HTML5 AND CSS3

TABLE OF CONTENTS

MODULE 0 - COURSE ORIENTATION

- I am your presenter, Richard Raney.
- You are here because of the wonder and amazement of today's web-based technologies.
- The Internet is a vast wealth of information. Think of it as a big-box store where everything can be found in one location.

COURSE OUTCOMES

- Within this video series you will learn the fundamentals of webpage development using HTML5 and CSS3.
- Learn best practices for text and color configurations.
- Enhanced focus on page layout, design, accessibility, and Web standards.
- The student will create webpages using a text-editor such as Notepad, Notepad ++ (recommended for Windows), TextEdit (Mac), or an editor of their choice.
- Do not use a word processor because they use special coding behind the scenes and browsers may not render the content properly.
- We have all at some point experienced the Internet for social networking, research, general information, or self-publishing (blogs, vlogs, podcasts, etc.). If you are watching this video series, you have found an interest or curiosity of how things work behind the scenes. Modules 1 through 10 should be viewed in order as to maintain structure for adaptive learning.

LEARNING IS DYNAMIC

- Although this course has multiple videos associated with this manual, you will see on occasion an ***ADD/EDIT*** within a section. These are for students to code the solution without a video instruction.
- This performs three important factors when coding.
    - Stare and Compare – This is a popular term in coding when there is a reference to use for comparing code between same elements, e.g., HTML, CSS, Java, JSON, etc.
    - Trial and error – Although a student can be shown the steps, it is a good practice for students to troubleshoot code.
    - During any copy/paste activity, it is quite common to miss a character.
        - copy – <div>some type of code</div>
        - paste - <div>some type of code</div
            - As you can see, the 'greater than' symbol was not pasted, leaving the <div> element open, thus creating an error which will render the webpage incorrectly.

MODULE 1 - INTERNET & WEB

THE INTERNET AND THE WEB

- The Internet is the interconnection of networks for the purpose of exchanging information.
- The World Wide Web is the collection of computers (servers and clients) that host webpages for anyone with Internet access can view.

WEB STANDARDS AND ACCESSIBILITY

- The Internet is not owned by any single individual or entity.
- There are standards that should be followed so that all people can access the information available – Accessibility. The Americans with Disabilities Act (ADA) states that anyone that creates webpages should make them available and understandable to anyone with disabilities.
- These standards are created and recommended by a group of people called the World Wide Web Consortium or “W3C” for short. Keep in mind there are no specific rules for creating webpages, however; in order to reach the most people from all walks of life these standards are highly recommended.

INFORMATION ON THE WEB

- Keep in mind that not all the information on the web is viable or true. Searching for specific information needs to be validated hopefully by more than one website. We hope that the information we are seeking is trustworthy, but we also need to realize there are many unethical uses for the Internet.

NETWORK OVERVIEW

- A network is a collection of two or more computers to share information. Within this collection is the Client/Server Model.

THE CLIENT/SERVER MODEL

- The client/server model is the basis of how the Internet works.
- When you sit at your computer, tablet, or Smartphone you are the Client.
- When you are browsing or working within an app and you request information you are accessing a Server.

INTERNET PROTOCOLS

- Protocols define how communications between a client and server are established.
- The three most popular protocols used by web developers are:
  - FTP (File Transfer Protocol)
    - The instructions needed to transfer files from one location to another.
  - Email Protocols

        - The most commonly used email protocols are SMTP (Simple Mail Transfer Protocol), and IMAP (Internet Message Access Protocol).
    - HTTP or HTTPS
        - HyperText Transfer Protocol is the set of rules needed for a browser to read the webpage code and display it as intended.
        - These rules are browser dependent, and the code should meet the requirements of each type of browser collectively. This is accomplished via Cascading Style Sheets (CSS).
        - The “s” at the end of HTTPS is to inform the user that the webpage is secure and encrypted.

URLS AND DOMAIN NAMES

- A URL or Uniform Resource Locator is basically where the resources such as images, videos, webpages, etc. are located.
- Domain names locate the server(s) that holds the information/resources users request.
- As seen in this image we have the http protocol, the web server, the domain name, a directory, and the file. This is the full URL or website address.

MARKUP LANGUAGES

- HTML (HyperText Markup Language)
    - The most popular of web development languages. It provides the instructions that a browser uses to display webpages.
    - This simple example will become your first webpage.
- XML (Extensible Markup Language
    - XML provides flexibility to a developer. It does not replace HTML but the two work together.
    - The use of XML allows developers to create their own tags that better describe their information as seen in the image below.
- XHTML (Extensible HyperText Markup Language)
    - This language uses both XML and HTML4.
    - Notice the html !DOCTYPE and the XHTML encoding.
- HTML5
    - HTML5 is the latest and greatest version of HTML.
    - It is intended to replace HTML4 and XHTML with new elements, form edits, and video rendering.
    - The W3C, as of 2017, approved its recommendations for HTML5.1, which is the syntax we will use throughout this series.

POPULAR USES OF THE WEB

- E-Commerce
- Mobile Access
- Social Networking
- Blogging
- Vlogging
- Podcasts
- Personal and Academic Research

MODULE 2 - HTML BASICS

HTML OVERVIEW

- Tags
  - Tags or elements are the coding used to manipulate the content of the webpage.
- Attributes
  - Attributes give tags more details.
  - A font tag <font></font> give the text a certain look, but an attribute provides additional manipulation of the content. <font color=”red”></font>. The color red is the attribute of the tag <font>.

DOCUMENT TYPE DEFINITION

- This is the first line in the webpage document.
- It defines the markup language used to properly render the webpage.
- HTML5 uses a rather simple doctype of <!DOCTYPE html>.

WEB PAGE TEMPLATE

- Nearly every HTML webpage will have the same six elements.
  - <!DOCTYPE>
    - As mentioned earlier this tells the browser what version of html is in use. It is a self-contained tag.
  - html <html></html>
    - This tells the browser this document is formatted for html.
    - Additional attributes can be included within the html tag such as what language (lang) to display.
    - It uses a closing tag at the end of the document.
  - Head <head></<head>
    - The head element contains information about the webpage. It uses opening and closing tags.
    - It also contains internal and external scripts, style sheets, character coding, etc.
    - It must have a closing tag before the body tag.
  - Title <title></title>
    - Coded within the head tag that displays the name of the webpage on the title bar of the browser.
    - It uses a closing tag.
  - Meta <meta>
    - Meta data describes characteristics of the page.
    - It is a self-contained tag meaning it does not need a closing tag.
    - <meta charset=”utf-8”>
  - Link <link> - optional
    - Link is used to retrieve or access external documents.
    - These documents can be internal or external to the website.
    - Stylesheets, JavaScript, etc.
    - It is a self-contained tag.

- Body <body></body>
    - The body tag contains the content of the webpage.
    - This includes all text, images, video, etc.
    - It has to have a closing tag before the footer <footer> (the footer tag(s) are optional) or before the closing html tag.
- Footer <footer></footer> - This is optional
    - This contains copyright information, disclaimers, etc.
    - It also contains plain-text navigation links for Accessibility.
    - It has to have a closing tag.
- Basic structure for standard HTML

```
<!DOCTYPE html>
<head>
        <title>Some Title</title>
        <meta charset=”utf-8”>
        <link **for external documents>
</head>
<body>
                Content, Content
                Content, Content
        <footer>
                copyright
                Text navigation links
                Additional information
        </footer>
</body>
</html>
```

YOUR FIRST WEB PAGE

- Let us get to the good stuff.
- Moving forward will be hands-on tutorials.
- You can pause the video at any time to work with the examples.
    - Let us create a folder for your webpages. I recommend placing this folder in My Documents (for Windows) or File Manager (for Mac) for ease of access.
    - Name the folder MyWebpages.
    - Open your text editor. Do not use a word processor, use Notepad, Notepad++ (highly recommended for Windows) or TextEdit for Mac.
    - You can pause this video at any time to keep up.

```
<!DOCTYPE html>
<html lang="en">
<head>
        <title>Hello World</title>
        <meta charset="utf-8">
```

```
</head>
<body>
        Hello World.
</body>
</html>
```

- Save the file as helloWorld.html
- Navigate to the MyWebpages folder and double-click the helloWorld.html document. It should open up in your default browser. Mine is Google Chrome.
- There are a few unwritten rules for naming conventions and writing code.
- Each of these are valid and many developers use these naming schemes.
    - Never use spaces when creating folders and filenames. The servers that store web elements do not like spaces.
    - Capitalize the first letter of each word in the folder name, MyWebpages.
    - Keep the first letter of the filename lowercase and capitalize the first letter of the remaining words, helloWorld.html.
    - Each new element within the document must have a closing tag with the exception of the html <html …>, <meta…>, and line-break tag, <br>.
    - When writing code, it is best to indent each beginning element and content. It makes it easier to follow and troubleshooting.
- That was not too difficult, now was it. Ready to move on?
- In the first series we talked about the Internet and the World Wide Web.
- You also created your first webpage using basic HTML elements.
- Now, we will create a template to save time on typing for the remaining of this series.

