HTML5

Lesson 2: **Fundamentals of HTML, XHTML, and CSS**

Windows, Bing, PowerPoint, Internet Explorer, Visual Studio, WebMatrix, DreamSpark, and Silverlight are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Other product and company names mentioned herein may be the trademarks of their respective owners.

The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious. No association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred.

The information contained in this tutorial is provided without any express, statutory, or implied warranties. Neither the authors, Microsoft Corporation, nor its resellers or distributors will be held liable for any damages caused or alleged to be caused either directly or indirectly by this tutorial.

Created for Microsoft by Avlade: [www.Avlade.com](http://www.Avlade.com)

Copyright 2011 by Microsoft Corporation.

Certain portions excerpted from the *HTML5 Digital Classroom* are copyright 2011 by AGI Training, and are used by permission. For more information about the *HTML5 Digital Classroom* book visit <http://www.digitalclassroombooks.com/Web-Design/HTML5-Digital-Classroom-Book.html>

Lesson 2  
Fundamentals of HTML, XHTML, and CSS

## Learning objective:

In this lesson, you’ll discover the fundamentals of HTML, XHTML, and CSS. Together, these form the structure and style of your web pages.

## Starting up

You will work with several files from the HTML5\_02lessons folder in this lesson. Make sure you have loaded the HTML5 lessons folder onto your hard drive. See “Loading lesson files” in the Starting Up section of this book.

## Web languages

In this lesson, you will explore the fundamentals of two languages: HTML and CSS. Although they have different syntax and rules, they are highly dependent on each other. By the end of this lesson, you will understand how to create simple HTML pages, add images, create hyperlinks from one page to another, and add simple styling to pages using CSS.

This lesson covers a lot of ground, and many of the core principles introduced in this lesson will help you learn how to build fundamentally sound websites today as well as prepare you for the future with the new HTML5 features.

## Web page structure is based on HTML

Hypertext Markup Language (HTML) documents use the .html or .htm extension. This extension allows a web browser or device, such as a smartphone, to understand that HTML content is on the page, and the content of the page is then rendered by the browser or device according to the rules of HTML.

Markup tags are used to define the content on an HTML page. Markup tags are contained between greater than (<) and less than (>) symbols, and they are placed at the start and end of an object or text that is used in an HTML page. Here is an example of two heading 1 tags for text. The tags are not seen by the viewer of the web page, but every web browser knows that the text between the tags is a heading 1.

<h1>Welcome to Worldwide Apparel’s intranet</h1>

In this example, the <h1> is the opening tag and the </h1> is the closing tag. So this entire line of code is an *element.* More specifically, it is referred to as the heading 1 element.

HTML and XHTML are closely related. There is a list of rules defined by the World Wide Web Consortium, or W3C that specify the perimeters of HTML and XHTML.

| HTML code as rendered in the browser |
| --- |
| To help you understand the relationship between the HTML code and what you see in your web browser, the following illustration will show you the connection between the two.    *A.* Doctype. This line instructs the browser to interpret all the code that follows according to a unique set of rules. *B.* HTML element. This element nests all the following elements and tells the browser to expect an HTML document. *C.* Head element. This section includes information about the page, but nothing is rendered on the page itself. *D.*Title element. Any content inside the title tags show up at the top of the browser. This is what is used when a user bookmarks a page in the browser. *E.* Body element. All content within the body can be rendered in the browser’s main window. *F.* Heading 1 element. The first of six heading elements. Content that is a heading 1 is rendered very large and bold. *G.* Image element. Links to a graphic file and displays it on the page. *H.* Paragraph element. By default, the browser adds space before and after this element which often contains multiple lines of text. *I.* Strong element. Formats the enclosed content as bold by default. *J.* Ordered list element. Defines the enclosed list items as numbered. *K.* List element. Multiple list items will automatically be numbered by the browser. |

## The details of XHTML syntax

There is little fundamental difference between HTML 4.0 and XHTML 1.0 - the two standards previously released by the W3C (World Wide Web Consortium). As XHTML was defined, it was created so that pages written in XHTML also work in browsers that render current HTML. The tags and attributes of XHTML and HTML remained the same, but the syntax of XHTML code is stricter.

The most significant differences between XHTML and HTML are as follows:

* In XHTML, all tags must be lowercase.
* XHTML requires all tags to be closed - meaning that there must be a tag at the start and end of the element being tagged - such as a headline, paragraph, or image.

All tags in XHTML must close, even special tags that technically don’t require an open and close tag. For example, the <br> tag, which creates a line break, uses a special self-closing syntax. A tag that self-closes looks like this (with a space and a forward slash): <br />

* XHTML requires proper nesting of tags. In the following example, the <em> tag to emphasize text opens within the <h1> headline tag. As such, it must be closed before the <h1> is closed.

<h1> Welcome to <em>Worldwide Apparel’s</em> intranet</h1>

We’ve used XHTML-compliant code throughout this book as we provide HTML5 examples, which help make your designs compatible with modern browsers and mobile devices and make it easier to troubleshoot if coding errors turn up.

## Doctype lets the web browser know what to expect

The start of every web page should include a Doctype declaration. Declaring the doctype tells the web browser a little bit of information about what it is going to see on the page. Because there are different specifications for XHTML and HTML, the web browser knows which language it’s about to see and render. Because a browser renders the page starting at the top line and then moves down, placing your doctype on the first line makes a lot of sense. While it’s not required, it’s good form to always use doctype at the start of your HTML pages. The doctype for HTML 4.0.1 looks like this:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

When a web browser sees a Doctype declaration, the browser expects that everything on the page that follows will use that language. If the page adheres to the specifications perfectly, it is considered valid. In Lesson 3 you will look at the change in doctype requirements that the HTML5 specification introduces.

## The W3C and page validation

The W3C is the World Wide Web consortium - a non-profit group that helps guides the evolution of the web. The W3C provides guidelines and rules for specifications including HTML and XHTML. One way to determine the validity of the HTML or XHTML code you generate is to use W3C’s free online validation service.

You will need access to the Internet for this exercise. If you do not have Internet access, you may read through the exercise to understand the validation process.

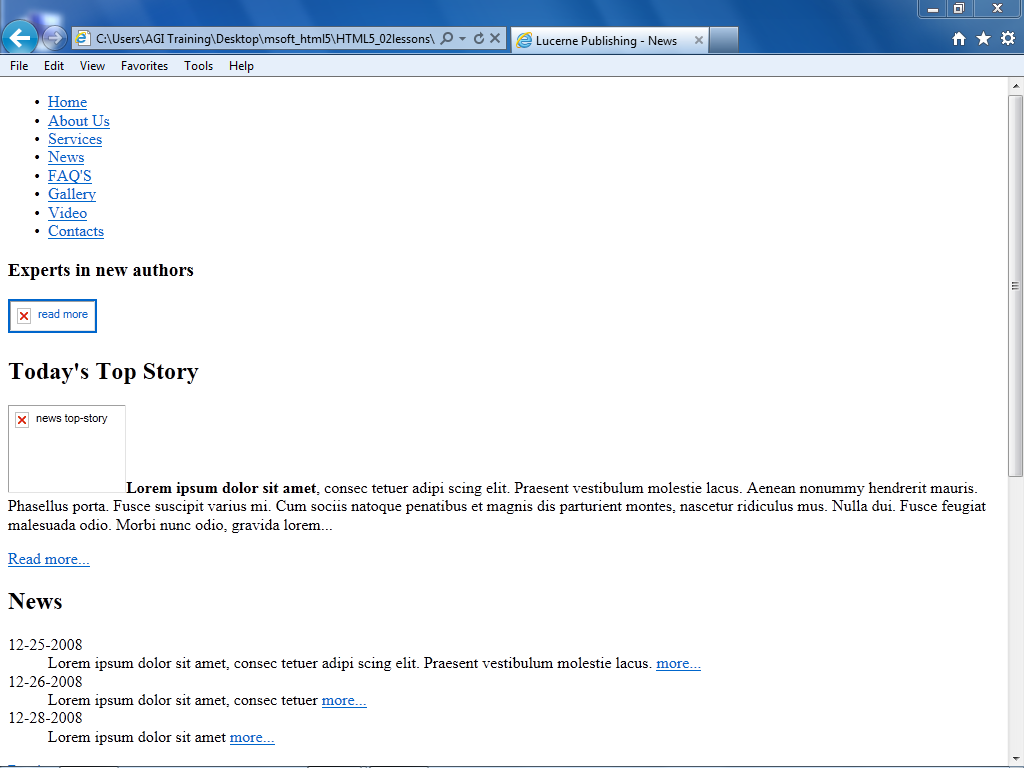
1. Open your web browser and navigate to <http://validator.w3.org>.
2. Click the Validate by File Upload tab.

The W3C validator allows you to check your HTML code for errors.

1. Click Browse and navigate to your HTML5\_02lessons folder, select the w3\_noncompliant.html file, then click Open. Click the Check button to validate the code.
2. The W3C site returns several errors. Scroll down the page and you can see in-depth information on the errors. Don’t worry about the errors at this point. You will now upload a nearly identical file without errors.
3. Click the Browse button, navigate to your HTML5\_02lessons folder, select the w3\_compliant.html file, and then click Open.
4. Click the Revalidate button. You now see a Congratulations message that the page has been checked and found to be compliant as XHTML 1.0 Strict.

Although the page is valid, it may not look good to a viewer. This example uses a page that has missing styles and missing images to emphasize this point. It’s important to understand that having valid code is only one step in a series to make certain your web pages can be viewed by the widest possible audience.

You can validate web pages that you’ve already placed online. Do this by using the Validate by URL option. You can also paste HTML code directly into the validator by choosing the Validate by Direct Input option.

1. In your web browser, choose File > Open and navigate to the HTML5\_02lessons folder. Select the same w3\_compliant.html document that you just confirmed was valid, click Open and then OK. In Internet Explorer 9, you can also navigate to the HTML5\_02lessons folder on your computer and drag and drop the w3\_compliant.html document into your browser window.

A “valid” page can have links to images that don’t exist and may have a poor visual design.

Because we know that the page uses valid XHTML, we know that whatever problems there are with the page, they are not due to improper XHTML code. We know that there are no missing tags or misspelled tags. This can be useful for troubleshooting, allowing you to quickly identify any syntax problems.

1. In your browser return to <http://validator.w3.org> and click on the “Validate by File Upload” button. Even though you will be discussing the differences between HTML5 pages and HTML 4.01 and XHTML pages in the next lesson, you can still see the results of a page that validates as HTML5.
2. Click the Browse button, navigate to your HTML5\_02lessons folder, and select the html5\_compliant.html file. Click Open and then click the Check button to receive a successful result for an HTML5 complaint document.

| Other benefits of standards-based design |
| --- |
| W3C page validation is the most tangible aspect of standards-based web design, but there are also other benefits to creating well-structured pages, including:  Less code: Using HTML and CSS allows you to create similar pages with fewer lines of code - less work for you and faster download times for the viewer.  Ease of maintenance: Less code means a website that is easier to maintain. This helps you, the author of a page, as well as any members of a team working on maintaining or revising a website.  Accessibility: Web documents marked up semantically, meaning those that use the best HTML tag for the job, can be easier to navigate by users with visual impairments and the information they contain is more likely to be found by a visitor to the site.  Search engine optimization: Web pages with clear and logically named sections, both within the code and also within page content, are easier for search engines to index and categorize because content that is organized and well-labeled is easier for search engines to evaluate content and relevance of content on the page.  Device compatibility: Websites that separate the structure from the style are more easily repurposed for mobile devices and other browsers. CSS also allows for alternative style sheets that optimize the appearance based on the device being used to view the page. |

## HTML structure

One of the most important concepts to understand when designing web content is the nested structure of HTML documents. Elements are often nested within each other. You will often start with the HTML structure first and then begin to style it with CSS.

As an example, let’s look at the basic elements that are in virtually every web page:

<html>

  <body>

  </body>

</html>

In this example, the <body> element is nested within the <html> element. In other words, <body> is placed between the opening <html> tag and the closing </html> tag, so nested tags are those that are placed between other opening and closing tags. These two elements, <body> and <html>, form the structure of all web pages; when a browser opens an HTML document, it looks for this structure.

Content within the body tag is visible on the page as it is displayed within the web browser.

<html>

  <body>

The internal resource and information center for all employees of Worldwide Apparel

  </body>  
</html>

In HTML documents, some of the content is displayed to the viewer in their browser, but there is also other code on the page that is hidden from view, but useful for the browser, search engine, or site developer. Examples of this hidden code include scripts to add interactivity, code to help search engines categorize the document, and the styles that define the appearance of the page. This code is often found inside of the <head> element, and the <head> element is nested within the <html> tags.

An example of this is:

<html>

  <head>

  </head>

  <body>

The internal resource and information center for all employees of Worldwide Apparel

  </body>

</html>

In the above example, there is no content in the <head> element just yet. Notice that the <head> element is nested within the <html> element, but is not nested within the <body> element. The <head> element opens and closes before the <body> element starts.

The <body> element contains text, but it is lacking context, so neither you nor a search engine can determine if it is a heading, list, quotation, or some other type of content. To define the text as a paragraph, the <p> tag is used:

<html>

  <head>

  </head>

  <body>

<p> The internal resource and information center for all employees of Worldwide Apparel </p>

  </body>

</html>

The paragraph element is now nested within the <body> element, which, in turn, is now nested within the <html> element. You will now open this document in a text editor and add to the file.

1. Open your text editor and then choose File > Open and navigate to your HTML5\_02lessons folder. Depending on which text editor you are using, you may need to select “All Files” instead of “Text Documents” in order to see the file. Choose the 02\_index.html file and then click Open.
2. Choose File > Save As, name the file **02\_index\_work.html** and then click Save. This creates a copy you can work off and maintains the original file as a backup.

To get a better understanding of the structure of HTML and nesting of tags, you will add a hyperlink to this document linking the words *Worldwide Apparel* to an external website.

1. In the last paragraph that reads “All content on this site is the copyright of Worldwide Apparel,” click once before the word Worldwide, and then type the following code: <a>. This <a> is the opening tag for the anchor element, which you use to link to other pages in your site or elsewhere on the Internet.
2. Click to the right of the word Apparel and type </a>. This is the closing tag for the anchor tag and is required in XHTML.

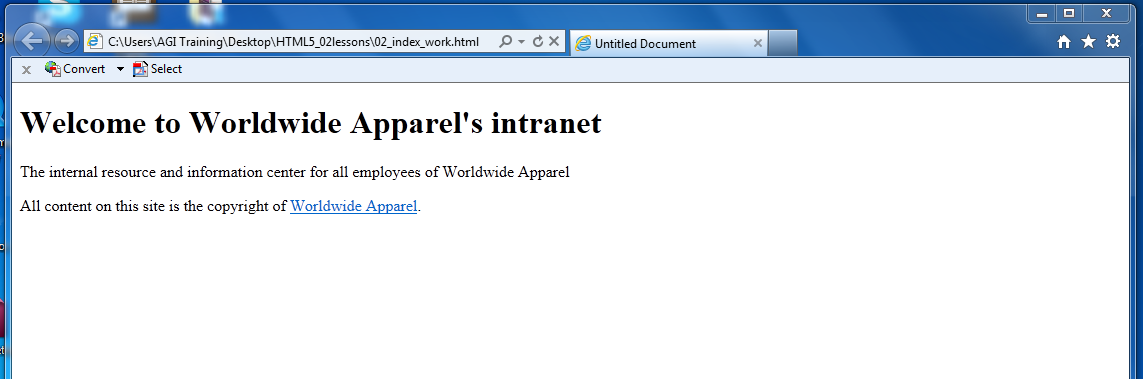
Some web editors, such as Microsoft’s Visual Web Developer Express, may be set to automatically complete closing tags. You may choose to manually delete this closing tag or cut and paste it into the correct location. Alternatively, most text editors allow you to change the “Auto complete” feature. It will be different for every text editor but in Visual Web Developer Express 2010 you can change this preset, by choosing Tools > Options > Formatting and then deselecting the “Auto insert close tag” checkbox.

To finish the job of creating a link, you need to add the destination of the link with the href attribute.

1. Click between the letter a and the closing bracket (>) in the opening tag. Press the spacebar once to add a space and type **href=""**. The complete code should now read <a href="">.

You now have an anchor tag and the href attribute. To finish the job of creating a hyperlink, you need to add the value of the attribute. In this case the value will be a URL - a web address.

1. Click inside the quotation marks and type **http://www.digitalclassroombooks  
   .com/worldwideapparel**. This completes the destination and with all the pieces in place, you now have a complete hyperlink.
2. Choose File > Save, and then preview the page in your web browser by either opening your browser and choosing File > Open and navigating to the file you just saved, or by Ctrl + clicking (Mac OS) or right-clicking (Windows) the file and directing your operating system to open the file with your web browser. The link has the standard blue underlined appearance of a hyperlink that you have not yet visited.



A simple hyperlink created by the use of the <a> tag and href attribute.

1. Close your browser and return to your text editor.

## Placing images in HTML

To add images to an HTML document, use the <img> tag. Like the anchor tag, the image tag does nothing by itself. The image tag relies on attributes and values that specify the image to display. Here you will insert an image into the HTML code.

1. Locate the first paragraph and then click once after the closing paragraph tag </p> and press Return or Enter to go to the next line. Type <img />.

The image tag is in a special category of HTML tags that are self-closing. You do not need a pair of tags with the image tag; one tag is sufficient, but it is important that you type this tag correctly. There is a space between the img and the /. This satisfies the requirements of XHTML syntax, and you will specify the exact image to use in the space between the img and /.

1. Click once to the right of the text img, press the spacebar, and then type src="".

src is the source attribute, and you will specify a value, which is the location (URL) of an image which will display on the page.

1. Click between the quotation marks that follow the src= code and type images/familysitting.jpg.

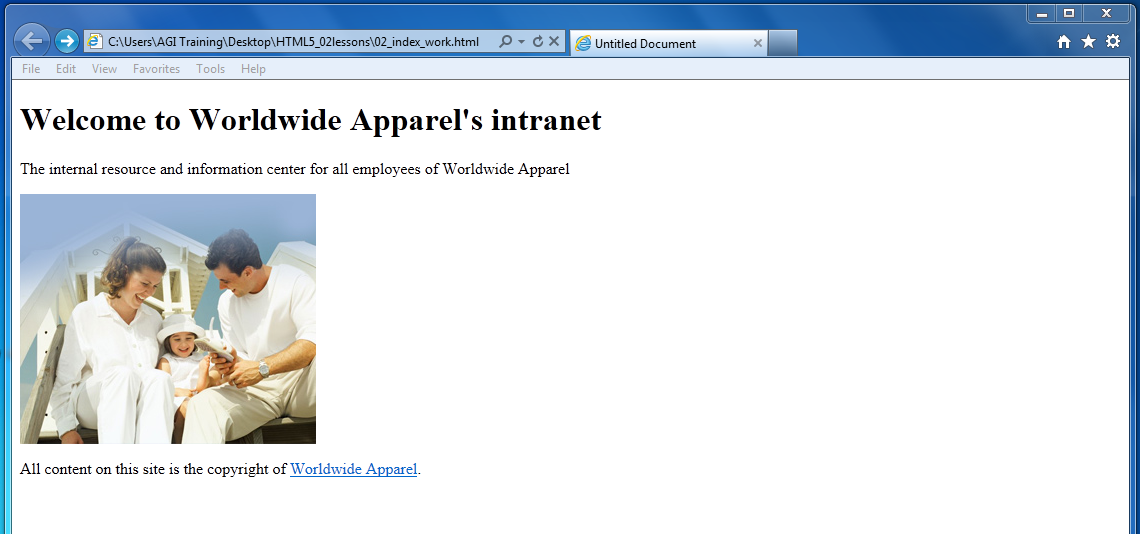
Your img code should now look like this:

<img src="images/familysitting.jpg" />

This code tells a web browser to look inside the images folder and display the file familysitting.jpg. In the next few steps, be sure to maintain the extra space between the last quotation mark and the closing tag. You will be adding an alt tag.

This alt attribute represents the text equivalent for the image and is required if you want your page to be valid. Alt attributes help those who use screen readers to navigate the web. They also appear in browsers if the image is broken or missing for some reason.

1. Click to the right of the last quotation mark that follows the familysitting.jpg file name and press the spacebar. Type alt="".
2. Click inside the quotation marks you added in step 4 and type Family sitting on steps.
3. Choose File > Save and then preview this page in your browser.



Images are added to pages using the src element and a reference to the graphic.

Both the src attribute and the alt attribute are required for fully valid XHTML. There are also optional attributes that you should consider. We’ll look at two optional attributes: height and width.

1. Return to your text editor and click to the right of the last quotation mark following the alt attribute, press the spacebar, then type width="296" height="250". These attributes tell the web browser how large the image should be displayed on the page, and although these values are optional, they do help the browser render the page more efficiently and should be used whenever possible.

| Using optional attributes |
| --- |
| Many of HTML’s optional attributes fall under the category of best practices. *Best practices* is an umbrella term used to describe the accepted way of doing something in web design. There are generally logical reasons behind best practices; for example, setting the width and height creates a placeholder for the images even if they haven’t loaded due to a slow Internet connection. Without the placeholder created by the width and height values, the page layout will change as the images load. |

1. Choose File > Save, and then preview your page in the browser to see your image. Keep this document open as you will be working with it in the next exercise of this lesson.

The role of CSS

Cascading Style Sheets (CSS) use a separate language from HTML. CSS allows you to apply consistent styling of elements across all pages on your site, so that all headings, lists, and paragraphs look and act the same on every page of a site.

| How we refer to CSS syntax in this book |
| --- |
| Before you begin to work with CSS, we need to explain how we will refer to the various parts of CSS syntax throughout this book. This is not as easy as it sounds because there is a gap between the official specification of the CSS language and the way designers often refer to CSS in the “real world.” Nevertheless, here are the fundamentals: the following code is what we refer to as a rule in CSS:  There are various components to this rule, as follows:  *A.* Selector. *B.* Declaration. *C.* Property. *D.*Value.  We will refer to each of the various components from time to time throughout the book, so if we ask you to change the value “blue” to “red”, you should know what to do. Or, if we ask you to locate and change the h1 selector to an h2 selector, it should make sense.  On a day-to-day basis, most designers aren’t always so specific. For example, the rule above might be referred to as a “style,” “style rule,” “the h1 rule,” or “the CSS rule for h1.” Also, as you can see above, the official name for the pair of the property and the value is called a declaration. Again, in everyday use, the use of the term “declaration” is not common and most designers will use the term property or properties interchangeably. |

## Styling a heading

To get a sense of how CSS works, you’ll create a simple CSS rule that changes the style of a heading in your page. In your 02\_index\_work.html page, you already have the content “Welcome to Worldwide Apparel’s intranet” nested inside an <h1> tag. Perhaps one of the best ways to begin thinking about how CSS works is to consider how the default style of this heading is rendered in the browser.

1. Within your browser, examine the heading. The style and formatting instructions are being provided by the underlying “rules” that your browser uses to display <h1> elements. The size, color, and font are provided by the browser because no other formatting instructions have been specified. The browser only knows that this is a heading 1 element and displays it accordingly. For example; the default size is 16 pixels, the color is black and the font is Times New Roman. This is basically true whether you are using Internet Explorer, Firefox or some other browser. You will now redefine this style with a CSS rule.
2. In your code, locate the <title> tag, click once at the end of the line following the closing tag, then press Return or Enter to add a new line of code. Type the following:

<style type="text/css">

1. Press Return or Enter three times and then type </style>. This is currently an empty style element which you will use to place your rule for the style of the <h1> element.

The <style> element is nested within the <head> tags of your page. In HTML, everything nested inside the <head> section is not rendered by the browser on the page. For example, there is also a <title> element inside this section; this title appears at the top of the web browser, not on the actual page.

1. In the empty line below the opening <style> tag, type the following:

h1 {

This is your selector. The selector is the HTML element you want to style, in this case, the heading 1 element.

1. Press Return or Enter and then press Tab to place your cursor below the curly bracket if your editor doesn't already move the cursor for you. The tab is optional, but it helps make your CSS more readable. As you will soon see, the number of lines in this rule will grow and it’s worthwhile keeping the code easy to read.
2. Type the following code below h1 {:

color:red;

The word *color* is referred to as a property in CSS syntax and the word *red* is a value*.* The combined pair of a property and a value is called a declaration.

1. Press Return or Enter again and on the next line, type } which is the right curly bracket character.

This closes the curly bracket you added in step 4.

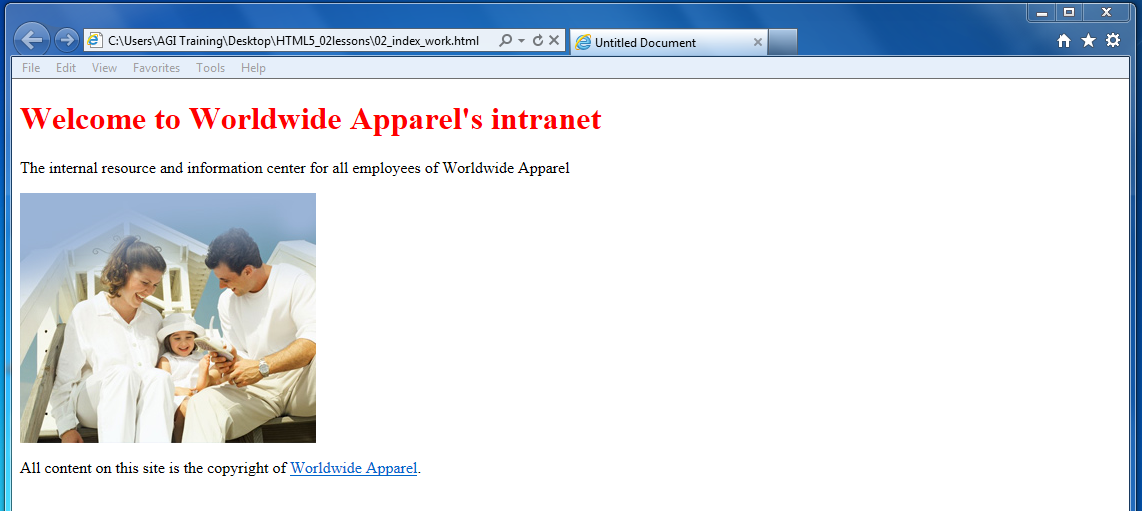
You now have three lines in this rule:

h1 {

  color:red;

}

1. Choose File > Save, and then preview your page in the browser. The heading is now a red color and you have successfully created your first CSS rule. Close your browser and return to your text editor.



Your H1 color is now being styled by a CSS rule.

1. In the HTML file, select the word red and replace it by typing the following for the color value: #FF0000. This hexadecimal color is the equivalent of the color name red. You can use either named colors or hexadecimal colors when defining colors using CSS.

Save your file and then preview it in the browser. The color remains the same. Hexadecimal colors are a more common method for describing colors on the web.

| Hexadecimal colors |
| --- |
| Color in both HTML and CSS is referred to by a six-character code preceded by a pound sign. This code is called hexadecimal code, and is the system used to identify and apply color to elements. You can reproduce almost any color using a unique hexadecimal code. For example, the following code is dark-red: #CC0000.  The first, middle, and last pair of digits in the hexadecimal code corresponds to values in the RGB spectrum. For instance, white, which is represented in RGB as R:255 G:255 B:255, is represented in HTML as #FFFFFF (255|255|255). A program like Photoshop will allow you to choose a specific RGB color in the Color Editor and give you the equivalent hexadecimal color for use in your code.  There are also online references you can use to locate or “mix” hexadecimal colors, such as: www.w3schools.com/Html/html\_colorvalues.asp |

The rule you just created uses what is known as a “type selector” since it targets every instance of the h1 element type in your document. Type selectors assign CSS properties to an existing HTML tag. In this case, the <h1> tag. All <h1> tags on this page will be displayed as red. Type selectors are more commonly known as tag selectors. It is rare that you will actually hear someone use the phrase “type selector”, but that is the official name for it, so we mention it here.

You will now get an introduction to another category of CSS styles known as a class. You will also work with the HTML <span> element, which separates and controls inline content, such as a sentence within a paragraph, or an individual word within a sentence.

## Understanding class styles and the **<span>** element

Tag selectors are frequently used, but they can only be applied to HTML elements. When you want to style something that does not map directly to a tag, such as changing the color of a single word within a paragraph, a tag selector is not the best choice (or even possible in some cases). In this case, you can use a class selector, which is a CSS rule that you can apply to any number of items on a page. Class selectors have flexible naming options, but you should choose names that describe what they do. For example, you may wish to name class selectors as .caption, .imageborder, or .redtext. In this exercise, you will create a class style that applies the color red to the words *Worldwide Apparel* in your paragraph.

1. Place your cursor on the line immediately below the closing curly bracket for the h1 rule, then type the following:

.red {

  color:red;

}

Note the period at the beginning of the class selector. The text following the period is the class name. You can use any name you wish, but the period is required at the start to identify it as a class. The declaration color:red is the same as in the last exercise, but in this case the selector is not the h1 element.

The class name can be anything you want, but it must have the period at the beginning to identify it as a class. Next, you’ll apply this class to the words *Worldwide Apparel* in order to style them red. To do this, you will need an HTML tag <span>.

