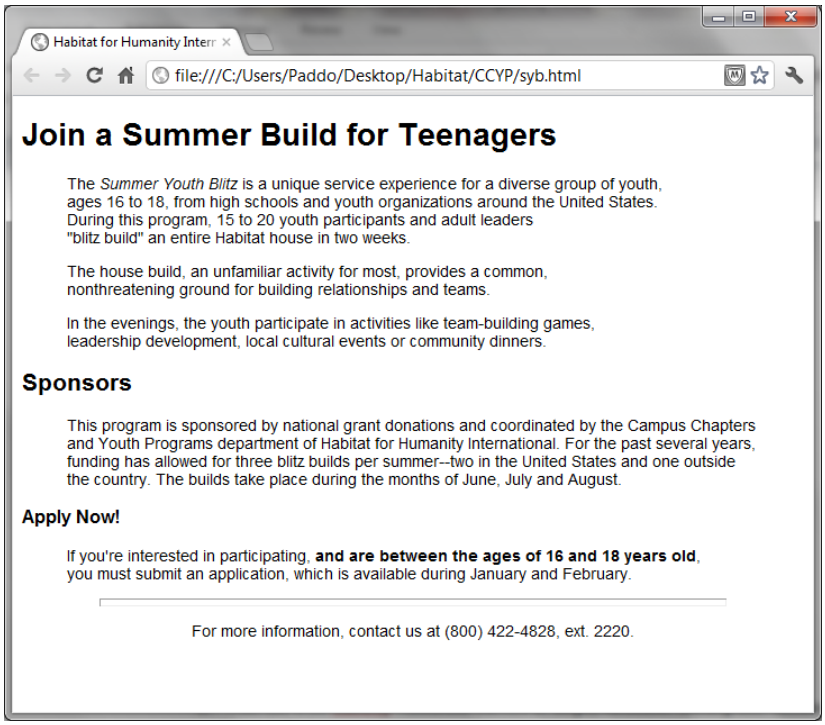


*Figure 3-2: Customizing <hr/> element with inline CSS style attributes*

6. Visit the W3C Markup Validation Service at ***http://validator.w3.org*** and validate your code.

In this lab, you inserted and customized a horizontal line in your HTML page.

 

Many Web authors choose to insert images and bars instead of using HTML horizontal lines. You may have a reason to choose the <hr> element in the future because it may designate some contextual meaning or introduce a thematic change in your Web page.

**OBJECTIVE**
2.2.1: Images

2.2.3: Background images and colors

# Images in Web Pages

You may have heard the saying that a picture is worth a thousand words. Most Web pages incorporate graphical images in their designs. Images add interest to a page, but they provide more than just aesthetics. Images are memorable and can be used to create a mood, emphasize a point or sell a product.

Images can be big or small; they can function as links; they can be used to launch script actions; and they can be used as image maps. Although scripting is not discussed in detail in this course, you will be introduced to clickable image maps in a later lesson. In this lesson, you will focus on using images purely as graphical enhancements.


*When chosen carefully, images can greatly enhance your pages. However, too many images can slow page loading, waste costly online time, and even displease users by making the page look too cluttered. Be creative but sparing in your use of images.*

The <img> tag displays a graphical image on your page. The key attributes that are required in this tag are *src* (abbreviation for source) and *alt* (alternative text for the image if it cannot be seen). You use the *src* attribute to specify the name and, if necessary, the location of your image file.

The <img> tag is an empty tag but should be treated as a stand-alone non-empty tag:

```
<img src="imagefile.png" alt="Alternative text goes here"/>
```

For the <img> tag to validate as HTML, it must include the *src* and *alt* attributes. You will learn more about the *alt* attribute shortly.


*If you upload your Web pages to a Web server and the images do not appear, check the <img src=" "> value of each image. If you created all of your images in a separate directory, make sure you uploaded that directory as well.*


## CIW Online Resources – Movie Clips

Visit CIW Online at http://education.Certification-Partners.com/CIW to watch a movie clip about this topic.

*Lesson 3: Inserting a Graphic into an HTML5 Page*

## Image file formats

**OBJECTIVE**
2.2.2: Graphic file formats

The three universally supported Web image formats are:

- Graphics Interchange Format (GIF).
- Joint Photographic Experts Group (JPEG).
- Portable Network Graphics (PNG).

Windows Internet Explorer and Mozilla Firefox also support the display of Windows Bitmap (BMP) images when used in HTML documents.

 

### JPEG

**NOTE:** Note the differences between GIF images and JPEG images.

The Joint Photographic Experts Group (JPEG) format supports up to 24 bits of color information (16.7 million colors), and is typically used for photographs and complex images.

This format also supports compression, meaning that you can reduce the image's file size. However, the more an image is compressed, the more its quality is reduced. For this reason, standard JPEG image compression is called "lossy" compression. JPEG compression is copyrighted in many countries. As a result, applications that use JPEG and other copyrighted materials may cost more or have limitations placed on them.

To learn more about the JPEG format, visit The JPEG Committee home page (*www.jpeg.org*) and the Independent JPEG Group home page (*www.ijg.org*).

### GIF

Graphics Interchange Format (GIF) files support 256 colors, rather than the millions of colors available to JPEG images. GIFs are best suited for line art, custom drawings and navigational images. GIF has two versions:

- GIF 87a
- GIF 89a

GIF 89a is more popular because it supports the following techniques:

- **Transparency** — the ability to make any part of the image invisible so the page background shows through. The image thus appears to blend into the background.
- **Interlacing** — the ability for an image to render gradually as it downloads.
- **Animation** — a series of images appearing in sequence to create the effect of motion.

GIF and its compression format are also copyrighted in many countries, making it somewhat controversial. You will learn more about animation, transparency and interlacing later in the course.

### PNG

**NOTE:** If you will be creating Web pages professionally, you may want to invest in an image-editing program, such as Adobe Photoshop. Photoshop is an advanced graphics application that can create almost any image the developer conceives.

Portable Network Graphics (PNG) has emerged as a standard format for images on the Web and is widely implemented. PNG bit depth can be adjusted, unlike GIF and JPEG, which must be 8-bit and 24-bit depth, respectively. PNGs also use compression filters that can support up to 48-bit color. The PNG format was developed using open standards, which means that it does not have the same legal liabilities as other formats (e.g., GIF).

PNG brings together the best features of the GIF and JPEG formats into one format. PNG images provide the following features:

- **Transparency** — similar to GIF 89a.
- **Interlacing** — similar to GIF 89a.
- **Compression** — lossless, unlike standard JPEG compression. Also, the compression used in the PNG format is not copyrighted, which has helped ensure developer and user agent vendor support.
- **Animation** — less popular than animated GIF, but gaining attention.

However, some older browsers offer incomplete support for the PNG format. Internet Explorer version 6 had incomplete support for transparent PNGs, and Internet Explorer 7

 

and 8 are unable to correctly display PNG images with color correction. PNG-compatible browsers include Apple Safari, Google Chrome, Mozilla Firefox and Opera, with partial support available in the various versions of Internet Explorer.

You can learn more about the PNG format at LibPNG.org (*www.libpng.org*), the free reference library for PNG images. Table 3-3 provides a summary of the features provided by the three common image formats.

*Table 3-3: Features common to major image file formats*

| Format | Transparency | Interlacing | Compression | Animation |
|---|---|---|---|---|
| **GIF 89a** | Yes | Yes | Yes | Yes |
| **JPEG (standard)** | No | No | Yes | No |
| **PNG** | Yes | Yes | Yes | Yes |


*If you spend time browsing images on the Internet, you may be tempted to use graphics created by others in your Web pages. Be aware that any content — text, sound files or images — is the sole property of the original owner. You may be subject to penalties under copyright laws if you use someone else's creation without express, written permission.*

### *WebP*

**NOTE:**
In early 2013, Google started using the WebP format in its Google+ app for Android. Since then, Google has documented a large decrease in bandwidth usage, credited to the smaller file sizes of WebP lossless images (compared to PNGs) and WebP lossy images (compared to JPEGs).

A new image format called WebP was introduced in 2010. The WebP image format was developed by Google and is supported natively in the Chrome browser. As of this writing, WebP is also supported in the Opera browser, but not in Internet Explorer, Safari or Firefox.

The WebP format offers image capabilities for:

- Animation.
- Image transparency with lossless compression.
- Image transparency with lossy compression.