BASIC HTML TEMPLATE

- Start with a new document using your text editor.
- Save the document in your MyWebpages folder and name it myWebTemplate.html.
- Add the following to the new document:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title></title>
    <meta charset="utf-8">
</head>
<body>
</body>
</html>
```

- Save the file.
- Now you have a template to save some time.
- I recommend creating several templates when creating webpages. I also recommend to copy/paste code rather than typing it repeatedly.
- In the next video we will talk about headings and their role in the structure of your webpages.

HEADING ELEMENT

- Heading elements use six levels of default sizes.
  - The largest is <h1> with <h6> being the smallest.
  - Do not confuse heading elements with the head tag. The two are very much different.
  - Let us practice.
  - Open the file named myWebTemplate.html
  - Save As (do not just Save because you will overwrite the template) myHeadings.html.
  - Type in “Heading Elements” without the quotes between the title tags. <title>Heading Elements</title>
  - Now, underneath the <body> tag type the following…
  - <h1>Heading Level 1</h1>
  - <h2>Heading Level 2</h2>
  - <h3>Heading Level 3</h3>
  - <h4>Heading Level 4</h4>
  - <h5>Heading Level 5</h5>
  - <h6>Heading Level 6</h6>
  - Your document should like very similar to mine. If it does not, pause the video and make corrections.
  - Save the file, navigate to your folder, and double-click the file. It should open a browser window and display like the one shown.
  - FYI, when using heading tags, there is a line-break automatically inserted after each closing tag.
  - TIP – Use headings to make your pages more usable. Headings help to maintain structure to the page and the content. We will see more examples in the proceeding videos.
  - Headings create visual sections within the webpage and the height of the header can signify its importance to the topic as a sub-topic.
  - In the next video, we will use the Paragraph Element to further provide additional structure.

PARAGRAPH ELEMENT

- Paragraph elements help to keep sentences grouped together.
- They are coded as <p> with a closing tag </p>.
- This leaves an empty space above and below the content within the tags.
- Let us do an example.
- Use the file you used for the headings. If it is not open, within the text editor click Open, navigate to the web folder, and double-click the headings file.
- Save As paragraphExample.html
- Go to the <title> tag and replace the title with “Paragraph Example” without the quotes.
- Below <h1></h1> tags type the following:
- <p> This is an example of a paragraph using the paragraph tag. It is good to use standardized methods to maintain a readable structure of your content. </p>
- Save the document, navigate to the file, and double-click it. It should look like the one shown here. If it does not go back and review your steps and make any necessary corrections.
- Save the file again and open the file.
- Congratulations. You have truly begun coding webpages with usable and readable structure.

- In the next video we will discuss the line-break element <br>.

LINE BREAK ELEMENT

- The line-break element is used to separate sentences within a paragraph.
- It is also used to add blank lines between elements of the document.
- <p>This is a paragraph. <br> This is the next line using a line-break.</p>
- Let us do an example.
- If it is not open, open the file paragraphExample.html.
- Save As lineBreak.html
- Change the title to “Using Line-Breaks” without the quotes.
- Insert a line-break <br> before ‘Heading tags…’ like shown.
- Save the file.
- Navigate to the web folder and double-click the file lineBreak.html. It should look like the one shown.
- If not, review the steps and make any necessary corrections.
- Line-breaks are a widely used element, probably the most used element.
- However, be careful when using <br> elements because the layout can change with unwanted results depending on the device, computer, tablet, Smartphone, etc. and the browser.
- Next, we will talk about the blockquote <blockquote> element.

BLOCKQUOTE ELEMENT

- The blockquote element is best used to display quoted text.
- It indents the entire quote as to create a visual separation from other content in the webpage.
- It requires a closing tag, <blockquote>text</blockquote>
- Let us do an example.
- Within your text editor open the file myWebTemplate.html.
- Save As blockquoteExample.html
- Use “Blockquote Example” without the quotes as the title.
- Insert heading 1, <h1> and enter William Shakespeare </h1>
- Insert a paragraph tag <p> and enter One of Shakespeare’s most famous quotes is:</p>
- Insert a blockquote tag <blockquote>
- To be or not to be, that is the question.
- Close the blockquote </blockquote>
- Insert another <p> tag and add, “This is what a blockquote looks like.” without the quotes and close the tag </p>.
- Save the file, navigate to it, and open it. It should look like the one shown.
- It not, review the steps, make any necessary corrections, and save and open.
- Blockquotes can be used for other purposes to differentiate specific content from other content.
- Next, we will discuss Phrase Elements.

PHRASE ELEMENTS

- Phrase elements are used to manipulate text depending on how the text is used.
- The resulting effect of the element is shown directly in the content of the webpage.
- These tags require closing tags to include the text being manipulated.
- There are numerous phrase elements that can be use, but the more popular elements are:
    - <b> or <strong> for **bold** text.

    - <cite> or <i> or <em> for *italicized* text.
    - <mark> for highlighting text.
    - Additional phrase elements can be found at https://www.w3.org/TR/html4/struct/text.html#h-9.2.1
    - Let us do an example
    - If it is not already open, open the file blockquoteExample.html.
    - Save As phraseElementsExample.html.
    - Let us make the paragraph bold.
    - After the <p> tag insert <strong> and before the closing </p> tag close the </strong> tag.
    - Next, Let us emphasize the quote by adding the <em> tag in front of the quote <em>To be….
    - Close the tag by inserting the </em> tag after the period in the quote.
    - Save the file.
    - Navigate to the file and double-click to open in a browser.
    - It should look like the one shown. If it does not, review the steps, make any necessary corrections, and save and reopen.
    - As you seen while performing the tutorial that Phrase Elements can draw attention to content.
    - This is relatively important when attempting to relay the importance of the content.
  - Next, we will discuss Ordered and Unordered Lists and create an example.

ORDERED LIST AND UNORDERED LISTS

  - Ordered Lists are used to numerically list items in a specific order.
  - The list attributes can be:
    - Numerals
    - Uppercase
    - Lowercase
    - Roman Numerals
    - Lowercase Roman Numerals
    - The above can also be in descending order by using the reversed=”reversed” ordered list attribute.
    - If necessary, the list can be started with a specific number by using the start=”10” ordered list attribute.
    - Let us do an example.
    - Open the myWebTemplate.html file.
    - Save As orderedListExample.html
    - Make the title “Ordered Lists” without the quotes.
    - Insert the ordered list tag <ol> (this is the letter “o” and the letter “L” in lowercase).
    - Insert the following for the list.
      - <li>Red</li>
      - <li>Blue</li>
      - <li>Green</li>
    - Now close the ordered list </ol>
    - Save and navigate to the file and double-click to open in a browser.
    - It should look like the one shown. If it does not, review the steps, make any necessary corrections, and save and reopen.
  - Unordered lists are nearly the same with the exception of numbering.
  - Unordered lists are there to list items without a hierarchy, so the visitor does not place any value of one item over another.

        - With the orderedListExample.html still open, Save As unorderedListExample.html.
        - Simply change <ol> to <ul> and the closing tag from </ol> to </ul>.
        - Save the file.
        - Navigate to the file and double-click to open in a browser.
        - It should look like the one shown. If it does not, review the steps, make any necessary corrections, and save and reopen.
        - Ordered and Unordered Lists are common in a number of webpages.
        - These two are greatly used when working with steps to accomplish a task.
        - Next, we will examine the use of Description Lists and create an example.

DESCRIPTION LIST – HTML5

- Description Lists are great for a description or definition of a word or phrase.
- It is similar to an ordered or unordered list with the exception there are no numerals or bullets.
    - Let us do an example.
    - Open the file myWebTemplate.html.
    - Save As descriptionListExample.html
    - Enter “Description List Example” without the quotes as the title.
    - Insert the <h1> tag after the body tag and enter “Example of a Description List” without the quotes and close the </h1> tag.
    - Insert the <dl> (lowercase ‘d’ and lowercase ‘L’) tag. Think of it as descriptionlist.
    - Insert the <dt> tag. This is the word or phrase that is to be described. Add HTML and close the </dt> tag.
    - Insert the <dd> tag. This is the definition. Add HTML stands for HyperText Markup Language and is used in nearly every webpage.
    - Close the </dd> tag.
    - Insert the <dt> tag. This is the term that is to be described. Add CSS and close the </dt> tag.
    - Insert the <dd> tag. Add CSS stands for Cascading Style Sheet. It is used to provide additional formatting to elements and ends with the .css extension. Close the </dd> tag.
    - Close the </dl> tag.
    - Let us add a <p> tag to give a finished look. Add This is how a Description List looks. and close the </p> tag.
    - Save the file.
    - Navigate to the file and double-click to open in a browser.
    - It should look like the one shown. If it does not, review the steps, make any necessary corrections, and save and reopen.
    - Description Lists are useful for FAQs, establishing a vocabulary for a specific topic, etc.
    - Next, we will examine the use of Special Characters and why they should be coded, and then create an example.

SPECIAL CHARACTERS

- The use of special characters requires the use of character codes. Each code for a character will begin with an & (ampersand) and end with a ; (semi-colon).
- The ampersand lets the browser know that this is a special character, and the semi-colon tells the browser this is the end of the character code.
- Here are a few special characters and the code needed to properly display them.
    - Quotation marks “ use the &quot; code.

    - Apostrophes ‘ use the &#30; code.
    - Empty or Blank spaces use the &nbsp; code.
    - Ampersands & use the &amp; code.
    - Copyright symbol uses the &copy; code.
- Let us do an example using special characters.
    - Open the file myWebTemplate.html.
    - Save As specialCharacters.html.
    - Add the title Examples of Special Characters.
    - Insert the <h1> and add Below are special characters and their proper code and close the </h1> tag.
    - We will use the <br> tag to place each on a separate line.
    - You will notice that we are adding the ampersand symbol twice so we can show the character and the code.
    - Add Quotation marks &quot; uses the &amp;quot; code.
    - Insert <br>
    - Add Apostrophes &#30; use the &amp;#30; code.
    - Insert <br>
    - Add Empty or Blank spaces &nbsp;nbsp; use the &amp;nbsp; code.
    - Insert <br>
    - Add Ampersands &amp; use the &amp;amp; code.
    - Insert <br>
    - Add Copyright symbol &copy; uses the &amp;copy; code.
    - Insert a <p> and add These are just a few special character codes. More can be found at https://www.w3schools.com/html/html_symbols.asp </p>
    - Save the file.
    - Navigate to the file and double-click to open in a browser.
    - It should look like the one shown. If it does not, review the steps, make any necessary corrections, and save and reopen.
- The more experienced you get the more you will understand why it is always better to code a special character than trying to natively insert one.
- Next, we will examine the use of Structure Elements to help your webpage standout.

STRUCTURAL ELEMENTS

- HTML5 introduced new structure element to arrange certain areas based on their use.
    - Div Element
        - This element creates “dividers” within a specific area of the webpage. It places empty spaces above and below the content within.
        - It uses closing tags <div></div>.
    - Header Element
        - This is generally first structure element within the document. It generally contains heading elements and its text.
        - It has a closing tag <header></header>.
    - Nav Element
        - This area contains navigation links for locating information within the website. This element can also contain links to external websites.
        - It has a closing tag <nav></nav>.
    - Main Element
        - It contains the main content of the webpage and should only be used once per document.
        - It has a closing tag <main></main>.
    - Footer Element

        - The footer section contains general links and Copyright/Trademark information.
    - Let us do an example.
- We are going to create a new HTML5 template to use during this and proceeding tutorials.
    - Open the file myWebTemplate.html.
    - Save As myHTML5Template.html
    - First, insert the <header></header> tags beneath the opening body tag <body>.
    - Next, insert the <nav></nav> tags beneath the closing </heading> tag.
    - Next, insert the <main></main> tags.
    - Next, insert the footer <footer></footer> tags.
    - Save the file for this new template.
    - Now we will begin coding using HTML5 elements.
    - With the new template still open, Save As sturctureExample.html.
    - Begin by adding the title “Conceptual Innovative Technologies” without the quotes.
    - Insert the <h1> tag inside the <header> tags and add Conceptual Innovative Technologies and close the </h1> tag.
    - Within the <nav> tags insert the <strong> tag to bold the text and add some spaces using the special character code for spaces. Use Home, About Us, and Contact within and then close the </strong> tag.
    - Let us place some content in the <main> tags.
    - Insert <h2> and add Innovative Concepts and Designs the close the tag </h2>.
    - Insert a <p> tag and add CIT uses multiple disciplines to create innovative technologies.</p>
    - Let us insert a <div> tag and add content. We will also use the <br> tag for each line.
    - Insert the <div> tag below the paragraph and add the following:
        - CIT<br>
        - 1234 Main St.<br>
        - Some City and State, 11223
        - 555-555-5555<br><br>
    - Close the </div> tag.
    - Next, we will add some copyright notices to the footer. Let us make the text smaller and italicize it.
    - Insert <small><i>Copyright &copy; 2020 Your Name Here and close the tags in the correct order </i></small>. Remember that tags need to be closed in reverse order.
    - Save the file.
    - Navigate to the file and double-click to open in a browser.
    - It should look like the one shown. If it does not, review the steps, make any necessary corrections, and save and reopen.
- Congratulations on creating your first HTML5 webpage. It only gets better and more interesting as we proceed.
- Next, we will discuss and add hyperlinks using the Anchor Element.

ANCHOR ELEMENTS

- Anchor elements <a></a> create hyperlinks to navigate to internal and external sources.

- All anchor tags will use the href attribute. This attribute is the actual URL (address) of the resource and must be enclosed in double-quotation marks.
- You may have seen hyperlinks as bright blue or bright blue underlined text.
- What makes anchor tags so versatile is that you can have short text or phrases that represent a web address that could be dozens of characters long.
- There are two types of hyperlinks:
    - Absolute hyperlinks
        - These links contain the exact address beginning with http:// or https://
        - We can use absolute links to navigate within our website.
    - Relative Hyperlinks
        - Relative hyperlinks are used to navigate to webpages within the website.
        - It is relative to the current webpage displayed in that it resides in the same directory.
        - FYI, you can navigate to internal webpages that reside outside the current directory by using ../. This only navigates to the upper level directory. We will explain this in more detail in the proceeding modules.
    - Let us open our myHTML5Template.html.
    - Save As anchorElementExample.html
    - Add the title Hyperlinks using Anchor Tags
    - Insert an <h2> tag inside the <header> tags and add Click the link below to navigate to W3C Standards and close the </h2> tag.
    - Leave the <nav> tags empty for this example.
    - Within the <main> tag insert the <a> tag and add the href=”https://www.w3.org/standards/”>W3C Standards</a>.
    - Save the file.
    - Navigate to the file and double-click to open in a browser.
    - It should look like the one shown. If it does not, review the steps, make any necessary corrections, and save and reopen.
    - Hyperlinks are the backbone of navigating the Internet.
    - It is vital that your visitors can link to the appropriate resources when navigating your website. It is imperative to test each link from time to time to ensure the link is still active.
    - Broken links can happen when organizations redo their websites and remove unnecessary links.
    - To this module, we will explore how to validate your webpage to ensure that it meets W3C standards and that your html code is correct and free of errors.

HTML VALIDATION

- Validation ensures your webpage is using the correct syntax.
- Errors are reported accurately as possible.
- A majority of errors have to do with tags not closed in the proper order.
- Tip – using a text editor that color-codes the syntax will greatly aid in troubleshooting errors.
- Let us create an error of an earlier tutorial.
    - Open the file structureExample.html
    - Save As validateExample1.html.
    - Remove the / from the closing </h1> tag.
    - Within the <strong> tag remove the semi-colon after the first &nbsp.
    - Next, remove the closing </div> tag.
    - Save the file.

    - Navigate to the file and double-click to open in a browser.
    - You will notice that it looks the same as the original file. However, the errors may not allow it to render properly in other browsers.
    - Next, open a browser and go to http://validate.w3.org and follow the instructions. Click the tab “Validate by File Upload” and browse to the validateExample.html.
    - You will receive a list of errors. Notice how the errors are reported. Because we created the errors, we can correct them. This is good practice to better recognize errors and how to correct them.
    - Now, use the validation tool to check the original file of structureExample.html
    - You should see there are no errors to report.
    - Writing HTML code is not as difficult as one would imagine. However, you must pay close attention to detail.
    - Practice creating your own webpages using the preceding examples and you will soon master the basics of creating webpages.
  - I look forward to showing you how to incorporate Cascading Style Sheets within your webpages in Module 3.

MODULE 3 - COLOR & TEXT, CSS

OVERVIEW OF CASCADING STYLE SHEETS

- Style sheets have been used for years.
- Its purpose is to provide flexibility and a single-point of configuration for multiple webpages.
- This means that when multiple webpages use the same formatting and a change is necessary, the change needs to be implemented manually for each webpage; this is time consuming, and errors are prone to creep in.
- Because a single style sheet can provide the formatting for multiple webpages the change only needs to be implemented in one location, the style sheet.
- Advantages of Cascading Style Sheets are:
  - Better control over typography and page layouts.
    - Use typography and structure images
  - Style can be separate from structure.
  - Styles can be store externally so that one change can propagate to multiple webpages.
  - Potentially can make the document smaller in size because the style is stored separately.
  - Maintaining the site is simplified due to one-change-affects-all.
    - Note: *CSS3 is not supported by all browsers. Additional information can be located at http://caniuse.com/#search=css3. We will focus on aspects that are supports by most browsers.*

CONFIGURING CASCADING STYLE SHEETS

- There are four specific means of configuring styles
  - Inline Styles – These are coded within the webpage tags.
  - Embedded Styles – These are coded generally inside the <head> tag and are also referred to as Internal Styles.
  - External Styles – These are styles that are configured in a separate document and linked to the webpage via a <link> tag. Link tags are self-contained and require additional attributes.
  - Imported Styles – These styles come from an external source not associated with the webpage/website. It is similar to an external sheet but requires special coding.

CSS SELECTORS AND DECLARATIONS

- Each CSS element requires a selector and a declaration.
  - CSS Style Rule Selector – This is an HTML element, a class name, or an id name.
  - CSS Style Rule Declaration – This is the property of the element being manipulated. This also requires a declaration value to validate the property.

THE BACKGROUND-COLOR PROPERTY

- This will configure the background color of an element, body, header, nav, etc.
- The style for the element is enclosed in braces.
- body {background-color: yellow} **

- Braces are always used to contain the declaration property and the declaration value.

THE COLOR PROPERTY

- The color property configures the color of the text inside an element.
- The example below will make all the text within the <body> tags blue.
- body {color: blue}

COMBINING PROPERTIES

- Styles allow us to use multiple properties within the same element.
    - body {color: white; background-color: orange;}
    - Notice the use of semi-colons to separate the properties. This is something important to keep in mind. Without the semi-colon to separate the properties, the styles will not render properly on the webpage.
    - There are several ways to format the elements and properties as seen below.
    - body {color: white; background-color: blue}
    - body {color: white;
        - background-color: blue;}
    - body {
        - color: white;
        - background-color: blue;
        - }
    - The latter is the preferred method. However, when using a single property, the single line method is quite acceptable.
    - To see more element properties, refer to the W3C's CSS Reference Properties at https://www.w3schools.com/cssref/default.asp
- Styling text color and background-color is quite acceptable and widely used in web development.
- Colors, especially background-colors can add some visual separation within sections of the webpage.
- Next, we will look at acceptable uses of color on webpages.

USING COLOR ON WEB PAGES

- In the old days, web developers were limited to 216 colors due to only having 8-bit color monitors.
- These days we can configure nearly 16 million colors.
- We achieve this by using hexadecimal values to make color.
- The color declaration value will begin with the # (hash-tag) followed by six hexadecimal characters; 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F (there are 16 characters that make up hexadecimal values).
    - Some basic values are:
        - #FFFFFF is white
        - #000000 is black
        - #FF0000 is red
        - #00FF00 is green
        - #0000FF is blue.
    - To make the background-color of an element using the hexadecimal code looks like:
        - body {background-color: #00FF00}

- TIP – It is common practice to use capital letters when coding colors using hexadecimal; #AA00BB, #12AB34, etc.
- Go to Color Picker at https://htmlcolorcodes.com/color-picker/ to look at color selection and the hexadecimal code.
- The use of colors can be complicated and over-thinking can create havoc.
- Keep in mind that certain colors will blend in and make the text difficult to read. When selection background-color and text color look for good contrast.
- In the next section we will look at Inline Styles and work a few examples.

INLINE CSS WITH THE STYLE ATTRIBUTE

- Inline styles are coded within the HTML element using the *style* attribute.
  - <body style="declaration property: declaration value>
- The value of the style is set in the declaration value of the declaration statement. Remember that there are three items needed to configure a style:
  - The Selector (the HTML element)
  - The Declaration Property
  - The Declaration Value
  - Also, when using multiple styles within a single element, a semi-colon is used to separate each property/value.
  - For Example:
    - <body style="color: #FF1100; background-color: #00FF00;>
  - It can get confusing when using inline styles because the use of braces { } is not needed and will improperly render the webpage.
- Before we do a few examples, there are rules for hierarchy.
  - Global styles are generally configured on the CSS document and linked within the <head> tag using the <link> tag.
  - Global styles are also styles that encompass child elements.
  - Embedded or Internal styles will override any external global styles.
  - Inline styles will override their parent inline styles, embedded, and external styles.
  - Inline styles can override other inline styles of the element is inside another element.
    - <body>
    -    <h1>
    - Any inline within the <h1> tag will override the same style within the <body> tag
- Now, Let us look at a few examples.
  - Let us begin by opening our template, myHTML5Template.html.
  - Save As inlineStyleExample.html
  - Make the title Inline Style Example
  - Within the <body> tag, configure the following:
  - <body style="background-color: #F5F5F5; color: #008080;">
  - Next, configure an <h1> style.
  - <h1 style="background-color: #008080; color: #F5F5F5;">Inline CSS Style</h1>
  - Now, we will add a paragraph
  - <p>This paragraph inherited the style used in the body tag.</p>
  - Save the file and navigate to the folder
  - Open the file to see it in the browser.
  - If it does not look correct, locate the inline.html file in the Module3 folder and compare the code.
  - Review the steps, resave, and refresh in the browser or reopen the file.

- You can see that the style in the <h1> tag overrode the <body> style but continued to the paragraph.
    - Let us override the <body> text color by placing a style inside another paragraph.
        - Insert another paragraph and configure the text color.
        - First, Save As inlineStyleP.html.
        - Next, add <p style=”color: #333333”>This paragraph’s text color overrode the text color from the body style.</p>
        - Save the file.
        - Navigate to it and open it.
        - It should look like the one shown.
        - If it does not, locate the inlinep.html file in the Module3 folder and compare the code.
        - Review the steps, make any corrections, save, and refresh the browser or reopen the file.
- The previous examples demonstrated how to use inline styles and the inherited values to child elements.
- Although this can be useful, it is not good practice.