1. In the first paragraph <p> within the <body> tag, locate the words *Worldwide Apparel*, click once to the left of it, and then type:

<span>

1. Click to the right of the words *Worldwide Apparel* and add a closing span tag </span>.

Your code should look like this:

<p>... employees of <span> Worldwide Apparel</span> </p>

Save your file. If you were to preview the page in the browser, you would see no change. The <span> tag in HTML is an empty tag; it does nothing on its own and needs to be paired with a style. The <span> tag defines the beginning and end of where the style will be applied within the paragraph, but it does not apply the style on its own, and does not define the style.

1. Locate the opening <span> tag you inserted. Click once after the word span but before the > bracket, press the spacebar, and then type the following:

class="red"

The code should now read:

<p>... employees of <span class="red"> Worldwide Apparel</span> </p>

1. Locate the second paragraph that contains the copyright information and within the open <p> tag type the following:

<p class="red"> All content on this site is the copyright of <a href="<http://www.digitalclassroombooks.com/worldwideapparel>"> Worldwide Apparel</a>.</p>

1. Save your page and preview it in your browser. The entire paragraph (with the exception of the hyperlink) text is now styled red. Here you can see the flexibility of CSS classes; they can be used and reused to style different parts of your page. The hyperlink did not change color because it has a different style affecting its appearance (the <a> tag). In a later exercise you will learn how to change a hyperlink style.



CSS class styles can be applied to inline elements as well as block elements such as the last paragraph.

1. Close your browser but keep the document open in the text editor, as you will be working with it in the next exercise.

## Three ways to use styles

In the previous exercise, your styles were located within the head section of the page. This type of style is called an *internal style sheet.* In addition to internal (or embedded) style sheets, there are external style sheets and inline styles.

An *external style sheet* is a separate document with the file extension .css. When using an external style sheet, all styles reside inside the style sheet document and you link it to your HTML pages. While internal style sheets affect only the page on which they exist, external styles can be applied to multiple pages.

*Inline styles* are the third type of style but are used with much less frequency than internal or external styles. With inline styles, the style rules are nested inside the HTML tags. An example of an inline style that colors a heading red would look like this:

<h1 style="color:red">Worldwide Apparel</h1>

Inline styles are powerful because they override both internal and external styles, although they only apply to a single tag at a time. This embedded nature of inline styles means they are not easily re-used. In the simple example illustrated above, you can see the style for the color red is nested inside the <h1> tag. If you had 50 <h1> elements throughout your website and were using inline styles, you would need to add this style code 50 times. Furthermore, if you decided to change the color to green, you would need to locate and modify all 50 uses of the style. Inline styles are useful for single overrides, or when an internal or external style sheet may not be available; a good example of this is HTML-based e-mail.

You will not be using inline styles very often in this or the following lessons, which is a reflection of the current state of web design. Working with a combination of internal and external styles is the most common practice of web designers today.

## Internal versus external style sheets

Internal style sheets are CSS rules contained directly within a document, using the <style> tag. The entire style sheet is contained within the opening and closing <style> tags and these style tags are located inside the <head> section of the document.

With internal style sheets, CSS rules apply only to the HTML in the current document. For example, if you had a 20 page website and were using internal style sheets, you would need to create a separate style sheet in each of the pages. A change to the style would require you to update the internal styles in each of the 20 separate pages.

External style sheets are CSS rules saved in a separate document with the file extension .css. External style sheets place CSS rules for a site in a single document. You can attach the .css file to an unlimited number of HTML pages. This provides more flexibility. If a style rule for the paragraph element is defined in the external style sheet, all paragraphs across the site can be modified with a single step. You will now make an external style sheet and then attach it to a new HTML page.

## Creating an external style sheet

An HTML page does not have to be limited to just one external style sheet, and many large websites will break-up their styles into separate pages, making them easier to organize and maintain. You can even use style sheets for specific functions such as printing a page or for displaying a site on mobile devices. Specific style sheets can even be used to make sites compatible with older web browsers when they are used to visit sites you create.

In this exercise, you will create a new external style sheet, move the style rules from your current document to the external style sheet, and then attach the style sheet to a new HTML page.

1. Choose File > New File and choose the Style Sheet template and open it if your web editor offers the choice.

The text editor you are using may have a different menu command. You may need to choose the equivalent command. For example, many editors will allow you to choose File > New CSS document.

1. Choose File > Save As and name the document styles.css and save the file into the HTML5\_02lessons folder. An external cascading style sheet has a specific .css file extension, but it is simply a text file.
2. Switch to the HTML document from the last exercise (02\_index\_work.html), but keep the style sheet open as well.
3. In the HTML document, locate the rules you created within the <style> tags, and then select them. Do not select the style tags themselves, just the rules that start with h1 and end with the closing bracket }.

  
Select just the style rules, not the <style> tag.

1. Choose Edit > Cut, then switch to the styles.css file. If there is any placeholder code, you should overwrite it by selecting it and then choose Edit > Paste to paste the rules into the external style sheet document. Choose File > Save to save your style sheet.

The entire external style sheet acts as a substitute for the <style> tags in the HTML document. Now that you have moved the rules to this document, you need to link it to your HTML page so that a web browser knows where to find the style rules that apply to the HTML.

1. Switch back to the 02\_index\_work.html page and choose File > Save. You will now add the <link> tag, and point to the styles.css document. This step is crucial because if you do not link to the style sheet, the HTML page will have no styles.
2. Place your cursor after the closing style tag </ style> then press Return or Enter to start a new line. Now type the following:

<link rel="stylesheet" type="text/css" href="styles.css" />

You have just added the rel, type, and href attributes. You may recall the href attribute from when you added the hyperlink in an earlier exercise. In order for your external style sheet to work properly, the name of the file, and the path to the file must both be accurate.

1. Choose File > Save and then preview the HTML page in your browser. The page should not change, as the same styles are being used; they are simply being applied from outside the document.
2. Close the browser and return to your text editor. You’ll now create a new HTML document, and add the same link to the external CSS file, seeing how the rules are applied.
3. Choose File > Open and open the file test.html in the HTML5\_02lessons folder. This is an empty HTML document.
4. Continuing to work in your text editor, switch back to the 02\_index\_work.html file and select the entire <link> element you typed in step 7:

<link rel="stylesheet" type="text/css" href="styles.css" />

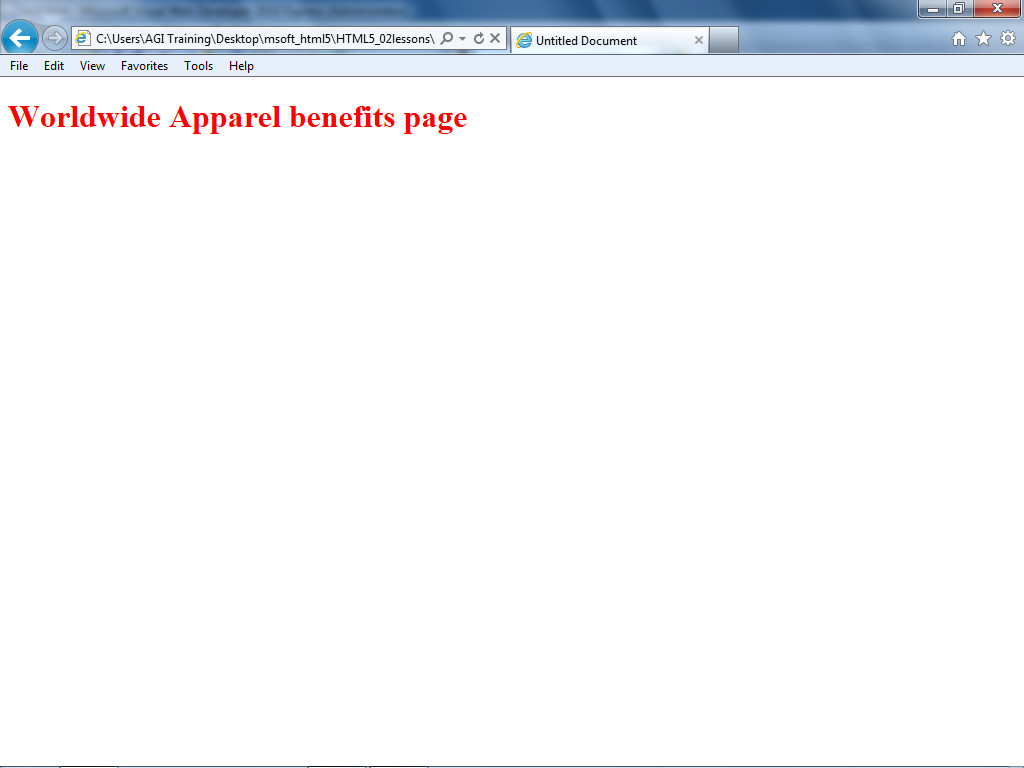
Choose Edit > Copy.

