Week 2:

Just like for the previous week, this week is going to be split in 2 sections. HTML and CSS. Anyways, what we have done this week was learn about new HTML tags and about CSS Text properties. Both of these will be explained in more details in their appropriate section!  
  
HTML:

This week for HTML, we have learned about 4 new tags! We have learned about the anchor tag, internal links, comments and tables, Although, let’s start off with the anchor tag!  
  
The anchor tag, written as <a> with its closing tag, is used to provide links in your webpage. A link is what you click to go to a different webpage. To use this tag you simply have to have text in between the opening and closing tag. This should now show text! Although, you might notice that nothing happens when you click it since you haven’t specified the link yet. To specify the link, you simply have to add an attribute called “href”. The href attribute is there to ask you the URL(the link) of the page you want to go to. Here is how you use it…  
  
<a href=”google.com”>My link</a>  
  
When you click this, it should change the current window of the browser(if you are making a website) to google. Another thing you can do with links is have a relative link. For now we have been using an absolute link which is essentially what you think of when you think of a link. Relative links are links that link to other pages of your website. You can do these by simply having the name of the html file instead, here is an example…  
  
<a href=”myexample.html”>My link</a>

There is another thing you can do with the <a> tag. There is another attribute that can be used, this attribute is the “target” attribute. What this does is to tell how it should open the link, in other words, will it open in a new window, tab, the same page, etc… It is very easy to use!  
  
<a href=”google.com” target=”\_self”>My link</a>

Inside of the target attribute, you might notice the text that says “\_self”. This is how you specify the properties of the “target” attribute. In this case, it opens it in the same page the link was clicked in. Here is what each one can do…

|  |  |
| --- | --- |
| \_self | opens the link in page where the link was clicked |
| \_blank | opens the link in a new window or tab |
| \_parent | opens the link in the parent frame |
| \_top | opens the link in the full body of the window |
| framename | opens the link in a named frame |

The property of the attribute really changes on your needs. The two most common one are “\_self” and “\_blank”. If you do not specify the property or the attribute, it will automatically act as if it used “\_self” by default. Anyways, when using a link to your own webpage you would want to use the “\_self” property. Although, if you are linking to a different website you might want to use “\_blank” instead. There is one last thing you can do with the links. You are not required to have text in between (you need something but not necessarily text!). Say you want an image to be a link, you can do so easily by replacing the text in between the opening and closing tag to be that image (or really whatever you want).  
  
  
  
  
Let’s now learn about internal links! This is going to be very quick and easy! An internal link is simply a link that will… well… link, to another part of the page(much like relative link except this is inside the same page and not to another page of the website). This only requires one thing different from the normal <a> tag. When specifying the URL(href) you simply put a “#” followed by the id of the element you want to go to. Let me show you what it looks like.  
  
<a href=”#my-id”>Goto section</a>

<p id=”my-id”>The new section!</p>

You might recognize the id tag from the CSS from the previous week! But anyways, what this will do is scroll all the way down to that paragraph since it has the ID it is searching for! Just a small warning though, the id cannot be used more than once, if it is it will simply go to the first element of the page that contains the id specified. One last thing! This <a> tag cannot have the “target” attribute. It technically can but if anything, but if “\_self” is specified it will open a weird internet page and scroll on the already previously opened page.  
  
  
Let’s talk about the next thing! Let’s do comments since Tables are a little more complicated! A comment is simply some text you can put in your file that will be ignored by the browser. This can be useful if you want to say exactly what something does without having it be read as actual HTML. They are very simple to use!  
  
<!-- Hello -->

This above is a comment. Everything between “<!--” and “-->” is the actual text, the two other parts of it is what marks the beginning and end of the comment. You can also have it take multiple lines, simply skip a line in between.  
  
<!-- Hello  
This is the rest of my comment!  
-->  
  
There you go! You can now use them everywhere in your code!

The last thing we learned about in HTML is Tables. These things \/

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

Tables can be very useful for organizing information when bullet lists won’t do the trick. Making the table is simple (adding things to it is less simple). To create the table, you simply need the <table> tags this will create the table itself. What you then need to do is create a header for the table. The header is where the names of the columns will go (the horizontal line at the top). This can simply by placing the <thead> tags inside of the <table> tags. Like so…  
  
<table>

<thead>

</thead>

</table>

Now you have the header column even though it is still empty.  
CSS: