Week 2: HTML Bootcamp

Agenda

- Overview
    - What's Due?
- HTML Bootcamp
    - What is HTML?
    - Semantic HTML
    - HTML Validation
    - Class Examples
- HTML Exercise
- Next Week

Overview

This week, we will have a basic recap of HTML and how it is used to establish a structure for our website. Students will also learn about validating their documents. After reviewing information about HTML elements and tags, students will build a basic web page and submit it for review. By the end of this week, students will be able to identify various tags and use them correctly on a live web page. **There are no discussion posts due this week.**

What's Due

- HTML Exercise

HTML Bootcamp

My assumption is that all of you have taken some sort of web development course in the past. Therefore, I don't want to spend too much time reviewing HTML as you should already understand it. However, I know it might have been a while since you took that course, so this week will serve as a quick refresher of the major concepts.

What is HTML?

HTML stands for Hypertext Markup Language and represents the structure of a web page. It is based on the concept of tags that "markup" the document. Those tags allow the browser to interpret .html files and present them correctly to users on the page. Each tag begins and ends with an angled bracket.

<p class="para">This is a paragraph.</p>

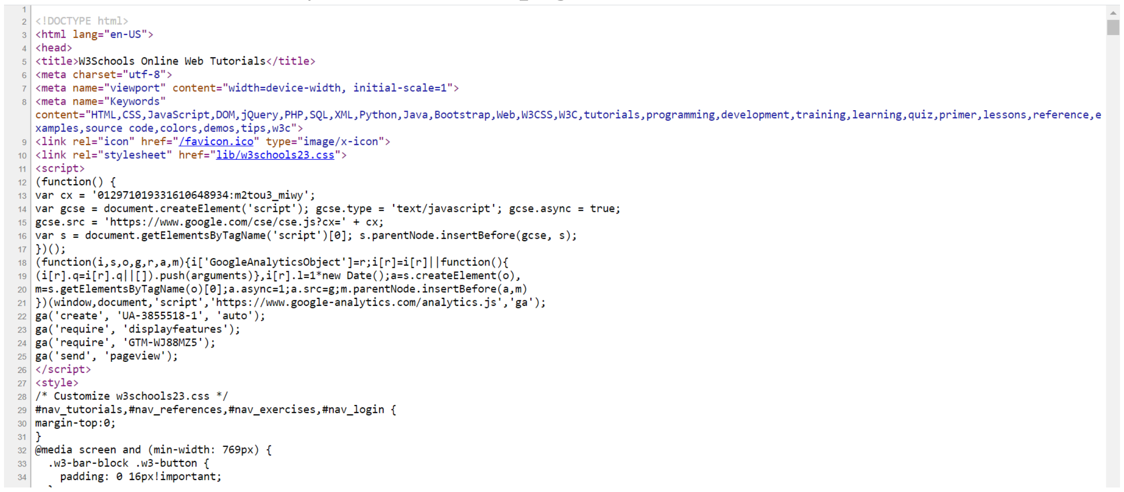
The bold text directly above is an HTML element. Elements are made up of tags, attributes, values, and content. The <p> and </p> parts are the starting and ending tags. "class" is the attribute and "para" is the value you are providing for the attribute. Everything between the starting and ending tags is the content.

Those 4 parts are what makes up the content for pages on the internet. Since every web page is just text and images, you can view the source of the code at any time. To do so, right-click a web page and select "View Source" or "Inspect Element."

View Source will usually bring up a separate web page with all the HTML being used. Inspect Element will pull up your browser's web developer tools and the object you clicked on the page will be highlighted.

The following screenshots are from W3 Schools. They include the "View Source" and "Inspect Element" screen. Inspect Element is a part of the developer tools which are built into browsers. Google Chrome has powerful tools and this is just one of them.