Thus, WebP combines the transparency support that the PNG format offers with the smaller file sizes afforded by JPEG. Smaller file sizes mean that browsers can load Web pages faster and that not as much data must be transferred over the network, thereby reducing bandwidth usage.

As of this writing, WebP does not support interlacing, layers or high color depth. However, these features are being considered for future versions of the format.

To learn more about the WebP image format, visit the Google Developer's WebP page (*https://developers.google.com/speed/webp/*).

## Using the *alt* attribute with images

**NOTE:**
Make sure you understand that you must use the *alt* attribute in order for documents to render in HTML5. This tag can be used by text-based browsers and other applications to provide information to those who cannot see images.

Every image used in HTML5 is required to contain the *alt* attribute with a corresponding value. The *alt* attribute specifies alternative text to appear while the graphic is loading, or in place of the graphic in non-graphical browsers such as Lynx. This alternative text will also display if the image fails to load or if the user has configured his or her browser not to display images.

The syntax for using the *alt* attribute is as follows:

```
<img src="image.png" alt="This text should describe the image."/>
```

 

The tag is closed using the HTML non-empty tag format. Any other use of the <img> tag will not validate as HTML.


*Search engines will rank a page higher in a search engine results page if it consistently uses the alt attribute effectively. Remember to include a short but useful description of every image.*

## Combining background images and background colors

You can specify both an image and a color for the background in a Web page. In fact, it can be advantageous to specify both, in case a background image becomes unavailable for some reason.

If you use a style sheet and specify both image and color as a background, then the background image will always render first. If the image cannot be found, a background color will then appear. All values specified in style sheets will override anything specified in the HTML itself.

In the following lab, you will incorporate images in a Web page. Suppose the marketing team and another member of your Web development team have collaborated to create an image for your Web site. You have been asked to insert the image into a page you have developed, and you have been given the same image in the three standard file formats. You must choose an image format and insert it in the appropriate location on the page.

### Lab 3-3: Incorporating images in an HTML page

In this lab, you will learn how to place and align an image relative to text in a Web page.

**NOTE:**
Spend as much time as possible inserting and aligning images.

1. Navigate to the **C:\CIW\Site_Dev\Lab Files\Lesson03\Lab_3-3** directory, and copy the following files to the **Habitat\CCYP\Syb** directory:

   **SYBcollage2.gif**
   **SYBcollage2.jpg**
   **SYBcollage2.png**

2. Open the **Habitat\CCYP\Syb** directory, and view each image by double-clicking on it. See if you notice any difference among the three image formats.

3. Right-click each image and select **Properties**. What is the file size of each image type (GIF, JPEG and PNG)?

**NOTE:**
The file sizes should be approximately as follows:
-SYBcollage2.gif: *51/52 Kb*
-SYBcollage2.jpg: *25/28 Kb*
-SYBcollage2.png: *111/112Kb*

4. As a class, discuss the following points about these image formats:

   - The larger the image file size, the longer it takes users to download.
   - Older browsers may not be able to render PNG images.
   - GIF and PNG images can be interlaced, animated or made transparent. Standard JPEG images do not support these features.
   - JPEG images offer the highest image quality.
   - PNG and standard JPEG images can be compressed; however, the higher the compression, the lower the image quality.

5. Now you will insert an image into your Web page. Open **syb.html** in a text editor.

 

6. Insert the SYBcollage2.png image by entering the code indicated in bold:

```
<blockquote>
<img src="syb/SYBcollage2.png" alt="Join a summer build!"/>
</blockquote>
<h1>Join a Summer Build for Teenagers</h1>
```

*Note: The code you typed inserted a file from the Syb\ subfolder. You will learn more about how to specify files in other directories later in this lesson.*

7. Load your edited page into a browser. You should see the image shown in Figure 3-3.

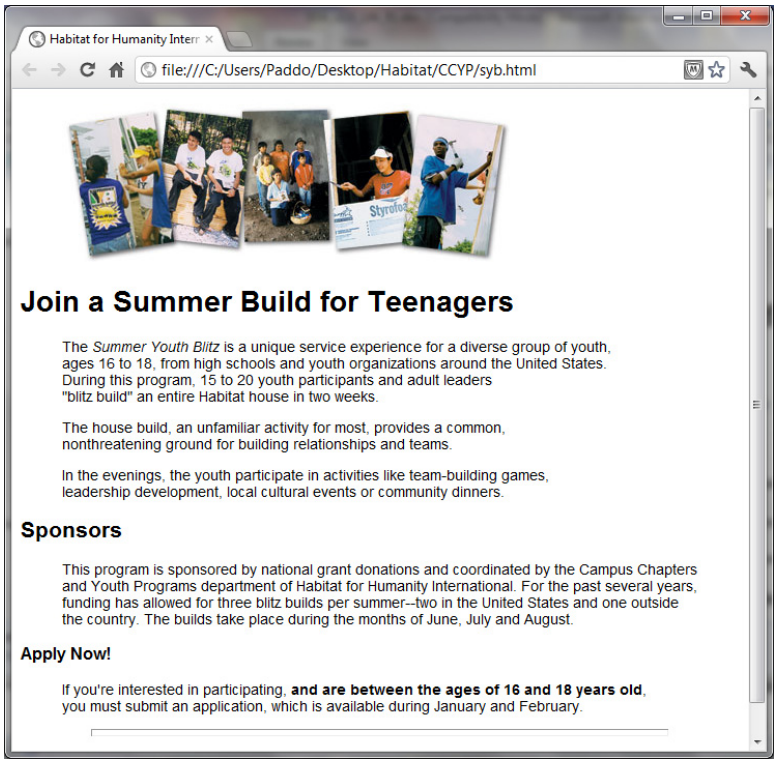


*Figure 3-3: File syb.html after adding image*

8. Validate your code.

9. Delete the code ***alt="Join a summer build!"***. Now revalidate your code. What were the results, and why? How does validated HTML help users with disabilities?

_____

_____

_____

10. Reinsert the code ***alt="Join a summer build!"*** where it was.

11. Comment out the **<blockquote>** tags that encompass the <img> element and its values using the ***<!-- ... -->*** comment tags. This will stop the .PNG image file from loading. **Save** your file.

12. Next, you will center the image using CSS. This task requires you to create a class in the CSS file and reference it within the HTML document. The class is simply a set of instructions. Once the class is set up, it will be easy to center your images.

 

*Note: The image element attributes used to align images are no longer supported in HTML5. CSS classes must be created in order to comply with HTML5.*

13. **Editor:** Open the **syb.css** file. To create a class, add the following CSS rule to the bottom of the css file:

```
    img.center
{
display: block;
margin-left: auto;
margin-right: auto;
}
```

You created instructions for a "center" class for the "img" element (img.center) that will be referenced within the HTML document. The image will display as a block element so that it does not merge with any text. The "auto" margin to the left and right creates an equal area on both sides of the image. This technique centers the image, regardless of the size of the browser window.

14. Save the **syb.css** file.

15. **Editor:** Open the **syb.html** file. Create a new line that inserts and centers the JPEG image using the *center* class you created for the <img> element:

```
<img src="syb/SYBcollage2.jpg" alt="Join a summer build!" class="center"/>
```

16. Save the **syb.html** file and open it in a browser. It should resemble Figure 3-4. Resize the browser window. Notice that the image repositions itself to the center of the page regardless of the browser window's size.

