Designing Web Pages with HTML

Instructor: Daniel Chong Email: dchong@mybpl.org

Course Objective: To gain an understanding of the structure of webpages and how to change that structure using HTML.

What is HTML?

HTML is the standard markup language for creating Webpages. It stands for Hyper Text Markup Language. It builds the structure for how websites are displayed and content is presented on the Internet. Within this HTML structure we use ***Elements***, represented by various tags, to render our content within the webpage. The browser will not display these tags, but uses them as variables to hold our content that we wish to display.

Although some will tell you that HTML is not a real programing language, the fact of the matter is, this is what makes up the backbone of the internet. Even large corporations like Amazon and Microsoft will rely on immense amounts of HTML to get their websites functioning. Just by knowing the basics you will cultivate a better understanding of how the World-Wide-Web works as a whole.

Getting Started Writing Your Own HTML

When we write our own HTML we want to use a simple text editor. If we are on a PC we can use Notepad; while on a Mac we would use TextEdit. Tonight in class we will be using Notepad, but if you are at home and use a Windows PC then I recommend looking into a program called Notepad++ (It's got a little gecko as the icon). This is just a more robust plain text editor, which allows you to specify which programing language you are using so it can look for syntax errors, preventing you from debugging simple typos and errors on your own.

While there are even more robust options out there such as Adobe Dreamweaver, those are more advanced and could take a whole college semester to learn the program in its entirety, but if this is something that interests you it would not be a bad thing to look into.

You can also write you HTML code in something like Microsoft Word, but I would tend to avoid it as the formatting rules of a word processor can be annoying to deal with. If this is something you would like to use, you would need to make sure when you save your document to save it as a .txt(plain text) instead of the standard .docx or .doc filetype.

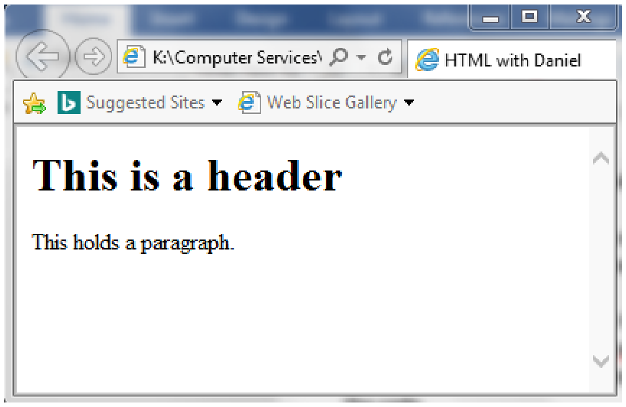
Tonight we will start by opening Notepad:
Open the Start Screen (the window symbol at the bottom left on your screen). Type Notepad.

If at home you are using Windows 7 or before it would be:
Open Start > **Programs > Accessories > Notepad**

If at home you are using a Mac it would be:
Open Finder > Applications > TextEdit | Preferences > Format > choose "Plain Text" |under "Open and Save", check the box that says "Display HTML files as HTML code instead of formatted text". | Then open a new document to place the code.

Example of a basic html page

```
<!DOCTYPE html>
<html>
<head>
<title>HTML with Daniel</title>
</head>
<body>
<h1>This is a header</h1>
<p> This holds a paragraph.</p>
</body>
</html>
```



This is a header

This holds a paragraph.

Tag Explanations

- <!DOCTYPE html> : This defines the document as HTML, only needs to appear once at the top of the page **– This is Required**
- <html></html> : These are the root elements of a webpage, meaning they are the first element to be opened, and the last element to be closed on every page. **– This is Required**
- <head></head> This contains meta information about the page, this is where you would define character types, link style sheets, and put things like the title of the page for Search Engines to catalogue. **– This is Required**
- <title></title> Specifies a title for the document. **– This is Required**
- <body></body> Container for any visible page content. **– This is Required**
- <h1></h1> Defines a large heading, first priority (headers can have priority 1-6)
- <p></p> Defines a paragraph

As you can see from the above example, most tags come in pairs. There is an opening <tag> and a closing </tag>. Notice how the closing tag is preceded by a /. **THIS IS REQUIRED** to properly close a tag, if you do not include this there will be mistakes that may cascade into other tags rendering the whole page a mess. It is good practice to put in the closing tag as soon as you open a tag. Tags are not case sensitive so <P> and <p> will both result in a paragraph, but it is good practice to only use lowercase with your tags.

Some tags are called “empty tags” and these are tags that hold no content and are instead just attributes. Two common examples of this are the horizontal rule and line break elements. These are elements that hold no “content” in the sense that they are instead used as just an attribute of the formatting. HTML5 does not require empty elements to be closed, so while <br> or <hr> is acceptable for a line break or horizontal rule tag, it is good practice to close these elements as some other versions of HTML require it. To properly close an empty tag you would do: <br /> or <hr />.

Saving our HTML Files

When we want to save the file to our computer it is important that we know exactly where we are saving our documents and also to make sure that we are saving it as the correct file type. If we are using Notepad the way we would do this would be as follows.

1. File > Save As
2. Name the file “index.html” – index is the standard for what you name your home page
3. Change the Save as type to “All Files (*.*)”
4. Change the Encoding to “UTF-8” – this is the preferred way of encoding for HTML files.

To view our html file, we can either double click on it in Windows Explorer or Finder and it will open with the default browser, or we can right click on it and choose "Open With." We can also open a file right in the browser, in Firefox you click the 3 bars in the upper right hand corner and go down to "Open File" or Ctrl+O and follow the same steps as before, as it will open your file explorer to search for files.

To see any changes we make to the file, we first have to make sure we save it (just a normal save, not a "save as") and then you can either press the refresh button on the browser or simply press F5 to refresh.

HTML Attributes

Attributes are used to provide extra information about an element. ALL HTML elements can have attributes, and these attributes are always specified within the opening tag. An attribute usually comes in a name/value pair. It is good practice to put quotes around the attribute value.

A common example of attributes can be seen using the <a></a> (anchor) and the <img /> (image) elements.

First we'll touch on the anchor elements, which can be called using <a></a>. This will allow you to place a link into your website to either redirect you within the webpage itself, or as an external link which takes you to a different page or website. If we wanted to have a link to say, Google, we would use the following syntax.

```
<a href="https://www.google.com">This is a Link</a>
```

NOTE: In this example, we have the anchor element, which is our <a> tag, but as you can see we also need to specify where exactly we want this link to go. This is where the href attribute comes in. href stands for Hyper Text Reference, and will allow us to designate a location for our link. Without using this attribute we would have what is called, dead links, which is a big no-no in web design.

Next we will cover the src (source) attribute, using the image element. If we want to have an image on our website, then we have to have the file on our local machine, and we also have to know where that file is saved or in other words the source. The path we follow to this source is called the Fully Qualified Domain Name (Think of this as all the folders and sub folders you have to step through, and the directions you would tell someone if they were looking for the file.) For this class we will have saved the images on the desktop so we just need to know the name and file type of the image, but that will not always be the case, if you do not know the full FQDN then your image will not render on the page. If we wanted to have an image of the Rocky Mountains and that image was saved as "rockyMountain.jpg" then we would use the following syntax.

```
<img src="rockyMountain.jpg" />
```

NOTE: There are other attributes we can use with the img element, such as width and height as well as alt. Width and height will allow us to choose exactly how big we want our image to be. While the alt attribute can be used to designate a text value for the image, which is good practice for accessibility for people who need to use screen readers or other devices. These attributes can be nested within each other, so we could end up having an img element looking something like this.

```
<img src="rockyMountain.jpg" height="500" width="300" alt="Rocky Mountain Sunset" />
```

The last common attribute we can use is called style. This can be used to specify the styling of an element, such as color, font, weight, or size. Usually this is done within a separate file with CSS but we can do inline styling on the fly if we need to make a rule a priority. To use this we would use the following syntax.

```
<p style="color:red"> The text of this paragraph is now red </p>
```

NOTE: Notice how in the value of our attribute we had to specify both the color and what color we want, and it is separated by a colon (:), this is true for all style attributes.

Common Attributes

Attribute	Description
alt	Specifies an alternative text for an image, when the image cannot be displayed
disabled	Specifies that an input element should be disabled
href	Specifies the URL (web address) for a link
id	Specifies a unique id for an element
src	Specifies the URL (web address) for an image
style	Specifies an inline CSS style for an element
title	Specifies extra information about an element (displayed as a tool tip)

*Table pulled form w3schools.com

Lists & Tables

Lists and Tables are used to organize out content within our webpage. They can be used with other elements within them, such as images or links and using the style attribute we can style them to be however we want, changing things like the width of table cells or aligning a list to be on the right side of the page.

First we will cover lists, and to do that we have to understand that we have two different kinds of lists used in two different situations. They are Ordered and Unordered Lists. We would use an Ordered List to convey a set of directions or something similar, while a Unordered List would be something like a grocery list. Both of these elements have different tags, <ol>item</ol> and <ul>item</ul> respectively, but within these elements we nest another tag which is the list item tag which can be called using <li>item</li>. The syntax for these two lists would look like this.

Unordered List

```
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

This would show up as a bullet list.

Ordered List

```
<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

This would show up as a numbered list.

NOTE: There is a third type of list called a Description List, it is used when we want to not only have a list item but also a description of that item. We will not be going over it tonight.

Next we will cover Tables. Tables are like lists in the sense that, like lists, we use nested elements within the table to set it up. The elements of a table are the Table itself, Table Rows, Table Headers, and Table Data. These can be called using the <table></table>| <tr></tr> | <th></th> | <td></td> tags respectively. To start a table we call the <table></table> tags. The next thing we want would be the first row, which will contain our headers. After that each Row will contain our respective data, and we can keep adding rows and headers ad infinitum. Say we wanted to have a table of our employees on our website and we wanted to have the data included be their First Name, Last Name, and Age. We would set up our syntax as follows.

```
<table style="width:100%">
  <tr>
    <th>Firstname</th>
    <th>Lastname</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td>50</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
    <td>94</td>
  </tr>
</table>
```

NOTE: Notice how in the opening table tag I gave it the style value of width and set that width to 100%, this will make the table be 100% of the screen width wise. We can change this value as we see fit, and we can also specify cell height and width, as well as add a border or other styles to it. Say we have data that might span the width of two columns or the height of two rows, we can change that with the table attributes colspan and rowspan. This will allow us to make changes to things such as a schedule with hourly time slots.

Divisions (Div)

The <div> tag defines a division or a section in an HTML document. It is often used as a container for other HTML elements to style them with CSS or to perform certain tasks with JavaScript. While we won't be covering this extensively, it is extremely useful and can be a very powerful tool for making your website look crisp and clean.
By default the browser will always place a line break before and after the <div> element. This can be changed using CSS. It will also display in a block format by default, but this can be changed once again using CSS.
An example of a div would be something like this.

```
<div style="background-color:lightblue">
  <h3>This is a heading</h3>
  <p>This is a paragraph.</p>
</div>
```

NOTE: This will make the header and paragraph appear on a light blue background, it will be separate from other elements.

Moving Forward with Web Design

Today we have covered only the very basics of what you can accomplish with HTML, as you move forward you will want to practice with styling your webpages by creating custom styles using Cascading Style Sheets. From there if you feel you really like this whole web design thing, you could start to learn about a myriad of other web-based programing languages such as JavaScript, PHP, SQL, or even AJAX. This is just the first step in a journey that could lead to a personal website or even a career. No matter how far you go with it, know that just from being here tonight you have become a more proficient user of the web.

While the best way to learn is to actual build websites, starting from scratch can be intimidating. The good news is, there are so many websites out there and you can inspect their Source Code to see how they built their website!

You can do this by visiting the site you wish to inspect, click the three lines or bullets in the upper right-hand corner and then choosing the "Developer Tools" and choosing "View Source." This can be an amazing tool to see how exactly a professional web developer did something, and this way you can replicate it!

If you are ever looking for tutorials or references I recommend checking out www.w3cschools.com, this is the website of the World-Wide-Web Consortium, the organization that makes the standards for today's web development cycle.

If you are looking for a real challenge, but the opportunity to learn a whole lot about web development, I would check out www.theodinproject.com. The Odin Project is an open-source, free curriculum of classes to become a Full-Stack Web Developer, complete this and you will most likely be able to land a job in the web development world. Although it is hard, and will take time to complete, the skills you learn from this project will push you far beyond adept to advanced users.

If you are ever stuck and don't know what to do, Google is your best friend, always remember that some else has had the same problem as you and has probably posted the solution somewhere online, such as www.stackoverflow.com.

On November 29th, we will continue this course with Creating a Web Site. This class will pick up where Designing Web Pages With HTML left off. It will teach students how to acquire web space online, transfer their files to the Internet, weave together multiple pages to make their web sites more efficient and easier to understand, and cover more advanced HTML topics more thoroughly.

I look forward to seeing you again! If you ever have any questions, feel free to email me.