**HTML and CSS**

**HTML and CSS**

HTML stands for **HyperText Markup Language**. **Hypertext** means "text with links in it." Any time you click on a word that brings you to a new webpage, you've clicked on hypertext!

A **markup language** is a programming language used to make text do more than just sit on a page: it can turn text into images, links, tables, lists, and much more. HTML is the markup language we'll be learning.

What makes webpages pretty? That's CSS—**Cascading Style Sheets**. Think of it like skin and makeup that covers the bones of HTML. We'll learn HTML first, then worry about CSS in later courses.

The first thing we should do is set up the skeleton of the page.

a. Always put <!DOCTYPE html> on the first line. This tells the browser what language it's reading (in this case, HTML).  
b. Always put <html> on the next line. This starts the HTML document.  
c. Always put </html> on the last line. This ends the HTML document.

**Basic terminology**

To learn more HTML, we should learn how to talk about HTML. Already you have seen we use <>s a lot.

1. Things inside <>s are called **tags**.
2. Tags nearly always come in pairs: an opening tag and a closing tag.
3. Example of opening tag: <html>
4. Example of closing tag: </html>

You can think of tags as being like parentheses: whenever you open one, you should close it. Tags also **nest**, so you should close them in the right order: the most recently opened tag should be the first one closed, like in the example below.

<first tag><second tag>Some text!</second tag></first tag>

**Make the head**

Add an opening <head> tag and closing </head> tag.

Between the <head> tags, add in an opening <title> tag and closing </title> tag.

Between the <title> tags, write in a title for your page. For example, "My Webpage."

**Paragraphs in the body**

Great job! To review, an HTML file has both a head and a body. The head is where you put information about your HTML file, like its title.

The body is where you put your content, such as text, images, and links. The content in the body is what will be visible on the actual page.

The body goes inside the <html> tags, right after the <head> tags, like this:

<html>

<head>

<title>My Webpage</title>

</head>

<body>

<p> Hello world! </p>

</body>

</html>

**Paragraphs and headings**

We're definitely making good progress! We've learned when and why we use HTML. We've also learned how to:

a. Set up an HTML file with tags  
b. Title the webpage (in the <head>)  
c. Create paragraphs (in the <body> with <p> tags)

The next step is to give our paragraphs headings using **heading tags**. Let's start with the <h1> tag. The content between this tag will be the biggest!

**More about headings!**

HTML actually lets us have more than one heading size. There are six heading sizes, where <h1> is the boss and <h6> is puny! Basically h1 has the largest size and h6 has the smallest size.

* <h1> - The CEO
* <h2> - VP
* <h3> - Director
* <h4> - Middle management
* <h5> - Lowly assistant
* <h6> - Gets coffee for everyone

**Adding Links!**

What if you wanted to send the user to another part of your website, or another website altogether? You use hyperlinks, or links for short!

<a href="https://www.codecademy.com">My Favorite Site!</a>

1. First, there's an opening <a> tag and that tag has an attribute called href. The href value tells your link where you want it to go, in this case [https://www.codecademy.com](https://www.codecademy.com/).
2. Then you have a description of your link between your opening <a> and your closing </a> tags. This is what you will be able to click on.
3. Finally, you have your closing </a>tag.

**Adding images**

You can add images to your websites to make them look fancy.

We use an image tag, like so: <img>. This tag is a bit different from the others. Instead of putting the content between the tags, you tell the tag where to get the picture using src. It's also different because there is no ending tag. It has /in the tag to close it: <img src="url" />.

Check out the tag to the right—it adds a picture of a rubber duck to the page! (You can see it by clicking on the Preview button.)

See the web address (or **URL**) after src=? It's "https://s3.amazonaws.com/codecademy-blog/assets/f3a16fb6.jpg". That tells the <img> tag where to get the picture from!

Every image on the web has its own image URL. Simply right-click on an image and choose "Copy image URL." Paste that URL in quotes after src= to insert with your <img> tag.

**Ordered lists**

Good! Now let's learn how to make **ordered lists**. An ordered list is simply a list that is numbered, like the one below.

1. On [line 8](javascript:void(0)), we begin the ordered list with the opening tag <ol>.
2. On [lines 9](javascript:void(0)) – 11, we **wrap** (i.e. surround) each individual item with <li>and </li> tags.
3. Because each listed item is only on one line, we put the entire **element** on one line.
4. On [line 13](javascript:void(0)), we finish the ordered list with the closing tag </ol>.

How cool is this? We can now add ordered lists to headings and paragraphs as things we can use in our HTML body.

**Unordered lists**

We just learned how to make ordered lists, but what if the order doesn't matter, what if we just want bullet points?

<h2>Taco Ingredients</h2>

<ul>

<li>Cheese</li>

<li>Sour Cream</li>

</ul>

Does this look familiar?

1. First, we open our list with an unordered list <ul> tag
2. For each item we wish to add to the list, we use a list item tag <li> with text in between
3. We then tell the browser we are done with our list by calling our closing </ul>tag

**Making comments**

Comments start with <!-- and end with --> like this:

<!-- This is an example of a comment! -->

**Font size**

Recall that <p> and </p> are opening and closing **tags**.

We can give tags more instructions by including **attributes** in the opening tag. An attribute is simply a characteristic or some description for the content in the element. You saw this with src in <img>and href in <a>.

Let's change the size of the text. How? We use the **style** attribute. We make it equal to font-size, followed by a colon, the size you want, and end it with px (short for "pixels"). For example:

<p style="font-size: 12px">

**Font color**

What is awesome about the **style** attribute is that we use it a lot! And we can use it with many different tags, not just <p>. Let's now change the colors of our text in a heading.

To change the color of text, simply add the style attribute in the opening tag, then make the style equal to "color:blue" (or whatever color you like). For example:

<h2 style="color:red">

What if you want to change the color and the size of the text? Simple! Just add a semi-colon between each bit. For example:

<h2 style="color: green; font-size:12px">

A full list of available colors can be found [here.](http://www.w3.org/TR/css3-color/#svg-color)

**Note**: do not type something like:

<h2 <h2 style="color: green; font-size:12px">

If you incorrectly nest your HTML tags like that, your code **will not pass**.

**Font family**

We've covered font colors and font sizes. But we want more power! We want to decide what font type to use. We can do this using font-family, like this:

<h1 style="font-family: Arial">Title</h1>

1. First we wrote <h1>Big title</h1>
2. Then inside the opening <h1> tag, we added a style attribute, and set it equal to "font-family: Arial".

This styles the <h1> tag with Arial font.

We can do the same for other tags. So we could have a li:

<li style="font-family: Arial">Hello!</li>

[Here's a list](http://www.w3.org/TR/CSS21/fonts.html#generic-font-families) of available fonts.

**Background color**

The previous section covered a number of nice tricks to control how the text looks. Now we want to learn about how to change the color of the webpage's background.

We can use the style attribute again, and set it equal to "background-color: red" (or whatever color you want).

For example, here's how to change the background color of a <p> tag to red:

<p style="background-color: red;">Hello! </p>

**Aligning the text**

Often it is nice to be able to move the text around. To do so, we again use the **style** attribute. And then we use "text-align:left" (or right, or center) to determine the location of the text.

<h1 style="text-align:center">

**Strong words!**

We can change the appearance of words. What if we want to make them **bold**?

Surprise! We don't have to use the style attribute. Here are the steps:

1. Identify the word or words you want to **bold**.
2. Surround those words with opening tag <strong> and closing tag </strong>.
3. Celebrate how awesome you are at HTML!

**Emphasize words!**

Aside from bolding words, we often want to italicize words for **em**phasis. (Hint, hint.)

Like bolding, we do not need to use the style attribute. Instead:

1. Identify the word or words you want to italicize.
2. Surround the word or words with the opening tag <em> and closing tag </em>.
3. Be humble and grateful for your newfound powers.

**What are tables?**

Tables are very useful. We use them to store tabular data so it is easy to read! When you want to present information neatly in a table with rows and columns—you guessed it—the <table> tag is what you need.

There are many tags associated with tables, but it all starts with the <table>tag, so let's add that first.

**Rows of information**

A table is just a bunch of information arranged in rows and columns.

We use the <tr> tag to create a **table row**. We'll learn how to create columns shortly, and everything will start to come together. (You don't really create columns in <table>s: instead, you tell each row how many cells to have, and that determines your number of columns).

**A single column**

Look at the HTML now. Can you tell that there are still three rows? We've added a little more whitespace to make it easier to deal with **table columns** and **table data**.

We've added a single <td> ("table data") cell to the first row, essentially creating a single column. If you view the Result tab now, you'll see that we've drawn a border around it. it's not that impressive, but don't worry: we're about to add more table data cells.

We're starting to add a lot of HTML elements now. Make sure to indent your tags as you nest them so you don't get confused!

**Head of the table**

Here's the table we made earlier. It's okay, but it just looks like we have a list of famous Hollywood people (monsters?) and their birth years. To make our table look a little more like a table, we'll use the <thead> and <tbody> tags. These go within the <table> tag and stand for **t**able head and **t**able body, respectively.

The <head> HTML tag contains information about a web page (e.g. its title) and the <body> tag contains the contents of the web page. In the same way, the <thead> tag can be thought of as containing information about a table and the <tbody> tag containing the actual tabular data.

**Table Heads**

We have just added a <thead> tag above the <tbody>.  
It will hold the heading for each column.

You add text to a <thead> similar to a <tbody>, like this:

<thead>

<tr>

<th>

Name

</th>

<th>

Favorite Color

</th>

</tr>

</thead>

1. First we have an opening <thead> tag for the table head.
2. Then we have an opening <tr> tag for the row. (to start the row)
3. After that, a <th></th> cell for the Name column heading. Notice that we use <th></th> for the table heading cells instead of<td></td>.
4. Then another <th></th> cell for the Favorite Color column heading.
5. Finally, we close the row element with a closing </tr> tag, and close out the table heading element with a closing </thead> tag.

**Naming your table**

Our table is missing a title. We want to add a title row that goes all the way across the top.

To do so, we need to use the **colspan** attribute for the <th> tag. By default, table cells take up 1 column. If we want a table cell to take up the space of 3 columns instead of 1, we can set the colspan attribute to 3.

It looks like this:

<th colspan="3">3 columns across!</th>

**'Div'ide and conquer**

One of the most versatile structure tags available to you is the <div></div> tag. Short for "division," <div> allows you to divide your page into containers (that is, different pieces). This will come in handy when you begin learning CSS in the next unit: you'll be able to style different parts of your website individually!

Check out the Result tab. You should see three blocks: one red, one blue, and one green. Each one is its own <div>container.

**Spantastic**

While <div> allows you to divide your webpage up into pieces you can style individually, <span> allows you to control styling for smaller parts of your page, such as text. For example, if you always want the first word of your paragraphs to be red, you can wrap each first word in <span></span> tags and make them red using CSS!