- It can get complicated to maintain, adds unnecessary code to the web document, and is just plain inefficient. However, there will be instances that inline styles can provide a temporary adjustment to a single document.
- Next, we will examine Embedded styles and work a few examples.

EMBEDDED CSS WITH THE STYLE ELEMENT

- Embedded styles are commonly placed in the <head> tag.
- Using the <style> element lets the browser know to use these styles are above external styles.
- The convenience of using embedded or external styles comes from making changes.
- Imagine having to change the same inline style for 10 or 20 paragraphs in the document. This image should be enough to realize the convenience of using embedded or external styles.
- Practice makes perfect, so Let us work a few examples.
    - Open the myHTML5Template.html
    - Save As embeddedCSS1.html
    - Add First Embedded CSS Example for the title.
    - Under the <meta> tag, add the following style.
    - <style>
    - body {background-color: #E6E6FA;
        - color: #191970;
    - }
    - Now, close the </style> element.
    - Inside the <header> element, add
        - <h1>Embedded CSS Example 1</h1>
    - Skip the nav tags and in the <main> tag add,
    - <p>This page uses an embedded style</p>
    - Save the file. Navigate to it and open in a browser.
    - If it does not look like the one shown, locate the embedded.html file in the 3.2 folder and compare the code.
    - Review the steps, make corrections, save, and refresh the browser or reopen the file.
- You can see how using the body selector element applied the styles throughout the webpage.

- Next, we will configure more of a real-world scenario.
- But first, Let us create another template for using embedded styles.
    - Open the file structureExample.html
    - Save As citEmbeddedTemplate.html
    - Now that we have a template with multiple elements, we can begin the next example.
    - First, Save As citEmbeddedStyles1.html
    - In the <head> section below the <meta> tag, add the following.
    - <style>
    - body {
        - background-color: #E6E6FA;
        - color: #191970;
    - }
    - h1 {
        - background-color: #191970;
        - color: #E6E6FA;
    - }
    - h2 {
        - background-color: #AEAED4;
        - color: #191970;
    - }
    - </style>
    - Save the file. Navigate to it and open in a browser.
    - If it does not look like the one shown, locate the index.html file in the 3.2 folder and compare the code.
    - Review the steps, make corrections, save, and refresh the browser or reopen the file.
- Our webpages are starting to take on a professional look and feel.
- Now that we have worked some examples with inline and embedded styles, we will continue forward using the files included with this series.
- Up to now we have explored the Internet, web standards, the client/server model, protocols used with web design, markup languages, HTML elements, and inline and embedded styles.
- Moving forward, because there is a lot to complete, we will be modifying existing files and creating folders on the fly.

CONFIGURING TEXT WITH CSS

THE FONT-FAMILY PROPERTY

- When configuring fonts, we need to be concerned with the typeface of the font.
- All browsers use a default font and may vary depending on the browser type and version.
- Verdana, Tahoma, and Georgia fonts are widely used because they were designed for computer monitors.
- It is common practice to use serif fonts such as Georgia or Times New Roman for headings.
- San-serif fonts such as Verdana and Arial are commonly used with general content.
- Font-Families
    - There are five basic font families.
        - serif
            - Times New Roman, Georgia, Palatino

            - san-serif
                - Arial, Tahoma, Helvetica, Verdana
            - monospace
                - Courier New, Lucida Console
            - cursive
                - Lucida Handwriting, Brush Script, Comic Sans MS
            - fantasy
                - Jokerman, Impact, Papyrus
        - Of the fonts listed, serif and san-serif are the most widely used font types.
    - Let us work an example of font-families
        - From within your text editor, open the file starter2.html from the Module3 folder.
        - Save as index.html in a folder named kayak3
        - Locate the <style> tag in the <head> section and add an embedded style.
        - body {font-family Verdana, Arial, sans-serif}
        - Add styles for h2 and h3
        - h2, h3 {font-family: Georgia, “Times New Roman”, serif;}
        - Save the file.
        - Navigate to it and open it.
        - It should look like the one shown.
        - If it does not, locate the index.html file in the 3.3 folder and compare the code.
        - Review the steps, make any corrections, save, and refresh the browser or reopen the file.

MORE CSS TEXT PROPERTIES

- There are numerous properties you can include to manipulate the text
    - font-size – sets the size of the font
    - font-weight – sets the boldness of the font
    - font-style – italics is the most used value
    - line-height – changes the line of text to a default height
    - text-align – values are left (default), center, and right
    - text-decoration – values include underline, line-through (strike-through), none, overline
    - text-indent – indents the first line of text within the specified element
    - letter-spacing – places space between characters
    - word-spacing – places spaces between words
    - text-transform – values are none (default), capitalize, uppercase, lowercase
    - white-space – sets the rules for whitespace (space characters and line feeds)
    - text-shadow – CSS3 only, adds depth and dimensions to text
        - horizontal offset – pixel value - negative values to the left, positive values to the right
        - vertical offset – pixel value – negative value go above, positive values go below
        - blur radius (optional) – pixel value – the higher the value the more blur
        - color value – use a valid color for the shadow
- Of the properties listed, 1 through 5 will be the most used.
- The values for the text properties are:
    - Text Value – xx-small, x-small, small, medium (default), large, x-large, xx-large.
    - Pixel Unit (px) – Numeric values such as 10px.

    - Em (em) – Numeric values such as .80em. This is recommended by W3C because it scales well when resized in a browser.
    - Percentage Value – Numeric value such as 75%.
  - Let us work some examples using some of these text properties.
    - Open the file index.html from the Module3, 3.2 folder and Save As embedded1.html in the Module3 folder.
    - Locate the index.html file in the 3.4 folder. Right-click and open with Notepad. This is the reference file.
    - Add font-family: Arial, Verdana, sans-serif; to the body selector.
    - Save the file.
    - Navigate to it and open it.
    - It should look like the 3.4 index.html file when opened in a browser.
    - If it does not, review the steps, make any corrections, save, and refresh the browser or reopen the file.
    - Before we continue, Let us rename the file, Save As index.html and place in a folder named trillium3.
    - Continue to use the reference file for the remaining of this exercise.
    - Next, configure the h1 selector
    - Next. the h2 selector
    - Next, configure the Nav area
    - Next, the paragraph
    - Next, unordered list
    - Save the file.
    - Navigate to it and open it.
    - It should look like the index.html file in the 3.4 folder when opened in a browser..
    - If it does not, review the steps, make any corrections, save, and refresh the browser or reopen the file.
  - Does not this web page look great. It has a good color scheme with the offsetting purple tones.
  - What a difference it made adding just a few additional lines of code.
  - We added formatting the body, h1, h2, nav, p, and ul elements.
  - Next, we will examine CSS Class, ID, and Descendant Selectors.

CSS CLASS, ID, AND DESCENDANT SELECTORS

THE CLASS SELECTOR

  - A class selector is used when you need to format an element, but it does not need to affect all the same elements.
  - You will use classes when the format is applied to two or more elements.
  - Class selectors begin with a period (.)
    - .feature {color: #11AA33;}
  - This class will be added anywhere the class name of ‘feature’ is used.
    - <li class=”feature”>ex 1</li>
    - <li class=”feature”>ex 2</li>

THE ID SELECTOR

  - An id selector is used when you need to format a single element of the webpage.
  - Id selectors begin with a hashtag #.
    - #feature {color: #333333;}

- Again, the id will be added to the single element within the document as an attribute of the element.
    - <div id=”feature”> </div>
    - Attempting to use the id in multiple elements within a single webpage will cause errors.
    - FYI – You can use the id selector in multiple webpages so long as it is only used once in each webpage.

THE DESCENDANT SELECTOR

- The descendant selector is used to format the same child elements with a parent element.
- It starts with the parent element a space and the child element.
    - main p {color: #222222;}
- We covered how to use class, id, and descendant selectors. This gives developers more flexibility while reducing the amount of additional coding to format the webpage.
- Next, we will have a look at the Span Element.

SPAN ELEMENT

- Similar to the use of the <div> element.
    - Also known as a block display
    - Used to separate sections of a webpage
- Used to define an area that is not separated from other areas of the webpage.
- Also known as an inline display.
- Let us do an example
    - From your text editor, open the file index.html located in the 3.5 folder within the Module3 folder.
    - Configure a class for the company name
    - Add the class to a span element within the <p> element that starts with Trillium Media…
    - Save the file.
    - Navigate to it and open it.
    - It should look like the index.html file in the 3.6 folder.
    - If it does not, compare the code, review the steps, make any corrections, save, and refresh the browser or reopen the file.
    - Now we are making good progress.
- However, it is highly unlikely that a website with have a single webpage.
- We used a <span> element to alter the format of a small group of text, the company name within the paragraph tag.
- To carry a style to multiple webpages, we will use an external style sheet in the next video.
- We also learned that it is unlikely that a website will have a single webpage.
- To minimize the amount of style reconfigurations, we are going to explore the use of External Style Sheets using the <link> element within the <head> tag.

USING EXTERNAL STYLE SHEETS

- External style sheets offer developers great flexibility
- It increases the efficiency of changes as one change can affect multiple elements throughout the website.
- External styles are configured in a single document and use the .css extension.
- We will use the <link> element with the rel (relationship) attribute as “stylesheet” and an href attribute to call the external style sheet.

  - Let us create an external style sheet using our text editor.
  - If you are using Notepad++, open a new file and Save As color.css in the Module3 folder.
  - Locate the external.html file in the 3.7 folder. Right-click and open with Notepad. This is the reference file.
  - Next, Let us make some changes to a webpage and link the color.css to this webpage.
  - Open the file template.html located in the Module2.
  - Save As external.html in the Module3 folder.
  - Using the reference file, make the following changes:
    - Change the title to External Styles
    - Add a <link> tag in the <head> section below the <meta> tag.
    - Add a paragraph inside the body tag.
    - Save the file.
  - Navigate to it and open it.
  - It should look like the one shown.
  - If it does not, review the steps, make any corrections, save, and refresh the browser or reopen the file.
- Now that you have created an external style sheet and linked it to webpage, Let us do an example with a lot more detail.
  - Locate the index.html file in the 3.8 folder. Right-click and open with Notepad. This is the reference file.
  - With your text editor, open the index.html file located in the 3.6 folder within the Module3 folder.
  - Save As index.html but place the file in a new folder called trilliumext.
  - Now we will remove the embedded style and place it in a new external sheet.
  - In the text editor, start a new file and Save As trillium.css in the trilliumext folder.
  - Click back to the index.html file
  - Copy all the content between the style tags and paste it in the new .css file and save.
  - Go back to the index file and delete everything between the style tags including the opening and closing tags and save.
  - Your .css file should look like the trillium.css file in the 3.8 folder.
  - Next, insert a <link> tag referencing the external trillium.css style sheet below the <meta> tag and Save.
  - Next, Let us link the trillium.css to another page.
  - Open the services.html located in Module3.
  - Save As and navigate to the trilliumext folder (should be located in the 3.6 folder) and click Save. This will place the file in the same directory as the style sheet.
  - Add a link in the services.html page to the trillium.css.
  - TIP – copy the link from the index.html and paste it in the same location in the services.html page.
  - Save the services.html page with the new <link>.
  - Navigate to it and open it.
  - It should look like the services.html file in the 3.8 folder when opened in a browser.
  - If it does not, review the steps, make any corrections, save, and refresh the browser or reopen the file.
  - You should still have the index.html page loaded in your browser. Click on the Services link to see the magic of external css.

    - It should look like the one shown when navigating between the Home link and the Services link.
- Although this tutorial was somewhat daunting and time-consuming, it demonstrates the efficiency of using an external style sheet.
- It also demonstrated how to convert an embedded style to an external style.
- In the next video, we will look at the center HTML element using the external sheet.
- Next, we will look at how to center elements of a webpage using an external style sheet.
- We will be introducing new selectors in the stylesheet.

CENTER HTML ELEMENTS WITH CSS

- Centering content is rather simple when using margin declaration properties and values.
- Remember that a margin is the empty area that surrounds an element.
- With regards to the <body>, the empty space is between the content and the browser edges.

MARGINS

- The margin properties are:
    - margin-left – the space to the left of the element.
    - margin-right – the space to the right of the element.
    - These margins can be code with:
        - “0”
        - pixels
        - em
        - percentages
        - auto
    - When both margins are set to auto, the available space is divided evenly thus centering the content in the browser.

CONTAINERS

- Containers are required to ensure only the elements needing centered are centered.
- Use <div> tags in the webpage with class or id selectors in the CSS.
- To avoid confusion when using class or id selectors, give the selectors descriptive names and use these names as class= or id=
    - bodyWrapper
    - navWrapper
    - mainWrapper
    - And so on.
- Let us work an example.
    - We will use the files located in the 3.8 folder within Module3
    - Create a new folder called trilliumcenter in the Module3 folder.
    - Copy the index.html, services.html, and the trillium.css files to the trilliumcenter folder from the 3.8 folder.
    - We will work within the trilliumcenter folder.
    - First, open the trillium.css file in your text editor.
    - Add an id selector named wrapper at the bottom of the stylesheet.
        - remember that id selectors begin with the hashtag #.
    - Save the stylesheet.
    - Open the index.html file within the trilliumext folder.

    - Insert the HTML <div> element to encompass the entire content of the webpage below the opening <body> tag using the id="wrapper"
    - Next, we need to close this </div> tag and save.
    - Navigate to it and open it.
    - It should look like the index.html file in the 3.9 folder when opened in a browser.
    - If it does not, view the code for the index.html in the 3.9 folder, review the steps, make any corrections, save, and refresh the browser or reopen the file.
  - Well done. Remember that practice makes perfect.
  - Creating stylesheets takes focus and attention to detail.
  - It also requires a firm understanding of the layout/format needed to present the content in a readable, sensible, and logic manner.
  - Hand coding the css and the changes to the webpages would take a long time. That is another reason I mentioned earlier about creating templates.
  - You can create .css templates also.
  - Next, we will look at the logical flow of styles and how the hierarchal nature of child styles can override their parent styles.
  - Before we move forward, one thing I want to mention is documenting within the webpage.
  - It is common and expected practice to add notations within the document, so you remember how, when, and why. It also helps other developers working in these webpages to have a better understanding of how, when, and why.
  - Next, we will examine the Cascading effect with regards to the hierarchal nature of overriding styles.

THE "CASCADE"

  - The Cascade is the order of precedence.
    - Browser defaults
    - External stylesheets
    - Embedded styles
    - Inline styles
  - This order is what allows child styles to override parent styles.
    - Inline can override embedded
    - Embedded can override external stylesheets
    - External stylesheets can override the browser's default styles.
  - It is important to fully understand the order of precedence when coding styles, otherwise your webpage will not have the desired appearance or formatting.
  - Let us have a look at the precedence of styles.
      - Follow along with the short video example.
      - Below is an exercise for the student to do without coaching.
    - ***The following is an exercise the student does without tutorial.***
    - Create a folder named mycascade in the Module3 folder.
    - Locate the mypage1.html and the site.css files in the 3.10 folder. Right-click both and open with Notepad. These are the reference files.
    - Open a new file in your editor and Save As site.css in the mycascade folder.
    - ***ADD/EDIT*** the code in the site.css file with the .css reference file.
    - Save and close.
    - Next, open a new document and Save As mypage1.html in the mycascade folder.
    - ***ADD/EDIT*** the code in the .html reference file to the mypage1.html file.
    - Navigate to it and open it.

    - It should look like the mypage1.html file in the 3.10 folder when opened in a browser.
    - If it does not, compare the code with the reference files, review the steps, make any corrections, save, and refresh the browser or reopen the file.
  - We are almost done with this module, and what a module it has been.
  - The last item on the agenda is validating the CSS.
  - This is much like webpage validation in that it looks for errors within CSS and shows any errors with possible recommendations.

CSS VALIDATION

  - Let us validate the color.css external sheet we completed in one of the previous sections.
  - Locate and open the color.css file. It should be found in the 3.7 folder.
    - Save As cssValidate.css in the Module3 folder.
    - Let us create an error by removing the dash between background and color.
    - Next, remove the # from the color: property.
    - Save the file.
    - Open a browser window and navigate to https://jigsaw.w3.org/css-validator/#validate_by_upload and follow the instructions.
    - Your results should look like mine. Now correct the errors we created, save, and re-validate.
  - What a time-consuming but incredibly useful module.
  - You received a lot of good information and tutorials to reinforce the importance and convenience of styles; inline, embedded (internal), and external stylesheets.
  - In the next Module, we will dive into Visual Elements and Graphics and see what CSS3 and HTML5 can do for you.

MODULE 4 - VISUAL ELEMENTS & GRAPHICS

- You will want to design the visuals in such a manner that it keeps your audience engages, but also allow others to enjoy your site without the use images. Remember Accessibility? Not all users will

CONFIGURING LINES AND BORDERS

- The use of visual elements helps to create separate areas of your webpage.
- The first element we will look at is the Horizontal Rule Element.
- It is a straight line across the page and is a self-contained tag. <hr>
- We can adjust the properties of this element with the use of styles.
  - Let us give it a try.
  - From within your text editor, open the file starter1.html located in the Module4 folder.
  - Save As hr.html – It should save the same location, which is fine.
  - Insert the <hr> tag above the <footer> tag.
  - Save the file.
  - Navigate to it and open it.
  - It should look like the one shown.
  - If it does not, review the steps, make any corrections, save, and refresh the browser or reopen the file.

THE BORDER AND PADDING PROPERTIES

- Border and padding can get quite frustrating. Pay attention to the dimensions you will add as they will distort the rendering.

BORDER PROPERTIES

- This configures the border of an element.
- It is 0 by default, which does not display a border.
- It can be customized for the top, right, bottom, and left areas by using:
  - border-top
  - border-right
  - border-bottom
  - border-left
- The above list is also the direction of configuration.
- It also be configured using a shorthanded method {border: 3px dashed #000033}

BORDER STYLES

- The border-style is required and can be formatting using the following options:
  - none (default)
  - inset
  - outset
  - double
  - groove
  - ridge
  - solid
  - dashed
  - dotted

BORDER WIDTH

- The border-width is used to control the width or thickness of the border

BORDER COLOR

- The border-color is used to color the border.
- TIP – There are a multitude of new border properties. These can be found at the link below:
- https://www.w3schools.com/cssref/pr_border.asp

PADDING PROPERTY

- This configures the space between the content and the border.
- Imagine standing in a room. You are the content, and the walls are the border. The padding is the empty space between you and the walls.
- Because you can move around in the room and change the space between you and the walls, padding too can be adjusted.
- TIP
    - Paddings and Borders are numerical values that change the dimensions, thus changing distances.
    - If the content and padding are too great for the space allocated, other visual elements will be directly affected.
    - Borders, Padding, and Margins (later in this series) require careful planning and simple math.
    - Strictly avoid the trial-and-error approach to figuring these properties. It is very time consuming and frustrating.
- Let us do an example working with borders and padding.
    - Within your text editor, open the file starter1.html within the Module4 folder.
    - Save As border.html.
    - Insert an h1 selector in the style and add {padding: 1em;}
        - TIP – When adding selectors try to keep them grouped together. In this example place the h1 selector above the h2 selector. This makes locating and change values or styles let time-consuming.
    - Within the h2 selector of the style add the following:
        - border-bottom: 2px dashed #191970;
    - Add styles to the footer selector:
        - border-top: thin solid #AEAED4;
        - padding-top: 10px;
        - color: #333333;
    - Save the file.
    - Navigate to it and open it.
    - Try it in a different browser. Edge and Chrome may have slightly different appearances.
    - It should look like the one shown.
    - If it does not, review the steps, make any corrections, save, and refresh the browser or reopen the file.
- Working with borders and padding can adversely change the visual dimensions of the webpage elements.
- It takes careful and detailed planning to avoid wasting time troubleshooting the page layout. Use a search engine to locate tips for efficient practices for configuring borders and padding.
- Next, we will work to make our webpages compelling with the proper use of graphics.

TYPES OF GRAPHICS

- There are numerous graphical formats. However, there are five that provide the best results:
  - .gif
  - .jpg or .jpeg
  - .png
  - .tiff
  - raw image files
- Of the five listed, gif, jpg/jpeg, and png are the most widely used image formats.
- There are attributes of each format that needs to be considered when choosing image types.

GRAPHIC INTERCHANGE FORMAT (GIF) IMAGES

- Best used for line drawings containing solid tones and simple images like clip-art.
- Attributes
  - Transparency – GIF takes and removes the background color to allow objects behind the image to show through. Image editing software can adjust the amount of transparency.
  - Animation – GIF animations consist of multiple images, slightly different, laced together, and are displayed in order. This give the image it is animated effect.
  - Compression – GIFs are considered lossless when saved because the image is not compressed. The image retains all of its original properties.
  - Optimization – This attribute allows the image to be optimized for the web by reducing its size. Reducing the size does affect the quality so it is necessary to balance the size for quicker downloading and the quality of the image.
  - Interlacing – Because browsers read the webpage from top to bottom, images are loaded in the order they are placed in the document. Interlacing allows for the image to be loaded progressively which means it becomes more visible as its elements are captured.

JOINT PHOTOGRAPHIC EXPERT GROUP (JPEG) IMAGES

- JPEG images are best used in photography because they can contain 16 million colors whereas GIF images are limited to 256 colors. These images have the extension of .jpeg or .jpg.
  - Transparency – None.
  - Animation – None.
  - Compression – Considered a lossy compression when saved due to the loss of pixels.
  - Optimization – There needs to be a balance of size versus quality when working with this image type. Larger sized images are crisper in their appearance whereas reducing the size will reduce the quality. Pixelation occurs when too much of the image has been removed and it appears somewhat fuzzy or out of focus. There are applications available that aid in optimizing the image.
    - Adobe Photoshop
    - Adobe Lightroom
    - GIMP
  - Lacing – None.

    - Progressive JPEG – This is similar to lacing with a GIF in that it progressively displays as the image is downloaded.
    - TIP – Consider using thumbnail quality images with JPEG/JPG and hyperlink them to the original image; either in another browser tab or another browser window.

PORTABLE NETWORK GRAPHIC (PNG) IMAGES

- PNG images capture the best of GIFs and JPEG/JPGs. This image type supports millions of colors, is lossless compression, and does support transparency levels. It also supports interlacing.

WEBP IMAGE FORMAT

- This is a new image format designed by Google. It is not yet ready for all browsers but is support in Chrome and Opera. It has an improved lossy compression which means more of the quality remains when saved.
- Who would have thought that there is so much to consider when choosing image types for webpages?