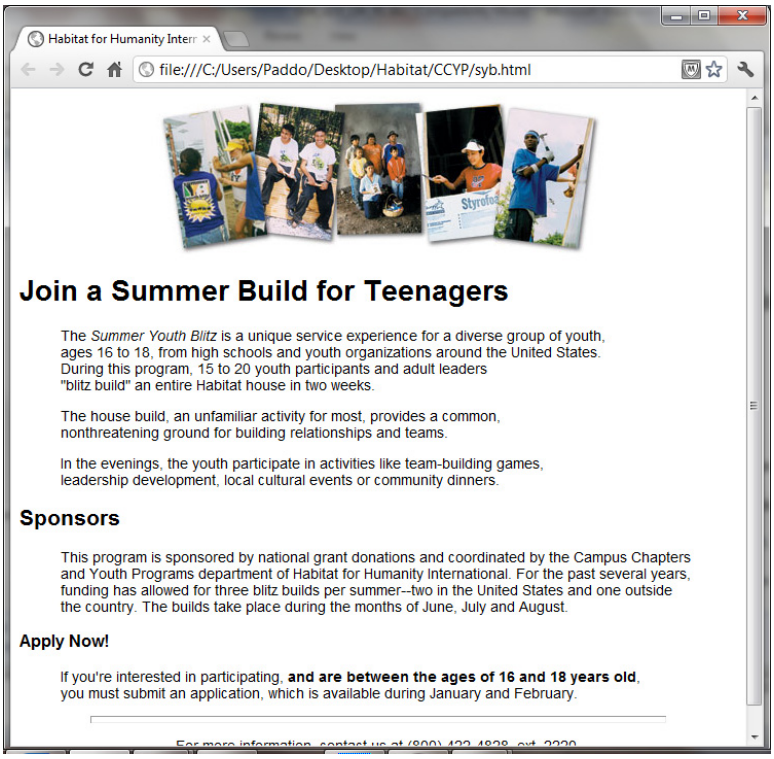


*Figure 3-4: Centering image by creating CSS center class*

 

**17.** When you are finished, comment out the <img> tag you just created for the .JPG image:

```
<!--
<img src="syb/SYBcollage2.jpg" alt="Join a summer build!" class="center"/>
-->
```

**18.** Remove the comment notations from the <blockquote> tags around the <img> tag for the .PNG image. Your code should now render the .PNG image.

In this lab, you added image files to a Web page. You used the required *alt* attribute, and you experimented with image placement using the <blockquote> element and the CSS *center* class.

**OBJECTIVE**
2.2.1: Images

2.2.3: Background images and colors

## Aligning images relative to text

After you start working with images, you will see that placing an image on your page is only the first step. You must know how to position images relative to text on a page.

### *External CSS*

External CSS classes are useful for floating images to the left or right of text in HTML5. The syntax for the *CSS class* entries are as follows:

```
img.floatleft
{
float: left;
margin: 5px;
}
```

In this example, the image class is defined as *floatleft*. The float property specifies that the image will float to the left of the text. The margin property specifies that the image will include a five pixel margin around it. This will provide a small cushion between the image and the text.

The HTML document needs the following code to access the CSS class:

```
<img src="syb/SYBcollage2.png" class="floatleft" />
```

You can use any name you would like for class: *floatleft* is a good descriptor. If the class was created to float text to the right, it could be named "floatright" with a float value of "right."

### *Inline CSS style attribute*

**NOTE:**
Aligning images relative to text using CSS can be tedious and can sometimes produce unsatisfactory results. Many developers use HTML tables instead because the image can be placed in one cell and the text in another. You will learn about HTML tables later in this course.

Although it is not the best method to use if you are developing sites with multiple pages, the inline CSS style attribute can also be used for floating images to the left and right of text:

```
<img src="syb/SYBcollage2.png" style="float:left; margin: 5px;" />
```

Table 3-4 lists the values that you can use to align images to text.

 

*Table 3-4 Using CSS to align images to text*

| Alignment Option | Description | Code |
|---|---|---|
| **bottom** | The default alignment. The bottom of the image is aligned with the baseline of adjoining text. | The CSS class can be written as:<br><br>CSS file:<br>`img.bottom {vertical-align:baseline;}`<br><br>HTML file:<br>`<img src="syb/SYBcollage2.png" class="bottom" />` |
| **middle** | A vertical — not horizontal — alignment option. This value aligns the middle of the image to the baseline of adjoining text. | The CSS class can be written as:<br><br>CSS file:<br>`img.middle {vertical-align:middle; margin:5px;}`<br><br>HTML file:<br>`<img src="syb/SYBcollage2.png" class="middle" />` |
| **top** | Aligns the top of the image with the top of adjoining text. | The CSS class can be written as:<br><br>CSS file:<br>`img.top {vertical-align:top;}`<br><br>HTML file:<br>`<img src="syb/SYBcollage2.png" class="top" />` |
| **left** | Floats the image to the left of the text paragraph into which the <img> tag is inserted. The top of the image will align with the left and top of the adjoining text. | The CSS class can be written as:<br><br>CSS file:<br>`img.floatleft {float:left; margin:5px;}`<br><br>HTML file:<br>`<img src="syb/SYBcollage2.png" class="floatleft" />` |
| **right** | Floats the image to the right of the text paragraph into which the <img> tag is inserted. The top of the image will align with the right and top of the adjoining text. | The CSS class can be written as:<br><br>CSS file:<br>`img.floatright {float:right; margin:5px;}`<br><br>HTML file:<br>`<img src="syb/SYBcollage2.png" class="floatright" />` |

 

The <img> element includes two traditional attributes that are helpful:

- *height* — resizes the image's original height dimension in pixels
- *width* — resizes the image's original width dimension in pixels

**NOTE:**
When the "left" or "right" value is specified, the image is aligned vertically to the top by default.

In the following lab, you will use CSS with the <img> tag to align images relative to text on your Web page. Suppose your project manager has notified you that several images will be incorporated into a Web page you are developing. You will need to experiment with aligning the images and text. Proper alignment of page components in relation to each other are an important part of Web design that can greatly improve — or degrade — a page's appearance and effectiveness.

## Lab 3-4: Aligning images relative to text with HTML

In this lab, you will practice CSS techniques to align your images relative to nearby text.

1. **Editor:** Open **syb.css** and add the following rule. This rule will float an image to the right of the text:

```
img.floatright
{
float:right;
margin:5px;
}
```

2. Save the file.

3. **Editor:** Open **syb.html**.

4. Find the <img> tag and edit it as follows so that it aligns to the right of the text:

```
<blockquote>
<img src="syb/SYBcollage2.png" alt="Join a summer build!" class="floatright"/>
</blockquote>
```

5. View your page in a browser. It should resemble Figure 3-5.

 

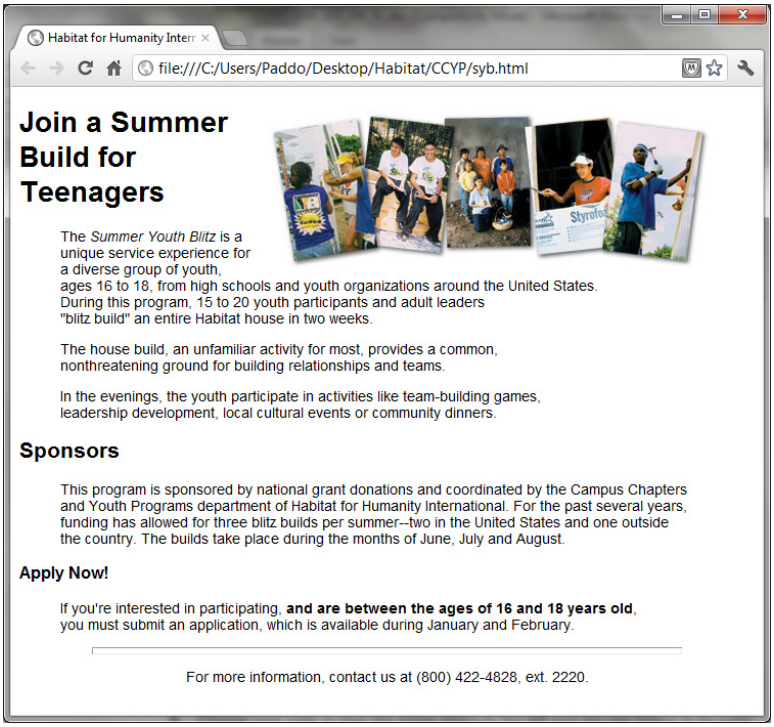


*Figure 3-5: Floating image to right of text*

6. Change your code so that the image aligns to the left and the text wraps to its right. Then change it so the text wraps to the top of the image, then again so text wraps to the middle of the image. Be sure to view the page each time in your browser, and note the differences.

7. Now, modify the *margin* value to "40" in the CSS file:

```
img.floatright
{
float:right;
margin:40px;
}
```

8. Save the CSS file and refresh your browser. Notice that the margin around the image increases. Experiment with a larger value, such as "**60**", "**100**" or "**200**".

   *Note: These values may seem large, and will probably make the image seem out of place. Simply experiment with these values. Remember that a pixel is a very small unit of measurement. The number of pixels per inch is determined by the monitor's screen resolution.*

9. When you are finished experimenting, change the *margin* value back to "**5**".

10. **Editor:** Finish this lab by positioning the image to the right of the text (as you did at the beginning of the lab).