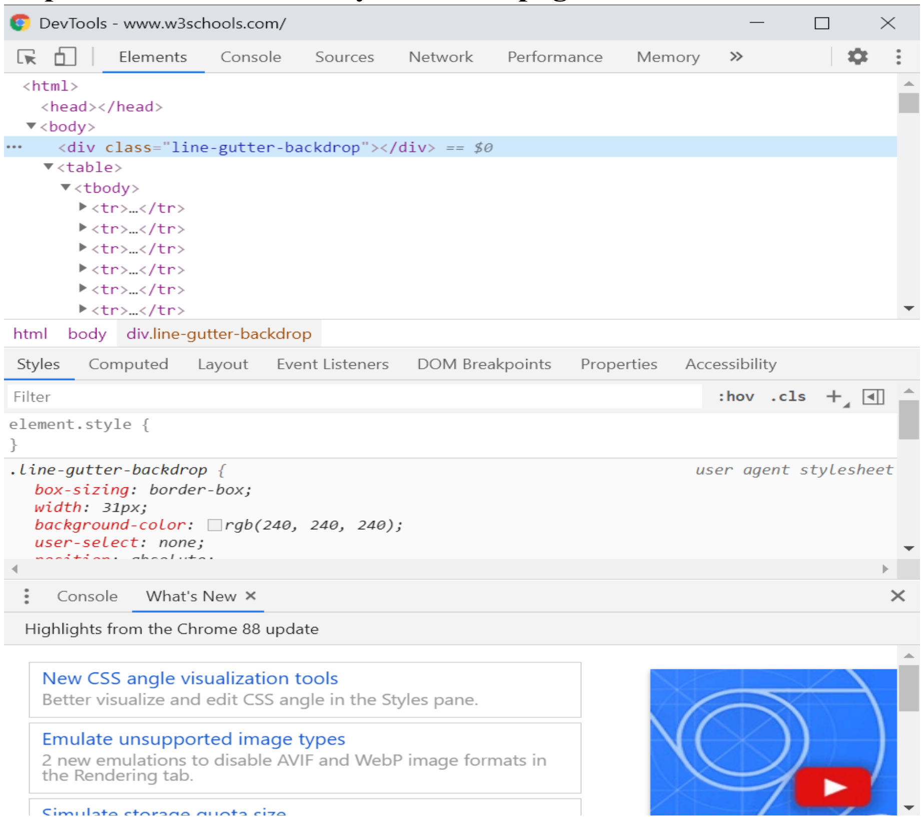
View Source – Allows you to view the page source, but not interact with it



```

<!DOCTYPE html>
<html lang="en-US">
<head>
<title>W3Schools Online Web Tutorials</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<meta name="Keywords"
content="HTML,CSS,JavaScript,DOM,jQuery,PHP,SQL,XML,Python,Java,Bootstrap,Web,W3CSS,W3C,tutorials,programming,development,training,learning,quiz,primer,lessons,reference,e
xamples,source code,colors,demos,tips,w3c">
<link rel="icon" href="/favicon.ico" type="image/x-icon">
<link rel="stylesheet" href="lib/w3schools23.css">
<script>
(function() {
var cx = '012971019331610648934:m2tou3_miwy';
var gcse = document.createElement('script'); gcse.type = 'text/javascript'; gcse.async = true;
gcse.src = 'https://www.google.com/cse/cse.js?cx=' + cx;
var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(gcse, s);
})();
(function(i,s,o,g,r,a,m){i['GoogleAnalyticsObject']=r;i[r]=i[r]||function(){
(i[r].q=i[r].q||[]).push(arguments)},i[r].l=1*new Date();a=s.createElement(o),
m=s.getElementsByTagName(o)[0];a.async=1;a.src=g;m.parentNode.insertBefore(a,m)
})(window,document,'script','https://www.google-analytics.com/analytics.js','ga');
ga('create', 'UA-3855518-1', 'auto');
ga('require', 'displayfeatures');
ga('require', 'GTM-WJ88MZ5');
ga('send', 'pageview');
</script>
<style>
/* Customize w3schools23.css */
#nav_tutorials,#nav_references,#nav_exercises,#nav_login {
margin-top:0;
}
@media screen and (min-width: 769px) {
  .w3-bar-block .w3-button {
    padding: 0 16px!important;
```

Inspect Element – Allows you to view page source and interact with it



```
<html>
  <head></head>
▼<body>
    <div class="line-gutter-backdrop"></div> == $0
  ▼<table>
    ▼<tbody>
      ▶<tr>…</tr>
      ▶<tr>…</tr>
      ▶<tr>…</tr>
      ▶<tr>…</tr>
      ▶<tr>…</tr>
      ▶<tr>…</tr>
```

```
element.style {
}
.line-gutter-backdrop {                        user agent stylesheet
  box-sizing: border-box;
  width: 31px;
  background-color: rgb(240, 240, 240);
  user-select: none;
```

Highlights from the Chrome 88 update

New CSS angle visualization tools
Better visualize and edit CSS angle in the Styles pane.

Emulate unsupported image types
2 new emulations to disable AVIF and WebP image formats in the Rendering tab.

Simulate storage quota size

In HTML, some tags require ending tags and others do not. <p>, <div>, <span>, <strong>, <em>, and many others require a closing tag which looks like the following: </tag>

"tag" is replaced by the name of the tag. <br>, <hr>, and <img> are examples of "empty" tags that don't require a closing tag and can be written without any slashes.

The table below contains some very common HTML tags along with an explanation of their purpose. Most of these tags are ones you have probably seen before, but it doesn't include some of the newer semantic tags in HTML 5.

Tag Name	Purpose
<!Doctype html>	Not really an HTML tag but more like information for the browser about what type of document it should expect. This one is for the latest version of HTML, which is HTML 5, and is the only acceptable one for this course. It always comes first in an HTML document and doesn't have an end tag.
<html>	This is the root of an HTML document and should wrap all other HTML elements. You should always use the lang attribute with this tag to define your web page's language. This is helpful for search engines/browsers.
<head>	This is the head of an HTML document and typically contains information about the document inside of metatags. Things in this tag do not appear on the web page.
<title>	This is the title of an HTML document and the content inside of the tags usually appears in the tab of the browser window. It goes inside of the <head> tag and is required for all HTML documents.
<body>	This defines the body of an HTML document. Elements inside of this tag will appear on the page.
<h1> - <h6>	Defines headings in a document. h1 is the most important while h6 is the least important. In general, only one h1 should be used per page.
<p>	Defines a paragraph.
<ul>	Unordered list. This will display a list with bullet points as the default style.
<ol>	Ordered list. This will display a list with numbers as the default style.
<li>	List item. This tag goes inside of <ul> and <ol> tags to display a single item.
<em>	Defines emphasized text that is displayed in *italic*.
<strong>	Defines text with strong importance and is usually displayed in **bold**.
<table>	Defines a table in a document. This must contain tr and th/td tags at the very least.
<thead>	Used to group header content in a table. It is normally used for the first row of a table.
<tbody>	Used to group the body content in an HTML table. If there is no footer for the table, this is used for the rest of the rows in a table.
<tr>	Defines a row in a table. It must contain th or td tags inside of it for columns

<th> or <td>	Defines a column in a table. <th> should be used for the first row in a table (which should be in a <thead> tag) while <td> should be used in rows after that (those rows should be in a <tbody> tag).
<br>	Defines a line break and is used to break text content apart. This tag does not have an end tag.
<hr>	Horizontal rule. This will display a horizontal line across the screen. This tag does not have an end tag.
<img>	Displays an image. This tag requires two attributes which are src and alt. Src is used to define the source of the image whether it's a local image or something on the internet. The alt attribute is important for accessibility and is used to describe the image for visually impaired users or screen readers. This tag does not have an end tag.
<div>	Defines a division or section in a web page. It is a container, and all kinds of tags can be placed inside of it. It is typically used with CSS to change the layout of a page.
<span>	Like a div, this is also a container. However, this is an inline container which is typically used to mark up a piece of text in a document. An empty div will take up the whole width of a screen while a span tag only takes up the width of the content inside of it.
<form>	Defines a form for user input. It can contain a number of tags including <label>, <input>, <textarea>, and <button>.
<label>	Defines a label for another HTML element. These are required for forms because it helps with accessibility for all types of users.
<input>	Defines an input area for users. The input can come in many types of way depending on the "type" attribute paired with the input tag. This tag does not have an end tag but requires a type attribute to define the input expected.
<button>	Defines a button in a web page. This is typically used with a form to submit user input. The type attribute is necessary to let browsers know what type of button it is.

Tags in HTML are often nested within each other. An example is <ul> and <li>. <ul> is unordered list and <li> is list item and they appear together as shown below.

```
<ul>
   <li>Item 1</li>
   <li>Item 2</li>
</ul>
```

Tags can be nested over and over, but you want your HTML to be tidy and should code in a way that allows you to keep your code uncluttered. If you open a tag inside of another tag, you must close it unless it doesn't require a closing tag.