- This is just the tip of the iceberg.
- Take some time to search the Internet for tips from seasoned developers on ways to optimize images for your webpages.
- In the next section, we will look at Image Elements.

IMAGE ELEMENT

- The image element <img> configures the graphics of your webpage. Some of the more popular graphics used are:
    - Photographs
    - Banners
    - Company Logos
    - Navigation Button
    - Much, Much more
- Below is a snippet of what an image element can look like when configured:
- <img src=”logo.gif” height=”200” width=”500” alt=”My Company Logo”>
- src Attribute
    - Specifies the name and the location of the image.
    - It can be local to the website, resides in a directory in the web server.
    - It can be located at a remote location, like an image hosting website.
- Additional Attributes
    - alt – This attribute is highly recommended by the W3C for text-only browsers. It will show a tooltip when hovered over and will place the name of the image in a placeholder until it is fully downloaded. Recommended to not be over 100 characters long.
    - height/width – This is used to control the physical size of the image.
        - By default, the values for this attribute are in pixels (px) and as such do not need to be notated; however, it is good practice to include it…height=”50px”
    - title – This attribute is more descriptive than alt in that a balloon will appear stating the text in the title when the cursor hovers over the image.

        - id – text name, alphanumeric, begins with a letter, no spaces, and the value must be unique within the webpage. Remember that ids are singular, and classes are multiple.
        - longdesc – URL of a resource that contains an accessible, more descriptive detail of a complex image.
        - Obsolete Attributes
            - Although the following attributes are obsolete for in-element use, they can be used as a style:
                - align – use the float attribute within a style.
                - border
                - hspace/vspace
                - name – use with the id attribute
    - In this exercise we are going place a logo graphic and a photograph on a webpage.
    - Let us do an example.
        - First, locate the index.html file in the 4.3 folder. Right-click and open with Notepad. This is the reference file.
        - In the Module4 folder, create a folder called kayakch4.
        - Copy/paste hero.jpg and kayakdc.gif of the starters folder to the kayakch4 folder
        - Copy/Paste the starter2.html into the kayakch4 folder.
        - While we are in the kayakch4 folder, Let us rename the starter2.html file to index.html.
        - First, open the index.html file for editing.
            - TIP – While you are in a folder and you want to edit a file, rather than switching to your text editor, you may be able to right-click the file and select edit. For me, Notepad++ placed a link in the menu to “Edit with Notepad++”
        - With the index.html file ready for editing, delete the text within the <h1> tags.
        - Next, we will code an <img> element for the kayakdc.gif, including the src, alt, and height/width attributes. Use the code in the text document next to my editor under the Hands on 4.3 area.
        - Save the file.
        - Navigate to it and open it.
        - Compare it to the index.html file in the 4.3 folder.
        - If it does not, review the steps, make any corrections, save, and refresh the browser or reopen the file.
    - We have absorbed a great deal of information thus far.
    - Now, we will take our image exercises a step further and see how images can be used as hyperlinks.

IMAGE HYPERLINKS

- If you know how to create hyperlinks using a word or phrase, then coding an image <img> with a hyperlink will be quite easy.
- Simply insert the <img> element and its attributes inside an anchor tag <a></a> like seen below.
- <a href=”index.html”><img src=”home.gif” height=”19” width=”85” alt=”Home”></a>
- Thumbnail images are a good to use as hyperlinks.
- The value points to another image rather than another webpage as seen below
- <a href=”sunset.jpg”><img src=”thumb.jpg” height=”100” width=”100” alt=”click to view larger image”></a>
- **Let us have a look at this. Go to the thumb.html file located in the Module4 folder and open it up in your browser.

- Let us work with a more challenging exercise.
- In this exercise we will code images as hyperlinks and add accessibility for text-based browsers and browsers with images disabled.
    - We will add more images to the kayakch4 folder and work from within this folder.
    - First, copy the images from the starters folder from within the Module4 folder to the kayakch4 folder.
        - home.gif
        - tours.gif
        - reservations.gif
        - contact.gif
    - Next, from your text editor, open index.html.
    - You will notice the <a> tags are already coded. We are going to replace the text and use images as hyperlinks.
    - To add accessibility, let copy the <nav> element to the footer section right above the copyright notice. Do not do this later because we are going to replace the hyperlink text with images.
    - Next, configure a green background color for an id called bar. Refer to my text document next to my editor.
    - To prevent a browser from placing a hyperlink border around the image, add a selector img and style the border for none.
    - Let us move our attention to the <nav> area of the document.
    - Insert the id name bar inside the opening nav element.
    - Next, replace the text between each anchor tag with an image.
        - home.gif for index.html
        - tours.gif for tours.html
        - reservations.gif for reservations.html
        - contact.gif for contact.html
    - Ensure you do not leave any space between the <a> tag and the <img> tag. Follow the example in the text document.
    - Be mindful that the width of each .gif is different and can be referenced in the text document.
    - Save the file.
    - Navigate to it and open it.
    - It should look like the webpage shown.
    - If it does not, open the index.html solution file in the 4.4 folder.
    - Compare/review the steps, make any corrections to your index file, save, and refresh the browser or reopen the file.
- That was a fantastic challenge.
- Working with images is challenging but extremely flexible with styles.
- In the next section, we will examine HTML5 visual elements and perform several exercises.

HTML5 VISUAL ELEMENTS

- Configure an image with a caption using <div> as a container.
- Then, configure the image with a caption using HTML5 elements “figure” and “figcaption”
- Let us move to the desktop.
    - First, create a folder named mycaption in the Module4 folder.
    - Next, copy the myisland.jpg from the starters folder to the mycaption folder.
    - Launch your editor and open the file template.html from the Module2 folder. Do not worry, this is the correct folder.

- Next, Save As index.html in the mycaption folder.
- Change the title to Tropical Island.
- Save the file and open it in a browser. If you are using Notepad++, click Run to drop the menu and Launch your default/favorite browser.
- Next, Let us configure a caption for the image using the <div> and include a border for the image.
- With the index file still open, embed a style in the <head> section called figure to correspond to an id called figure. Remember that id selectors always begin with a hashtag #.
- Add the <div> tags to the body and with the id of figure.
- Next, insert the <img> element and include the values in the text doc.
- Add the text Tropical Island Getaway above the closing </div>.
- Save the file.
- Navigate to it and open it.
- It should look like the one shown.
- If it does not, review the steps, make any corrections, save, and refresh the browser or reopen the file.

- Wow, now that was interesting.
- Next, we are going to examine the figure and figcaption elements.
  - First, Let us create a new folder called mycaption2 in the Module4 folder
  - Next, navigate to the starters folder in Module4 and copy the myisland.jpg image to the mycaption2 folder.
  - Again, we will use the template file located in the Module2 folder and Save As index.html in the mycaption folder.
  - Modify the title element with Tropical Island.
  - Add an <img> element to the body to display the myisland.jpg image.
  - Save and launch the index.html.
  - It should show an image with no border or captions.
  - Next, add an embedded style using the code in the text document.
  - Next, add the figure element with the figcaption element in the <body> section.
  - Save the file.
  - Navigate to it and open it.
  - It should look like the one shown.
  - If it does not, review the steps, make any corrections, save, and refresh the browser or reopen the file.
- The new HTML5 figure and figcaption element make labeling images much easier than using the <div> element.
- As you can see the exercises are becoming more in-depth as we move forward.
- Next, we will talk about the Meter Element.
- Let us examine the meter and progress elements.

METER ELEMENT

- This element displays a visual gauge.
- It uses known values within a range.
- The attributes for this element are:
  - value
  - max
  - min
- We can view an example by navigating to the Module4 folder and opening the meter.html file. The meter element is a great way to create a visual representation of numeric values.

THE PROGRESS ELEMENT

- Similar to a progress bar for uploading and downloading files, this too represents a percentage based on the values configured.
- The attributes are:
    - value – this is the numeric value that we want to represent.
    - max – this the numeric value that uses the value to display a percentage based on the max value.
- We can view the progress element by navigating to the Module4 folder and opening the progress.html file.
- These elements are not automatic but could be if they were associated with a database and a server-side scripting language.
- That was an adventure**.**
- Are you beginning to realize that styles are a much better way to format a webpage rather than the old ways?
- Next, we will have a look at background images.

BACKGROUND IMAGES

- We have configured the background-color of a webpage using a style for the <body> tag.
- Thanks to styles, we can more easily use images as backgrounds.
- Background images can add a great deal of pleasing aesthetics as long as they are not overbearing or remove focus from the web content.
- Let us look at the properties we can use for background images.
- The background-image element
    - The background-image element can be used along with the background-color element.
    - The page will load the color before the image
    - If the image fails to load, the color will remain as the background-color.
- The background-repeat Property
    - Remove tiling that occurs when images are smaller than the browser window.
    - Can repeat rather than tile.
    - Values are
        - repeat – default
        - repeat-y – vertical
        - repeat-x – horizontal
        - no-repeat
- Let us do an exercise with background images.
    - We will use the files from our earlier kayakch4 folder.
    - Copy the image heroback.jpg from the starters folder to the kayakch4 folder
    - Launch your text editor and open the index.html file from the kayakch4 folder.
    - Locate the style tags in the <head> section.
    - Add the code see in the text document next to my editor.
    - Next, remove the <img> tag from the body of the webpage.
    - Save the file.
    - Navigate to it and open it.
    - It should look like the one shown if using any browser other than Internet Explorer.
    - You can nudge IE by using the display: block; declaration with a main selector in the <style>.
    - An example for the IE fix is in the 4.7 folder, iefix.html.
    - The background-position Property

    - Specifies other locations for the background-image to reside.
    - Value are:
      - Percentages;
      - Pixels;
      - left, top, center, bottom, right
    - Requires two values
    - If only one value is provided the second value defaults to center.
  - The background-attachment Property
    - Determines if the background-image remains stationary or scrolls with the page.
    - Valid values are:
      - fixed;
      - scroll;
- The background-image properties and the values are quite useful in HTML5
- They provide flexibility and better control of background images.
- Next, we will examine more elements for images.

MORE ABOUT IMAGES

- Images are powerful tools to enhance the appearance and validity of your website.
- Let us explore additional methods used with images.
- Image Maps
  - Image maps allow an image or parts of an image to be used as a hyperlink.
  - Selectable areas of an image are called hotspots and are configured in one of three shapes:
    - rectangles
    - circles
    - polygons
  - Requires the use of
    - image element
    - map element
    - one or more area elements.
  - Map Element <map>
    - This is the container tag for the beginning and ending or the map description
    - The id and name attributes must be the same
    - usemap attribute associates the image with a map element.
  - Area Element <area>
    - Defines the coordinates or edges of the clickable area.
    - It is a self-contained tag
    - Attributes are:
      - href
      - alt
      - title
      - shape
      - coords – define clickable areas
    - Shape coordinates
      - rect – x1, y1, x2, y2
      - circle – x,y,r
      - poly – x1, y1, x2, y2, x3, y3, etc.
    - Coordinates for the rect attribute
      - upper-left corner, x1, y1

            - lower-right corner, x2, y2
    - We can see an example if you examine the code in the map.html file in the Module4 folder.
- TIP – use a descriptive alt attribute for each area element in play for accessibility.
- Rarely do developers hand-code image maps.
    - Free online map generators:
        - www.maschek.hu/imagemap/imgmap
        - http://image-map.com
        - http://mobilefish.com/services/image_map/image_map.php
    - The Favorite Icon – favicon
        - An image seen in the address bar, browser tab, favorites, and bookmark lists.
        - 16x16 pixels or 32x32 pixels
    - Online tools to create a favicon
        - http://favicon.cc
        - http://www.favicongenerator.com
        - http://www.freefavicon.com
        - http://www.xiconeditor.com
- Image Slicing
    - A method of creating complex webpage images.
    - Most graphic applications can slice
    - Reasons
        - Mouse overs
        - Script activations

CSS SPRITES

- Sprite is an image file made with smaller graphics
- CSS properties
    - background-image
    - background-repeat
    - background-position
- Saves download time due to a single http request.
- Images are quite the useful tool other than for aesthetic purposes.
- We covered a lot of material thus far regarding images.
- Let us explore Sources and Guidelines for Graphics

SOURCES AND GUIDELINES FOR GRAPHICS

- The sources and guidelines noted here will save you time by learning from seasoned developers.
    - Obtaining Graphics
        - There are numerous ways to attain graphics for your website:
            - Self-creation
                - Photography
                - Designing
                - Scanning
                - Etc.
- Free or low-cost image sources for stock images
- Free Sources
    - http://www.freeimages.com
    - http://www.everystockphoto.com

    - http://www.freedigitalphotos.com
- Free or low cost image sources
    - http://pixaba.com
    - http://www.istockphoto.com
- Guidelines for Using Images
    - Reuse images when you can.
    - Consider size vs. quality
    - Consider the time to load the image
    - Use appropriate resolution
    - Specify the image dimensions.

CSS3 VISUAL EFFECTS

- CSS3 introduces quite a number of visual effects.
    - rounded corners
    - box and text shadows
    - transparent color with RGBA and HSLA
    - gradients.

THE CSS3 BACKGROUND-CLIP PROPERTY

- Confines the background image display
    - Attributes
        - content-box
        - padding-box
        - border-box
    - We can see an example of the background-clip in the clip folder in Module4. Open the file in your editor to see the code. Notice that the border is a dashed border.
    - ** comment openly regarding the code in the file.

THE CSS3 BACKGROUND-ORIGIN PROPERTY

- This object positions the image using:
    - content-box – positions relative to the content area
    - padding-box – default, positions relative to the padding area
    - border-box – positions relative to the border area
- This property is supported by current browser versions.
- *** The caveat of the background-origin is that it will have no effect if the background-attachment has a value of “fixed”.

THE CSS3 BACKGROUND-SIZE PROPERTY

- Used to resize or scale the background image.
- Valid values are:
    - width, height in percentages
    - width, height in pixels
    - cover - preserves the aspect ratio when scaling to the smallest size
    - contain – preserves the aspect ratio when scaling to the largest size.

- If we move our attention to the sedona.html file we will see that the image is scaled to 100% to fill the space within the browser.
- Multiple Background Images
- Multiple images are supported
- Let us have a look.
    - Create a new folder called trilliumch4
    - Copy all the files from the trillstarters folder to the trilliumch4 folder
    - Update the index.html file
    - Configure a style to display the trilliumfoot.gif and the trilliumgradient.png and a background-color of #F4FFE4 and a color of #333333
    - Configure the trilliumfoot.gif to not repeat and show in the bottom-right corner.
    - Save the file.
    - Navigate to it and open it.
    - It should look like the one in the 4.9 folder.
    - If it does not, review the steps, make any corrections, save, and refresh the browser or reopen the file.

CSS3 ROUNDED CORNERS

- Softens up corners with the border-radius Property
- Valid values
    - Can have 1 to 4 numeric values to set the radius
        - pixels or em
        - percentage
    - One value sets all four corners
    - Four values in order of:
        - top left
        - top right
        - bottom right
        - bottom left
    - Corners can be configured individually
        - border-bottom-left-radius
        - border-bottom-right-radius
        - border-top-left-radius
        - border-top-right-radius
    - See box.html in Module4
    - Let us try it.
    - Create a new folder called borderch4
    - Copy lighthouselogo.jpg and background.jpg from the starters folder to the borderch4 folder
    - Copy the starter3.html file to the borderch4 folder.
    - View the file in your browser before making changes.
    - Launch your editor and open the starter3.html from your borderch4 folder.
    - Save As index.html
    - Set an embedded style for the h1 selector. Refer the index.html file in 4.10 for the code.
    - Save the file and open in a browser.
    - If yours looks different, review the steps and the code, make corrections if needed, save, and refresh the browser or reopen the file.

CSS BOX-SHADOW PROPERTY

- Used for a shadow effect on block-displays like <div> and <p> elements
- It uses the following attributes
    - horizontal offset
        - positive values shadow to the right
        - negative values shadow to the left
    - vertical offset
        - positive values shadow below
        - negative values shadow above
    - blur radius (optional) is pixel only
        - defaults to 0
        - higher values, more blur
    - spread distance (optional) is pixel only
        - defaults to 0
        - positive values expand
        - negative values contract
    - It does require a valid color value.
- Let us put this to practice
- Let us apply the box-shadow and text-shadow to a centered content area.
    - Create a new folder called shadowch4
    - Copy the lighthouselogo.jpg and the background,jpg files from the starters folder to the shadowch4 folder
    - Open the index.html file in 4.10 from your editor and save it to the shadowch4 folder.
    - Configure a <div> element with the id=”container” inside the <body> tags. Close the </div> tag before the </body> tag.
    - Edit the embedded CSS with the id selector named “container.”
        - White background-color with padding at 1.25em
        - Use width, min-width, margin-left and margin-right
        - Add a style declaration to the h1 element selector for a dark-gray text shadow
        -
        - Refer to the index file in 4.11 to see the code.
    - Edit the CSS to add a style declaration with the container id
    - Add a style declaration for the h2 element selector with a light-gray text shadow with no blur.
    - Save the file and open it.
    - It should look like the sample file from 4.10 when opened. If not, review the steps and make corrections. Save and refresh or reopen.
- Shadows around images and text will gives a site a warmer feel. However, with most objects in a development environment it can get out of control. Only add the aesthetics to an element if it would emphasis the element.
- Next, Let us explore CSS3’s opacity property.

THE CSS3 OPACITY PROPERTY

- Used to configure transparency
- Value ranges
    - 0 is completely transparent
    - 1 is complete opaque
        - (range is 0 to 1 only. 0.2, .80, etc.)

        -

- The opacity controls the background and the text equally
- See index.html in the 4.13 folder for an example.
- Let us move to the desktop and work an example.
    - Create a new folder called opacitych4
    - Copy fall.jpg from the starters folder to the opacitych4 folder.
    - Launch your editor and open the template.html file from the Module2 folder.
    - Save As index.html in the opacitych4 folder
    - Change the page title to Fall Nature Hikes
    - Next, create a <div> container holding the <h1> element. Refer to 4.13 index.html for the code.
    - Add style tags to the head section.
    - Insert an id selector named “content” that will show the fall.jpg as the background image and that does not repeat, has a width of 640 pixels, left and right margins set to auto, and a top padding of 20 pixels. Refer the index.html file from 4.13.
    - Next, configure the h1 element selector to have a white background-color, opacity set to 0.6, font size to 4em, and 10 pixels of padding. Refer the index.html file from 4.13.
    - Save the file and open it in the browser. It should look like 4.13 index.html. Make corrections if necessary, save, and refresh or reopen.

CSS3 RGBA COLOR

- New syntax for the RGBA transparency color.
- Does not use hexadecimal
- Decimal values are 0 to 255 for each RGB.
- Uses the format rgb(xxx,xxx,xxx)
- Let us work an example for the RGBA Color.
    - Create a folder called rgbach4.
    - Copy fall.jpg from the Module4 starters folder to the rgbach4 folder.
    - From your editor, open the index.html found in the 4.13 folder within Module4.
    - Save As rgba.html to your rgbach4 folder.
    - Delete the h1 declarations for the element selector from the embedded styles.
    - Refer to the code in rgba.html from the 4.14 folder.
    - Insert the h1 selector in the embedded style sheet.
    - Add the declarations
        - 10 pixels of right padding
        - use a sans-serif font that is right aligned and white.
            - (hint – Verdana, Helvetica)
        - Make the text 70% opaque with a font size of 5em.
        - Add two color values
            - modern browsers – rgba(255, 255, 255, 0.7)
            - older browsers - #FFFFFF
    - Save the file and open.
    - It should look like the index.html file found in 4.14 from within the Module4 folder.
    - If not, review the code, make corrections as necessary, save and refresh or reopen.
- Next, Let us have a look at the next CSS3 color property, HSLA.

CSS HSLA COLOR

- Colors are normally configured with hexadecimal or rgb decimal values. (#333333, rgb(xxx,xxx,xxx)
- CSS3 introduced the HSLA color
  - Based on a color wheel, aka, circle of color.
  - The order of configuration is
    - hsla(hue, saturation, lightness, alpha)
    - hue is a decimal value, 0 to 360
    - saturation is a percentage of 0 to 100
    - lightness is a percentage of 0 to 100
    - alpha uses the transparency range of 0 to 1 (0.2, 0.88, etc.)
- Let us do an example
  - Refer to the hsla.html file located in the 4.15 folder within Module4.
  - Create a folder called hslach4.
  - Copy the fall.jpg from the Module4 starters folder to the hslach4 folder.
  - Launch your editor and open the rgba.html file from the 4.14 folder.
  - Save As hsla.html in the hslach4 folder
  - Delete the declarations for the h1 selector from the embedded styles.
  - Embed the code for the h1 selector.
    - 20 pixels of padding
    - use a serf font (Georgia, “Times New Roman”) with a size of 6em that has a light yellow color
      - Hint – Use the older hexadecimal color for the conversion to rgba.
    - set the alpha to 0.8 with a full saturation.
  - Save the file and open. It should look like the hsla.html file found in 4.15 within Module4.
  - If not, review the code, make necessary corrections, save, and refresh or reopen.

- Before we move on, I have a tip. Practice coding colors for modern browsers by using the hexadecimal code and convert it to the rgba code.
- Colors are difficult to work with because there are many factors to consider with colors and color schemes.
- Next, Let us look at CSS3 Gradients.

CSS3 GRADIENTS

- Color transitions blend smoothly
- Defined purely with CSS3
- No image file is needed
- Refer to the index.html from the gradient folder in the Module4 folder.

LINEAR GRADIENT

- Single direction, smooth color blending
- Top to bottom or right to left using a phrase below:
  - “to bottom”
  - “to top”
  - “to left”
  - “to right”

- List two or more hexadecimal color values for the start and stop color blends.

RADIAL GRADIENT

- Smooth blend from a single point outward.
    - radiant-gradient is the background-image value.
- Starts at the center of the page by default.
- Let us create a gradient webpage.
    - First, refer to the linear.html file in the 4.16 folder. Right-click and open with Notepad. This is the reference file.
    - Create a new folder called gradientch4.
    - Copy the starter4.html file from the Module4 folder to the gradientch4 folder. Rename it to index.html.
    - Code an embedded style for the body of the webpage for a fallback orchid background-color of #DA70D6 with a linear gradient blend of white to orchid that does not repeat from the top to the bottom. Save and open in a browser. It should look line linear.html from the 4.16 folder.
    - Next configure a radial gradient.
    - Edit the body section and code change the text within h1 to radial gradient using the same white and orchid
    - Edit the CSS and modify the value of the background-image property so that it configures a radial gradient from white to orchid from the center.
    - Refer to the code in the radial.html file found in the 4.16 folder within Module4.
    - Save the file and open it in a browser. It should look similar to the one shown.
- Having these types of effects handy for color manipulations will definitely make your webpages stand out.
- In this module we explored, examined and practices creating and configuring CSS3 Visual Elements and Graphics. It was a challenging road, but we make it through.
- The next module, module 5 we will exam web design ad its aspects of purpose.

MODULE 5 - WEB DESIGN

- How often have you noticed that some websites perform better than others? In this module we will explore what is and what is not good, standard practices to make your website better.

DESIGN FOR YOUR TARGET AUDIENCE

- Design your website for its purpose, to attract and keep your target audience.
- Consider your audience and how they will use your site.
- Consider the objects, images, information, etc., that your target audience will be looking for.

WEBSITE ORGANIZATION

- How will your information be organized?
- The three architectures are:
  - Hierarchical
  - Linear
  - Random
- The use of a site map will you organize your information.
  - Hierarchical
    - Uses a grouping method of relevance.
    - Grouping methods may include:
      - Product
      - Alphabetical
      - Categorical
      - Functional
      - Many types of grouping methods.
    - Use “chunking” to keep relevant objects together.
    - Maintain a “three-click rule” when planning the depth.
    - May need to blend a linear and hierarchical design.
  - Linear
    - Best for sequential navigation.
      - Tutorials
      - Presentations
  - Random
    - Not suited for commercial use.
    - Has very little logical structure
    - There is no logistical approach
- Explore methods for planning
  - Brainstorming
  - Stare and compare
  - Modeling
    - Maintain a design similar to competitors.
    - Keep up on the latest trends in navigation.

PRINCIPLES OF VISUAL DESIGN

- Visual design principles are cross-industry practices.
  - Repetition
    - Keeps your audience on track by repeating common objects.
      - Colors
      - Images

        - Actions/Activities
        - Fonts
        - Styles
    - Contrast
        - Keep objects visible and easy to read.
        - Font colors and background colors or image colors should not fade into or overbear one another
    - Proximity
        - Keep like objects/elements contained within a region
    - Alignment
        - Keep display blocks even horizontally to keep the eyes from having to move randomly between blocks

DESIGN TO PROVIDE ACCESSIBILITY

- Disabled visitors
- Text-only browsers
- Voice narrations
- Give options for zoom
- Use descriptive alt and titles for images, links, blocks, etc.
- May benefit search engine listings
- Giving disabled visitors a means to successfully navigate your site can increase traffic and encourage your visitors to return.
- Keep the site:
    - Perceivable – easy to read, see, and hear.
    - Operable – keep navigation and actions seamless
    - Understandable – Ease to read, has logical flow, etc.
    - Robust – up-to-date with modern elements that help to future proof your site.

WRITING FOR THE WEB

- Keep content concise and to the point
- Avoid large blocks of text, also known as fluff.
    - Organize your content
        - Easy for visitors to scan the webpage
    - Font Selection
        - Use common, acceptable, and easy to read fonts.
        - Make the fonts appropriate for the content
            - Do not use comic fonts for obituaries.
    - Font Size
        - To small causes eye strain.
        - To large forces unnecessary eye movement
    - Font Weight
        - Bold or *italicize* important information
        - Do not make everything important
    - Font Color and Contrast
        - Make the font colors and background colors have an appropriate blend of contrast
        - Avoid bright colors with bright backgrounds
        - Keep the font colors and the background colors within the same color types.
        - Search for sites that offer great advice or standards for color contrasts.

- Line Length
  - Keep your sentences short and concise
  - Avoid long words unless they are truly meaningful
  - Maintain the standard of 6 to 10 words per sentence.
  - Longer sentences are acceptable as a summary or point-of-fact.
- Alignment
  - Avoid centering blocks of text. The eyes have to jump to much from line to line.
- Text in Hyperlinks
  - Use short and descriptive phrases for hyperlink text.
  - Adjust navigation of touch screens.
    - Monitors
    - Smartphones
    - Tablets
    - Etc.
- Reading Levels
  - Match the content levels to your target audience.
  - Do not use long, complicated words unless they point-of-fact.
- Spelling and Grammar
  - Review your spelling and grammar.
  - Good spelling and grammar will help avoid a judgmental audience.

USE OF COLOR

- Determining color schemes may be the most difficult aspect of website planning.
  - Color Scheme Based on an Image
    - Research how human eyes perceive color.
    - Locate websites tailored for color schemes based on images and background colors.
    - Use personal experience on color contrast and blending

COLOR WHEELS

- Color wheels may be a good starting point.
  - Shades, Tints, and Tones
    - Shade – Making the color darker by mixing in black.
    - Tint – Making the color lighter by mixing in white.
    - Tone – Making the color less saturated by mixing in gray (blending white and black).
  - Color Scheme Based on the Color Wheel
    - Monochromatic Color Scheme
      - Shades, tones, and tints of the same color.
    - Analogous Color Schemes
      - Adjacent colors on the color wheel.
      - The main color is the dominant color.
      - The offset colors are used for accents.
    - Complementary Color Schemes
      - Colors that are opposite on the color wheel.
      - One color must be selected as the dominant.
      - The other is the complement.
    - Split Complementary Color Scheme
      - Select a dominant color

  - Complement color is opposite
  - The two colors adjacent to the complement are secondary complements.
- Triadic Color Scheme
  - Colors that are equally spread apart.
    - Imagine an equilateral triangle and the three corners make up the scheme.
  - Tetradic Color Scheme
    - Four colors that are complementary pairs.
- Implementing a Color Scheme
  - Select a dominant color.
  - Choose additional colors complementary to the primary.
  - Plan, Plan, Plan your color scheme before implementing.
  - Trial and Error is time-consuming and frustrating.
- Accessibility and Color
  - Keep in mind that not everyone sees colors the same way.
  - Consider someone who is colorblind.
    - Researching the colors that are often missing from colorblindness may help to determine a color scheme based on the results.
- Colors and Your Target Audience
  - Consider color schemes based your audience.
  - Research demographics to see what schemes best match the demographic of your target audience.
  - White will always be a strong background color.
  - A ***hero*** image is a large or compelling image that grabs the attention of the audience. Hero images will hold a majority of the colors of your scheme.

USE OF GRAPHICS AND MULTIMEDIA

- Graphics and multimedia should be used sparingly. However, this too is dependent on your target audience.
- Keep close considerations for accessibility.
  - Text-based browsers
  - Narrative browsing (voice overs)
  - Keep the graphics and multimedia valid for your site and your target audience.
- File Size and Image Dimensions Matter
  - You do not want your audience to go away because of slow loading of images or multimedia.
    - The unwritten rule is three seconds. The page should be available to begin navigation and reading within three seconds.
    - If actions are held up due to slow downloads because of resource intensive images or multimedia, consider alternate objects. Once your visitor is vested in your page, meaning they are “hooked” they are less likely to leave if they encounter slower load times.
- Antialiased/Aliased Text in Media
  - Jagged images or text can be uneasy to read.
    - They may appear blurry.
    - They may bleed into other graphics or text.
  - Use image tools to smooth the edges of graphic or text, Antialiased.
- Use Only Necessary Multimedia

        - Use animations, video, sound to a minimum unless the site requires it for entertainment or capturing an appeal.
        - The use of multimedia is extremely dependent on your target audience.
    - Provide Alternative Text
        - As it was discussed earlier in this series, alternate text is necessary for accessibility
        - When you cannot provide meaningful text as an alternate, consider audio or a video. Again, this is determined by your target audience.

MORE DESIGN CONSIDERATIONS

- Load time
    - As mentioned previously in this module, visitors may leave if it takes too long for the page to become accessible. Remember the three-second rule.
    - Each visitor will have a different perceived load time. This is the time a visitor waits before actions or activities can resume.
    - Some techniques to help minimize load time.
        - CSS Sprites
        - Reduce image and multimedia sizes.
            - Weigh between quality and accessibility
        - Break up larger pages into smaller pages.
- Above the fold
    - Keep the most relevant information towards the top of the page.
    - People do not read as often as they scan pages. Keep information flowing from top left to bottom right.
- White Space
    - White space is the space or area around the objects on the webpage.
    - Additional white space may be more appealing and capturing to your visitors, helping to ensure they return.
- Horizontal Scrolling
    - Avoid letting your page elements extend beyond the right edge of the browser window when shown in full-screen.
    - Scrolling horizontally is not optimal because the information has to be “chased” rather than scanned.
- Browsers
    - Not all browsers will perform the same on your website.
    - Consider how the more popular browsers will interact with your code.
    - Using HTML5 combined with CSS3 can help to provide the best of both worlds; using multiple settings for different browsers within the same style elements.
- Screen Resolutions
    - Designing webpages around screen resolutions can limit what you can do with your webpages.
    - Consider alternate methods of getting your information to your target audiences.
        - Mobile sites for mobile users
        - Visitors using tablets.

NAVIGATION DESIGN

- Ease of Navigation
    - As mentioned earlier, planning and maintaining an operable website revolves around a visitor’s ability to seamlessly navigate your site.
    - Clearly label or explain optimal ways to navigate your site.

        - Consider using back buttons on the page rather than having the visitor click the browsers back button. Browser caching could interfere with reverse navigation.
    - Navigation Bars
        - These elements use images or text displayed in a logical order so your audience can have the same pathways from each page.
    - Breadcrumb Navigation
        - This is similar to the browser history but is generally display at the top within the <nav> element.
        - You may have seen this type of navigation towards the top of the page when there is a visual reference showing the order of navigation.
            - Home > Contact > etc.
    - Using graphics for Navigation
        - There are pros and cons to using graphics for navigation.
            - Pros
                - More descriptive with the use of buttons, images, or icons that symbolize the action or activity.
                - Images can be switched when needed for trends.
            - Cons
                - May not be recognized in text-only or image disabled browsers.
                - Images may get miscoded whereby losing their relationship to the destination.
    - Skip Repetitive Navigation
        - We will discuss this consideration in more detail in Module7.
    - Dynamic Navigation
        - This is the type of navigation that uses cascading menus.
        - When the visitor hovers over a link, a menu comes to focus giving the user additional link choices. This continues until the visitor clicks a link or removes the cursor from the menu area.
    - Site Map
        - Site maps are a text-based hierarchical layout of the website’s path of navigation.
        - Resembles an outline with varying levels of based on parent links.
    - Site Search Feature
        - This is an acceptable method of locating specific content on a website.
            - Pros
                - Additional links can be implemented as the searchable information becomes more available.
            - Cons
                - It needs to be updated regularly to remove or add relative links which can be time consuming and frustrating
                - Based on keywords and metadata. If these are not correct or relevant to the information be sought, the visitor may leave and not return.

PAGE LAYOUT DESIGN

- Wireframes and Page Layout
    - Wireframes are sketches or diagrams in block form that gives the designer(s) a visual clue as to how the webpage will look after coding.
    - Wireframes are dynamic in that they provide a starting point, but the considerations for the page layout will make wireframes iterative.

- Fixed Layout
    - A fixed layout is static in that once it is coded, the browser or the user have no control over the look and feel.
    - Changing the window size does not dynamically change the layout. This often leads to horizontal scrolling if the area of the webpage exceeds the width of the screen based the relative size of the screen.
    - The size of objects such as images, containers, and other graphics are hard-coded meaning the values of the dimensions are numerical.
- Fluid Layout
    - This is a dynamic layout in such the layout changes as the viewport changes.
    - Downsizing or upsizing the page alters its visual layout without removing its purpose of functionality and operability.
    - The size of objects such as images, containers, and other graphics are soft-coded meaning the values of the dimensions are set as percentages thereby giving browser the flexibility of arranging these objects the best use of the available space.

DESIGN FOR THE MOBILE WEB

- Desktops are becoming less used by the casual surfer.
- The use of portable, hand-held devices is becoming more and more appealing. The behind-the-scenes technologies for these devices are constantly improving.
- This approach to user convenience is a good indicator that website designs need to have a flexible and dynamic approach for operability on mobile devices.
- Three Approaches
    - Develop a mobile site strictly for hand-held devices using the .mobi extension.
    - Push mobile devices to mobile sites based on device access type, e.g., desktop versus hand-held.
    - Maintain a dynamic site that will adjust its properties based on the accessing device.
- Mobile Device Design Considerations
    - Smaller screen sizes
    - Low bandwidth
    - Font, color, and media issues
    - Awkward controls with limited processing power and lack of memory for resource intensive websites.
    - Functionality that provides ease of access through hyperlinks or search objects.
- Mobile Design Checklist
    - Be aware of smaller screen sizes
    - Remove nonessential elements of the page to load.
    - Use background images tailored to mobile devices. These images will still work on full-sized devices.
    - Be very descriptive with alt for text for images.
    - Do not exceed a single column due to screen size and resolution of mobile devices.
    - Choose colors that maximize contrast because hand-held devices are more difficult to read on sunny days.
- Responsive Website
        - There are dozens of screen resolutions in play with regards to desktop, tablet, or Smartphones.

- Design the website so hand-held devices can have dynamic control over the actual webpages.
- More on this will be examined in Module7.

WEB DESIGN BEST PRACTICES CHECKLIST

- There are numerous websites that offer website development checklists.
- Checklists are iterative and flexible giving the developer(s) a roadmap for their website.
- In this module we discussed
  - The most common type of organization.
  - Visual design principles
  - Tailoring to your target demographic
  - Navigation principles
  - The use of text
  - Graphics
  - Universal webpage design
  - Design Techniques
  - Design best practices
- In Module 6 we will expand on the design aspects with page layout techniques.

MODULE 6 - PAGE LAYOUT

- This module will lay the foundations for best page layout considerations.
- We will also code new HTML5 elements.

THE BOX MODEL

- Each element in a web-document is considered a rectangle.
- The rectangle has content, padding, a border, and a margin.
  - Content
    - This is the area of a webpage or container that can hold images or text.
    - The visible width of an element consists of the content, padding, and border of the element.
    - The width property does not consider the margins and therefore care has to be taken when calculating the total area needed for content, images, and multimedia.
  - Padding
    - Padding is the area directly around the content, image, or multimedia. It is configured using the padding property.
  - Border
    - This is the area between the padding and the margin.
    - Defaults to a value of 0 and is not displayed.
  - Margin
    - The margin is the space between the border of one element to the border of another element.
      - margin-property configures the margin. There is a set margin depending on the element or object but is not displayed.
      - Changing the margin dimensions (px or em) will affect adjacent elements and any elements outward.
      - To completely remove the margin, use 0 as a value without the px or em.

NORMAL FLOW

- Normal Flow
  - The normal flow is the order in which the browser reads the code, from top to bottom.
  - This is especially important when working with block elements like <div> tags.
- Let us look at an example.
  - First, locate the file box1.html in the 6.1 folder. Right-click, Open with, Notepad. This is for reference to the code. Opening the file in a different application helps to remove any overwriting. However, when in Notepad, always click No when closing the file if you get a notification asking if you want to save the file.
  - Launch your editor and open the starter1.html.
  - Save the starter1.html (Save As) box1.html
  - In the <body> tags enter the code found in box1.txt.
  - Next, embed a class style for the div1 and div2 selectors using the code from box1.txt between the style tags.
  - Save the file and open in a browser.

    - You can see the blocks….
    - Next, save the box1.html (Save As) box2.html for an example of nested div containers.
    - Next, change the div1 and div2 class code in the body as shown in box1.txt.
    - Save the box2.html and open it in a browser.
    - You can see……
    - This is how normal flow works.
    - This example should give you a good understanding of the normal flow.
    - This is an important aspect when writing code.
- Now, Let us have a look at the CSS Float property

CSS FLOAT

- The float property by default pushes the object as far left as possible within the boundaries of the container or the viewport.
    - Values are:
        - float: right;
        - float: left;
    - Note – it is good practice to include a width value for the object if one does not exist, like an <img>
- Let us have a look at an example.
    - First, launch Notepad or FileEdit (Mac) and 6.2 index.html …….
    - Next, create a folder….
    - Next, copy files……Module6 to ….ch6float…..
    - Launch your editor and open starteryls.html…..
    - Notice no CSS…
    - Open the starteryls.html in a browser…..
    - Notice the normal flow of the objects in the page.
    - Let us add some CSS.
    - Add a style rule for a class named “float”. Refer to the file from Notepad or FileEdit for the code.
    - Assign the selector to assign the float class to the image element.
    - Save and open.
- Take a moment to switch back and forth between the code and the webpage to better understand the normal flow and how it is rendered.
- Notice the image is vertical in the page and “floated” outside the normal flow.
- Next, Let us look at the CSS’s way of clearing floats to render the normal flow as intended.

CSS: CLEARING A FLOAT

- The clear property is used to end a float.
    - Values
        - left
        - right
        - both
- Clear using a line break <br>
    - A common practice to clear floats is to use a class within the <br> element. <br class=”some name”> with a class selector in the CSS.
    - Generally, the <br> class element is coded prior to the closing </div> tag.
- Clear using the overflow property
    - It is widely used to clear floats.
        - Values

            - visible – default – content will overflow its intended boundaries.
            - hidden – the content is clipped, meaning some will not be visible, to contain within the boundaries.
            - auto – attempts to fit the content but will add scrollbars if the object is too large for the container.
            - scroll – the content is rendered with the use of scrollbars.
    - Look at the code below:
        - <div>
        - <img class="float" src="yls.jpg" alt="Yellow Lady Slipper" height="100" width="100">
        - <p>The Yellow Lady Slipper grows in wooded areas and blooms in June each year. The flower is a member of the orchid family.</p>
        - </div>
    - We see the image and the first paragraph are within the <div> tags.
    - However, this will not allow the background-color to extend beyond the container.
    - Use the overflow property for resolution.
        - div {background-color: #F3F1BF;
            - overflow: auto;
            - width: 100%}
            - *** the width will allow all of the content to display, even the background-color, thus clearing the float.
    - The overflow property is the more common practice of clearing float.
- Using Scrollbars
    - Using overflow: auto will automatically include scrollbars if the content exceeds the allocated container space.
    - It is best, at times, to hard-code scrollbars when the design requires the content outside the oversized element remains unaltered by the browser.
        - div { background-color: #F3F1BF;
            - overflow: scroll;
            - width: 300px;
            - height: 100px;

CSS3 BOX SIZING

- When using static objects like images, browsers do not consider the space outside of the dimensions of the object, like padding or borders. Thus, the rendered code will not perform as expected.
- Using hard-coded values using the box-sizing property will force the browser to render the code as wanted.
    - Values
        - content-box – default
        - border-box
- Open the files boxsizing1.html and boxsizing2.html found in the Module6 folder.
- Notice boxsizing1 is set to default and the browser renders the elements according to the viewport.
- In boxsizing2 the browser keeps the elements intact because the border-box value maintains a strict rule for maintain the wanted appearance.
- One trick that developers use to maintain an expected rendering is to universally apply the box-sizing property to all HTML elements by using the universal selector, the asterisk.
        - * {box-sizing: border-box}

        - This will force the browser to consider all dimensions when rendering the content/objects within the container.
      - Note: Be careful using the universal selector as it will be applied to all the elements. You may need to embedded or inline styles to override the universal selector to achieve the desired rendering.
- Maintaining the elements within the containers using box-sizing takes careful planning and simple math. However, when used properly and with intent rather than act of laziness will save hours of frustrating trial-and-error and troubleshooting.
- Next, Let us look at how CSS can make configuring a two-column layout less complicated.

CSS TWO-COLUMN LAYOUT

- Nothing is better than jumping right in. This next exercise will convert a single layout to a two-column layout.
- This is a rather lengthy exercise so pause the video when needed to work back and forth from the samples to your editor.
  - Launch your editor and open the **singlecol.html** file from the Module6 folder.
  - Take note of the code structuring the HTML elements. Also notice this is a standard HTML5 structured layout. We see the:
    - Head
    - Styles
    - Body
    - Nav
    - Main
    - Footer
  - Save the file as (Save As) index.html. It should default to the Module6 folder. When you open up the file it should look ……
  - Next, Let us convert this single-column design into a two-column design.
  - We will move the <nav> element to the left of the page, removing it from the top.
  - Embed a class nav selector with a float: left; and a width: 90px;
  - We need to break the horizontal layout by adding a line-break after the first two nav links.
  - Add the <br> tags… one after the Home and one after Page 1.
  - Save the file and open it in a browser. It should look like the one shown.
- This code automatically floats the nav menu to the left, however, we still have a single-column layout.
- Let us add some CSS to the nav and the body, thus creating a two-column layout.
  - First, for this to render properly we need to make sure the margin of the main element is greater than the floating element, the nav container.
  - Within the embedded CSS, code the following:
    - main {margin-left: 100px;}
  - Save the file and open in your browser or refresh if still open.
  - You can see the main content adjusted fully to the right, leaving the nav to the left.
  - To get a better visual, Let us add some pizazz.
  - Navigate to the index.html file found in the 6.3 folder. Right-click, Open with, Notepad. This will display the HTML code. Reference this code to finish this exercise.
  - Type in the remainder of the code from the reference file.
  - Save the file and open it in a browser.

- Internet Explorer needs an extra push to get the look right, so we need to add 'display: block; to the main element selector in the CSS.
- You can see an example of the IE fix by opening the iefix.html file in the 6.3 folder.
    - Adding a bit of pizazz makes this page looks cozy and warm.
    - Next, we will explore using hyperlinks in an unordered list.

HYPERLINKS IN AND UNORDERED LIST

- One of the more popular reasons for using CSS is it gives the developer better control of semantic coding.
- Semantic coding is writing code in such a manner as to give the visitor a visual clue as to what the purpose of certain elements represents. Examples are navigation, menus, dynamic menus, etc.
- This is especially pertinent with regards to navigation.
- Configure List Markers with CSS
    - List markers are the bullets, alphanumeric, Roman numerals, and images used in ordered and unordered lists.
    - The list-style property is used -type, -images, and/or -position to configure the markers you want to use for lists.
        - list-style-type – these are what we normally see at the beginning of lists.
            - Values
                - none
                - disc
                - circle
                - square
                - decimal
                - upper-alpha
                - lower-alpha
                - lower-roman
        - List-style-images – we use this property to retrieve the image we wish to use for our markers.
            - Value
                - url(location to image)
                    - This location can be internal or external to the web server.
        - List-style-position – sets the position of the marker. Can be used in conjunction with -type and -images properties.
            - Values
                - inside – indents the markers and wraps the text.
                - outside – default – markers do have a default placement.

VERTICAL NAVIGATION WITH AN UNORDERED LIST

- A normal list like this would default to using bullet points as markers.

```
<ul>
        <li><a href="index.html">Home</a></li>
        <li><a href="menu.html">Menu</a></li>
        <li><a href="directions.html">Directions</a></li>
        <li><a href="contact.html">Contact</a></li>
    </ul>
```

- Let us look at a snippet of code using CSS to remove the markers. So that we do not inadvertently remove all markers on all <ul> lists globally, Let us use a descendant selector to make sure only the ul markers in the nav container is removed.
    - nav ul {list-style-type: none}
- Do you want to remove the line seen under most hyperlinks? Use the text-decoration property. We can make this globally, embedded, or inline.
    - nav ul {text-decoration: none;}
- For all hyperlinks
    - nav a {text-decoration: none;}

HORIZONTAL NAVIGATION WITH AN UNORDERED LIST

- Use CSS to shift your ul list to a horizontal position.
    - Values
        - none – removes the element
        - inline – places the list horizontally
        - inline-block – places the list in a block fashion with horizontal aspects.
        - block – will have a margin at the top and bottom

```
<nav>
    <ul>
        <li><a href="index.html">Home</a></li>
        <li><a href="rooms.html">Menu</a></li>
        <li><a href="directions.html">Directions</a></li>
        <li><a href="contact.html">Contact</a></li>
    </ul>
</nav>
```

-
    - Make the list horizontal
        - nav li {display: inline;}
- Let us look at some pseudo classes

CSS INTERACTIVITY WITH PSEUDO-CLASSES

- Every wonder how links changed colors or shape when you hover over them?
- CSS is the answer
- A pseudo-class is simply a class name with a colon in front. This tells the browser to make a change upon a listed action. Think of it as JavaScript, but this is quicker and more efficient.
    - pseudo-class names
        - :link
        - :visited
        - :focus
        - :hover
        - :active
            - Note – These must be coded in this exact order to work. Although some can be eliminated, you must use the remaining class names in this order.

        - This what it may look like
        - nav a:link { color: #ffffff; }
        - nav a:visited { color: #eaeaea; }
        - nav a:hover { color: #000066; }
    - Let us work an example.
        - First, Let us open the file for our reference code.
        - Find index.html in the 6.4 folder. Right-click, Open with, Notepad. This is our reference file.
        - Next, create a folder called ch6hover
        - Copy the lighthouseisland.jpg, lighthouselogo.jpg, and the starter.html files to the ch6hover folder.
        - Next, open the starter.html file with your editor.
        - Save the file as (Save As) index.html to the ch6hover folder.
        - Using the reference file, type the missing code to the new index.html file.
        - Save and open it in a browser to view these changes.
        - Next, type in the missing CSS code, save and open it in a browser.
- CSS Buttons
    - Using buttons for navigation are excellent for semantic use.
    - Using CSS is quicker and does not use bandwidth to download the button images.
- Let us dive right in an exercise for CSS buttons.
    - First, locate the reference file **button.html** in the 6.5 folder. Right-click, Open with, Notepad. This is our reference file.
    - Next, create a folder named button in the Module6 folder.
    - With your editor, open the template.html file in the Module2 folder.
    - Save As index.html in the button folder.
    - Using the button.html file as your reference, type in the code to the new index.html file.
    - Save and open it in a browser.
    - We can also use CSS button classes and some fancy code and configure horizontal buttons for hyperlinks.
    - Open up the navbuttons.html file found in the 6.5 folder.
    - See how the buttons are running horizontally.
    - Right-click to view the source.
    - Notice that the configuration is almost exactly the same as configuring normal buttons, with the exception of the nav li style with the inline value. That is the only change needed on this particular page for the buttons to align horizontally.
    - Let us try an experiment
    - Open the navbuttons.html in the 6.5 folder with your editor.
    - Let us comment out the nav li
        - /* nav li {display: inline;} */
    - Was not that an interesting coding session? Let us move on to another exercise coding a CSS two-column layout.

PRACTICE WITH CSS TWO-COLUMN LAYOUT

- Let us get started on configuring another two-column layout.
    - First, locate the file index.html file in the 6.6 folder for the reference html code. Right-click, Open with, Notepad.
    - Next, from your editor, open the file lighthouse.css in the 6.6 folder. This is the reference code for the CSS.

    - Open a new document in your editor and create a new file named lighthouse.css and save in the ch6practice folder. This is the external style sheet for this exercise.
    - Next, open the starter2.html file in your editor.
    - Save the file as (Save As) index.html in the ch6practice folder.
    - Using the index.html reference file, and the lighthouse.css reference file, type in the missing code.
    - Save and open the file and move the mouse over the nav links and see the magic.
- What a way to reinforce your skills.
- Next, we will work in the header for a header text image replacement

HEADER TEXT IMAGE REPLACEMENT

- There will come a time that the font in the header needs to be changed, for whatever reason.
- Using images requires creating a new image with the new desired font. This is very time consuming.
- Using CSS and an improved header text image replacement technique, banner fonts can be changed at any time within the CSS.
- Let us try a quick example.
    - First, create a folder called replacech6.
    - Copy the starter3.html file in the Module6 folder and rename it index.html.
    - Next, locate and copy the background.jpg and the lighthousebanner.gif from the starters folder in Module6 to the replacech6 folder.
    - Locate the index.html file in the 6.7 folder. Right-click, Open with Notepad for the reference code.
    - Open the index.html from the replacech6 folder in your editor.
    - Type in the missing code from the reference file to the new index.html file.
    - Save and open the file in your browser.
    - That is all there is to it.
    - Next, we will practice using an unordered list for an image gallery.
    - First, create a folder named gallery in the Module6 folder.
    - Next, locate the index.html file in the 6.8 folder. Right-click, Open with, Notepad, this is our reference file.
    - Next, from your editor, open the template.html file in Module2 and Save As index.html in the gallery folder.
    - Now, type in the missing code from the reference file to the new index.html file.
    - Save and open the file. The images should be equally spaced and realign as the page size changes.
    - That was short and sweet. This is something that you should practice. It will be used quite a bit with image organization.
    - Next, Let us look at using CSS to position elements and objects.

POSITIONING WITH CSS

- .We may need full control of an element or object to keep it in a particular location.
- Used to have additional control over elements and objects.
- There are four position properties used in CSS
    - Static – This is the default value and is rendered in normal flow.
    - fixed – Keeps the element or object fixed in a location, even when the page is scrolled.

  - relative – Positions the element or object relative to where it would be in normal flow.
  - absolute – Set the element or object in an exact location, regardless of normal flow.
- Static
  - position allows the webpage control of an element or object for normal flow positioning.
- Fixed
  - This keeps elements or objects in place, not affected by normal flow.
  - Examples
    - Right or left navigation menus
    - Images
    - Multimedia
    - Alerts
  - nav {position: fixed}
- Relative
  - Keeps objects and elements as close as possible relative the normal flow.
  - The browser still keeps the area reserved
  - Normal flow position for elements and objects may be adversely affected.
  - Values
    - left – offsets the relative position of the element or object to the left – numeric or percentage
    - right - offsets the relative position of the element or object to the right. – numeric or percentage
    - top - offsets the relative position of the element or object to the top – numeric or percentage
    - bottom - offsets the relative position of the element or object to the left – numeric or percentage
    - h1 { background-color: #cccccc;
      - padding: 5px;
      - color: #000000;
    - }
    - p { position: relative;
      - left: 30px;
    - font-family: Arial, sans-serif;
    - }
- Absolute
  - Specifies the exact location
  - Dependent on the first non-static parent element.
  - If none exists, the position is relation to the webpage body.
  - Values
    - left
    - right
    - top
    - bottom
  - h1 { background-color: #cccccc;
    - padding: 5px;
    - color: #000000;
  - }
  - p { position: absolute;
    - left: 200px;
    - top: 100px;
    - font-family: Arial, sans-serif;

            - width: 300px;
        - }
- Let us practice positioning
    - First, open the gallery.html file in the 6.9 folder. Right-click, Open with, Notepad. This is our reference code.
    - Create a folder named gallery2
    - Copy the image files (photo1, 2, 3, 4, and the photo thumbs) to the gallery2 folder.
    - Launch the editor and open the template.html file in Module2, Save As index.html in the gallery2 folder.
    - Using the reference code, type in the missing code to the index.html file.
    - Save and open the file.
- Practice this technique and play with the various values to see what happens.
- Next, we will look at CSS debugging techniques.

CSS DEBUGGING TECHNIQUES

- There will always be issues when creating new code.
- Having a good footing on debugging (troubleshooting) will relieve some frustration when errors occur.
- Remember, there are many, many sources to help you troubleshoot.
    - Verify Correct HTML Syntax
        - The HTML validator tool would be a great place to start when debugging those “I can’t find the problem.” The validator will give you clues, if not tell you exactly what is wrong.
    - Verify Correct CSS Syntax
        - CSS code statements are short so locating a missing punctuation mark is not too difficult. However, again use the CSS validator tool to check for syntax errors.
    - Configure temporary background colors.
        - This is good to view the elements and their locations.
    - Configure temporary borders.
        - This technique is extremely useful as we demonstrated earlier with the two-column layout.
    - Use “commenting out” for the CSS code
        - This helps to narrow down the errant code.
        - This is a great way to debug if your CSS file or embedment is long and complicated.
- Use the web to locate other techniques from seasoned professionals.
- Create your own errors so you can see what happens.
- Take notes for reference.

MORE HTML5 STRUCTURED ELEMENTS

- HTML5 introduced four new elements
    - Section
    - Article
    - Aside
    - Time
- Let us explore these a little more.

- Section
    - This states a section within the webpage, sort of like category.
    - It can have the normal HTML5 elements inside with the exception of head, style, and body tags.
    - Sections will remain inside the body tags
- Article
    - Used to present independent entries like blogs, comments, etc.
    - It can have the normal HTML5 elements inside with the exception of head, style, and body tags.
    - Articles will remain inside the body tags
- Aside
    - Block display element that indicates sidebar elements.
    - It can have the normal HTML5 elements inside with the exception of head, style, and body tags.
    - Aside will remain inside the body tags
- Let us so an example using these new structure elements.
    - First, locate the index.html file in the 6.10 folder. Right-click, Open with, Notepad. This is our reference code.
    - Next, create a folder called ch6blog in the Module6 folder.
    - Now, copy the index.html file and the lighthouselogo.jpg to the ch6blog folder.
    - Open the new index.html file in your editor. Compare the code to the reference file and type in the missing code.
    - Save the file and open in your browser.
- See how these new elements can have an advantage in the structure of your pages?
- What about how these new elements react in older browser version?
- CSS of course.

HTML5 AND COMPATIBILITY WITH OLDER BROWSERS

- Older browsers will not recognize these HTML5 elements.
- Use a block display statement in the CSS.
- header, main, nav, footer, section, article, figure, figcaption, aside {display: block}
- I hope you enjoyed Module6 as much as I did.
- Having a good and functional page layout is better for your visitors.
- Next, we will look at more information on links, layout, and mobile coding.

MODULE 7 - LINKS, LAYOUT, & MOBILE

- One of the more important considerations is how webpages will be rendered on mobile devices.
- We have to have good links and layouts to give mobile users as close to the same positive experience we want for desktop users.

ANOTHER LOOK AT HYPERLINKS

- Hyperlinks make the Web what it is today.
- Let us look at more coding with CSS to help keep the Web connected. But, first..
  - More on Relative links.
    - Recall that relative hyperlinks link to resources within your site.
    - We know how to code moving within the same folder or directory.
    - Hyperlinks need to move vertically (in and out of the current directory) as well as horizontally (within the same directory).
    - Links inside the current folder.
    - **<a href="weekend.html">Weekend Events</a>**
    - Links outside the current folder but in its parent directory
    - **<a href="days/weekend.html">Weekend Events</a>**
    - Links up one directory (outside the current folder and its parent directory)
    - **<a href="../weekend.html">Weekend Events</a>**
    - Links up two directories
    - **<a href="../days/weekend.html">Weekend Events</a>**
    - Let us look at an example of hyperlinking outside of the current folder.
- We will create a new webpage and configure hyperlinks in other webpages that will take us to our new webpage.
  - First, create a folder called Casita.
  - Next, copy all the files from the CasitaExample folder to the Casita folder.
  - Using your editor, open the canyon.html file. Save As juniper.html and make sure it is saved in the rooms folder. This is the new file we are adding to the website.
  - Open the index.html file in your browser and view source. Study the links and see how the code is to move from file to file and folder to folder.
  - Open the canyon.html file in your browser and view source. Again, study the links to see the code for moving in and out of files/folders. We will use the canyon.html file as a reference.
  - Back to the juniper.html file. Change the title and the <h2> text to Juniper.
  - Add the new <li> element with the new link between the Javelina room and the Weekend Events hyperlinks:
    - <li><a href=”juniper.html”>Juniper</a></li>
  - You can use the files and folders in the 7.1 folder for code reference.
  - We need to add the new link to all the pages in the same location within the lists.
  - Save each file after the changes are made. Afterwards, open the index.html file in your browser and navigate to each page and verify the juniper.html links are good. Again, if there are issues, refer to the 7.1 folder for the code references.

- That was not too difficult, just repetitive.
- Coding webpages can be time-consuming.
- Make using the copy/paste feature second nature. It will make coding less frustrating; however, make sure the syntax is correct or else you will copy/paste errors.

FRAGMENT IDENTIFIERS

- These provide a means of navigating to a specific location within a page.
- The element needs an id
  - <div id=”content”?>
- We now can click an anchor tag that pushes us to a specific element in the webpage.
  - <a href=”#content”>Go To Content</a<
- This hyperlink will move the visitor to the contents of the element <div> that has an id of “content.”
- We can also use this technique to move to a certain area of a different webpage as long as the target webpage’s element uses the same id=”name” that is referenced in the hyperlink.
  - <a href=”links.html#soccer”>Go to Soccer</a>
- Let us practice with fragment identifiers.
  - First, locate the favorites.html file in the 7.2 folder. Right-click, Open with, Notepad. This is the reference code.
  - Next, launch your editor and open the starter1.html file and Save As favorites.html in the Module7 folder.
  - Using the reference code, add the missing code to the new favorites.html file.
- This link trick can help your visitors get to a specific area of any webpage in your site.
- How many ways could you use this for linking your site together?
- The Target Attribute
  - This is mainly used to open a new browser tab or window.
  - Prevents a user from having to click the back button.
  - The visitor just closes the new tab or window, thus not losing their place on the website.
    - <a href=”someSite.html” target=”_blank”>Some Site</a>

BLOCK ANCHOR

- This anchor can configure one or more block elements as hyperlink.
- Let us look at a snippet of code.
  - <a href="http://www.w3.org/TR/html-markup">
  - <h1>HTML5 Reference</h1>
  - <p>Bookmark this site for a handy HTML5 reference.</p>
  - </a>
- See how the <h1> and <p> tags are the “keywords/phrase” inside the anchor tags?
- Look at the target.html file in the 7.3 folder. Open it in your browser and view the source code. You can see the target attribute in use.

TELEPHONE AND TEXT MESSAGE HYPERLINKS

- Used to make a phone call or send a text message inside a webpage.
- RFC 3966 refers to telephone hyperlinks.
- RFC 5724 refers to SMS text messaging.
- We will not explore this feature in this lesson; however, do your own research for additional information.

CSS SPRITES

- Used to combine smaller images into a larger single image.
- Save loading time because the browser only needs to make one image request rather than multiple.
- Works well when you have the same image within a nav menu.
- It will use the same CSS code for background images.
  - Sprite Generators
    - Check out the sites below regarding making your own sprites.
    - CSS Sprite Generator
      - http://csssprites.com
      - http://spritegen.website-performance.org
      - SpriteMe http://spriteme.org
      -
  - Let us quickly have a look at configuring sprites.
  - First, locate the index.html file in the 7.4 folder. Right-click, open with Notepad. This is our reference file.
  - Create a new folder called sprites in the Module7 folder.
  - Go to the starters folder in Module7 and copy the lighthouseisland.jpg, lighthouselogo.jpg, and sprites.gif to the sprites folder.
  - Use your editor and open the starter2.html file and Save As index.html in the sprites folder.
  - Now, add the missing code to the nav a selector/descendent and the nav a:hover selector/descendent/pseudo-class.
  - Save the file and open it in your browser. When you hover of the menu the image changes.
  - The reason we code a position of -100px in the nav a:hover is so the sprite image will lower 100px so the other lighthouse image in the sprite is visible.
- Can you see the efficiency of using sprites?
- Visit the sites noted earlier or search on your own.
- Practice this section to find beneficial ways to use sprites.

THREE-COLUMN CSS PAGE LAYOUT

- A majority of today's websites have three columns.
  - Navigation
  - Main Content
  - Sidebar
- We see this a lot on websites that have navigation on the left and ads, relevant links, archives, etc. on the right.
- This section is a long exercise.
  - First, locate the index.html file in the 7.5 folder, right-click and open with Notepad. This is our reference file.
  - Create a new folder called wildflowers3.
  - Copy the three images from the 7.5 folder to the new wildflowers3 folder.
  - Open the template.html file from Module2 and Save As index.html in the wildflowers3 folder.
  - Using the reference file, add the html missing code. Do not enter the embedded css code. You will add it later.
  - When you are finished with the html code, save the file, and open it in a browser. You should see that all of the elements are placed vertically rather than a three-column layout. Do not close this webpage, leave it up to compare it after the css is added.

    - Now add the embedded css code from our reference file to the head element of the index.html. Take a moment and review and see what the css is going to do. Try to imagine what the page will look like.
    - Save the file and open it in your browser.
    - Switch back and forth a few times and see how the css moved the elements to the right locations.

- Now that was an exercise.
- This page encompassed a great deal of what we have learned thus far.
- In the next section we will examine CSS styling for printing.

CSS STYLING FOR PRINT

- Gives you more control of how and what a visitor can print from your webpage.
- You may have seen a new print window pop up to show you what it will look like when it prints.
- This uses the media attribute
    - media=” “
- Along with the media type
    - media=”screen”
    - media=”print”
    - media=”handheld”
        - FYI – the handheld type is unstable and is not widely used.
- The next section will describe practices for designing for mobile devices

PRINT STYLE BEST PRACTICES

- Hide non-essential content
    - use the {display: none} to remove unwanted content for print.
- Configure font size and color for printing
    - Style the font in points (pt) rather than em or px. Printers will work better with font points.
    - Code the background-color to white.
    - Code the font to black.
    - To remove background images from printing use the.
        - {background-image: none}
- Control page breaks
    - You can use page-break properties to control the printing page breaks.
        - page-break-before
        - page-break-after
        - Values
            - Always
            - avoid
            - auto
        - .newpage {page-break-before: always;}
- Print URLs for Hyperlinks
    - Visitors may wish to see the URL on the printed page for future reference.
    - It takes two properties
        - A CSS pseudo-element
        - CSS content
    - The pseudo elements are
        - :after

        - :before
        - :first-letter
        - :first-line
        - FYI – CSS3 added a second colon preceding the name - ::after. It is not widely supported but will be implemented fully in the near future

- Let us put these to work by creating two css links, one for a screen view and one for printing.
    - First, navigate to the 7.6 folder and right-click and open with Notepad the index.html file, the wildflowers.css, and the wildflowerprints.css. These are the reference files.
    - Next, create a new folder called wildflowersPrint.
    - Now, copy the four files from the 7.5 folder to the new wildflowersPrint folder.
    - Using your editor, open index.html file from the wildflowersPrint folder.
    - Next, copy/paste (Ctrl+x to remove the styling) the embedded css between the <style></style> tags to a new document. Save the new document as wildflowers.css in the wildflowersPrint folder. This css version is for viewing the webpage on a screen.
    - Now, save the wildflowers.css as (Save As) wildflowersprint.css. This is the css version for printing.
    - Delete the <style></style> tags from the index.html file.
    - Next, add two <link> tags for the style sheets in the head section of the index.html file. One for the regular css using the attribute media=”screen” and the other link with the attribute of media=”print” and save. Refer to the reference index.html file for the code.
    - Next, we are going to edit the wildflowersprint.css file for printing.
        - As noted above, use the wildflowersprint.css from the 7.6 folder as the reference file.
    - Change the text-size to 24pt from 3em in the <h1> selector.
    - In the Remove the background image declarations and change the background-color to #FFFFFF (white) in the header section. Refer to the reference file for proper removal.
    - Under the header selector, add a new selector and descendent (header a) to remove the link (Skip to Content) from the printable page.
    - Set the nav area not to display by removing the declarations and adding display: none;
    - Remove the ‘header a’ selectors and declarations.
    - Alter the main selector (under the ‘aside’ selector) by changing the margin: 0 40% 0 0;
    - Add a font-size of 12pt.
    - Make the font-color #000000; (black).
    - Add a declaration width: 40% to the aside selector.
    - Change the font-size to 10pt for the .news class selector.
    - Add aside a selector and descendent to remove the underlines from the links in the sidebar. aside a {text-decoration: none;}
    - Now, make the URL printable with black font color and 10pt size.
    - Look through the css and make the background-color white for all h2 and h3 selectors.
    - Remove the min-width declaration from the #container id selector.
    - Save the file. Open the index.html file in your browser and do print preview.
    - Open the index.html file in the 7.6 folder in your browser and compare the two. They both should look very similar.

    - If there are differences in the appearance, the student should carefully and with detail compare the wildflowersprint.css reference file (7.6 folder) with the wildflowersprint.css file (wildflowerPrint folder) for errors.
- These aspects are important aspects for visitors when printing.
- Let CSS do the work rather than give the browser full control.
- Next, we will look at considerations for the mobile web

DESIGNING FOR THE MOBILE WEB

- This is particularly important due to the fact that nearly every person has a mobile device they use regularly.
- There are a few methods to design for the mobile web.
    - .mobi – this is the mobile domain extension.
    - Have a mobile directory in your website.
    - Use stylesheets, one for the desktop and one for the mobile.

MOBILE WEB DESIGN BEST PRACTICES

- There are several aspects of mobile design we need to consider.
    - Small screen sizes
    - Low bandwidth – low signal strength can slow rendering
    - Font, color, and media – Common fonts, good contrast between text and background.
    - Awkward controls – Touch displays need to be considered
    - Functionality – Hyperlinks easily readable and accessible.
- Optimize Layout for Mobile Use
    - Single column design with a smaller header
    - Important navigation links
    - Main content first
    - Footers for additional navigation and information.
    - Limit scrolling in one direction
    - Use heading elements
    - Use lists
    - Avoid tables at all costs.
    - Label form controls
    - Avoid using pixel units
    - Avoid absolute positioning
    - Hide non-essential content.
- Optimize Navigation for Mobile Use
    - Provide navigation at the top
    - Provide consistent navigation
    - Avoid targeted hyperlinks (open in new window)
    - Balance the number of links and the number of levels.
- Optimize Graphics for Mobile Use