1. Switch back to the test.html document, click below the <title> element, and then choose Edit > Paste to place the <link> element. Save the file by choosing File > Save.

The external style sheet is now attached to this HTML document. Any HTML tags you add to this new document will be styled if there is a corresponding rule in the CSS file. For example, you will now add a new heading 1 element and the content will automatically have the color red.

Click inside the <body> element and type:

<h1> Worldwide Apparel benefits page </h1>

Save the file and preview it in your web browser.

The <h1> tag gets its style from the external CSS style sheet you created.

The heading is red because the style rule for the <h1> element is color:#FF0000 and because this rule is located in an external sheet and linked in two places: the 02\_index\_work.html and test.html pages. Because of this, you can control the style of both HTML documents from a central location.

## What makes styles cascading

You’ve seen three different places where CSS rules are found: inline, internally, and externally. If there are conflicting definitions of styles between inline, internal, and external styles, the inline style will be used because it is closer to the HTML source. The internal style sheet takes precedence over an external style sheet, and definitions used in an external style sheet are used only if they don’t conflict with either inline or internal styles.

## Working with text and fonts on the web

When designing for the web, you can format text in a way that is similar to desktop publishing and word processing applications, but there are important differences to keep in mind. When you specify that a specific font be used, that font needs to be installed on the user’s computer when the web page is rendered on the viewer’s computer or device. If the user does not have this font, the browser replaces it with another font.

Because you don’t know what fonts are installed on viewers’ computers, and because the web browser of a viewer might substitute fonts, your design intentions for text might not be faithfully reproduced. One option is to use fonts that you are sure will be found on most computers. Unfortunately, only a handful of fonts can reliably be found on virtually all computers around the world.

| Web-safe fonts |
| --- |
| Following is a list of the most reliable fonts for web use:   * Arial * Verdana * Georgia * Times New Roman * Courier * Trebuchet * Lucida * Tahoma * Impact   The list is small because it takes into account both Mac and Windows platforms and assumes that there may still be older computer systems that are active and accessing the web. These older systems had a more limited font selection than today’s systems, and so a designer needs to consider this when choosing fonts.  The above list is also limited for stylistic reasons. Both Courier and Impact, for example, are used infrequently because although they are widely available, their distinctive styles limit their everyday use. |

One of the solutions to the lack of fonts on the web is to use a font stack. In CSS, a font stack is a list of multiple fonts that the web browser uses in an attempt to display text onscreen.

The following CSS code shows an example of a font stack:

font-family:"Helvetica Neue", Helvetica, Arial, sans-serif;

In this example, the browser first looks for the Helvetica Neue font on the user’s system. Notice the quotation marks in this example. In most cases, when specifying a font, quotation marks are unnecessary, but in some cases, the quotation marks are needed to help the user’s computer choose the right version of the font. If the user doesn’t have Helvetica Neue, then the browser looks for the more generic version of Helvetica. If Helvetica is absent, the browser uses Arial, which is a font that is extremely similar to Helvetica. If for some reason Arial is not on the system, the last choice is sans-serif, which allows the system to use any sans-serif font it can find on the system. Sans-serif is the generic definition for all fonts that do not have small strokes (called serifs) at the end of each character. Examples of serif fonts are Times New Roman and Georgia.

## Setting a font-family

In this exercise, you will set your font-family for an entire page, and then set the font family for your headings.

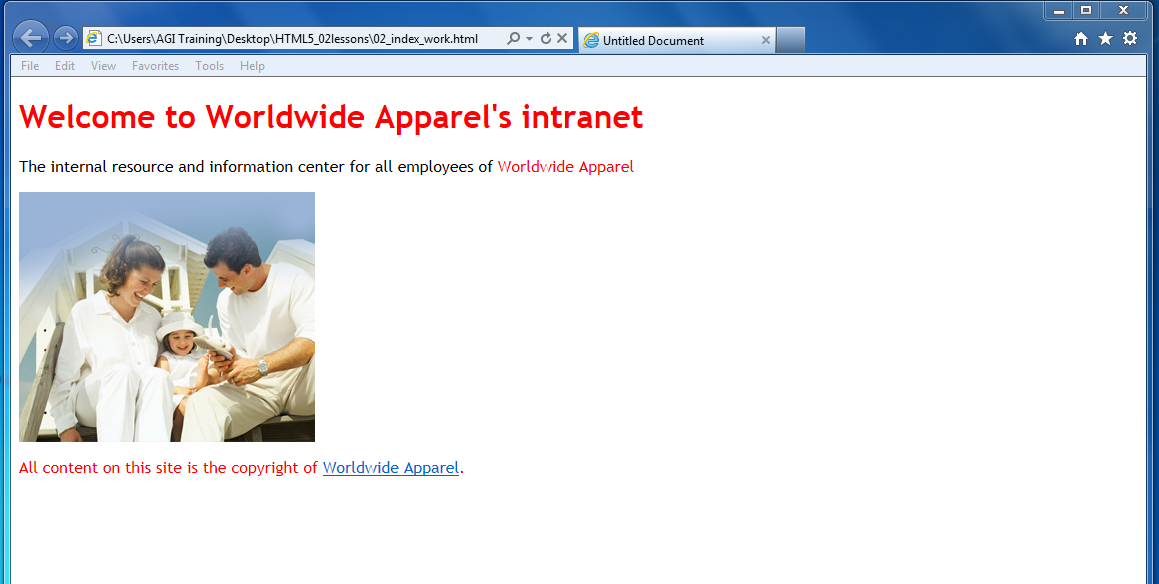
1. In your text editor, switch over to the 02\_index\_work.html file if it is still open, or open it if necessary. You will be styling the text on this page but since your styles are located within the external style sheet you will need to add any new styles to that style sheet.
2. Switch over to the styles.css document you created in the last exercise. You currently have styles for a heading 1 and a class named .red. You will now add a style for the body of the page.
3. At the top of your document, above the rule for the h1 style, type the following :

**body {**

**font-family:"Trebuchet MS", Tahoma, Arial, sans-serif;**

**}**

1. Choose File > Save, and then preview your 02\_index\_work page in the browser. As noted above, your web browser renders Trebuchet if you have it on your system; if you do not, it displays Tahoma; and if you don’t have Tahoma, Arial will be chosen and if Arial is not present you still see a sans-serif font.



When you define the font family Trebuchet for the body rule, all your text is set in this font.

All of the text on your page is rendered in Trebuchet because the only style set is for the body. Remember that the HTML body tag defines all of the elements on the page. Now you will set a specific font family for the paragraph element.

1. Return to the style sheet in your text editor and underneath the rule for the body add a new style rule for the paragraph <p>, as follows:

**p {**

**font-family:Georgia, "Times New Roman", Times, serif;**

**}**

1. Save your document and preview 02\_index\_work.html in your browser. Now there is a specific rule for paragraphs, they are styled as Georgia. The heading is still using Trebuchet, which you defined in the body style. Keep in mind that although you could have referenced any font on your own system, Trebuchet and Georgia are good primary choices because they are installed on the majority of user’s computers around the world.

| The promising future of web fonts |
| --- |
| The lack of choices for using fonts on the web has been a source of frustration for web designers for many years. The situation is improving as several companies have created solutions to enable your pages to display on a browser with the fonts you’ve specified as a designer.  The landscape of fonts on the web is changing, however, with the rise of web fonts and in particular the CSS3 @font-face feature. It is now possible to add a broader range of fonts than ever before and still make sure that your users will see the desired effect.  For a more advanced discussion of this topic as well as step-by-step instructions on how to add web fonts to your web pages refer to Lesson 11, *Styling with CSS3* in the book “HTML5 Digital Classroom”. |

## Sizing text with CSS

When using CSS to style text for the web, you have a few options for the unit of measurement. The CSS property that controls the size of your text is named font-size.

You can control the font-size property in a few different ways:

* **Absolute-size**: A set of keywords that indicate predefined font sizes. Named font sizes scale according to the user’s font setting preferences. Possible values include xx-small, x-small, small, medium, large, x-large, and xx-large.
* **Length**: A number followed by an absolute units designator (cm, mm, in, pt, or pc) or a relative units designator (em, ex, or px).
* **Percentage**: An integer followed by a percent sign (%). The value is a percentage of the font size of the parent object.
* **Relative-size**: A set of keywords that are interpreted as relative to the font size of the parent object. Possible values include larger and smaller.

Choosing the unit of measurement for the font-size in a web page is an important decision and not as easy as it might seem. The main difficulty in selecting a size has to do with monitor resolution and the different way that web browsers over the years have rendered text.

Text on smaller monitors often looks different than text on larger monitors; with a bit of forethought, you can address this. In addition to the monitor resolution issue, you must also consider the way that different web browsers interpret how text is rendered. For example web browsers allow users to resize their text manually. Furthermore, there is a growing audience that browses the web with mobile devices, which makes sizing your text even more important.

## Pixels and points are not ideal choices for sizing text

Setting font size in points might come naturally to you if you have worked in print design, or if you have created web graphics, you might be comfortable measuring using pixels. The font-size property in CSS allows you to use both forms of measurement. In the following example, the first CSS selector shows you a paragraph rule for points, while the second one shows you a paragraph rule for pixels:

p {

font-size:12pt;

}

Points

p {

font-size:12px;

}

Pixels

Even though points are supported, it is bad practice to use them and not advised for web design. Points are a system of measurement designed for print, and although available for use, they indicate an absolute unit of measurement and they don’t translate well to the screen. Pixels, on the other hand, are the unit of measurement often used for screen-based graphics. Monitor resolution sizes are measured in pixel units. In an ideal world, designers could reliably use pixel sizes for their fonts because they are relative units and are designed to scale natively. Throughout the years, web browsers have addressed the way text is resized in different ways, for example, Internet Explorer 6 and 7 do *not* resize pixel based text if the user chooses to override the default settings.

Virtually all web browsers include a text resize option. This option is often found in the View menu. In some modern browsers, the text-resize option is located in a submenu called Zoom. Many browsers also use the keyboard shortcut Ctrl + + (plus) and Ctrl + – (minus) to increase and decrease the text size, respectively. On the Mac OS, these shortcuts are Command + + [plus] (and Command + - [minus].

## Using a combination of percent and the em measurement

One way to sidestep the issue of text-resizing for units based on pixels is to use alternative units. Here you will create reliable font sizing using a combination of percents and ems. To get a sense of how these work, you will apply some CSS styling to an HTML file that we have begun styling for you.

1. In your text editor, choose File > Open and navigate to the HTML5\_02lessons folder. Locate the 02\_sizing.html file and click OK. Choose File > Save As and name this file 02\_sizing\_work.html.

This file has four blocks of text: a heading 1 <h1>, a heading 2 <h2>, and two paragraphs <p>. This file uses an internal style sheet for convenience and Trebuchet has been set as the body style. You will start by setting different properties for the body to see their effect.

1. Before making any changes, you should know what the page looks like in its default state. Preview the page in your default browser. All browsers set a default size for the text if there is no user styles being applied; in most cases 16 pixels is the value used for the body (in this case, the paragraphs are inheriting the body’s value). Close your browser and return to your text editor.
2. In the style rule for the body, type the following line:

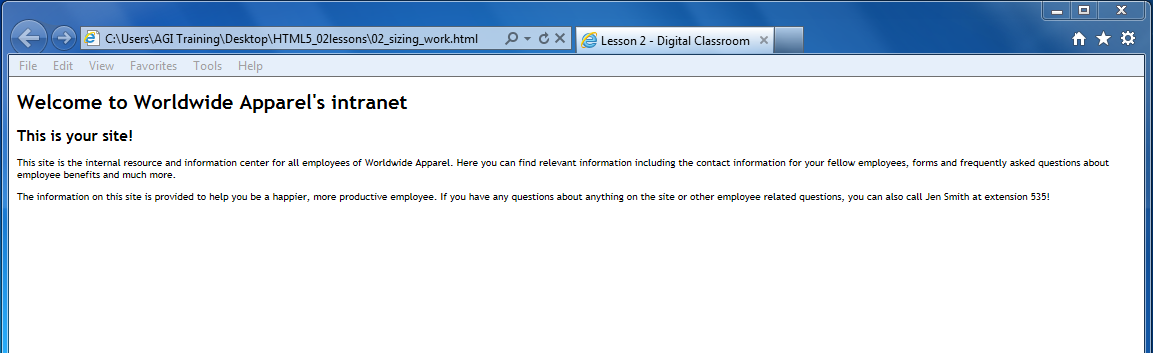
body {

font-family:"Trebuchet MS", Tahoma, Arial, sans-serif;

**font-size:10px;**

}

1. Save your file and then preview your page in the browser. Note that all your text is smaller. This is because the body style defines the baseline size for text on your page and the other elements such as the paragraphs and headings respond accordingly.



Setting the font size for the body style makes all the text on your page smaller.

1. Return to your text editor and change the following value in your font-size property:

font-size:small;

1. Again, save your file and preview the page in your browser. All your text is slightly larger than the 10-pixel value you set in step 3. As noted above, the value small is an absolute size unit of measurement called a keyword. Web browsers have pre-defined sizes assigned to keywords, and though keywords are preferred by some because they avoid the whole issue of using units, they often don’t offer the level of control that designers prefer.
2. Return to the text editor and change the following value in your font-size property:

font-size:100%;

1. Save your file and preview the page in your browser. You might notice that there is no difference between this size and the size of the text at the beginning of the exercise (when no font-size was defined). This step explicitly defines the font-size for the body to be the same size as the browser-defined font-size.

You will have to take a small leap of faith here and realize that the technique you are learning addresses some particular resizing problems in older browsers. Taking care of these problems now will mean fewer problems in the future and provide better compatibility for all of your pages.