11. Validate your HTML and CSS files using the W3C validation services.

In this lab, you learned to position images in relation to text with HTML and CSS.

 

## Resizing images

**NOTE:**
In some cases, it is preferable to resize images with an image-editing application (such as Adobe PageMaker, The GIMP or Corel Paint Shop Pro) rather than on the Web page.

At times, you will want to use an image in a size other than its natural size. If you need to resize an image, you must maintain its relative measurements. For example, if you have an image that is 200 pixels wide by 300 pixels tall, you probably would not want to change the size to be 100 pixels wide by 300 pixels tall because this would distort the image, making it appear taller and narrower than it was originally. If you were trying to shrink this image to one-half its size, you would instead change the width to 100 and the height to 150. By shrinking both dimensions by an equal percentage (in this case, by 50 percent), you maintain the original ratio of the image.

The syntax for specifying image height and width information is as follows:

```
<img src="imagename.gif" height="HeightInPixels" width="WidthInPixels"/>
```

**NOTE:**
It is considered good practice to specify the width of an image even if you are not changing the image from its original dimensions. This practice allows the browser to load surrounding text faster, because the browser then knows how much space to allot for the image before it loads.

To determine image dimensions, right-click the image file and select Properties then click the Details tab.

If you are not certain of the original dimensions of your image, you can ensure that the size will be changed proportionately by specifying either the height or the width; the other measurement will then be calculated proportionately for you based on the original image size.

### CIW Online Resources – Course Mastery

Visit CIW Online at *http://education.Certification-Partners.com/CIW* to take the Course Mastery review of this lesson or lesson segment.

*SDA Lesson 3 - Part A*

# HTML Entities

Occasionally, you will need to include a non-keyboard character in your Web page. These special characters are called HTML entities. For example, mathematics professors who use HTML pages to show math problems may want to use the "less than" (<) and "greater than" signs (>). And most companies use the © and ® symbols to indicate their copyrights and trademarks on their Web sites.

You can include non-keyboard characters in Web pages by using either the ANSI character value or the special HTML code for the character. These special character values can be read by HTML interpreters, which cannot otherwise recognize non-keyboard characters. The HTML code combination, called the escape sequence, consists of the ampersand ( & ), followed by a code for the specific character you want to generate on the page, followed by a semicolon ( ; ). For example, to generate the "less than" symbol on a Web page, you would use the following special character code as text on your HTML page:

```
&lt;
```

**NOTE:**
See **Optional Lab 3-1: Reviewing HTML5 tags**.

Using the escape sequence is also called "escaping" because it does not allow the browser's HTML interpreter to read the characters as literal text or HTML commands. In the Resources directory of the supplemental files for this course, the file named charcodes.html provides a list of codes for special characters. Table 3-5 defines some of the most commonly used special characters. As you can see, each special character code begins with the ampersand and ends with the semicolon character.

 

**NOTE:**
You need only be familiar with the copyright and registered trademark symbols. The rest of the characters are not required knowledge. Print a copy of the charcodes.html file to use as a reference, or keep the file open on your computer for quick reference.

*Table 3-5: HTML entity codes*

| Character | Description | Code |
|---|---|---|
| © | Copyright symbol | `&copy;` or `&#169;` |
| ® | Registered trademark symbol | `&reg;` or `&#174;` |
| é | Acute accent (over letter e) | `&eacute;` To create an acute accent over another character, such as o, enter: `&oacute;` |
| < | Less-than symbol | `<` |
| > | Greater-than symbol | `>` |
| & | Ampersand ("and") | `&` or & |
| £ | Pound sterling sign | `&#163;` |
| ü | Umlaut (over letter u) | `&uuml;` To create an umlaut over another character, such as i, enter: `&iuml;` |
| ñ | Tilde (over letter n) | `&ntilde;` To create a tilde over another character, such as o, enter: `&otilde;` |
| " | Quotation marks | `&quot;` |
| @ | At symbol | `&#64;` |
| **Non-breaking space** | Inserts an extra space. Often used to create indentations in a paragraph, or create additional spaces between words. | ` ` |

For a list of additional HTML entities, visit *www.w3schools.com/tags/ref_entities.asp* or *www.w3.org/MarkUp/html-spec/html-spec_13.html*.

## Non-breaking spaces

You will often see * * in HTML code, especially in code created by GUI-based Web authoring applications. The HTML special character code for a non-breaking space can be used to insert more than one space in succession when needed. You will find this character to be useful because more than one successive space is ignored in HTML. However, a non-breaking space is never ignored. Use non-breaking spaces sparingly. The following code ensures an indentation before the line "This begins an indented paragraph":

```
<p>
      This begins an indented paragraph.
</p>
```

In the following lab, you will create a symbol on a Web page using an HTML entity. Suppose your project manager mentions that the Web page you will post to the public

 

contains original text and images that could be subject to copyright infringement. You can use special character code to add a copyright statement to your page.

## Lab 3-5: Adding a copyright statement with an HTML entity

In this lab, you will add a copyright statement to your page using the HTML entity code to create the copyright symbol.

1. **Editor:** Open **syb.html**.

2. Insert the following statement at the bottom of the page, just above the </body> tag:

```
<p style="text-align:center">&copy; 2012 Habitat for Humanity
International</p>
```

3. Save your changes, then view your page in a browser. You will see the copyright symbol at the bottom of the page, as shown in Figure 3-6.

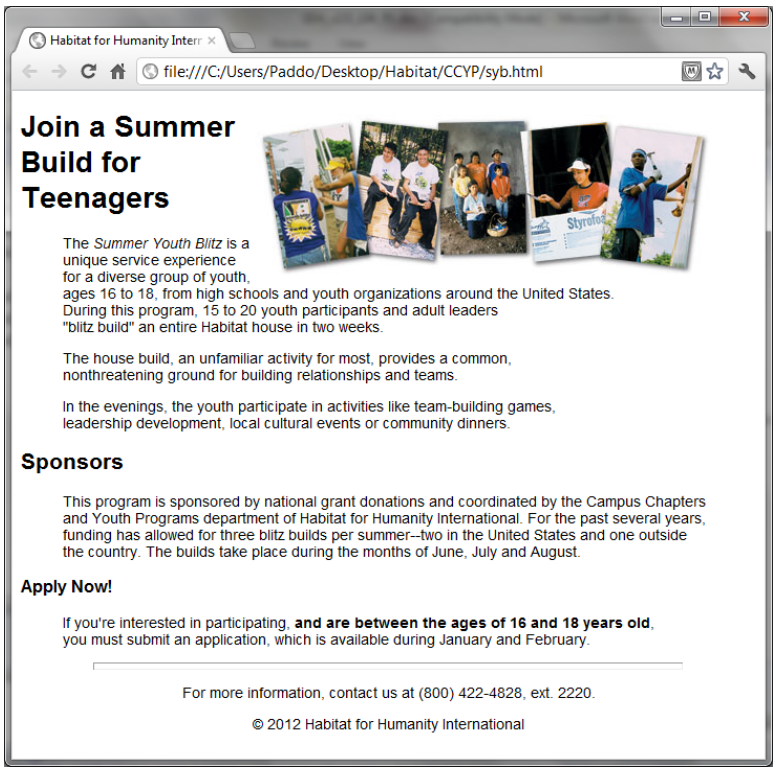


*Figure 3-6: Adding copyright symbol to Web page*

4. Validate your code, and test it in multiple browsers.

In this lab, you added a copyright statement using HTML special character code.

 

**OBJECTIVE**
2.2.3: Background images and colors

# Specifying Colors

You have learned that you can specify colors for the page background in HTML and CSS documents. In the previous examples, you used words for values, such as "teal" and "tan". Alternatively, you can also use a **hexadecimal** code to specify color values.

**hexadecimal**
A base-16 number system that allows large numbers to be displayed by fewer characters than if the number were displayed in the regular base-10 system. In hexadecimal, the number 10 is represented as the letter A, 15 is represented as F, and 16 is represented as 10.

Colors are often specified in terms of their RGB values. RGB stands for Red Green Blue. You may know that if you were mixing paint, the mixture of red, green and blue together creates a rather muddy color. But on a monitor screen, you are mixing light, and the mixture of red, green and blue light produces white, which is the presence of all colors. Black is the absence of all colors. In RGB code, the higher the numeric value representing a color, the lighter that color will be. The lower the value, the darker the color. Figure 3-7 demonstrates the composition of a hexadecimal color code.