Semantic HTML
For web pages, the browser interprets the tags used in the .html file and displays them as it comes across them. We can do things like create headings, paragraphs, images, etc. using

specific tags intended for those purposes. If we choose to use tags for purposes they weren't intended for, it could have consequences for users. Of course, we can use things like CSS/JS to cover up poor HTML structure but that only works for entirely visual users.

One example of improper tag usage is when a developer uses table-related tags to create a layout for a page. Div, section, main, aside, and other semantic tags are better suited for the purposes of creating a layout. Screen readers most likely will not read the content in the order you intended if you use a table for layouts.

If we want our page to be as accessible/usable for as many users as possible, it is important to use semantic markup correctly. Any tags we use should help define the document structure in a clear manner and separate content from styles. This will allow users that utilize tools like screen readers to access our content and create a better experience overall.

A lot of web developers tend to use div tags to markup their content as they are useful as containers. However, divs aren't very useful for non-visual users as they don't offer any information as to the nature of the content inside of them. It can be difficult to build a complete page without using divs, but you should do your best to use semantic markup when possible.

Examples of common semantic tags:

- Article
- Aside
- Footer
- Header
- Main
- Nav
- Section
- Summary

Additional Reading

- https://html.com/semantic-markup/
- https://www.lifewire.com/why-use-semantic-html-3468271
- https://internetingishard.com/html-and-css/semantic-html/

HTML Validation
For our pages to present themselves correctly and behave in expected ways, it is important to validate our documents. A document's code is considered valid when the tags are nested correctly, and everything is placed according to W3C specifications. An example would be a <ul> tag containing only <li> elements inside of it. One rule that trips up students a lot is that <ul> cannot be a direct child of another <ul>. The sample code below demonstrates good and bad structures for lists.

Good List Structure	**Bad List Structure**
<ul>	<ul>
<li>Item 1</li>	<li>Item 1</li>
<li>Item 2	<li>Item 2</li>

`        <ul>`	`        <ul>`
`            <li>Sub Item 1</li>`	`            <li>Sub Item 1</li>`
`        </ul>`	
`    </li>`	`        </ul>`
`    <li>Item 3</li>`	`        <li>Item 3</li>`
`</ul>`	`</ul>`

In the example above, the left side has the 2nd <ul> tag nested **inside** of the 2nd <li> tag. The </li> tag is closed **after** the 2nd <ul> set is closed. In the bad list portion, the 2nd <ul> is a direct child of the 1st <ul> and therefore not valid. This is just a rule that is a standard in HTML and there are many others like it. Look at the example below for an explanation of list nesting:

https://cinf362.000webhostapp.com/examples/nesting-example.html

I don't expect you all to memorize all rules which is why you place your content in a validator. It will tell you what lines of code are wrong so you can fix them and improve your document's markup. To check your code, copy all of it and paste it directly into the validator. You can also upload files or provide a link, but copy/pasting code is quicker.

Use the website below to validate your code in the assignment mentioned below. I would recommend bookmarking it in your favorite browser for the semester. Chrome is my browser of choice, but Edge and Firefox are pretty good as well. A green bar means your document validates perfectly.

https://validator.w3.org/#validate_by_input

Class Examples
https://cinf362.000webhostapp.com/examples/nesting-example.html (nesting)
https://cinf362.000webhostapp.com/examples/html-basics-example.html (HTML in general)
https://www.albany.edu/~cv762525/cinf362/videos/ (HTML-Basics.zip – It's a zipped video file)

There is also content on Blackboard to help you with understanding/debugging HTML code. These items are located on Blackboard → Resources → HTML and will help further explain HTML and demonstrate some best practices you should adopt moving forward.

Resources/Online Documents
Tutorials
https://www.htmldog.com/guides/html/
https://html.com/ (Read up until "Our Other HTML Tutorials" and that should suffice)
https://www.codecademy.com/learn/learn-html (Interactive, full-blown tutorial – you **DO NOT** have to complete this, but if you want to learn on the side, it helps)

Reference Guides
- https://html.spec.whatwg.org/multipage/ (very technical guide to the whole specification)
- https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML5

- http://html5doctor.com/element-index/

Before completing the assignment(s) for this week, please read the "Viewing Your Web Pages.docx" file on Blackboard. You will not be able to submit anything for the assignment without completing that portion first. It is located directly in the Lecture Notes folder.