    - Avoid large images
    - Configure alternate smaller background images
    - Avoid using images with text due to downsizing.
    - Specify image sizes custom for mobile displays
    - Provide alternate text for graphics.
- Optimize Text for Mobile Use
    - Have good contrast between font and background colors.
    - Use common font types
    - Use em or percentages for font sizing
    - Use short page titles

- Design for One Web
  - Try to design one site that all devices can use.
    - That pretty much sums up Designing for the mobile web.
    - Next, Let us have a look at the viewport meta tag.

VIEWPORT META TAG

- Meta tags are quite useful if configured correctly
- The viewport meta tag was developed by Apple for the iPhone and Android Smartphones.
- Helps to properly scale websites for mobile devices.
  - <meta name=”viewport” content=”width=device-width, initial-scale=1.0”>
- Directives (properties)
  - width – sets the width of the viewport in pixels
  - height – sets the height of the viewport in pixels
  - initial-scale – Numeric multiplier, set to 1 of 100%
  - minimum-scale – numeric multiplier, default to 0.25 for Mobile Safari
  - maximum-scale – numeric multiplier, default to 0.25 for Mobile Safari
  - user-scalable – Yes/no – determines if user has zoom control.
- This did not seem too complicated. Although there are many options available.
- Next, Let us have a look at CSS Media Queries.

CSS MEDIA QUERIES

- Progressive enhancements to improve viewing contexts (responsive web design from Module5)
- A Media Query
  - Determines factors of the mobile device.
  - Screen resolution
  - Screen orientation
  - Browser type
  - Code
    - <link href=”somestyle.css” rel=”stylesheet” media=”only screen and (max-width: 480px)”>
  - Media type
    - all – all devices
    - screen – computer screen
    - only – hides the media query from older browsers
    - print – printing the webpage
    - <link href=”somestyle.css” rel=”stylesheet” media=”only screen and (min-width: 768px) and (max-width:1024px)”>
  - Use the @media rule directly in the css
    - @media only screen and (max-width: 480px) {header {background-image: url(mobile.gif)
    - Search additional @media uses on the Web.
  - Let us take a two-column page and configure it to display as a single-column page for mobile devices. We will see three different layouts by resizing our browser to emulate a tablet and Smartphone.
    - First, locate the index.html file in the 7.7 folder. Right-click and open with Notepad. This is our reference file.
    - Create a folder named query7 in the Module7 folder.
    - Copy the lighthouseisland.jpg image from the starts folder in Module7 to the query7 folder.

        - Now, from your editor, open the starter3.html file from the Module7 folder and Save As index.html in the query7 folder.
        - Type in the missing embedded <style> code from the reference file to the index.html embedded <style> section.
        - Save the file and open in in a browser. You should be able to resize the window inward and see the layout change from the original layout to two mobile type layouts. One or tablets and another for Smartphones.
- Doing media queries and coding accordingly will take some of the stress of rendering the webpage
- In the next section, we will look at Responsive Images

RESPONSIVE IMAGES

- Flexible images with CSS
    - HTML and CSS code needs changed to accommodate viewport changes.
    - Remove the height and width attributes from the image properties.
    - Make the max-width to be 100%
    - Maintain the aspect ratio as closely as possible. The height can be set to height: auto;
- Let us work a responsive image example.
    - First, locate the index.html file from the 7.8 folder. Right-click and open with Notepad. This is the reference file.
    - Next, create a new folder called flexible7 in the Module7 folder.
    - Copy the two images from the 7.8 folder to the flexible7 folder.
    - Open your editor and open the starter4.html file and Save As index.html in the flexible7 folder.
    - As you edit and/or add code to the embedded styles from the reference file, image what is going to happen when the browser is resize.
    - Save the file and open it in a browser. Resize the viewport and see the changes. You can open the index.html file from the 7.8 folder and compare the actions of resizing.
- Keep this section in mind for flexible imaging.
- Next, Let us explore the HTML5.1 picture element.

HTML5.1 PICTURE ELEMENT

- Picture Element <picture></picture> – Gives developers a method to display images base on specific criteria. It requires opening and closing tags.
- Source Element is used with a container element.
    - A self-contained tag.
    - Attribute and values of the source element property
        - srcset – Required – is the source of the image.
        - media – Optional – media query
        - sizes – Optional – Numeric or percentage value to configure the image dimensions.
    - Let us try one.
        - First, locate the index.html file in the 7.9 folder. Right-click and open with Notepad. This is our reference file.
        - Create a folder called ch7picture.
        - Copy the large.jpg, medium.jpg, small.jpg, and the fallback.jpg from the Module7 starters folder to the ch7picture folder.

        - Using your editor, open the template.html file from the Module2 folder and Save As index.html in the ch7picture folder.
        - Next, using the reference file, add the missing code to the index.html file. Notice there is an <img> element contained in the <picture> element. This is for browsers that do not support the <picture> element.
        - Save the file and open it in a browser. Resize the browser and see the images swap depending on the viewport size.

HTML5.1 RESPONSIVE IMG ELEMENT ATTRIBUTES

- The srcset and the sizes attributes are new to HTML5.1
    - The size Attribute
        - informs the browser how much viewport is to be allocated for the image.
        - Defaults to 100vw (viewport width) which is 100%.
    - The srcset Attribute
        - Gives developers a method to display images based on specific criteria.
    - Let us have a quick look
    - Locate the index.html file in the 7.10 folder. Right-click and open with Notepad. This is the reference file.
    - Create a new folder called ch7image.
    - Copy the four images from the ch7picture folder to the ch7image folder.
    - Using your editor, open the template.html file from the Module2 folder and Save As index.html in the ch7image folder.
    - Edit/add the code from the reference file to the index.html file.
    - Save the file and open it in a browser. Again, resize the browser viewport and see the images change depending on the size. You may have to resize the browser, close, and reopen in the new size to see the appropriate changes.
- That was short and explanatory.
- In this next section we will cover some techniques for testing the displays of mobile devices.

TESTING MOBILE DISPLAY

- Obviously, the best way to test your webpage(s) for mobile use is to publish it and then look at them through a hand-held device.
- Some other methods are
    - Opera Mobile Emulator
        - This is for Windows only – http://www.opera.com/developer/movile-emulator
    - Mobile Phone Emulator
        - Runs in a browser window. Supports media queries – http://www.mobilephoneemulator.com
    - iPhone Emulator
        - Runs in a browser window. Supports media queries – http://www.testiphone.com
    - iPadPeek

    - Runs in a browser window. Supports media queries – http://ipadpeek.com
  - Testing in a Desktop Browser
    - You can open the webpage and resize the viewport to the approximate size of a hand-held.
    - If you have coded media queries within a <link> tag, modify the <link> tag to point to your mobile CSS.
    - Again, resize the viewport to the approximate size of a hand-held.
    - You can approximate the viewport size but search out these tools:
      - Chris Pederick's Web Developer Extension – http://chrispederick.com/work/web-developer
      - *Select resize > Display Window Size*
      - Viewport Dimensions Extension for Chrome – https://github.com/CSWilson/Viewport-dimensions
- Serious developers – locate SDKs for iPhone and Android mobile devices.
- We were introduced to methods and practices for designing with the mobile world in mind.
- Next, Let us look at CSS and the flexible box layout.

CSS FLEXIBLE BOX LAYOUT

- This will build on the float properties we examined earlier.
- This is a rather new flexbox technique to layout webpages for mobile devices that will not interfere with desktop devices.
- Flexbox is well supported in modern browsers
  - Check it out at – http://caniuse.com/flexbox for the current level of browser support.
- Configure a Flexible Container (flexbox)
  - The flex-direction property.
    - The flex container is configured using the display property. This will create a flexible container.
      - Some values are;
        - inline-flex – for flexible inline containers
    - The direction of the flex is configured using the flex-direction property.
      - #demo {display: flex}
      - flex-direction: row;}
  - The flex-wrap property
    - If we need to display the content on several lines, we use the flex-wrap property.
      - The default value is nowrap which maintains a single-line display.
      - The value wrap will allow the flex items to use multiple lines as needed. Think of it the same way as text wrapping.
    - The justify-content property will tell browser what to do with extra space.
      - See www.w3.org/TR/css3-flexbox/#justify-content-properly
      - The value of center will place the flex item in the center.
      - The value of space-between with evenly distribute the flex items and allocates the empty space.
- Configure the Flex Items
  - All items in a flex container are flexible.
  - They all have the same aspects
    - flexible in size

        - allocated the same amount of display area in the container
        - display in the same order as coded.
        - modifications take place with CSS
    - Flex property
        - Refer to www.w3.org/TR/css3-flexbox/#flex-common
        - This shows a list of value and attributes.
    - Order property
        - Will adjust the loading precedence for the flex items. The default value is 0. The motion values are -1 and 1.
        - Additional information is at
            - http://css-tricks.com/snippets/a-guide-to-flexbox
            - http://demo.agektmr.com/flexbox
            - https://developer.mozila.org/en-US/docs/WEb/Guide/CSS/Flexible_boxes
    - Let us put this section to some good use. We will add @media queries for hand-held devices.
    - First, locate the index.html file in the 7.11 folder. Right-click and open with Notepad. This is the reference file.
    - Now, create a folder called flexbox7 in the Module7 folder.
    - Copy the images header.jpg and the pools.jpg from the Module7 starters folder to the flexbox7 folder.
    - Using your editor, open the starter5.html file from the Module7 folder and Save As index.html in the flexbox7 folder.
    - Remember that all elements inside the flex container are flexible.
    - Follow along with the short video example, pausing as needed, then move to the following exercise.
    - ***The following is an exercise the student does without tutorial.***
    - ***ADD/EDIT*** the code in the index.html file with the reference file. Notice that <div> element directly above the <nav> element. The <div> element is the flexible container.
    - ***ADD/EDIT*** the embedded style and body code using the reference file.
    - Save the file and open it in your browser. Resize the browser and see the magic of flexboxes. Scroll up and down to see the changes in the nav element, the images, and the content.
- We are almost finished with this module. Only one more section, CSS Grid Layout.

CSS GRID LAYOUT

- We have seen the CSS float and flexbox properties at work.
- There is a new method to design a layout with CSS called CSS Grid Layout.
- This is a new property still under candidate status at W3C.
    - http://www.w3.org/TR/css-grid-1/
- It is designed to let you configure a two-dimensional grid layout.
- It can be designed to be fixed-size or flexible
- It can also change the order of precedence for display elements and objects.
    - Configure a grid container
        - First, we need to define the grid container. This is the element that contains the grid area.
        - Values
            - grid – indicates a block container
            - inline-grid indicates an inline-display container.
        - Grid lines define the grid rows and grid columns.
        - Grid cell is the intersect points. Just like the cells in a spreadsheet.

    - The grid area is the container that contains one or more grid items.
- Configure grid columns and rows using
    - grid-template-columns
    - grid-template-rows
        - These tell the browser how much space to allocate for each row and column
        - Values
            - pixels
            - percentages
            - keywords (none, auto, etc.)
            - flex factor units
            - https://www.w3schools.com/cssref/pr_grid-template.asp

- Let us put this in action by creating a three-column layout for a desktop view, a two-column layout for tablets, and a single-column layout using a grid configuration.
    - First, locate the index.html file in the 7.12 folder. Right-click and open with Notepad. This is our reference file.
    - Next, create a folder called grid7.
    - Copy the header.jpg and the pools.jpg from the starters folder to the grid7 folder.
    - Using your editor, open the starter6.html file and Save As index.html.
    - Review the image of the GRID LAYOUT in the Diagrams slide before you begin. You can also review a very comprehensive grid explanation at:
        - https://www.w3schools.com/css/css_grid.asp
    - Review the reference code before you begin. You need to understand the grid configurations for the body and the css.
    - As you add/edit the code, imagine what the performance will be when resizing the browser viewport.
    - Save the file and open it in a browser. Chrome and FireFox support grids.
- What a journey Module7 has been.
- Next, Let us explore Module8 regarding Tables and CSS

MODULE 8 – TABLES

TABLE OVERVIEW

- There are three main tags in constructing tables. These all use opening and closing tags
    - <table></table> - table container
    - <tr></tr> - begins each row
    - <td></td> - the data.
- The first <tr> element will display at the column names using the <td> elements
- Each <tr><td> combinations afterwards will list each row of data.

TABLE ELEMENT

- The table element is the container for the data in the table.
- There are several attributes that can be used to create the table.
    - Align – obsolete - left is default, right, center – use CSS
    - bgcolor – obsolete- the table background color – use CSS
    - border – 0 default – no border
        - 1 to 100 stated in pixels
    - cellpadding – obsolete – use CSS
    - cellspacing – obsolete – use CSS
    - title – description of the table – may show as a tooltip
    - width – obsolete – use CSS
- The border Attribute
    - HTML 4.0 and earlier and XHTML used the border attribute to display the area of the table.
    - HTML5 uses the border property to render default borders around the table and cells.
- Table Captions
    - Used to display the table name above the table.
    - Must be coded first under the <table> tag.
        - <table border=”1”>
        - <caption>This table</caption>
        - <tr>
        - <td>
        - </td>
        - </table>

TABLE ROWS, CELLS, AND HEADERS

- Table Row Element
    - <tr></tr> is the row.
        - Align – obsolete – use CSS
        - bgcolor – obsolete – use CSS
- Table Data Element
    - <td></td> is the data that is placed in the cell.
        - Attributes
            - align – obsolete
            - bgcolor – obsolete
            - colspan – the number of columns spanned by a cell
            - headers

    - height – row height
    - rowspan – number of rows spanned by a cell.
    - scope – row, col
    - valign – obsolete
    - width – obsolete
- Table Header Element
  - <th></th> will place the column names in bold by default.
- Let us make a table.
  - First, locate the index.html file in the 8.1 folder. Right click and open with Notepad. This is the reference file.
  - Next, using your editor open the template.html file in the Module2 folder and Save As mytable.html in the Module8 folder.
  - ***ADD/EDIT*** the mytable.html using the reference code.
  - Save the file and open in a browser.
- That was not so bad, now was it.
- Let us move on to spanning rows and columns.

SPAN ROWS AND COLUMNS

- The span is used to make the table look like it has merged cells, similar to a spreadsheet.
  - colspan – the number of columns a cell will occupy
  - rowspan – number of rows a cell will occupy.
- Let us try it.
  - First, locate the index.html file in 8.2 folder. Right-click and open with Notepad. This is the reference file.
  - Using your editor, open the template.html file in the Module2 folder. Save As myrowspan.html in the Module8 folder.
  - ***ADD/EDIT*** the myrowspan.html using the reference code.
  - Save the file and open it in a browser.

CONFIGURE AN ACCESSIBLE TABLE

- It is recommended to use the <th></th> and the <caption></caption> tags.
- Another highly W3C recommendation is to us id names to associate the proper row cell with the appropriate header.
- Let us see what this looks like
  - Navigate to the table5.html, the table6.html, and the table7.html files in the Module8 folder. Open all three in a browser and view the source code.
  - Table5.html is the old method and is not the preferred method for accessibility.
  - Table6.html is the preferred method for accessibility. Note the ‘headers=’ in the <td> tags. They match the id names in the <th> tags.
  - Table7.html is an alternate method for accessibility; however, this method does not work will with text-only or text-to-speech browsers.

STYLE A TABLE WITH CSS

- Let us jump right in and style a table with CSS
  - First, locate the index.html file in the 8.3 folder. Right-click and open with Notepad. This is the reference file.
  - Create a folder called ch8table in the Module8 folder.
  - Open the starter.html in a browser and view the table. View the source code.
  - Using your editor, open the starter.html file in the Module8 folder and Save As menu.html in the ch8table folder.

    - Open the menu.html file in a browser. We will see the changes as they take place.
    - ***ADD/EDIT*** the code in the menu.html file using the reference file. Save the file after each change and refresh the menu.html window in the browser.
    - When you are finished, save the file and refresh. It should look like the index.html file in the 8.3 folder.

CSS3 STRUCTURAL PSEUDO-CLASSES

- These classes allow us to efficiently color rows to give the table a real spreadsheet feel.
    - Class names
        - :first-of-type
        - :first-child
        - :last-of-type
        - :last-child
        - :nth-of-type(n)
- Let us have a look at this.
    - First, locate the index.html file in the 8.4 folder. Right-click and open with Notepad. This is the reference file.
    - Open the menu.html file from the previous exercise and Save As menu2.html.
    - ***ADD/EDIT*** the code in the menu2.html file using the code in the reference file.
    - Save the file and open it in a browser. You can compare it to the index.html file in the 8.4 folder.
- pseudo classes are great tools and work well with tables.
- Next, Let us move on to Configure Table Section

CONFIGURE TABLE SECTIONS

- There are three option for coding table sections
    - <thead>
    - <tbody>
    - <tfoot>
    - The <thead> must be used if the <tbody> and the <tfoot> are used; however, the <tbody> and the <tfood> are optional.
- These elements are good for configuring the areas of the table in different ways using attributes or CSS.
    - Let us have a look at some sample code.
    - Locate and open the tfoot.html file in the Module8 folder in your browser.
    - Take a look at the source code. See how the three elements are used inside the <table></table> tags? This is another method of configuring tables using css along with the table sections coding
- Congratulations on making it through Module8.
- Next, we will explore forms in Module9

MODULE 9 - FORMS

- Forms are used in nearly every website world-wide.
- You have used them quite often for:
    - Shopping
    - Memberships
    - Contacts
    - Surveys
    - Many, many more
- Let us get started.

OVERVIEW OF FORMS

- Forms are used everywhere
- Some uses are:
    - Registering
    - Signing In
    - Emailing
    - Etc.
- Every time you hit a submit button after entering data you have more than likely used a form.
- Two main components of a form
    - The HTML code that displays the form.
    - The server-side processing that manipulates the data.
    - Data:
        - Data is not information and information is not data.
        - Data is ambiguous, raw, and truly meaningless.
        - It takes a least two pieces of data to create information.
        - If we say red, it does not define anything. There are too many variables missing. We would generally refer to this as color. It could also be a nickname. It could be the first word in a street name.
        - If we add car then we have red car. Now we have something we can relate too.
        - Having this bit of clarification will make this module more meaningful than just talking about forms.

FORM ELEMENT

- The form element is the container that holds the form on the webpage.
- Forms are independent and cannot be nested.
- Forms themselves are not complicated but there are complicated processes that happen in the background. We will not get heavy in this aspect, but we will interject when this can reinforce this module.
- There are five attributes that make up a form.
    - action – The is the URL or path/filename to the server
    - autocomplete – HTML5 attribute
        - On – The browser will fill the form
        - Off – The browser will not fill the form
    - id – optional – alphanumeric - used to provide a unique identifier. The id name cannot be used elsewhere on the webpage.

      - ***Note – The id name can be used in other webpages as an identifier; however, if there are multiple instances of the webpage or website open, there is the possibility of cross-contamination of data. I recommend documenting each id name and its location as to not repeat the id name.
    - method - required
      - get – appends the form data to the web address and sent to the web server.
        - This is the string of extra characters, words, etc. we see following the filename. Sessions, etc.
      - post – this is the private method of sending data to the server or another webpage. Preferred value for the method attribute.
    - name – optional – alphanumeric and must start with a letter. Be descriptive but short.
        - Form1 – not good
        - RegForm – okay
        - OrderForm – good
        - SurveyForm – good
        - FormForRegistering – not good but could be used.
    - syntax
      - <form name="OrderForm" method="post" id="OrderForm" action="processData.php"
        - the action is the server-side script that processes the data.

FORM CONTROLS

- The places on the form that allows for user control
  - Input field
  - Radio button
  - Check boxes
  - Buttons
- New HTML controls allow for
  - Email
  - URLs
  - Dates
  - Times
  - Date selection (calendar scripting)

INPUT ELEMENT FORM CONTROLS

- The input element is a self-contained tag and does not require a closing tag.
- The type="reserved action" attribute defines the purpose of this field
  - type="text" – is the long space we often see when entering data. First Name, Last Name, etc.
  - type="radio" – is the radio button
  - type="file" – initiates a "choose" for uploading/downloading.
- For a list of form types and attributes go to w3schools
  - https://www.w3schools.com/html/html_form_input_types.asp
- The new HTML5 attribute "required" Let us the browser validate that data has been entered
  - Not support in older browsers – ignored.
  - Also use client/server side scripting for additional validation.

SUBMIT BUTTON

- This button is used to “submit” or begin executing the method get or post.
    - <input type=”submit>
        - Attributes for both buttons are the same
            - type
            - name
            - id
            - value
            - accesskey – hot key
            - tabindex – the order of advancement when the tab key is used.

RESET BUTTON

- Use to clear the form and set fields to their default value – generally restarts the form.
    - <input type=”reset”>
        - Attributes for both buttons are the same
            - type
            - name
            - id
            - value
            - accesskey – hot key
            - tabindex – the order of advancement when the tab key is used.
- Let us see this in action.
    - First, locate the form.html file in the 9.1 folder. Right-click and open with Notepad. This is the reference file.
    - Using your editor, open the template.html file from the Module2 folder and Save As form1.html in the Module9 folder.
    - ***ADD/EDIT*** the code in form1.html using the code from the reference file.
    - Save and open it in a browser.
    - If you were successful, when you enter an email address and ‘Sign me up’ you should see the email address at the end of the URL in the address bar.
- This is a quick and short, but good look at a basic form. All forms begin with these simple attributes.

CHECK BOX

- Used to select two or more options. Can also be cleared by unchecking
    - <type=”checkbox>
- Attributes
    - type
    - name
    - id
    - checked – checks the box by default
    - value – used for client-side/server-side processing
    - disabled – disables form control
    - readonly – control is active but cannot be changed from initial value
    - autofocus – HTML5 – places the cursor at this point upon access to form

    - required – HTML5 – explained previously
    - accesskey – hot key for access (shortcut key)
    - tabindex – the order of advancement when the tab key is used.

RADIO BUTTON

- Allows for a single selection from two or more options.
    - <input type=”radio”>
- Attributes
    - Same as checkbox

HIDDEN INPUT CONTROL

- Stores information that is not visible in the viewport.
- Can be accessed by client/server scripting
    - <input type=”hidden” name=”sendto” id=”sendto” value=email@some.com>
    - Can be used as the email address that is used to submit comments, contact info, etc.
- Attributes
    - type
    - name
    - id
    - value
    - disabled – a form control is disabled.

PASSWORD BOX

- Same as the text box with the exception that characters entered in the field are masked, usually by asterisks.
    - <input type=”password”>l
- Attributes
    - Same as text box

SCROLLING TEXT BOX

TEXTAREA ELEMENT

- Similar to the text box but adds scrolling action.
- The textarea element defines the start and stop of the scrolling text box.
- The height and width can be tailored for specific needs.
    - <textarea name=”comments” id=”comments” cols=”40” rows=”2”>Enter your comments here</textarea>
