



# HTML & CSS CRASH COURSE

**Learn html and css with easy to follow-  
step-by-step tutorials**

**Krishna Puyed**

# HTML Tutor

## Simple HTML Editor

HTML means Hypertext Markup Language. Its basically just a text file with codes that tell the browser how to display the information. For example, you can let the browser know that a certain string of text should be displayed as a header with bold font, or that the text should be centered on the page. To let the browser know the text file contains HTML, we use the file extension .html rather than .doc or .txt or .rtf.

Since a HTML document is nothing but a text file, you can use any text editor to make one. You can use Microsoft Word, Pages, or your built in text editors provided by the operating system. However if you are a Mac user I'm going to recommend a special HTML editor which is free called Kompozer. The nice thing about Kompozer is that it allows you to preview your file in real time inside the application without having to save your html file and loading it in a browser. Although in this book we will be focusing on teaching HTML and CSS, Kompozer allows WYSWYG editing of web pages. You can download it free here:

<http://www.kompozer.net/>

A website is made up of multiple HTML pages. So each HTML file is a single web page. When you type in a websites home address, such as <http://cnn.com> or <http://nytimes.com>, what happens is the browser opens a special file named index.html. In a nutshell this is an html file no different than any other, but it has the name index that tells the browser to load this file when someone visits the website. On your server you will place the index.html file in the home directory. There are some exceptions to this but for now that is how you can view the home page of a website.

The other web pages on the site will have different names pageone.html, pagetwo.html etc. These other pages can be in the home directory or you can make a folder on your server and place the pages in there. So for example, suppose you have a website <http://acmeincorporated.com>. In the public\_html folder on the server, you would place the home page here. This would be the file index.html. You could place an about.html page in this folder as well. Then it would be referenced in the browser as:

<http://acmeincorporated.com/about.html>

Alternatively, you could create a folder in your home directory and place the about.html file in there. Lets say that we called that folder info. In that case, the web address would be:

`http://acmeincorporated.com/info/about.html`

Like a word processing document, an HTML file can include different fonts, colors, images, and links to other html pages. An HTML page can also have a style format which is done using CSS. We will see how to enter the appropriate codes to do these tasks in future chapters.

## **Your First Webpage**

Now that we have an idea of what a web page and HTML file is and how to create one, let's get our feet wet and start creating simple web pages. The first thing you need to know is how to give instructions to the browser. This is done by using tags. The format used to enter a tag is to enclose it in `<>`. You will need an opening tag and a closing tag. Inside the `<>` characters, you give the browser and instruction. So for illustration, to tell the browser that a block of text is tag\_one, the opening tag would be:

`<tag_one>`

A closing tag is indicated with a forward slash /. So to tell the browser that we are finished with tag\_one, we would write:

`</tag_one>`

Now let's get into the structure of an actual HTML document and real tags that are used. The first line in your file is this one:

`<!DOCTYPE html>`

As the name implies, this is called the Document Type Declaration. It lets the browser know what type of HTML you are using. As written above, this tells the browser we are using the most recent version of HTML which is HTML 5.

Now we need to tell the browser where our HTML actually is. That might seem strange since we just let it know that the document was an HTML5 document. However you have to specify what every block of text is. To tell the browser that the following text is html, we use the html tag as

. We also need the closing tag, so our file should look like this:

```
<!DOCTYPE html>
```

```
<html>
```

```
</html>
```

Hence anything that falls between `<html>` and `</html>` is interpreted by the browser as being html. Unfortunately we aren't done. We need a second tag called the body tag that indicates where the content of the web page is placed. So we update our file as follows:

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
</body>
```

```
</html>
```

Now we can put the actual content of our web page in between

and. To put text, you simply type it as if you were using a web processor. Following the usual practice in teaching computer topics, we'll add "Hello World" to our web page.

```
<!DOCTYPE html>
```

```
<html>
```

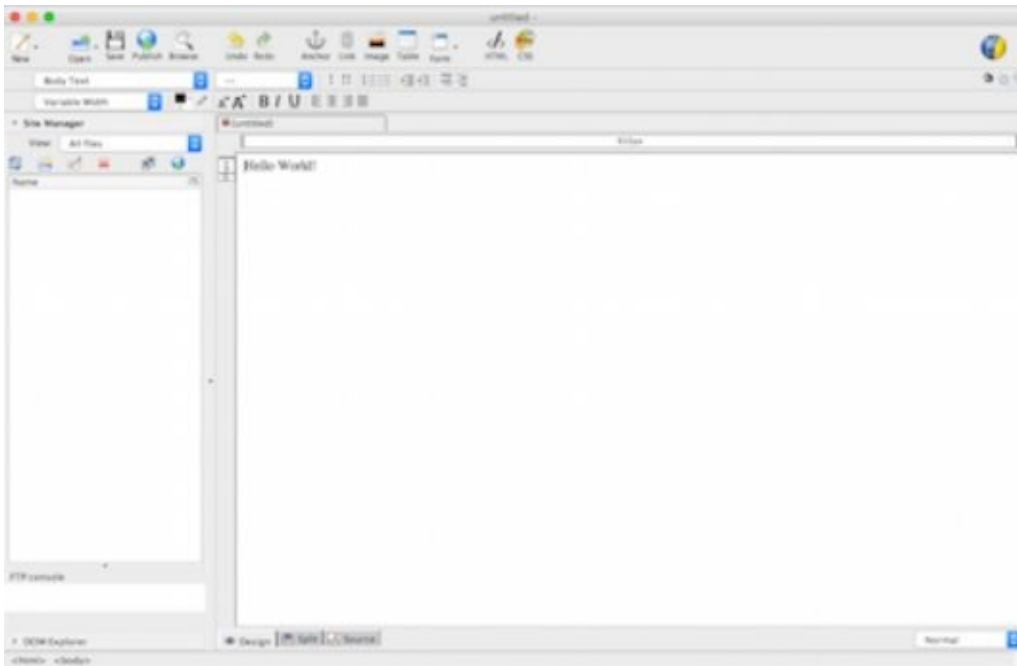
```
<body>
```

```
Hello World!
```

</body>

</html>

Here is our webpage preview in Kompozer. Not very impressive but its a start!



Now that we've learned to add text to a web page, lets learn how to use HTML to add line breaks and center text.

# Linebreaks and Center Tags

In our first tutorial we created a Hello World web page. It wasn't too exciting it just printed our Hello World message to the screen:

Hello World!

Now let's move forward by adding some text formatting. First let's add a couple more lines of text. Maybe we want to print the following:

Hello World!

My name is Joe.

My friend is Sally.

Our HTML looks like this:

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

Hello World!

My name is Joe.

My friend is Sally.