1. In the style rule for the paragraph, type the following line:

p {

font-size:1em;

}

The unit of measurement called an em is very similar to pixels in that it is designed to scale; the main difference is that ems are not tied to the monitor resolution, while pixels are related to the monitor resolution. Ems may not be intuitive at first, but understanding how to use them will pay off in the future.

1. Save your file and then preview the file in your browser. Depending on which browser you are using, you will probably not see any changes in your page. This is because an em value of 1 is tied to the font-size of 100% that you defined in the body.

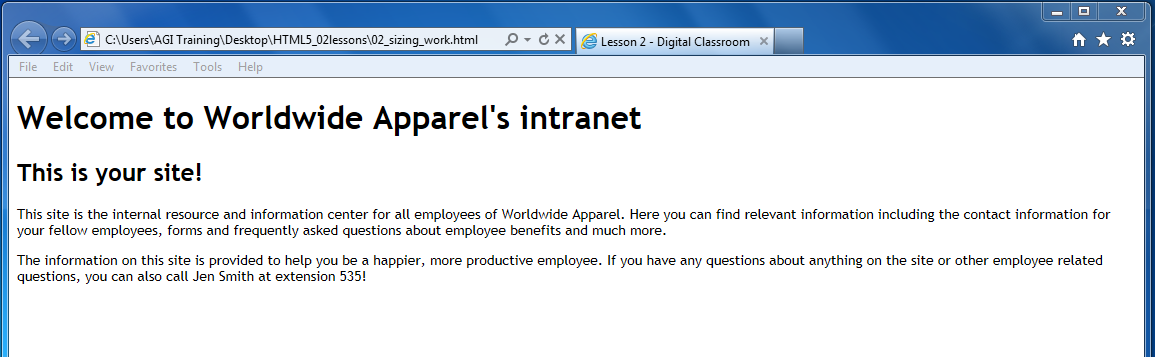
It may help to understand this relationship in an equation form: 1 em = 100% = 16 pixels.

Here, the paragraph size is the 1em value, the font-size for the body is 100%, and the default font-size for the web browser is 16 pixels. Once you understand this relationship, you can begin to change the value of the em in order to enlarge or reduce the size of your text.

1. Close your browser and return to the text editor. In the rule for the paragraph, change the following value:

font-size:0.875em;

1. Save your file and preview the page in your browser. Your paragraph text is now smaller.



Setting the font size for paragraph in em units.

The reason for using the precise 0.875 value is because it is the font-size equivalent to 14 pixels.

If you’re starting to think that web design is all about math, don’t worry too much. It all gets easier from here. If you are interested in understanding the math more deeply, you multiply the em value (0.875) by the browser’s default pixel value (16) to arrive at the 14-pixel number.

1. You will now size your headings using ems as well. For the h1 property, add the following line:

h1 {

**font-size:1.5em;**

}

This scales the top heading to 1.5 times the size of your body text; in this case, it is the equivalent of 24 pixels. Save your file and preview the page in your browser to see the effect.

Now the h1 is approximately the same size as the h2, which isn’t particularly logical, so you will reduce the size of the h2 heading as well.

1. For the h2 property, add the following line:

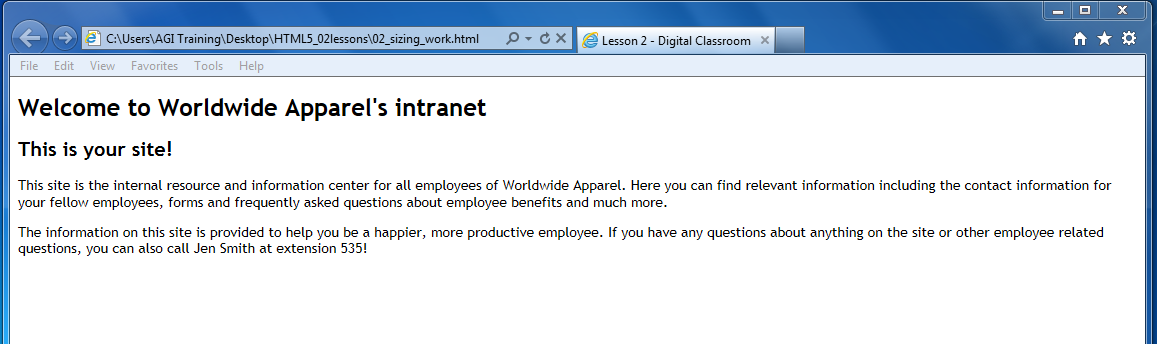
h2 {

**font-size:1.25em;**

}

This scales the top heading to 1.25 times the size of your body text, which is the equivalent of 20 pixels.

1. Save your file and preview it in the browser. You now have text proportioned as needed.



Setting the heading 2 element to 1.25 ems.

Remember that one of the main reasons why ems are used is to adjust for users who resize the text in their browser. You can simulate this by going into your browser and enlarging the text size. For example, in Internet Explorer 9, you can choose View > Text Size and then choose from a number of options. You can see that the text responds well to this enlarging and reducing. When you are finished, be sure to return the text size to the default setting. Most browsers have a command to allow you to do this.

The issue of browsers resizing text is a bit more complicated because some browsers use a zoom feature that increases or decreases magnification of the entire page. Zoom-enabled browsers may also have a text-only resize option.

There is another benefit of using ems, and this has to do with the scaling relationship between all elements that use ems.

1. Return to your text editor and in your body property, modify the following value:

font-size:**85%;**

1. Save your file and preview your page. All your text is now smaller, even though you just changed one value! This is because of the linked relationship the em has to the body element. Some designers adjust this base size if, for example, a client wants larger or smaller text across the entire site. Rather than modifying all the individual properties, having one rule control multiple font-sizes makes it easy to do.
2. Return the body property font-size to the original 100% value:

font-size:**100%;**

1. Save your document. You have completed this lesson.

In this exercise, you’ve discovered many ways to format text. When you want to style text, it is almost always best to use actual text rather than an image of text. Using actual text rather than a picture of text created in programs like Photoshop or Illustrator makes your sites more accessible to the widest audience of users, devices, and search engines.

## Self study

1. Add another heading below the last paragraph in 02\_index\_work.html as follows:

**<h2> Worldwide Apparel volunteer opportunities </h2>**

1. Create a tag style in your external style sheet that defines this heading as the color orange.

## Review

### Questions

1. What is a doctype and how does it relate to page validation?
2. In the following XHTML code, what is the attribute and what is the attribute value? What other attributes would you often find in an img element such as this?

<img src="images/familysitting.jpg"/>

1. Define the purpose of an external style sheet and one of the benefits of using an external style sheet.

### Answers

1. A doctype is a declaration at the start of your HTML document. It is used by a web browser to determine what markup language and version is used on the page. Page validation tests the syntax of your code against the specifications of your doctype. Page validation is a good way to check your page for problems such as missing tags or typographical errors in your code.
2. In this line of code, src is the attribute and the attribute value is images/familysitting.jpg. Nested inside the <img> tag is the src attribute and its value. It links to an image that is then rendered on the page. Other examples of image attributes are the alt attribute, which provides a text version of an image to devices such as screen readers, and the width and height attributes, which define the size of the image on the page.
3. An external style sheet is a text document with the extension .css. This document contains CSS rules that define the appearance of HTML elements. Because external style sheets can be linked to multiple HTML pages, they provide one central location for your styles. One benefit to this is the ability to update the style of an entire site with a single change to a CSS rule. Other benefits include the ability to use multiple style sheets for organizational purposes, and to specify specific style sheets for printing or optimize the display for mobile devices.