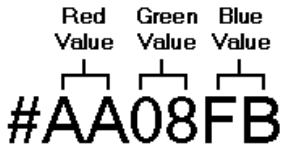


*Figure 3-7: Red hex value + green hex value + blue hex value = hexadecimal color code*

**NOTE:**
You are not required to memorize the hexadecimal codes but should understand how they are calculated and denoted. If possible, print copies of the 216color.html file, or visit *www.lynda.com/resources/webpalette.aspx* or *www.visibone.com/colorlab*.

Colors are specified in RGB values ranging from 0 to 255. The hexadecimal value FF represents 255. Therefore, the hexadecimal value #FFFFFF represents the highest possible value for all three RGB colors, producing white. The hexadecimal value #000000 represents the absence of all colors, or black. The number symbol ( # ) is not required by current generation browsers, but you should include this symbol for full backward-compatibility. The Resources directory from the supplemental files for this course contains a file named 216color.html that provides the RGB and hexadecimal codes for browser-safe colors, which will be discussed further in the next section. You can also visit the Browser-Safe Web Palette page at *www.lynda.com/resources/webpalette.aspx* (provided by Lynda Weinman) or the Visibone Webmaster's Color Laboratory at *www.visibone.com/colorlab* (provided by Bob Stein).

## Browser-safe color palette

**OBJECTIVE**
2.3.1: Browser-safe color palette

When you use a color in a Web page (whether for a background, font or image), you are enabling a combination of RGB values. This limited color palette is necessary because many computer screens have limitations.

**dithering**
The ability for a computer to approximate a color by combining the RGB values.

You will want your pages to render consistently no matter which browser or operating system is used to view them. In other words, if you define a blue background color, you will want it to appear the same in a Macintosh system using a version of Safari or Firefox as it would in a Windows system using a version of Internet Explorer. You also want your image colors to appear consistently. If your HTML code asks for a color that the browser or operating system cannot support, the computer will compensate through **dithering**. The results of dithering are unpredictable and often unattractive.

**inline images**
Images rendered in a Web page.

When Netscape Corporation marketed the first browser that supported **inline images**, it created a standard of 216 colors that would render consistently, known as the browser-safe color palette. Chrome, Firefox, Internet Explorer and other browsers conform to this list of colors.


*Sometimes the browser-safe color palette is called the Web-safe color palette.*

 

As mentioned, the file in your Resources directory named 216color.html contains a list of the 216 colors in the browser-safe palette. To further ensure cross-browser capability, you can specify colors in hexadecimal format, rather than by name.

**CIW Online Resources – Online Exercise**

Visit CIW Online at *http://education.Certification-Partners.com/CIW* to complete an interactive exercise that will reinforce what you have learned about this topic.

*Exercise 3-2: Specifying colors*

# Page Colors and Backgrounds

**OBJECTIVE**
2.2.3: Background images and colors

2.3.2: Color vs. line, value, shape and form

You can add color information to the <body> tag in an HTML page to control the colors of the page background, as well as the colors of the text and links on the page. In addition, you can tile an image across the page for a background.

## Background color

To specify a color for a page background, you add the *background-color* property to the *body* selector in an external CSS file using the following syntax:

```
body
{
font-family:arial, verdana, helvetica;
color:black;
font-size:14px;
background-color:white;
}
```

If you change the *background-color* value to the hexadecimal value #AA08FB, the background color of the page will become bright purple.

## Background image

To specify an image for a page background, add the *background-image* property to the body selector. You must specify the location of the image using the following syntax:

```
body
{
background-image:url('paper.gif');
}
```

## Text color

To designate the color of text on a page, use the *color* property in the body selector of the CSS file. View the preceding background color example for the syntax. Various color values, such as hexadecimal values, can be used instead of words.

## Hyperlink color

Later in this lesson, you will learn how to create hyperlinks. If you are using hyperlinks, you can control the colors of links depending on whether the link has been visited or not. Hyperlinks use the anchor, or <a> element, in HTML files. CSS allows different link states to be defined in the anchor selector. Table 3-6 describes two of the link states.

 

*Table 3-6: Color attributes available for various hyperlink states of anchor selector*

| Anchor Link State | Sample Declaration | Description |
|---|---|---|
| ***a:link*** | {color:#0000FF;}<br>{color:blue;} | Determines the color of unvisited links |
| ***a:visited*** | "{color:#FF0000;}<br>{color:red;} | Determines the color of visited links |

In the following lab, you will change Web page colors and backgrounds. Suppose your project manager has asked you to experiment with color combinations for your Web site's pages, text and links. She has also given you an image to use as a page background. Which background will work better for the Web page? And should you specify colors using color names or hexadecimal codes?

## Lab 3-6: Changing page colors and backgrounds with HTML

In this lab, you will use hexadecimal code to specify Web page colors. You will also use an image for a page background.

1. **Editor:** Open **syb.css**.
2. **Editor:** Change the following value to the *background-color* property of the body selector:

   ```
   background-color:#CCCCCC;
   ```

3. View the page in a browser. You will see that the background is gray.
4. Edit the CSS body selector to add the text *color* property as shown:

   ```
   color:#0099FF;
   ```

5. Load your page into the browser again. Notice that the text color has changed.

   *Note: Remember that it is always considered best practice to use style sheets instead of inline CSS style attributes whenever possible*

6. Return the *background-color* and *color* properties in the body selector back to their original values:

   ```
   body
   {
   font-family:arial, verdana, helvetica;
   color:black;
   font-size:14px;
   background-color:white;
   }
   ```

7. Save the **syb.css** file.

**NOTE:**
A copy of the edited style sheet is available in the Lab Files/Lesson03/Finished/Lab_3-6/syb directory. The file name is syb.css.

8. Go to the **C:\CIW\Site_Dev\Lab Files\Lesson03\Lab_3-6** directory, and copy the file **background.jpg** to the **syb** subfolder you have been using.
9. Now add this image as a background image by adding the *background-image* property to the CSS body selector as follows:

 

```
body
{
font-family:arial, verdana, helvetica;
color:black;
font-size:14px;
background-color:white;
background-image:url('background.jpg');
}
```

**10.** Refresh your browser. Your page will now have a purple background, due to the JPEG image you just inserted. The image is tiled by default.

*Note: The background image overrides the background color. Also, the URL reference for the background.jpg image does not require a directory path to the syb folder because the .css file and the background.jpg image are located in the same folder.*

**11.** Remove the purple background image (background.jpg) declaration from the style sheet so that your document's background is white again.

**12.** When your page renders as expected, save and validate the HTML and CSS files using the W3C validation services.

In this lab, you used hexadecimal color values and inserted a background image using HTML.

# Specifying Font Information

CSS allows you to change the size, color and typeface of the text. CSS supports many font-related selectors and properties, including:

- *font-family* — specifies the typeface (i.e., font name) to be used. A *font-family* value must include quotation marks if it has multiple words, such as "Times New Roman." You should also list backup fonts in the *font-family* value, such as Times New Roman, Arial, and Verdana. If the first font is not supported, the next value will attempt to render.
- *font-size* — takes values in pixels, with 16 being the normal size of default text. The *font-size* value is specified with the "px" abbreviation for "pixels." No spaces can exist between the number value and the abbreviation. For example, *font-size=16 px* would not render properly, but *font-size=16px* (with no space) would work correctly. Headings (h1 through h6) should be used whenever possible instead of the *font-size* property.
- *color* — uses the same values that you learned for specifying a page background color.

**NOTE:**
To review elements and attributes from this lesson, see **Activity 3-1: Identifying HTML and CSS terms**.


*If you specify a font that is not available on all systems, some users will not see the font face you chose. Instead, the font will display as a default font face. Be aware of this when planning your page and include backup font-family values.*

In the following lab, you will format text on a Web page. Suppose your Web team agrees that you should experiment with one of the site pages to see if a different font style would work well. You can use CSS to try some different font formatting styles.

 

## Lab 3-7: Formatting text with CSS

In this lab, you will learn how to change the font style of text using CSS.

1. **Editor:** Open **syb.css**.

2. Delete the comments from the **h1** heading selector, as shown:

```
/*
h1
{
font-family:arial, sans-serif;
color:black;
font-size:32px;
}
*/
```

3. **Browser:** Open the **syb.html** file. Did the h1 style change? It may be difficult to notice.

4. **Editor:** Open the **syb.css** file and add "courier new" with quotation marks as the first font-family value:

```
h1
{
font-family:"courier new", arial, sans-serif;
color:black;
font-size:32px;
}
```

5. **Browser:** View the **syb.html** file. The h1 font-family change is noticeable with the courier new font.

6. **Editor:** Delete the comments from the h2 heading selector. Change the font-size value to **20** pixels. Add "**courier new**" with quotation marks as the first font-family value, as shown:

```
/*
h2
{
font-family:"courier new", arial, sans-serif;
color:black;
font-size:20px;
}
*/
```

7. **Editor:** Delete the comments from the h3 heading selector. Change the color value to **blue**. Add "**courier new**" with quotation marks as the first font-family value, as shown:

```
/*
h3
{
font-family:"courier new", arial, sans-serif;
color: blue;
font-size:16px;
}
*/
```

8. When you are finished, your page should resemble Figure 3-8.

 

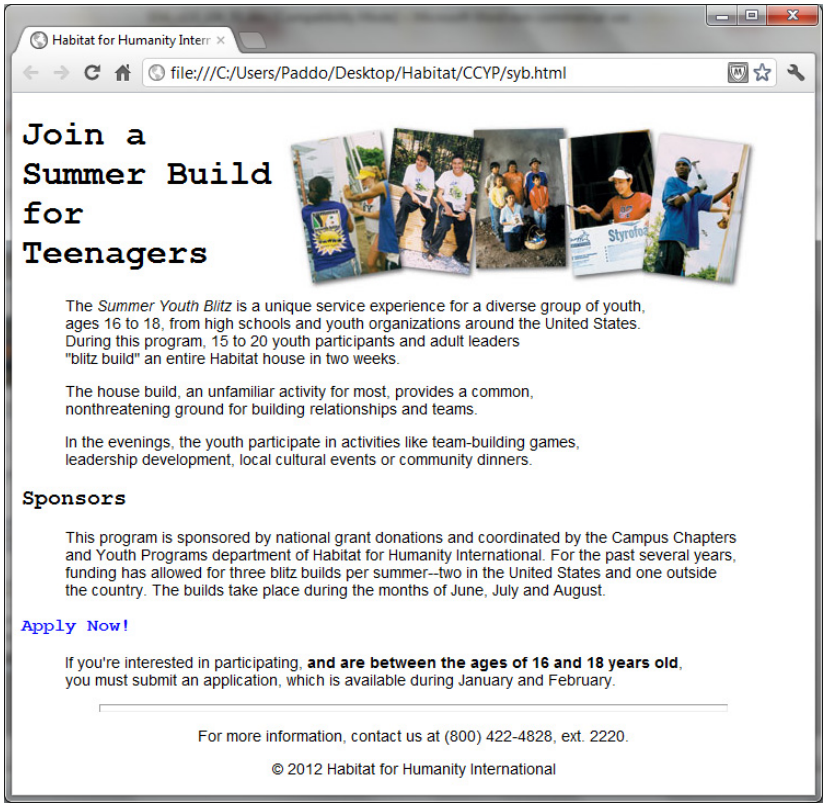


*Figure 3-8: Modifying heading styles with CSS*

**9.** When your page renders as expected, validate your code at ***http://validator.w3.org***.

In this lab, you used CSS to format Web page text.

With practice, you can embellish your HTML pages quickly and easily by using CSS to add graphical elements and styles.

# Web Design Issues

Thus far, you have created a Web page that validates to HTML5. You have also learned many page elements. Before continuing, it is important to consider some Web design issues, including color combinations and ways to structure documents.

**OBJECTIVE**
2.2.3: Background images and colors

2.3.3: Impact of color choices

## Color combinations

Color combinations are important to a Web site's look and feel because they can impose tone and mood. Color choices can convey the personality of a site, and thus its sponsoring organization, as serious, playful, trendy, conservative, creative, studious or authoritative, for example. Certain color combinations can also make a site easier — or more difficult — to read and view. Following are some examples of popular color combinations for Web sites:

- Gray, green and white (e.g., *www.w3schools.com*)
- Blue and white (e.g., current live Habitat for Humanity site, *www.habitat.org*)
- Red and white (e.g., *www.linux.org*)

 

- Red, white and gray (e.g., *www.cnn.com*)
- Red and gray (e.g., *http://espn.go.com*)

Once a color scheme is chosen, most sites use lighter shades of chosen colors for background. A lighter background acts as a foil (i.e., contrast) to the foreground text and images, making the site appear more polished and professional.

### *Culture and audience issues*

Remember that the Web pages and sites you help develop may be available to anyone in the world with a browser and an Internet connection. Consider the following issues:

- From what culture(s) are the people who will primarily view this site?
- Is your chosen color combination effective in the cultures this site targets or in most cultures?
- What is considered "professional" for the audience that will most likely view this site?

In the following lab, you will consider some color combinations that could be used for a Web site. Suppose the marketing department is in charge of branding for your organization. That team will determine the best color combinations for all company materials, sites and advertising. Although you are not responsible for choosing the color scheme, your project manager has asked you to research and compare some color combinations so that you can present information to the marketing team about the way that colors in the company Web pages will render in browsers.

**Lab 3-8: Comparing Web site color combinations**

In this lab, you will compare color schemes used in organization and corporate Web sites.

1. Open your browser and visit ***www.w3schools.com***.
2. What color scheme is used for this Web site? Is it easy to read, distracting or unnoticeable? What type of mood or personality does this site's color scheme convey? And what does the look and feel of the home page make you think about its sponsoring organization?
3. Visit the following sites and consider the same questions for each:

   ***www.habitat.org***
   ***www.microsoft.com***
   ***www.CIWcertified.com***

In this lab, you compared common color combinations used in Web sites, and you considered their effects on users.

**OBJECTIVE**
2.3.4: Page design and layout

## Page layout

Web page layout is the placement of all page elements — including text, images, headings, navigation menus and so forth — relative to each other. A good page layout makes the page aesthetically pleasing, and easy to scan, read and navigate. Following are some layout guidelines to consider when designing your Web pages:

 

- **Be succinct** — Limit words to clear, necessary verbiage, especially on the home page. Most users simply scan pages quickly looking for specific information or links to it. Let users navigate to additional pages if they want more information.
- **Make sure that each page focuses on one topic** — No tangent message, regardless of its importance, should be added to a page. Use links to point users to appropriate related topics on separate pages.
- **Divide the page into three sections** — Use the left side of the page for navigation, the upper section of the page for a topic title (as well as navigation), and the middle section of the page for the information.
- **Include navigation aids** — A common way to enable navigation is to place links at the top and bottom of the page, and within the body of the page, to reduce the need for users to scroll.
- **Place comments in each section of code** — Comments help explain changes you have made to the code or page. You can indicate the nature of the change, including the date you made the change and your name or initials, or you can explain the nature of the markup. For example, the syb.html page has three headings (h1, h2 and h3). You could comment each of these sections of the narrative.


*Effective page layout is essential for effective search engine optimization.*

**OBJECTIVE**
2.8.1: Structuring documents with CSS

2.21.1: Document structure with HTML5 and CSS

## *Document structure and style sheets*

When Web pages were first being developed, early designers would use the <table> tag to format pages. Tables were necessary to divide the pages into sections because the Cascading Style Sheets (CSS) standard had not yet been developed. By using tables, designers could ensure that all page elements aligned and rendered consistently.

Today, the use of the <table> tag to format pages is considered improper practice by the W3C. Pages formatted with the <table> tag will not pass W3C validation tests, no matter what standard you use.

Following the use of tables, developers began to add structure to an entire page by using the  tag. It provided a way of dividing a page using an *id* attribute. Once a name was specified for a document section using the *id* attribute, developers could define this section's place in the document (as well as its contents) in a linked style sheet. That strategy gave developers granular control over the document, and ensured that the style sheet — rather than the HTML tags — governed the document's structure.

HTML5 with CSS has introduced a more effective and simple way to structure Web pages. Instead of using the  tag as a "jack of all trades" element with its array of attributes, the developers of HTML5 created specific elements to define the document structure. These elements include <header>, <footer>, <nav>, <article>, and <aside>. A basic Web page structure is shown in Figure 3-9.