```
</body>
```

```
</html>
```

When you do this and save your html file, and then open it in a browser, what you see is this:

Hello World! My name is Joe. My friend is Sally.

So even though we put line breaks and some spacing, the browser ignores it. The browser sees one long string of text unless you add tags to tell it how to display that text.

## Centering Text

To center text, what you do is add a *center* tag. The open tag to use is `<center>` and the closing tag is `</center>`. So we can center the Hello World! string in the following way:

```
<!DOCTYPE html>
<html>
<body>
<center>Hello World!</center>
My name is Joe.
My friend is Sally.
</body>
</html>
```

This produces a web page that looks like this:

Hello World!

My name is Joe. My friend is Sally.

Notice that closing the center tag added a line break. To center all the text, we would write:

```
<!DOCTYPE html>
<html>
<body>
```

```
<center>
Hello World!
My name is Joe.
My friend is Sally.
</center>
</body>
</html>
```

Now we obtain:

Hello World! My name is Joe. My friend is Sally.

## Line Breaks

We can change the appearance of the page again and make it more readable by adding some line breaks. This is done with the tag `<br>`. Unlike other tags, a closing tag isn't necessary. You just add one for each line break you want. For now lets remove the center tag and just add line breaks. So lets add a line break after the Hello World! string:

```
<!DOCTYPE html>
<html>
<body>
Hello World!<br>
My name is Joe.
My friend is Sally.
</body>
</html>
```



This produces:

Hello World!

My name is Joe. My friend is Sally.

We can add more line breaks to put each sentence on its own line:

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
Hello World!<br>
```

```
My name is Joe.<br>
```

```
My friend is Sally.<br>
```

```
</body>
```

```
</html>
```

And we get this:

Hello World!

My name is Joe.

My friend is Sally.

To make it double spaced, we can add extra `<br>` tags.

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
Hello World!<br><br>
My name is Joe.<br><br>
My friend is Sally.
</body>
</html>
```

And now we see:

Hello World!

My name is Joe.

My friend is Sally.

Remember for line breaks you don't need a closing tag. To add more line breaks, just add more

tags. Suppose we want:

Hello World!

My name is Joe.

My friend is Sally.

To get this result our HTML looks like this:

```
<!DOCTYPE html>
<html>
<body>
Hello World!<br><br><br><br><br>
My name is Joe.<br><br>
My friend is Sally.
</body>
</html>
```

In the next lesson, we'll learn how to add a headline to our webpage.

# Header Tags

Header tags enable us to easily create nice bold text to spruce up the appearance of our web pages. Header tags use the format `<hx>` where  $x$  is an integer 1,2,3,4... The smaller the number, the larger the header. The truth is you already know about this since you've probably used different levels of headers in your word processing program.

Header tags have an opening and closing tag. The general syntax is:

```
<hx> Your text here </hx>
```

Working on our previous example, we can make the phrase "Hello World!" a header by using the `<h1>` tag:

```
<h1>Hello World!</h1>
```

If our entire html looks like this:

```
<!DOCTYPE html>
<html>
<body>
<h1>Hello World!</h1><br>
My name is Joe.<br>
My friend is Sally.
</body>
</html>
```

The result we see is:

# Hello World!

My name is Joe.

My friend is Sally.

When you add a header tag, the text between the header tags is not automatically centered. We have to do that manually. Let's change our html to:

```
<!DOCTYPE html>
<html>
<body>
<center><h1>Hello World!</h1></center><br>
My name is Joe.<br>
My friend is Sally.
</body>
</html>
```

Now the page appears as:

# Hello World!

My name is Joe.

My friend is Sally.

We can also enhance the appearance of the page by adding *subheadings*. This is done by specifying the header level with an integer larger than 1. For example <h1>,<h2>,<h3> etc. where <h1> is the highest header level, so will have the largest text.

Let's add a header and put a subject heading as follows:

```
<!DOCTYPE html>  
<html>  
<body>  
<center><h1>Hello World!</h1></center><br>  
<h2>Friends</h2>  
My name is Joe.<br>  
My friend is Sally.  
</body>  
</html>
```

Now the web page looks like this:

**Hello World!**

## **Friends**

My name is Joe.

My friend is Sally.

Notice that some spacing is automatically placed in between the header and the text that follows. Basically this is working like you were just typing away in a word processor, except we are putting in the formatting commands behind the scenes.

In the next chapter, we'll see how to format text colors.

# Font Color

A plain black and white web page might be functional, but its not very interesting. Let's see how we can improve the appearance of our web pages using the font tag and color attribute. It's very easy. Let's return to the html we had in the last lesson:

```
<!DOCTYPE html>
<html>
<body>
<center><h1>Hello World!</h1></center><br>
<h2>Friends</h2>
My name is Joe.<br>
My friend is Sally.
</body>
</html>
```

To have a text string appear in a particular color, we use the syntax:

```
<font color="color">Text String</font>
```

Hence, we can have text appear red using:

```
<font color="red">Some text</font>
```

Changing our html file with this in mind we can make the main header red:

```
<!DOCTYPE html>
<html>
<body>
```

```
<center><h1><font color="red">Hello World!</font></h1></center><br>
<h2>Friends</h2>
My name is Joe.<br>
My friend is Sally.
</body>
</html>
```

The web page now appears as:

**Hello World!**

## Friends

My name is Joe.  
My friend is Sally.

Let's say we wanted the "Friends" sub-header to appear as blue text. All we have to do is add another font tag with the color attribute, for the "Friends" text string:

```
<!DOCTYPE html>
<html>
<body>
<center><h1><font color="red">Hello World!</font></h1></center><br>
<h2><font color="blue">Friends</font></h2>
My name is Joe.<br>
My friend is Sally.
</body>
</html>
```



Note that we need a closing tag `</b>` when using different font attributes.

Now our web page looks like this:

**Hello World!**

## **Friends**

My name is Joe.

My friend is Sally.

Besides changing color, you might want to make other changes to fonts such as setting the type face. We will explore this in the next lesson.

# Font Size and Type Face

In this lesson we'll learn how to change font size and type face. This is also done using the font tag. To specify a font size, we simply write:

```
<font size="x">This is some text.</font>
```

where x is an integer. For example we can write:

```
<font size="3">My name is Joe.</font>
```

You can set different attributes using the same font tag. Let's suppose that we wanted the text to appear red and set the font size to 5. This could be done by writing:

```
<font size="5" color="red">My name is Joe.</font>
```

This produces:

**My name is Joe.**

To change the typeface, we can set the *face* attribute in a font tag. For example, to set the font of a text string to *verdana*, we write:

```
<font face="verdana">My other friend is Bob.</font>
```

And we obtain:

My other friend is Bob.

As before we can combine multiple attributes including the typeface. So let's set the text to green, the face to verdana, and also set the size attribute of the text.