HTML Exercise

Due February 6th at midnight
Your task is to build a webpage based on requirements given to you. After you are done creating the page, you will upload it to the UAlbany or 000webhost server so it can be submitted as a live link. To get started, download the "HTML-Exercise.zip" folder from Blackboard. It's attached to the assignment which can be found in this week's lecture notes or the Assignments folder.

Inside of that folder, you will find an HTML file and an image. Open the HTML file inside of a text editor and add code to it as mentioned in the list below Also, complete the research portion of it below the list. All code should be submitted in one page to Blackboard for credit. The image is only there to show you what a potential final page could look like; feel free to use your own information. **Your table will have more content and my page looks thinner because I took the screenshot while the browser was at half width.**

1. Fill in the **title tag** with an appropriate title for the assignment (HTML Exercise is fine)
2. Add an **h1 tag** with "About Me" inside of it
3. Add an **image** of something you like below it, make sure to include an alt tag attribute
4. Add a **paragraph** below the image with the following content inside of it:
    a. A brief explanation as to why you chose the image or what it's about
    b. A sentence about what your dream job would be. When you mention your dream job, use em tags to make the job appear emphasized
    c. A sentence about your favorite food. When you mention the food, use strong tags to make it appear bold on the page
5. Add an **h2 tag** with "Hobby Section" inside of it
6. Add a **p tag** which says, "Below are some hobbies that I enjoy."
7. Create an **unordered list** with three of your favorite hobbies/interests
    a. Two of those hobbies should have another list nested inside with one related website
    b. Each website should be wrapped in **a tags** to hyperlink them. Your hyperlink should point to the website you mention (href attribute) and the text should be related to the website's name
8. Add an **h3 tag** with "Definition Section" inside of it
9. Create a definition list using the **dl tag**
    a. Inside of this list, pick any two terms/words that you like (**dt tag**) and then define them (**dd tag**). For this part, one definition is enough for each word/term.
10. Add an **hr tag** beneath the list so that a line appears next
11. Follow the next set of instructions for a **table**

Look up the tags listed below and build a table based on them. The table will contain three columns in it. The first column should have the name of the tag, the second column will explain the purpose of the tag, and the third column will let us know if it is still in use. The possible values for the third column are Yes or NO with one of the following: "Not in use", "Use CSS instead", "New to HTML5", "Not supported in HTML5", or Replaced by <tagname>. A tag can have any number of these designations and I've provided an example below.

Name	Purpose	Still in Use?
Center	A tag used to center align items it is wrapped around.	No – Use CSS
Embed	Defines a container for external applications (video, etc.)	Yes – New to HTML5

- Font
- Aside
- Footer
- Strike
- Acronym
- Canvas
- Dir
- Applet
- Nav
- B
- Header
- Section

Please note that I have added styles to the page so your tables will have lines/spacing in them. They are located in the style tags in the <head> tag of the document. Make sure you don't remove them. Tables by default don't have any sort of padding or borders on them.

Your webpage is **due on Sunday, February 6h at midnight.** To submit the work for this exercise, visit Blackboard → Course Materials → Lectures Notes for this week's class. You can also go into the "Assignments" folder and the submission area will be titled "HTML Exercise." You should be submitting a live link to me; it can be through the UAlbany server or 000webhost. The work submitted will be evaluated based on the rubric explained below.

HTML Exercise Rubric – 10pts

- Readability of HTML: 2pts
    - Do you use tabs/spaces consistently?
- Validity of HTML: 2pts
    - Does the HTML validate in the validator?
- All required content is included: 6pts

Next Week

We will have the second of our "bootcamp" sessions. These are designed to catch you up on the core web technologies used in this course. If you're still familiar with CSS, this week's work shouldn't be too difficult. If you've never seen CSS before, this content will help you understand the basics of CSS so that you can style your pages.

- https://www.htmldog.com/guides/css/ - CSS Beginner Tutorial only
- https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps/What_is_CSS
- https://www.tutorialspoint.com/css/css_syntax.htm - Specific syntax examples