 

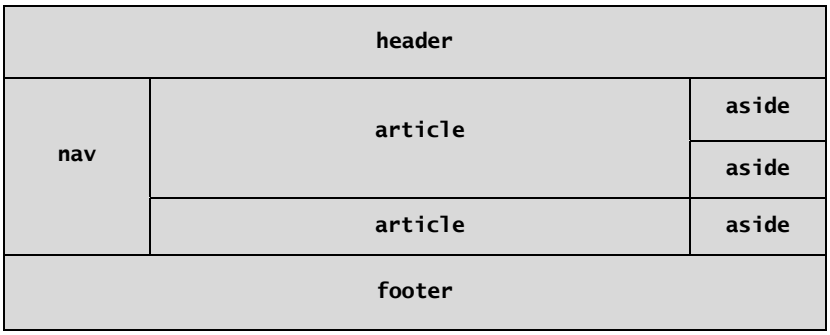


*Figure 3-9: Sample HTML5 structural elements of Web page*

As shown in the figure, a developer can structure the Web page with elements that are easily interpreted and native to any HTML5-compliant browser, regardless of whether the browser is on a mobile device, laptop or tablet. The structure elements can be defined as:

- **<header>** — Top of the Web page, similar to the header in a word-processing document.
- **<nav>** — Defines navigation links, such as hypertext menus to access various pages of the Web site.
- **<article>** — Web site content, such as company services, articles, blogs, images and videos.
- **<aside>** — Content that is "aside" from the article content, such as advertisements or news feeds.
- **<footer>** — Bottom of the Web page, similar to the footer in a word-processing document.

The header might contain the company logo; the footer could include copyright and contact info; the nav section may contain navigational links to other resources on the site; the articles could be blogs, tweets, video, social networking posts, news articles, or any other content; and the aside could be advertisements, highlighted resource links, and even more content.

The following HTML sample demonstrates a way to define part of a document's overall structure and add images using the <nav> tag:

```
<!-- NAVIGATIONAL SIDEBAR -->
<nav>
<img height="129" src="syb/SYBSumReptCV.jpg" width="100" alt="Link to a PDF
file. Go to www.adobe.com to download Adobe Reader"/>
<br/>
<br/>SYB
<br/>Summary Report
</nav>
```

**NOTE:**
Note that the <nav> tag is in the HTML file, and *nav* is in a style sheet. Also note that the HTML nav entry points to the CSS nav entry.

This HTML code can now be controlled through an external style sheet. The following style sheet entry defines formatting for the section of code created by the <nav> tag in the preceding HTML sample:

```
nav
{
float:left;
width:165px;
background:#fc3 url(navbg_04.gif) repeat-y top right;
height:662px;
}
```

 

In this CSS code, the navigational sidebar is made to appear (i.e., float) on the left side of the document. A background image (navbg_04.gif) is specified and will appear at the top right of the sidebar. The *repeat-y* entry ensures that the image is tiled vertically in the background. The sidebar will always appear 165 pixels wide and 662 pixels high. You could repeat these structural properties and values for each structural element of the HTML document, then use CSS to control the rendering of each section in the document.

If necessary, you can adjust the formatting any time by modifying the CSS entries. For example, suppose you were to add more text to a page, which makes the page longer. If you were using a sidebar image, this image may then be too short, making the page look awkward. To solve this problem, you could increase the sidebar image's height to accommodate the change by adjusting the *height* property's pixel value in the style sheet. You can also use CSS to move images to specific locations on your pages.

Alternatively, the following example shows an inline CSS style attribute declared inside an HTML <p> tag:

```
<p style="text-align:center; margin-right:12px;"/>
```

This tag uses the inline CSS style attribute to align text to the center of a right margin that begins 12 pixels into the document. This style will apply only to this <p> tag, and not to any others.

For more information about using proper CSS layout, visit the following sites:

- W3C — Home Page of Cascading Style Sheets (*www.w3.org/Style/CSS/Overview.en.html)*
- w3schools.com — CSS tutorials and reference (*www.w3schools.com/css/css_intro.asp*)

You will learn more about applying styles with CSS throughout this course.

### *Fixed-width vs. liquid design layouts*

**OBJECTIVE**
2.21.2: Fixed-width vs. liquid design

Web designers have no control over their site visitors' browser window sizes, the Web browsers used or the fonts installed on visitors' computers. Yet despite this, many designers try to control the way that Web page elements will render on the screen.

There are two page-layout methods that designers use to control the placement of Web page elements when rendered in the browser:

- **Fixed-width layout** — also known as absolute positioning. Achieved by assigning specific pixel widths to elements using the HTML5 structural elements or the  tag. This layout ensures that the text, images and layout will not vary from browser to browser. The problem with using a fixed-width layout is that the elements may not render as expected when users change the size of their browser windows.
- **Liquid layout** — also known as relative positioning. Achieved by assigning percentage values to elements. With this layout, the size of an element is flexible and will change dynamically depending on the size of the browser window. For example, you can specify in the CSS width property that Element A will occupy 35 percent of the screen and Element B will occupy the remaining 65 percent. If the user resizes the browser window, the elements will resize correspondingly.

There is great debate concerning the use of fixed-width versus liquid design layouts in Web pages. General industry consensus indicates that implementing a liquid design layout using CSS is the preferred method of designing Web pages.

 

### *Relative path names*

Most Web developers use subfolders to organize images, style sheets and Web pages. Currently, your HTML pages are configured to refer to all images and style sheets in the same directory (e.g., a folder on your Desktop). As the site grows, your HTML pages will refer to subdirectories. A reference to a file within a directory or subdirectory is called a relative path.

A relative path statement allows you to specify subdirectories (i.e., subfolders), as well as directories above the one where your page currently resides. A relative path assumes that the directory in which the HTML file resides is the current (i.e., "home") directory. All other directories exist either beneath the current directory (i.e., subdirectories) or above the current directory (i.e., parent directories) in a hierarchical structure.

For example, if you place the SYBcollage2.png file into the Syb\ directory, and your Web page's <img> tag needs to reference this image file, then you must change your *src* attribute value to include the new subdirectory location, as follows:

```
<img src="Syb/SYBcollage2.png" alt="PNG image"/>
```

If you omit the subdirectory reference to Syb/ before the image file name, then your HTML page will look for the SYBcollage2.png image file in its existing directory, and not in a subdirectory named Syb. Thus the page will render without the image file, and a small box with an X will appear in its place.

### *White space, the <img> tag and HTML*

With HTML, if you add a space or use the ENTER key to create a return within the <img> tag, the additional space will not be rendered in a browser. In XHTML, however, sometimes adding spaces or hard returns within or even between <img> tags will cause white space to appear on the page. Therefore, be careful when working with <img> tags in XHTML so that you do not add unintentional white space.

 

## HTML5 and Older Browsers

# Support for Internet Explorer 8, 7 and 6

HTML5 is not supported by Internet Explorer 8 or earlier. This is a challenge because Windows XP and Vista cannot run IE9, which is the only Internet Explorer version that supports HTML5. The simple solution is to install the latest version of the Chrome or Firefox browsers on Windows XP and Vista systems.

But what about your site visitors? You cannot force them to install Chrome or Firefox. They may be using IE8, 7 or 6. The only way to support these older IE browsers is to add JavaScript code to your HTML documents and a new rule to CSS.

Because older IE browsers will not recognize the new HTML5 elements, such as <header>, <footer>, <article>, etc. JavaScript must be used to create these elements in your Web page so older IE browsers will recognize them.

Note: JavaScript coding is beyond the scope of this course. This code is provided to demonstrate how JavaScript is used to support older browsers.

### *HTML5 document additions*

At a minimum, the basic HTML5 structural elements should be created by adding JavaScript in the <head> element:

```
<script>
document.createElement("article");
document.createElement("footer");
document.createElement("header");
document.createElement("nav");
</script>
```

### *CSS document additions*

The new HTML5 structural elements must also become block-level elements in CSS for consistent styling. A block-level element is a large block of content, such as a paragraph or a structural element that starts a new line of text or a new section.

```
header, nav, article, footer
{
display:block;
}
```

### *Adding all HTML5 elements*

Each new HTML5 element must be added. This can be a long and laborious task, so the best solution is to implement a mini-script developed by Remy Sharp, available at *http://remysharp.com/2009/01/07/html5-enabling-script*.

Instructions for enabling the script are included at the Web site. Remy Sharp's code has been utilized by Google, MIT and nearly all Web developers across the globe. It is the *de facto* standard for supporting HTML5 for older IE browsers.