```
<font color="green" face="verdana" size="4">My other friend is Bob.</font>
```

This gives us:

My Other friend is Bob.

Being able to change these font attributes gives us the power to improve the appearance of our web pages. Referring to the html of the last chapter, suppose that we changed it to:

```
<body>
<center>
<h1><font color="red">Hello World!</font></h1>
</center>
<h3><font color="blue">Friends</font></h3>
<font color="red" size="5">My name is Joe.</font>
<br>
<br>
<font color="blue" size="4">My best friend is Sally.</font>
<br>
<br>
<font color="green" face="verdana" size="4">My other friend is Bob.</font>
<br>
<br>
</body>
</html>
```

Now we see:

**Hello World!**

**Friends**

My name is Joe.

My best friend is Sally.

My other friend is Bob.

# Paragraph and Div Tags

Next we explore two more formatting tags that can be used to improve the appearance of your web pages and present content in a meaningful way. The first is the *paragraph* tag. It does exactly what it says, it creates a formatted paragraph for any text enclosed within the tags (by adding margins and spacing). The paragraph tag is defined by `<p>` and you must add a closing tag `</p>` where you want the paragraph to end.

For example we can write:

```
<body>
<p>Hello world this is some text. Here is a second line.</p>
<h2>This is my heading</h2>
Another line of text. <br>
Line two of new section. <br>
Some more text.<br>
<h2>A Different Section</h2>
Text for the second section.
<br>
<br>
</body>
</html>
```

This will put “Hello world this is some text. Here is a second line.” into a paragraph and automatically put spacing between it and the heading on the next line.

The *div* tag is used to group together elements into a section and apply formatting to them. So in short it defines a section in your html document. For a simple example, we will take some of the text above and define a section out of it using the div tag, and color the font blue. The div tag is written as `<div style=...>` and you use a closing tag to end the section as `</div>`.

You can see how to use the div tag here, where we color a section blue:

```
<body>
<p>Hello world this is some text. Here is a second line.</p>
<div style="color: rgb(0, 0, 255);">
<h2>This is my heading</h2>
Another line of text. <br>
Line two of new section. <br>
Some more text.<br>
</div>
<h2>A Different Section</h2>
Text for the second section.
<br>
<br>
</body>
</html>
```

We obtain this result:

Hello world this is some text. Here is a second line.

## **This is my heading**

Another line of text.

Line two of new section.

Some more text.

## **A Different Section**

Text for the second section.

# Hyperlinks

The internet wouldn't have much functionality if you could only look at one web page at a time. To ease the ability of users to move about your website and to visit related pages of interest we need to add hyperlinks to our html documents. This is done with the hyperlink tag.

Oddly the hyperlink tag is denoted with the letter a. The letter a is used because it means *anchor text*. Hence the `<a>` tag defines a hyperlink which will open a new web page when the user clicks on it. As you probably know hyperlinks are displayed to the user as underlined text, with blue color for a page they have not opened before and purple for web pages they've already visited.

Several attributes can be specified with the hyperlink tag. The most important is the *href* attribute which tells the browser which link to open when the user clicks on the text. So href is just the url of the target web page.

HTML 5 introduces some new attributes. For example, you can use download to tell the browser to begin downloading a file when the user clicks on the link.

The target specified in the href attribute can be a local file (relative to the web page) or any URL. The syntax you will use is:

```
<a href="url to open">Text displayed to user</a>
```

The text displayed to the user can be anything, most web page developers put descriptive text that makes it more readable and more amenable to search engines. But you can just put the URL there if desired. These days people automatically know that underlined blue text is a hyperlink so putting descriptive text is preferred.

Here is an example that will display a link to the New York Times website:

```
<body>
```

```
Hello World!
```

```
<br>
```

```
<a href="http://nytimes.com">New York Times</a>
```

```
<br>
```

This looks like so:

Hello World!

[New York Times](http://nytimes.com)

If we wanted to just display the actual link to the user, we could write:

```
<body>
```

Hello World!

```
<br>
```

```
<a href="http://nytimes.com">http://nytimes.com</a>
```

```
<br>
```

Then we would get:

Hello World!

<http://nytimes.com>

Now suppose you want to open a file on your own server. Consider the case of a web page named about.html in your same directory. Then you would write:

```
<body>
```

Hello World!

```
<br>
```

```
<a href="about.html">About Us</a>
```

```
<br>
```

If the file was in a folder called "Info" you'd write:

```
<body>
```



Hello World!

<br>

<a href="about.html">/Info/About Us</a>

<br>

Often links are placed in the middle of a text string, for example:

Here is some more text linking to <a href="http://cnn.com">cable news network</a> find out what the latest news is.

This displays as:

Here is some more text linking to [cable news network](http://cnn.com) find out what the latest news is.

# Displaying Images

Images are displayed in a web page using the `img` tag. The syntax is:

```

```

You can link to images on your own server or to any image on the web if you know the URL. To give a specific example I have obtained a link to a picture of the actress Selma Hayek. We can display the image in our web page as follows:

```
<body>
<h1>Selma Hayek!</h1>
<br>
<br>

<br>
<br>
</body>
```

If the image doesn't render, the text "Selma Hayek Not Here" will be displayed in its place. Here is how the web page looks:

## Selma Hayek!



To see how the alt attribute works, you can copy and paste from the preview in Kompozer to a text document, and you will see:

Selma Hayek!

Selma Hayek Not Here

You can put other tags around the image to improve the appearance of your web page. For example, if we wanted to center the image, we would write:

```
<body>
```

```
<h1>Selma Hayek!</h1>
```

```
<br>
```

```
<br>
```

```
<center>
```

```

```

```
</center>
```

```
<br>
```

```
<br>
```

```
</body>
```

```
</html>
```

# Video Players on Web Pages

Our next topic is the video tag. Using <video you can put a link to a video you want to use and display it using HTML 5 in a full featured video player. The best way to teach this is to simply show an example.

```
<body>
<video width="400" controls="">
<source src="savagearchives.info/video/20-Timer-sm.mp4" type="video/mp4">
Your browser does not support HTML5 video.
</source>
</video>
<p>Video courtesy of <a href="http://www.xcode-training-and-tips.com/"
target="_blank">Xcode Training</a>.
</p>
</body>
```

So we start with the <video> tag, then follow it with the <source> tag which tells the browser the URL where the video is located. As usual the URL can be a local file or any URL on the internet.

The code has a bit of text that is displayed if the browser does not support HTML 5. Kompozer has not been updated in awhile so when I preview this code in Kompozer I see this:

Your browser does not support HTML5 video.

Video courtesy of [Xcode Training](http://www.xcode-training-and-tips.com/).

On the other hand, opening it in the browser we see:



# Ordered and Unordered Lists

In this lesson we are going to learn how to display bulleted and numbered lists using HTML. This is done using the *unordered list* or *ordered list* tag. An unordered list is just a bulleted list of items. To create an unordered list, you will enclose your list items within an opening `<ul>` tag and a closing `</ul>` tag. Each list item is denoted with a list item or `<li>` tag. You need a closing `</li>` tag at the end of each list item.

As an example, let's create a list of friends names. In most cases you will have a header of some kind to denote the list. This can be done by putting the text introducing your list in between paragraph tags. So let's start by putting a title for our list as follows:

```
<p>My Best Friends:</p>
```

Now we will need to enclose the list with `<ul>` and `</ul>`:

```
<p>My Best Friends:</p>
<ul>
</ul>
```

Each bulleted list item will appear as `<li>item text</li>`. Let's make our list of names:

```
<p>My Best Friends:</p>
<ul>
  <li>Sally</li>
  <li>Jose</li>
  <li>Paul</li>
</ul>
```

What this produces is the following:

My Best Friends:

- Sally
- Jose
- Paul

Using <ol> in place of <ul> produces a numbered list. The tag <ol> means ordered listing. For example:

<p>Countries:</p>

<ol>

<li>China</li>

<li>United States</li>

<li>United Kingdom</li>

<li>France</li>

</ol>

Gives us:

Countries:

1. China
2. United States
3. United Kingdom
4. France



# Big and Small tags

The Big and Small tags provide you with a means to carefully control text size. You can nest big and small tags to get a string of text to be the size you want.

To use a Big tag simply place an opening `<big>` at the start of the text you want to be larger font. To make it larger add as many `<big>` tags as necessary. At the end of the text you want to have larger font, place a corresponding closing tag `</big>`.

The small tag works in an analogous manner. For example:

```
<h1>My Web Page</h1>
```

```
<p>Video <big><big><big>is</big></big></big> <small><small>  
<small>courtesy</small></small></small>
```

```
of Xcode training</a></p>
```

Produces:

Video **iS**<sub>courtesy</sub> of Xcode training

# Basic Text Formatting Tags

In this lesson we will introduce tags used to make text:

- Bold
- Italic
- Underlined

To make a string of text bold, enclose it in `<b>` and `</b>` tags. For example:

`<b>Barak Obama</b>` is President of the United States

produces:

**Barak Obama** is President of the United States

To put a string in italics enclose it in `<i>` and `</i>` tags:

Barak Obama is `<i>President</i>` of the United States

gives us:

Barak Obama is *President* of the United States

To underline a string of text, enclose it in `<u>` and `</u>` tags:

Barak Obama is President of the `<u>United States</u>`

This gives us:

Barak Obama is President of the United States

You can of course, nest tags to apply more than one type of formatting. We can make “United States” bold and underlined:

Barak Obama is President of the `<b><u>United States</u></b>`

Giving:

Barak Obama is President of the **United States**

To also make it italic, just add `<i>` and `</i>` tags:

Barak Obama is President of the `<i><b><u>United States</u></b></i>`

This gives:

Barak Obama is President of the ***United States***

# Superscript and Subscript Tags

Text often requires superscripts and subscripts. You can do this in html using the *sup* and *sub* tags. Each also requires a closing tag otherwise any text that follows will also be altered by the tag

Often superscripts are used to indicate references. To create a superscript enclose the desired superscript text in the tags `<sup>` and `</sup>`. For example if we wanted the text:

***Barak Obama*** won the 2012 *Presidential Election* in the **United States**.<sup>19</sup>

We would use the following html:

```
<b><i>Barak Obama</i></b> won the 2012 <i>Presidential Election</i> in the <b>
<u>United
States</u></b>. <sup>19</sup>
```

To create a subscript, simply enclose the text in `<sub>` and `</sub>` tags. To obtain:

$X_i$

We would use:

```
X<sub>i</sub>
```

# Quoting Passages of Text

There are two tags that are useful for displaying quotes on your webpages. The first is the `<q>` tag which simply encloses text in quotes. Suppose that we had this phrase of plain text in our html file:

Not only our future economic soundness but the very soundness of our democratic institutions depends on the determination of our government to give employment to idle men.

If we enclose it in `<q>` and `</q>` tags, on the web page it will be displayed like this:

“Not only our future economic soundness but the very soundness of our democratic institutions depends on the determination of our government to give employment to idle men.”

We can also enhance the display of the quote using the `<blockquote>` tag. What this does is set off and indent the quote. So if we had:

Here is a line.

Not only our future economic soundness but the very soundness of our democratic institutions depends on the determination of our government to give employment to idle men.

Here is another line.

Changing our HTML to:

Here is a line.`<br>`

`<blockquote><q>`Not only our future economic soundness but the very soundness of our democratic institutions depends on the determination of our government to give employment to idle men.`</q></blockquote>`

Here is another line.

Will give us:

Here is a line.

“Not only our future economic soundness but the very soundness of our  
democratic institutions depends on the determination of our government to  
give employment to idle men.”

Here is another line.

# Creating Tables

Tables allow you to display data in a row and column format. There are three tags you will need to create a simple table:

- `<Table>`
- `<TR>` -table row
- `<TD>` -table data cell

`<TR>` is a tag that tells the browser to begin a new row in the table. Anything in between the opening tag `<TR>` and the closing tag `</TR>` constitutes a single row. Individual cells in the row are denoted by `<TD>` with the closing tag `</TD>`.

To see how to set up an example, let's suppose that we are displaying sports scores. We can add a centered headline:

```
<center>
<h1>Basketball Scores from Final Four</h1>
</center>
```

Now lets add our table. We begin by adding the table tag, specifying border width:

```
<center>
<table summary="Example" border="1">
</table>
```

The summary is not displayed on the page. Now lets add our rows:

```
<center>
<table summary="Example" border="1">
<tr>
<td> Game One </td>
```

<td> Michigan 75 </td>

<td> Kentucky 73 </td>

</tr>

<tr>

<td> Game Two </td>

<td> Arizona 68 </td>

<td> Duke 61 </td>

</tr>

</table>

</center>



# Drawing Lines on Web Pages

In this lesson we are going to learn how to draw horizontal and vertical lines. Drawing horizontal lines with html is very easy using the <hr> tag. All you need to do is place an <hr> tag where you want the line to appear. For example, our html looks like this:

```
<body>
<center>
<h1>Basketball Scores from Final Four</h1>
</center>
<hr>
<br>
<br>
<center>
<table summary="Example" border="1" cellpadding="16" width="100%">
<tbody>
<tr>
<th bgcolor="#ccffff" width="50%"> Game One </th>
<td bgcolor="#ffffcc" width="25%"> Michigan 75 </td>
<td width="25%"> Kentucky 73 </td>
</tr>
<tr>
<th> Game Two </th>
<td> Arizona 68 </td>
<td> Duke 61 </td>
</tr>
</tbody>
</table>
</center>
</body>
```

We see:

You can also draw vertical lines using the `<hr>` tag, but it takes a bit of a trick. The



The screenshot shows a web browser window with a single tab titled 'Test'. The address bar shows 'DropDown' and the page title is '(untitled)'. The main content area displays a table titled 'Basketball Scores from Final Four'. The table has two rows and three columns. The first row is for 'Game One' and the second row is for 'Game Two'. The scores are: Michigan 75, Kentucky 73 for Game One; and Arizona 68, Duke 61 for Game Two. The table is styled with a light blue header row and a light yellow body row. The browser's status bar at the bottom shows 'Design', 'Split', and 'Source' tabs, and a 'Normal' font style selector.

Game One	Michigan 75	Kentucky 73
Game Two	Arizona 68	Duke 61

following code using 2 `<hr>` tags in a row will draw a vertical line down the center of the web page:

```
<hr
```

```
style="padding: 0px; height: 400px; width: 2px; margin-bottom: -8px;">
```

```
<hr style="padding: 0px; height: 100px; width: 2px; margin-top: -8px;"><br>
```

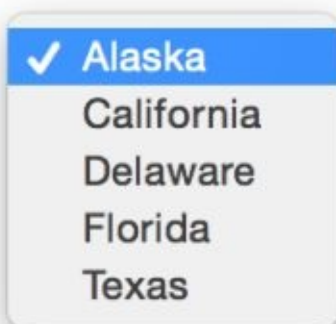
# Drop Down Lists

On many occasions you will want to give website visitors the ability to select an option from a drop down list Lets see how to set a list like this one up for your web page:

## My Great Header

Some text for my great webpage.

---



This is done using the `<select>` tag. You enclose the choices you want to present to the user in between and opening and closing select tag, using the `<option>` tag for each item. The format used for the option tag is to present the value and text displayed to the user. For example:

```
<option value="Alaska">Alaska</option>
```

The entire code for the page shown above, including header, horizontal line and drop down list with the code relevant for the drop down list highlighted in red is as follows:

```
<body>
<h1>My Great Header</h1>
<p>Some text for my great webpage.</p>
<hr>
<br>
<br>
<select>
<option value="Alaska">Alaska</option>
<option value="California">California</option>
<option value="Delaware">Delaware</option>
```

```
<option value="Florida">Florida</option>
```

```
<option value="Texas">Texas</option>
```

```
</select>
```

```
</body>
```

# Playing Audio Files

In this lesson we're going to learn how to set up an audio player on your web page. It will be complete with play/pause buttons, volume control, and time played displayed. These controls are displayed by default.

To play audio you need 2 tags:

- `<audio>` tag
- `<source>` tag with a path to your file and telling the browser what type of audio file you have

Setup is easy. Begin with the audio tag:

```
<audio controls="">
```

Next, we tell it where our source file is and what type of file we have. You can include multiple lines to include different file formats if you need to provide those for different browsers.

```
<source src="Genesis.mp3" type="audio/mpeg">
```

In my example, the file is in the same folder/directory as the web page. If yours is not be sure to provide the correct path to the file for the src attribute.

Next you need to provide a text string that will display if the users browser cannot display the audio controls.

Our entire code looks like:

```
<audio controls="">
```

```
<source src="Genesis.mp3" type="audio/mpeg">
```

Your browser does not support audio controls

</source>

</audio>

If we include this in the code for the previous web page we were working on:

<body>

<h1>My Great Header</h1>

<p>Some text for my great webpage.</p>

<hr>

<br>

<br>

<audio controls=""> <source src="Genesis.mp3" type="audio/mpeg"> Your browser does not support audio controls

</source></audio>

</body>

Then our page looks like this:

# My Great Header

Some text for my great webpage.

---



# iFrame

Using the iframe tag allows you to embed another web page or URL inside your web page. In the simplest case, we can just pass the URL as follows:

```
<iframe src="http://nytimes.com"></iframe>
```

However there are many properties we can specify. For example, we can set the width and height of the frame:

```
<iframe src="http://nytimes.com" width="700" height="400"></iframe>
```

You can also use the style attribute to set a border. We could for example set our embedded web page inside a dotted red box with the border given a 5 pixel width:

```
<iframe src="http://nytimes.com" style="border:5px dotted red"></iframe>
```

Here, we've set the width and height and added a solid blue border:

```
<iframe src="http://nytimes.com" style="border: 5px solid blue;"  
height="500" width="700"></iframe>
```

Adding this to the html we've used in several recent examples:

```
<body>  
<h1>My Great Header</h1>  
<p>Some text for my great webpage.</p>  
<hr>  
<br>  
<br>  
<iframe src="http://nytimes.com" style="border: 5px solid blue;"  
height="500" width="700"></iframe>
```



</body>

Gives us this web page, with the New York Times embedded inside the blue frame:

# My Great Header

Some text for my great webpage.

## Bill Cosby in His Own Words: Sex, Drugs and Deception


By GRAHAM BOWLEY and SYDNEY EMBER 12:52 PM ET

In a previously unpublicized deposition obtained by The Times, Mr. Cosby testified 10 years ago in a lawsuit filed by a young woman who accused him of drugging and molesting her. It details his calculated pursuit of young women, using fame and drugs.

Comments

### Excerpts From Bill Cosby's Deposition

### Your Weekend Briefing




## After Years of Drought, Wildfires Rage in California

By HAEYOUN PARK, DAMIEN CAVE and WILSON ANDREWS

The drought in the western United States has contributed to a devastating start to fire season.

## A Wizard at Prying Secrets From the Government



By RAVI SOMAIYA 3:40 PM ET

Jason Leopold of Vice News uses an encyclopedic knowledge of the Freedom of Information Act to obtain tens of thousands of

Brum: 1  
Donald  
Dowd: 1  
Lone R  
Kristof:  
Melinda  
Pillow 1  
Luhma  
Anxiou  
America  
Our Rac  
Momen  
Truth  
Join us  
Facebo

# Forms input

The `<form>` tag allows you to create an input form to do things like collect names, email addresses and so on. Everything that appears on the form is enclosed in between form opening and closing tags:

```
<form>
```

*your stuff here*

```
</form>
```

For example we can create input text boxes to let the user type in their city and state:

```
<form>
```

```
City:<br>
```

```
<input type="text" name="city">
```

```
<br>
```

```
State:<br>
```

```
<input type="text" name="state">
```

```
</form>
```

Other options allow you to add radio buttons and checkboxes. Using radio buttons as an example, to have the user select gender as male or female, we would add the following lines in between the form opening and closing tags:

```
<input type="radio" name="sex" value="male" checked>Male
```

```
<input type="radio" name="sex" value="female">Female
```

Of course you have to provide a means for the user to submit the data. You can add a submit button with this line:

```
<input type="submit" value="Submit">
```

To make something happen when the user clicks the submit button, such as take the info entered above and sticking it into a database, you need to write a script to handle the submit clicked action. The script itself is out of the scope of this tutorial, but lets say it was *myaction.php*. Then we would modify the opening form tag to have the form execute this action when the button is clicked:

```
<form action="myaction.php">
```

## CSS for Bi-Colored Web Page

In this simple example we introduce the use of style sheets to change the appearance of your web pages. A common task is formatting the page to display different areas, you might want to display a 2 or 3 column web page or add headers and footers or a sidebar. In this example we show how to set up a simple side bar, distinguished from the main web page by color:

The first thing to do in order to set this up is to put your style sheet in between the head tags. Our very simple style sheet will set the width of the sidebar and its background color. We also set the background color and left margin for the main part of the page:

```
<head>
```

```
<style type="text/css">
```

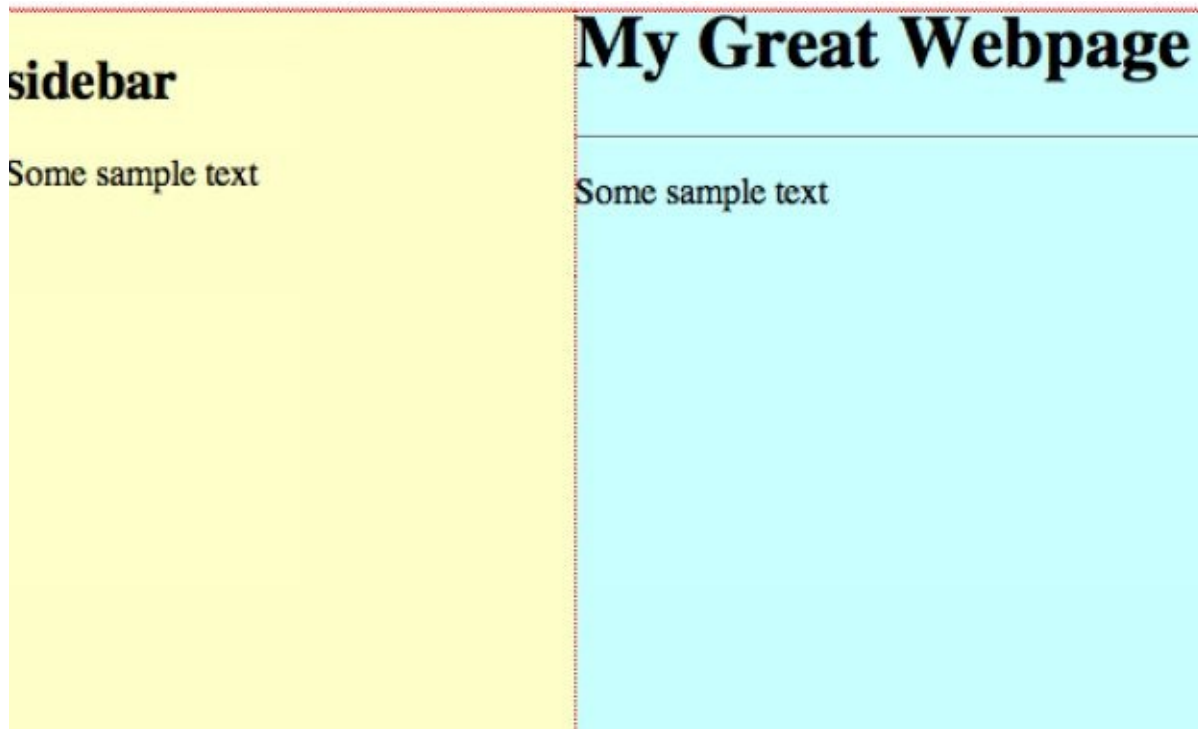
```
html, body {height: 100%; margin: 0}
```

```
#content {width: 100%; height: 100%}
```

```
#left {width: 250px; height: 100%; float: left; background-color: #FFFFCC}
```

```
#right {margin-left: 250px; height: 100%; background-color: #CCFFFF}
```

```
</style>
```



`</head>`

In the body section of the html file we use `<div>` tags to set up each section. There will be a `<div>` for the sidebar and main part of the page. We also need an extra `<div>` within which these are nested. This is shown here:

```
<div id="content">  
</div>
```

In between the opening and closing tags, we put the `<div>` tags for each section. First here is the sidebar. We have to tell the browser what style to use. This is done by setting the id to left:

```
<div id="left">  
<h2>sidebar</h2>  
<p>Some sample text</p>  
</div>
```

We do the same for the main part of the page:

```
<div id="right">
<h1>My Great Webpage</h1>
<hr>
<p>Some sample text</p>
</div>
```

Now of course you can put anything in between the div tags of each section, but we are only showing some text here for some simplicity. The entire code is shown here:

```
<html>
<head>
<style type="text/css">
html, body {height: 100%; margin: 0}
#content {width: 100%; height: 100%}
#left {width: 250px; height: 100%; float: left; background-color: #FFFFCC}
#right {margin-left: 250px; height: 100%; background-color: #CCFFFF}
</style>
</head>
<body>
<div id="content">
<div id="left">
<h2>sidebar</h2>
<p>Some sample text</p>
</div>
<div id="right">
<h1>My Great Webpage</h1>
<hr>
<p>Some sample text</p>
</div>
</div>
</body>
</html>
```

## More Text Formatting Tags

To highlight a section of text, use the mark tag. For example, to have the string “Queen of England” highlighted in yellow we write:

```
<p>Everyone came from far and wide to see the <mark>Queen of England</mark> as she  
rode through London.</p>
```

We can strikethrough text by either using the <s> tag or the <strike> tag. To strike through the phrase “Prince of Wales” in this sentence we can use either of the two examples:

```
<p>Rumors about who would be the next <s>Prince of Wales</s> filled the air.</p>
```

```
<p>Rumors about who would be the next <strike>Prince of Wales</strike> filled the air.  
</p>
```

Suppose you want to enclose part of a webpage or form in a box. This can be done using the fieldset and legend tags.

```
<fieldset>  
<legend>Some Boxed Stuff:</legend>  
<p>Some sample text.</p>  
</fieldset>
```

Finally, the preformatted tag <pre> allows us to type text and have it appear on the web page as typed. If we just type:

```
Hello from Times Square  
located in New York City
```

the finest city in all the  
world except maybe Paris or  
London.

It will appear as:

Hello from Times Square located in New York City the finest city in all the world except  
maybe Paris or London.

If we instead enter our html as:

```
<pre>Hello from Times Square  
located in New York City  
the finest city in all the  
world except maybe Paris or  
London.</pre>
```

That tells the browser to use our formatting, so it appears as:

Hello from Times Square  
located in New York City  
the finest city in all the  
world except maybe Paris or  
London.

# CSS Style Sheets

CSS means *cascading style sheet*. It is simply a file that goes along with your html file which specifies the look and formatting of the html file. We will illustrate this with a simple example.

First create a file called *thecssfile.css* and place it in the same folder as your html file. In this example, we will see how to specify how text with certain tags appears - what font is used, what style, and what size.

Note: Comments in a css file are lines which are ignored by the browser. These are enclosed between the characters */\*...\*/*.

```
/* Example comment that would be ignored */
```

What you do in a css file is list tags used in the html file and specify values for various attributes enclosed in curly braces.

```
tag_name{  
  
    properties  
  
}
```

For example, we can specify the font used and text size for an h1 header.

```
h1 {  
    font-family: Arial, 'Helvetica Neue', Helvetica, sans-serif;  
    font-size: 24px;  
    font-style: bold;  
    font-variant: normal;  
    font-weight: 700;
```



```
line-height: 26.3999996185303px;
}
```

If you wanted all text in your html file that was enclosed in paragraph tags <p> to have helvetica neue font with size 12 pixels, you would write:

```
p {
  font-family: Arial, 'Helvetica Neue', Helvetica, sans-serif;
  font-size: 12px;
  font-style: normal;
  font-variant: normal;
  font-weight: 400;
  line-height: 20px;
}
```

Or suppose that we want all quotes on our web page to have italic font. We can do this by specifying the properties of the block quote tag in our css file:

```
blockquote {
  font-family: Arial, 'Helvetica Neue', Helvetica, sans-serif;
  font-size: 21px;
  font-style: italic;
  font-variant: normal;
  font-weight: 400;
  line-height: 30px;
}
```

As you learn more css you will be able to make very fancy web pages, but with this simple example we are seeing some advantages of using css. One is that we can have all block quotes appear in italics with other specified font properties without having to type in tags every single time in our html file.

Here is the complete code of this css file:

```
/* The CSS File */
```

```
h1 {  
    font-family: Arial, 'Helvetica Neue', Helvetica, sans-serif;  
    font-size: 24px;  
    font-style: bold;  
    font-variant: normal;  
    font-weight: 700;  
    line-height: 26.3999996185303px;  
}
```

```
h3 {  
    font-family: Arial, 'Helvetica Neue', Helvetica, sans-serif;  
    font-size: 14px;  
    font-style: normal;  
    font-variant: normal;  
    font-weight: 500;  
    line-height: 15.3999996185303px;  
}
```

```
p {  
    font-family: Arial, 'Helvetica Neue', Helvetica, sans-serif;  
    font-size: 14px;  
    font-style: normal;  
    font-variant: normal;  
    font-weight: 400;  
    line-height: 20px;  
}
```

```
blockquote {  
    font-family: Arial, 'Helvetica Neue', Helvetica, sans-serif;  
    font-size: 21px;  
    font-style: italic;  
    font-variant: normal;  
    font-weight: 400;  
    line-height: 30px;
```

```
}  
pre {  
    font-family: Arial, 'Helvetica Neue', Helvetica, sans-serif;  
    font-size: 13px;  
    font-style: normal;  
    font-variant: normal;  
    font-weight: 400;  
    line-height: 1.85714302062988px;  
}
```

To use the css file, you need to add a reference to it in your header section of the html file. This is placed in between the header tags <head> and </head>. The important line to add is:

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

Be sure that the href is correct, or that the .css file is in the same folder as your html file. There are some other codes necessary to include in the header section but we won't worry about what they do just yet and just make sure they are listed:

```
<html>  
<head>  
<title>CSS Example</title>  
<meta http-equiv="Content-Type"  
content="text/html; charset=iso-8859-1">  
<link href="thecssfile.css" rel="stylesheet" type="text/css">  
</head>
```

Then we can add some things to our html body:

```
<body>  
<h1>My Great Web Page</h1>
```

```
<pre>This is some random text written for<br>display on the web page. Hello  
world<br>hello world hello world hello world.<br></pre>  
<br>  
<q></q>  
<blockquote><q>This is the best day in the world!</q></blockquote>  
<br>  
<br>  
</body>  
</html>
```

Now that we have specified the css file in the header, all the text listed in the body will automatically take on the properties specified, so for instance the quoted text will be in italics even though we didn't add `<i>` or `<em>` tags.

## Element and id Selectors

Although in the last example we created an external css file, this is not always necessary. You can include your style sheets directly in the `<head>`, `</head>` tags. Let's look at this and do so examining two ways to proceed.

An element selector is like we did in the previous lesson. We pick an element (or think of it as a type of tag), such as a header, block quote, or paragraph, and then specify its properties. If you create a style sheet and specify the properties for an h3 header, you are creating an element selector.

For example, we can specify styles for `<p>`, `<h1>`, and `<h2>` for a given web page:

```
<head>  
<style>  
p{
```

```
color: blue;
text-align:center;
}
h2{
text-decoration: underline;
text-align:center;
color: red;
}
h1{
text-shadow: 2px 2px #0000ff;
}
</style>
<title>Example Web page</title>
</head>
```

Another way to set styles is to use an id selector. In this case rather than defining properties for a specific element you simply define the properties you want and assign it an identifier.

```
#mytag {
text-align:center;
color:green;
}
```

Here we have created an element with identifier “mytag”. In this case you set an html element in your page to have these properties by setting its id attribute:

```
<p id=“mytag”>Hello from vactionland.</p>
```

Element and id selectors can be mixed in the same style sheet.

```
<html>
<head>
```

```
<style>
p{
color: blue;
text-align:center;
}
h2{
text-decoration: underline;
text-align:center;
color: red;
}
#mytag {
text-align:center;
color:green;
}
h1{
text-shadow: 2px 2px #0000ff;
}
</style>
<title></title>
</head>
```

In this case, if we just use <p>, the text will be blue, but if we use <p id=“mytag”>, the text will appear green. Hence, we can apply the style we need at will:

```
<body>
<h1>My Great Web Page</h1>
<p>Example text for a paragraph</p>
<br>
<br>
<h2>Second Header</h2>
<br>
<p id=“mytag”>Hello from vactionland.</p>
<p>Here is a second paragraph string.</p>
```

```
<h2 id="mytag">Third Header </h2>
</body>
```

The resulting web page is:

**Web**

**My Great Web Page**

Example text for a paragraph

**Second Header**

Hello from vactionland.

Here is a second paragraph string.

**Third Header**

## Page Backgrounds

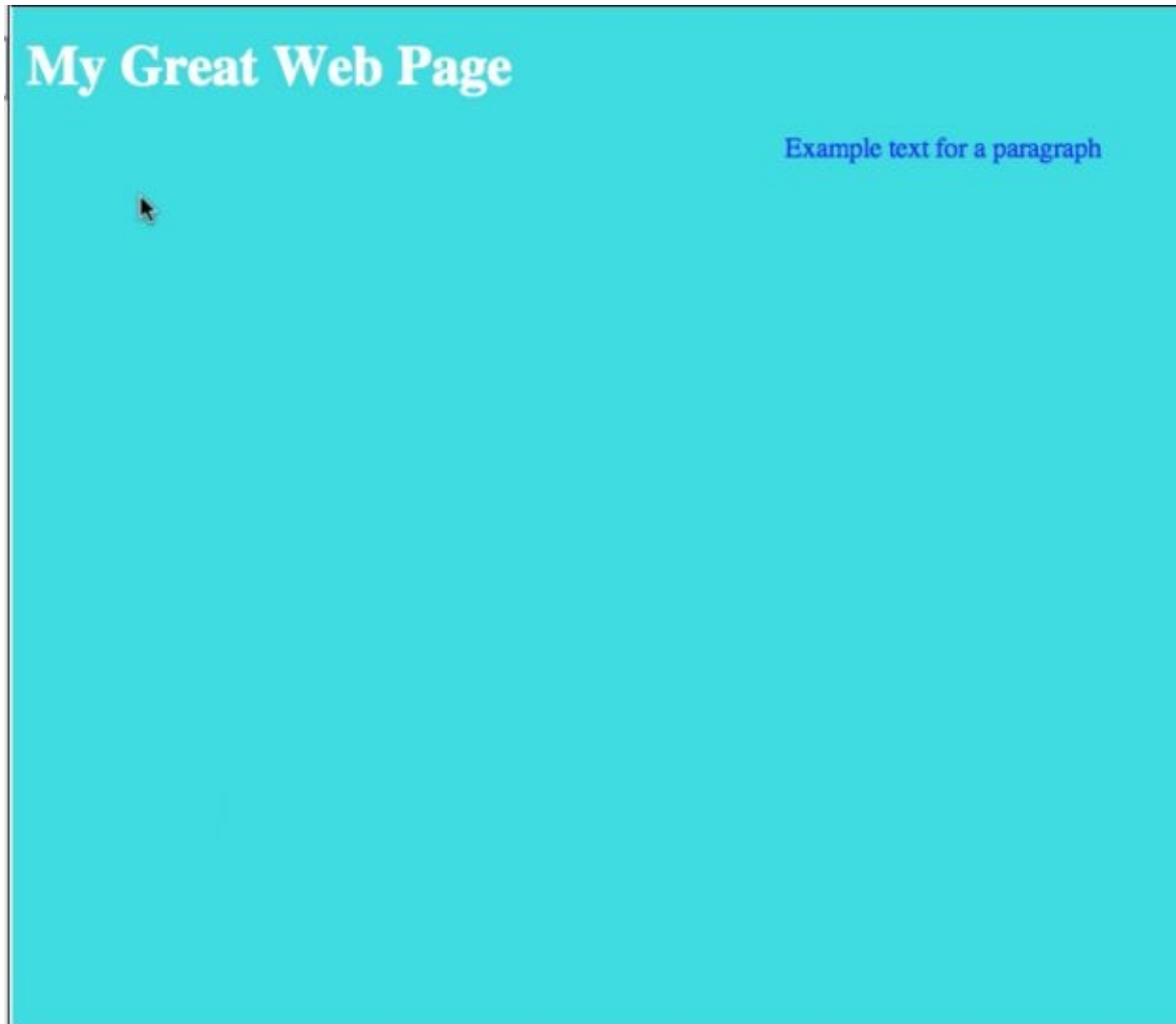
Let's say you want to set a background color for your web page and make headers appear as white text. We can choose a cyan color for our background:

This is easy to do using `<style>`. Let's look at the details and then put it together for the entire page. You can set the background color of an object using *background-color* in your css. In the example here:

```
background-color: #01c4de;
```

Will give the color displayed in the image. To set the background color for the entire web page, we specify that we want this to be the background color for the *body* tag.

```
body{
    background-color: #01c4de;
}
```



We can tell the browser that we want all h1 headers to be

rendered in white text using:

```
h1{  
  color: white;  
}
```

To tell the browser that you want h2 headers to be red, centered and underlined you would write:

```
h2{  
  text-decoration: underline;  
  text-align: center;  
  color: red;  
}
```



All together the html and css for the above page is:

```
<html>
<head>
  <style>
    body{
      background-color: #01c4de;
    }
    p{
      color: blue;
      text-align: center;
    }
    h2{
      text-decoration: underline;
      text-align:center;
      color: red;
    }
  </style>
  <title>Cyan Web Page</title>
</head>
<body>
  <h1>My Great Web Page</h1>
  <p>Example text for a paragraph</p>
</body>
</html>
```

It's also possible to use an image for your background rather than a solid color. Suppose that we had an image named forest.jpg that we wanted to use for the background. To do this change the body tag in your style sheet to:

```
body{
  background-image: url("forest.jpg");
}
```

Make sure to include the complete path to your image. In the example here, the image has been placed in the same folder as the web page. With this change, the web page now loads as:



## Classes in CSS

In this lesson we're going to take a step towards more advanced CSS by introducing the notion of classes. We will use this to build a web page like this:



A class  
is just  
another  
way to

identify elements in your HTML and give them certain attributes. The flexibility of the class allows you to apply characteristics to multiple items in your HTML if they are identified as being a member of that class. So in short, a class selector selects HTML elements in your web page that have been identified as being that class and applies the attributes you specify to it.

No doubt you've already seen classes in HTML files you've looked at. In your CSS a class is denoted with a period followed by the name assigned to the class. You can have a class without specifying the type of HTML tag it applies to such as:

```
.myclassname{  
    attributes here  
}
```

Or you can apply it to a specific tag, such as the paragraph tag. This example will set the font color of any paragraph marked "myblueclass" to blue:

```
p.myblueclass{  
    color: blue;  
}
```

To specify that a paragraph in your html file is a member of this class,

```
<p class="myblueclass">This is a sample paragraph that uses the myblueclass CSS class.  
</p>
```

Remember that you can apply a class to multiple elements in a page. For example, we can specify that all HTML elements with class="center" are center-aligned and have blue text:

```
.center {  
    text-align: center;  
    color: blue;  
}
```

Then you can apply this to multiple elements:

```
<h1 class="center">Header with the center class</h1>
<p> A normal paragraph without special formatting</p>
<p class="center">A formatted paragraph.</p>
<p>Another normal paragraph</p>
```

This produces:

## Header with the center class

A normal paragraph without special formatting

A formatted paragraph.

Another normal paragraph

So we've saved some labor. We are able to apply the same attributes (center and blue text) to different HTML elements. So lets see how to create the web page shown at the beginning of this chapter.

First lets add the flower in the upper left.

```
<style>
body{
  margin-left: 200px;
  background: #5d9ab2 url("flower.jpg") no-repeat top left;
}
```

Now lets create a class with the property of centered text. The class will be called *container*:

```
.container{
```

```
text-align: center;
}
```

The next class we create will draw the red box:

```
.my_box{
  border: 2px solid red;
  margin-left: auto;
  margin-right: auto;
  width: 80%;
  text-align: center;
  padding: 8px;
}
```

Finally we close out our <style> sheet setting h1 headers to white.

```
h1{
  color: white;
}
```

Now let's write the html for the page:

```
<body>
<div class="container">
  <div class="my_box">
    <h1>My Great Web Page</h1>
    <p>Some text to display. This is our most extensive example of css yet.</p>
  </div>
</div>
</body>
```

# Fixed Attachments

In this lesson we'll learn to fix an element on the page. In other words when you scroll the page, the element will remain in place while the rest of the page scrolls past. We will use the web page from the previous lesson and leave the flower in place.

Its actually pretty simple. We only need to change the CSS for the body tag, specifying the value of the background-attachment attribute.

Change your CSS to:

```
body{  
  margins-left: 200px;  
  background: #5d9ab2 url("flower.jpg") no-repeat top left;  
  background-attachment: fixed;  
}
```

That's all there is to it! Save your file and load in the browser again and you'll find that the flower stays in the upper left corner of the page. Add a large amount of text to your page to test the scrolling.