In the following lab, you will see how the HTML structural elements and CSS can be used to create document sections. Suppose members of your Web team have added images and page structure to an HTML page. The new document structure includes four sections:

 

- **A navigation bar in the header of the page** — with links to the rest of the site. This section will contain several images.
- **A navigation bar on the left side of the page** — with links that explain additional Habitat youth programs. This section will contain a background image, an image that will eventually link to a PDF file, and several button-style images, the first of which will eventually link to another page.
- **A body content section** — with the narrative (which you have already marked up) about the Summer Youth Blitz program.
- **A footer section** — with text-based navigation to help ensure accessibility.

The page and a style sheet have been developed already. You have been assigned to review the page and add comments to make sure that the document is properly structured. You have also been asked to verify that the document refers to the correct style sheet.

## Lab 3-9: Using HTML5 and CSS to structure a page

In this lab, you will review the structure of a Web page. Specifically, you will see the way that HTML5 and CSS can be used to create document sections.

1. **Editor:** Close **syb.html**.
2. **Windows Explorer:** Create a subdirectory off of **Habitat\CCYP\Syb** named ***Old***.
3. **Windows Explorer:** Move the contents of the Habitat\CCYP\Syb\ folder to the **Habitat\CCYP\Syb\Old** folder.

   *Note: These actions will not delete or move your existing syb.html page, because it is not in the Habitat\CCYP\Syb\ folder.*
4. Move the existing **syb.html** file into the **Habitat\CCYP\Syb\Old** folder.
5. **Windows Explorer:** Go to the **C:\CIW\Site_Dev\Lab Files\Lesson03\Lab_3-9** directory, and copy the contents of the Syb\ folder into the **Habitat\CCYP\Syb** folder.

   *Note: Copy the contents of the Syb\ folder, but not the folder itself.*
6. Go to the **C:\CIW\Site_Dev\Lab Files\Lesson03\Lab_3-9** folder, and copy the **syb.txt** file from this folder to your **Habitat\CCYP** directory.
7. Open **syb.txt**. Verify that the style sheet links to the **\Syb\syb.css** file.
8. In **syb.txt**, scroll down to the following code:

   ```
   <header>
   ```
9. Add the following comment immediately above the code block:

   ```
   <!-- HEADER -->
   ```
10. You have identified the top section of the page. Now, find the following text:

    ```
    <nav>
    ```

 

**11.** Immediately above this text, enter the following comment:

```
<!-- NAV -->
```

**12.** You have just marked the navigation section of this page. Next, find the following text:

```
<article>
```

**13.** Immediately above the text you just found, enter the following comment:

```
<!-- ARTICLE -->
```

**14.** Find the following text:

```
<footer>
```

**15.** Immediately above the text you just found, enter the following comment:

```
<!-- FOOTER -->
```

**16.** You have now commented each of the sections of your page. Review each section carefully. Notice that HTML5 structural elements are used to create each section.

**17.** Save syb.txt as **syb.html** in the Habitat\CCYP directory, and view it in a browser. Your page should resemble Figure 3-10.


*Figure 3-10: syb.html Web page using HTML5 structural elements and CSS*

 

18. **Editor:** Open **syb.html** in your editor. Find the section marked with the comment <!-- HEADER-->. Notice the <header> tag immediately beneath the comment. Each HTML5 structural element has a counterpart in the Syb\syb.css file because each tag, in conjunction with your linked style sheet, helps define the structure of this HTML page.

19. **Editor:** Open the **Syb\syb.css** file. Compare the structural tags with the corresponding entries in the style sheet. For example, notice that the *nav* entry in syb.css inserts an image named navbg_04.gif into the file. No reference to this image exists in syb.html. Therefore, you can use style sheets to add images.

20. Review the other sections of your page, and compare them to the style sheet.

21. Cut one of the HTML5 structural tags, then load the page in a browser to see the impact. Return to the editor, and Undo your changes to restore the document.

    *Note: You can restore the document by using the Undo feature in WordPad or Notepad. If necessary, you can also copy the complete file from the Finished folder for this lab.*

22. From **syb.html**, comment the reference to the style sheet, then reload the page in the browser. Notice that the document's structure is no longer intact because the style sheet, in conjunction with the HTML5 structural tags, now defines the structure of this HTML page. Remove the comment notation.

23. Validate your code. Correct all problems so that the code validates as CSS and HTML5.

24. If you like, remove the background image (background.jpg) from the page. What background color would you recommend for this page? Add it now.

In this lab, you added structure to your page using images and CSS.

## CIW Online Resources – Course Mastery

Visit CIW Online at *http://education.Certification-Partners.com/CIW* to take the Course Mastery review of this lesson or lesson segment.

*SDA Lesson 3 - Part B*

 

**Case Study**

## A Sharper Image

Iain works on the Web development team for a prominent community college system in the Southwest part of the United States. He was assigned to create several new Web pages, most of which would include detailed images. These images would aid navigation and provide vital registration instructions for the college students.

Iain was experienced with HTML and did not feel it was important to spend time validating all code because he was very busy. After creating and posting the new pages, however, Iain began receiving complaints from students who used text-based browsers. These students complained that they could not see (or hear) the registration instructions.

To solve this problem, Iain added the *alt* attribute to each <img> tag to describe each image with text. Although it would have taken him more time to include the *alt* attribute in his initial page development, Iain found that the *alt* attribute resolved the complaints.

* * *

As a class, discuss the following issues that might arise when incorporating images in Web pages:

- Most people use browsers that support images. Why is it important to support all browser types in this situation?
- How could validation have saved Iain development time? What other advantages would have come from initial validation?

 

# Lesson Summary

## Application project

When you incorporate an image into your Web page, you should consider the image's usefulness to the rest of the page. Images are generally effective only if they complement the text or overall message of the page.

Many Web sites offer copyright-free graphic files for site designers to use. These sites provide coordinated graphics such as buttons, bullets, rules and backgrounds that can help create a theme for your site. Use a search engine to locate free Web graphics, or visit the Yahoo! Directory at *http://dir.yahoo.com/*, and navigate to Arts & Humanities | Design Arts | Graphic Design | Web Page Design And Layout | Graphics. Add a new, free, non-copyrighted image to your Web page. After you finish, revert your file to its original state for the remaining lessons.

## Skills review

In this lesson, you were introduced to CSS and graphical Web page elements such as horizontal rules, images and colors. You learned how to position graphics relative to text on a page, and how to resize images for display using CSS. In addition, you applied CSS to font families, font size and font colors. You also learned about special characters, and about using HTML5 structural elements to add structure to a Web document.

Now that you have completed this lesson, you should be able to:

- ✓ 2.2.1: Incorporate graphical images into HTML pages.
- ✓ 2.2.2: Distinguish among and identify the uses and benefits of various graphic file formats, including GIF, GIF89a, JPEG, PNG, TIFF, BMP.
- ✓ 2.2.3: Add tiled images and colors to Web page backgrounds.
- ✓ 2.2.6: Insert horizontal rules into Web pages.
- ✓ 2.3.1: Define the browser-safe color palette.
- ✓ 2.3.2: Identify ways that color affects the principles of line, value, shape and form in Web pages.
- ✓ 2.3.3: Identify and demonstrate the impact of color combinations to various audiences and cultures.
- ✓ 2.3.4: Evaluate Web page design and layout.
- ✓ 2.8.1: Explain how to structure Web documents with CSS.
- ✓ 2.8.2: Identify ways to apply styles with CSS.
- ✓ 2.21.1: Use CSS and HTML5 elements to create document structure.
- ✓ 2.21.2: Distinguish between fixed-width and liquid design layouts.

 

## CIW Practice Exams

Visit CIW Online at *http://education.Certification-Partners.com/CIW* to take the Practice Exams assessment covering the objectives in this lesson.

*SDA Objective 2.02 Review*

*SDA Objective 2.03 Review*

*SDA Objective 2.08 Review*

*SDA Objective 2.21 Review*

*Note that some objectives may be only partially covered in this lesson.*

# Lesson 3 Review

1. What attribute allows inline CSS to be applied within HTML document tags?

2. What is the term for the grouping of a CSS selector, property and value?

3. Name the alignment options available for aligning images relative to text.

4. What is the function of the *alt* attribute?

5. What standard of 216 colors was introduced to render Web page colors consistently across different browsers?