        - cols – specifies the width based on the number of characters that can be entered, including spaces.
        - rows – specifies the height based on the number of characters stacked
- Attributes
    - name
    - id
    - cols
    - rows
    - maxlength – number of characters that can be entered
    - disabled

    - readonly
    - autofocus – HTML5
    - placeholder – HTML5 brief information to help the user
    - required – HTML5
    - wrap – hard/soft – configures line breaks based on the number of characters enters.
    - accesskey – HTML5
    - tabindex
- Let us create a comments form
    - First, locate the form.html file in the 9.2 folder. Right-click and open with Notepad. This is the reference file.
    - Using your editor, open the form.html file in the 9.1 folder and Save As form2.html in the Module9 folder.
    - ***ADD/EDIT*** the code in the form2.html file using the code in the reference file.
    - Save and open the file in your browser.
    - Add content in the fields and click the Contact button. You should see the information at the end of the URL in the address bar.
    - ***follow book

SELECT LIST

- This is also known as a drop-down menu, drop-down option box, etc.
- Let us you select from a list of values

SELECT ELEMENT

- This is the container element that holds the values.
- Uses open/close tags
- Attributes
    - name – alphanumeric – no spaces
    - id – alphanumeric – no spaces
    - size
        - 1 – sets the visible selection to one line and acts as a drop-down list
        - 2 or more – make it a scrolling list. Best to use if the visit needs to make more than selection (ctrl key)
    - multiple – default is one selection – Let us visitor select multiple options.
    - disabled – The is used when the first option is an input statement “Select Options Here”
    - tabindex

OPTION ELEMENT

- This is the list in the Select Element
- Uses opening/closing tags
    - <option></option>
- Attributes
    - value – is the placeholder for the selected option. Used when the form is trigger.
    - Selected
        - is the first option by default
        - can be any option in the list that shows first.
    - disabled

        - Note – This not a good option for navigation links. Best uses to select a value for processing

IMAGE BUTTONS AND THE BUTTON ELEMENT

- These two objects can add a bit of pizazz to your forms by creating custom buttons using CSS3

IMAGE BUTTON

- This is an image that replaces the standard submit button.
    - <input type="image" src="login.gif" alt="Login Button">
        - This is an actual image that when clicked will trigger the method="" and the action=""

BUTTON ELEMENT

- This is element will allow a button or a selected portion of text to be the action point. This is not as common to configure as the standard submit button.
- Attributes
    - type – submit, reset, button
    - name
    - id
    - alt – for accessibility
    - value – this is the value passed to the form handler

ACCESSIBILITY AND FORMS

- We will look at methods and elements that make the use of forms easier for vision and mobility impaired visitors

LABEL ELEMENT

- This is a container tag (<label></label>) that associates text with a form control.
- This makes our impaired visitors a broader area to click than just a small button.
- Two methods to associate labels with form controls.
    - First method – places the label element as a form control. It gives the visitor a large clickable area.
        - <label>Email: <input type="text" name="email" id="email"></label>
    - Second method – Use the "for" attribute to associate.
        - <label for="email">E-mail</label>
        - <input type="text" name="email" id="email:">
            - The (for="email") is the association for the (id="email")
- Let us see this in action
    - First, locate the form.html file in the 9.3 folder. Right-click and open with Notepad. This is the reference code.
    - Open the form.html in the 9.2 folder with your editor and Save As form3.html in the Module9 folder.
    - ***ADD/EDIT*** the form3.html file using the code from the reference file.
    - Save and open it in a browser. It should look similar to the form.html file in the 9.3 folder when opened in a browser.

FIELDSET AND LEGEND ELEMENTS

- This is a technique to give your webpage a bit of clarity and separation.
- It also provides a visual clue to the form because of the legend element
- Let us see this in action.
  - First, locate the form4.html in the 9.4 folder. Right-click and open with Notepad. This is the reference file.
  - Using your editor, open the form.html file in the 9.3 folder and Save As form4.html in the Module9 folder.
  - ***ADD/EDIT*** the form4.html file with the code from the reference file.
  - Save and open it in a browser. It should show Customer Information as part of the border enclosing the name and email fields.
  - Now, locate the form5.html file in the 9.4 folder. Right-click and open with Notepad. This is the reference file for this portion.
  - Next, save the form4.html as (Save As) form5.html in the Module9 folder.
  - ***ADD/EDIT*** the css and html code in the form5.html file using the code from the reference file.
  - Save and open it in a browser. It should have a red border enclosing the name and email fields.
- This exercise demonstrates how the fieldset and legend elements place a form that is descriptive but appealing because the legend offers a “no brainer” clue to the forms purpose.

THE TABINDEX ATTRIBUTE

- This is for users that must use the tab rather than a mouse.
- They tab through the webpage
- It will place the cursor anywhere inside the webpage based on the number 1 to n that is assigned to the tabindex attribute of the element.
- This can also be used with anchor tags to rotate through hyperlinks, simply hit the Enter key to execute.
  - <input type=”text” name=”Email” id=”Email” tabindex=”1”>
- If the tabindex is set to zero (0), this element will be the last visited.

THE ACCESSKEY ATTRIBUTE

- This codes a character on the keyboard as a “hot key” that will move the cursor to the field when the hot key is pressed.
  - <input type=”text” name=”email” id=”email” accesskey=”E”> the would be a two-key combination Shift+E
  - This is not completely reliable as the hot key might be a reserved key combination for the OS.
    - ***Note – It is imperative to thoroughly test the hot key combinations to ensure it is not used elsewhere.

STYLE A FORM WITH CSS

- Developers of old would use tables to format forms and form fields.
- Tables as forms is truly outdated and CSS is the better way to style a form.
- The box model is used as the framework for styling forms.
- A good example of a CSS styled form can be seen by loading the formcss.html file in the Module9 folder.
- After the formcss.html file is opened in a browser, view the source to see the CSS and the HTML code. Take a moment to stare and compare the coding with the results.

- Server-side Processing
  - Server-side processing consists of a request from a web browser, the request is processed, and the server returns the requested data/information.
  - Examples of server-side processing scripts/languages are:
    - .php – open source sql scripting platform
    - .asp – Microsoft's active server processing language
    - Ruby on Rails
    - Adobe ColdFusion
    - Oracle JavaServer
  - Today's server side scripting languages use direct execution
    - The script is run on the web server or an extension of the web server.
  - The processing is triggered by an attribute on the form and an action call.
  - Let us see this in action.
    - Locate the contact.html file in the 9.5 folder. Right-click and open with Notepad. This is the reference file.
    - Using your editor, open the formcss.html file in the Module9 folder and Save As contact.html in the Module9 folder.
    - ***ADD/EDIT*** the code in the contact.html file with the code from the reference file.
    - Save and open it in a browser. When you enter information in the fields and Submit, you should receive a response from the server.
    - ***Note – The server will not return any input information. This is a server that has been online for long time. It is strictly used for demo purposes and does not store any data. If you get the confirmation page, the code is correct.
    - If you do not get the confirmation, compare your code with the example code from the contact.html file.

PRIVACY AND FORMS

- Privacy of personal information is at the forefront of any Internet activity, especially when submitting personal information.
- When developing webpages that request information from the visitor, make sure you some type of policy in place. This will visitors an understanding of what happens to information they submit.
- Example privacy policy template can be found at
  - https://www.bbb.org/article/news-releases/21390-bbb-tip-writing-an-effective-privacy-policy-for-your-small-business-website

- For those of you that want to learn and use server-side processing, I recommend using
  - WAMP – Windows, Apache, MySQL, PHP. This is an open-source platform for Windows. It installs all the packages needed to run a webserver on your computer. It uses minimal resources and is great for writing and testing server-side scripts.
    - *** I use WAMP before I publish my scripts.
    - https://www.wampserver.com/en/download-wampserver-64bits/
    - or https://ampps.com/download
  - LAMP – This the same as WAMP except it is for Linux.
    - https://ampps.com/download
  - MAMP – For Mac
    - https://ampps.com/download

HTML5 FORM CONTROLS

- HTML5 offers numerous new controls for developing interactive forms.
- Older browsers that do not support HTML5 will display the input types as text boxes and ignore unsupported elements.

E-MAIL ADDRESS INPUT

- The email address input control checks to make sure the information entered is in appropriate email format.
- Open the email.html in the Module9 folder.
  - Type a few characters and click Sign Me Up!
  - You should get an error message with a stylish looking alert.

URL INPUT

- This input type is much like the email input type.
- It verifies the URL that is entered has the correct formatting.
  - Open the url.html file in the Module9 folder.
  - Enter a few random characters and Send Form!
  - You will get an alert similar to the email alert.
  - Try different combinations
    - google.com
    - www.google.com
    - http://www.google.com

TELEPHONE NUMBER INPUT

- This is an input element with the type of tel
  - <input type=”tel”>
- It is basically the same as the email and url.
- Checks for proper input format.

SEARCH FIELD INPUT

- Similar to a text box but has the type=”search”.
- Open search.html in the Module9 folder

DATALIST FORM CONTROL

- This opens a full list of the options when clicked in the text box.
- Open color.html in the Module9 folder

SLIDER FORM CONTROL

- Uses the input=”range” and allow the user to set a range on a slider.
- Open slider.html in the Module9 folder.
- There are three attributes
  - min – the lowest number in the range
  - max – the highest number in the range
  - step – incremental numeric step value

SPINNER FORM CONTROL

- Creates an alert if the value entered is not numeric

- Uses the same attributes of the text box plus max, min, step

CALENDAR FORM CONTROL

*****This is truly an excellent non-scripted calendar.**

- Attributes
    - date
    - datetime
    - datetime-local
    - time
    - month
    - week
- Open date.html in the Module9 folder in your browser. View the source code.

COLOR-WELL FORM CONTROL

- This is simple color palette to select color.
- It provides the hexadecimal code, the RGB, and the HSL.
- This form would be useful in picking color for your webpage.
- Let us do an exercise to create a form that will accept a name, email, rating (slider), and comments
    - First, locate the form.html file in the 9.6 folder. Right-click and open with Notepad. This is the reference file.
    - Using your editor, open the formcss.html file from the Module9 folder and Save As form6.html in the Module9 folder.
    - ***ADD/EDIT*** the code in the form6.html file using the code in the reference file.
    - Save and open it in a browser.
    - You should get a confirmation from the server if you coded the file correctly.
- This is a great demo of a server-side script.
    - give error alerts.
    - does receive a response from the server.
- In module10 we will review standards and best practices for the website development.

MODULE 10 – WEB DEVELOPMENT

- This module is intended to provide additional tools for a successful web development project.

SUCCESSFUL LARGE-SCALE PROJECT DEVELOPMENT

- Web development projects require a team of professionals that are skilled in various aspects in development.

PROJECT JOB ROLES

- Project Manager – Oversees the project phases including but not limited to:
  - Team selection
  - Maintains
    - Scope
    - Time
    - Cost
  - Excellent communication skills
- Information Architect – Clarifies the purpose and keeps the project moving down the right path.
- User Experience Designer – Makes development suggestion base on user’s interactions to the new design.
- Marketing Representative – Leads the marketing team to keep up with the latest trends and recommends changes based on various demographic variables.
- Copywriter and Editor – Oversees the preparations and evaluates copy.
- Content Manager – This person plays a key role in overseeing the content.
  - Checks for accuracy.
  - Participates in creative development
- Graphic Designer – This person creates the visual content, including but not limited to logos, images, clip art, etc.
- Database Administrator – The person will maintain the company’s databases and the enhancements for better storage.
- Network Administrator – Maintains the web server.
- Web Developer/Designer – This person or team writes the code, tests, debugs, etc.

PROJECT STAFFING

- Ensuring the project as the right people in the right places.
- Some web projects are outsourced to companies with the human resources that have the right skills for the various web development skillsets.

THE DEVELOPMENT PROCESS

- The series of phases to complete during the project timeframe.
- SDLC provides a roadmap for successfully creating a website from conception to realization.
- This involves
  - Gathering requirements
  - Planning the phases for milestones and goals.
  - Iterative changes as needed.

METHODOLOGIES

- Prototyping -
    - A working model that give the stakeholders an example of the functionality of the site. This iterative.
- Spiral Systems Development
    - The project is broken down into smaller projects.
- Joint Application Development JAD
    - This method focuses on the collaborative efforts of the team/teams of the project.
- Agile Software Development
    - A methodology that acts on responsiveness between the client and the project team.
- Organization-Specific Development Methodologies
    - In-house development and coordination of the company’s website.

CONCEPTUALIZATION

- This is taking a concept and evaluating if the concept will succeed.
- The purpose of the site should be well evaluated to provide the groundwork for developing.

ANALYSIS

- This is the critical portion of the design phase.
- We need to determine what demographics we want to reach.
    - Determine Information Topic – This will have a great deal to do with the design. How will information be used? How will it be presented?
    - Determine Functionality Requirements – What will it take to fulfill the functionality of requirements around the purpose of the site?
    - Determine Environmental Requirements – What hardware is needed? How much bandwidth do we have?
    - Determine Content Requirements
        - What do want to convey?
        - What topics are currently the “bubble?”
    - Compare the old approach – This may only be an update the content. What new benefits are there?
    - Review competitor’ websites.
        - What elements of their website are successful?
        - What is their target market?
    - Estimate Costs – Web development can get expensive. Start with the minimum resources it will take to accomplish the goals?
    - Cost/Benefit – Will the cost override the benefit”

DESIGN

- Prepare and plan the concepts, purpose, functionality of the site.
    - Choose a site organization – This means how will the site be organized?
        - Hierarchical
        - Linear
        - Random
    - Design the prototype – give the owners a brief look at what the site will do.

    - Design a page layout – Use sketches and wireframes to determine the layout.
    - Document each page – What does the page specifically supposed to do? Keep track of changes.

PRODUCTION

- Bringing the smaller pieces together.
- What platform are we developing?
- Choose an authoring tool
    - WordPress
    - Google
    - Hand-code
- Organize your site
    - How do we store elements?
    - What is the directory structure?
    - What are the naming schemes?
    - What is the directory structure?
- Develop and Test
    - It is best to break the functionality down into smaller aspects and code according.

TESTING

- Test on different browsers
- Test on different devices/platforms
- Test with different screen resolutions
- Test the bandwidth
- Test from outside the hosting location
- Test, test, test
- Create a physical document for the testing requirements
- Automated Testing
- Accessibility Testing
    - Rehabilitation Act
    - Content Accessibility
    - Perceive
    - Operable
    - Understandable
    - Robust
    - Test using more than one method.
    - Usability Testing
        - Intuitive Design
        - Ease of learning
        - Efficiency of use
        - Memorability
        - Error Frequency and Severity
        - Subjective Satisfaction

LAUNCH

- Publish the site on the World Wide Web.

MAINTENANCE

- Making the updates to keep standards intact
- Updating based on user experiences

EVALUATION

- Does the final design meet the considerations it was supposed to meet?
- What iterations can be made for feature/functional improvements?

DOMAIN NAME OVERVIEW

- Describe your business
    - come up with several names that best describe the purpose.
    - Keep the domain name short.
    - Avoid Hyphens
    - Explore different top-level domains.
        - .org
        - .net
        - .me
        - etc.
    - Brainstorm potential keywords
        - Avoid trademarked words and phrases.
    - Know the territory
    - Verify Availability through a web hosting service
- Register the Domain before someone else uses your domain.
    - Keep the registration private
- Web hosting
    - Research different hosting services
    - Virtual or physical hosting.
- Choosing a Virtual host
    - Considerations
        - cost
        - bandwidth
        - subscription features
        - tech support
        - know the hosting platform
            - Windows
            - Linux
            - Apache
- Research and document
    - Compare with other sites
    - Keep track of changes

MODULE 11 - WEB MULTIMEDIA AND INTERACTIVITY

- In this module we will discuss multimedia and interactivity on the web.
- We will perform a few examples to bring you up to speed.

PLUG-INS, CONTAINERS, AND CODECS

- Plugins are small helper applications that work with specific file types.
- If a plugin for a certain file type is not part of the browser, the user is generally prompted to download and save the file.
- Popular plugins are:
    - Adobe Flash Player
        - .swf extension – Flash players can enhance audio, video, animation, and interactivity.
    - Adobe Shockwave Player
        - .swf extension – used for high-performance multimedia.
    - Adobe Reader
        - .pdf extension – used to read .pdf files directly in the browser.
    - Java Runtime Environment
        - The JRE is used to run applications that have Java as the language.
    - Window Media Player
        - Is used to play video, streaming video, animation, and presentations.
- What is great about HTML5 is that it is plugin independent. This means that plugins are not required with HTML5 coding practices.
- When working with multimedia we need to be aware of the container (the space needed by the file extension) and the codec (the compression algorithm).
- Not all codecs are compatible with modern browsers.
- More can be found at:
    - https://blog.filestack.com/thoughts-and-knowledge/complete-list-audio-video-file-formats/
- Common video types include:
    - .mov - apple
    - .avi – Microsoft’s original format
    - .wmv – Microsoft’s streaming type
    - .mpg – Moving Pictures Experts Group (mpeg)
    - .m4v and .mp4 – QuickTime and iTunes
    - .3gp – streaming over high-speed wireless networks
    - .ogv or .ogg – open source format
    - .webm – open media file format.
- Common audio types include:
    - .wav – Microsoft’s standard for PCs
    - .aiff and aif – used on Mac computers
    - .mid – midi interface used to create digital sounds
    - .au – older type for use on Unix machines
    - .mp3 – most common for music files
    - .ogg – open source file format
    - .m4a – audio only mpeg-4 format.

GETTING STARTED WITH AUDIO AND VIDEO

- There are numerous methods that developer use to bring in multimedia.
- The easiest is to provide a hyperlink

PROVIDE A HYPERLINK

- When a visitor clicks on a link, the audio or video is loaded and played in the current window or a new window or tab.
- Let us see an example of a hyperlink for a podcast.
  - First, navigate to the index.html file in the 11.1 folder in Module11. Right-click and open with Notepad. This is our reference file.
  - Next, create a folder in the Module11 folder called podcast.
  - Next, copy/paste the podcast.mp3 and the podcast.txt to the podcast folder from the starters folder.
  - From within your editor, open the template.html file in Module2 and Save As index.html in the podcast folder.
  - ***ADD/EDIT*** the index.html code using the code from the reference file.
  - Save and open the file in a browser.

WORKING WITH MULTIMEDIA ON THE WEB

- Working with multimedia on the web can be tricky.
- Do not think that because something is on the web that it is free to use or distribute.
- The only files that you can truly publish on the web are the files you create yourself.
- You may need permission or a license to publish copyrighted files on the web.
- Audio files can get large, and you need to consider the download time.
- Keep your audio files as small as possible using a sampling rate that compresses the file but still keeps the quality.
- More on audio sampling rates can be found at:
  - https://manual.audacityteam.org/man/sample_rates.html
- The same applies to video files.
- When you create a video, keep in mind the size of the file and the final format. MP4 files are the more common and can be played on most modern browsers.
- Probably the most common site to upload your videos is YouTube. The most comprehensive instructions can be found at:
  - https://creatoracademy.youtube.com/page/home
- Let us look at Adobe Flash and the HTML5 Embed Element

FLASH

- Flash has been around for a long time.
- It is used to enhance a user's experience by adding visual effects and interactivity.
- It is extremely versatile, and more can be found at:
  - https://www.adobe.com/products/flashplayer.html
- Flash files have the extension .swf (shockwave format).

HTML5 EMBED ELEMENT

- The element is self-contained to provide a container for the external content.
- The element was not a standard until HTML5.
- Quickly navigate to the flashembedded.html file in Module11 and open it in a browser.
- You can see the text scroll across the banner. This is flash.
  - Right-click and view the code. It is rather simple to embed a .swf file.
- There are attributes for the embed element
  - src
    - This is the location of the file.
  - height/width
    - This configures the dimensions of the container.

    - type
        - This is the mime type. For flash we use
            - type=”application/x-shockwave-flash”
    - bgcolor
        - Optional – It configures the background color of the flash.
    - quality
        - Optional – sets the quality of the flash, generally set to “high.”
    - title
        - Optional – This is used for displaying the title in a tooltip or for text only or text to speech browsers.
    - wmode
        - Optional – Used to configure the transparency of the background for supporting browsers.
- Now that we have seen flash in action and the source-code, Let us add flash to a webpage.
    - First, navigate to the index.html file in 11.2. Right-click and open with Notepad. This is the reference file.
    - Next, create a folder called embed in the Module11 folder.
    - Copy/paste the lighthouse.swf file from the starters folder in Module 11 to the embed folder.
    - Open the template.html file from Module2 in your editor and Save As index.html in the embed folder.
    - ***ADD/EDIT*** the index.html code from the code in the reference file.
    - Save the file and open it in a browser.
    - See how the video zooms out.
- There are multiple sites for free flash animations. Just search for flash creating software.
- Now that you have seen the grace and ease of embedding a flash file, next we will see the HTML5 audio and video elements.

HTML5 AUDIO AND VIDEO ELEMENTS

- Welcome back. Let us explore the HTML5 Audio and Video Elements.
- These elements allow audio and video files to play natively in the browser.
- We need to be aware of the container and the codec.

AUDIO ELEMENT - <AUDIO></AUDIO>

- This requires an opening and closing tag.
- The element allows for the playing of audio files without the need for plugins.
- You may need to supply multiple formats of the file for older browsers, it is recommended to have at least two with the two being:
    - mp3 and ogg.
- Attributes of the audio element are:
    - src – file name
        - Optional -The location of the audio file.
    - type – MIME type
        - Optional - The MIME type, may be audio/mpeg or audio/ogg
    - autoplay - autoplay
        - Optional – used to play the file automatically.
    - controls - controls
        - Optional – if the controls should be displayed, this is recommended.
    - loop - loop
        - Optional – This lets the file repeat

        - preload
            - Optional – How the file is to be retrieved
                - None – no preload
                - metadata – only the metadata is downloaded
                - auto – the audio file and metadata are downloaded.
        - title
            - Optional – This should be included for accessibility.

THE SOURCE ELEMENT

- This is a self-contained element that specifies the file and the MIME type.
- Remember to code at least two sources (files) of differing formats for accessibility.
- Let us quickly look at an example.
    - Navigate to the audio.html file in Module11 and open it in your browser.
    - Notice the controls.
    - View the source code. There are two sources for the audio.
- Let us do an example.
    - First, create a folder called audio5 in the Module11 folder.
    - Next, copy the podcast.mp3, podcast.ogg, and the podcast.txt files from the starters folder to the audio5 folder.
    - Next, navigate to the index.html file in the 11.3 folder. Right-click and open with Notepad. This is the reference file.
    - Now, open the template.html file from Module2 in your editor and Save As index.html in the audio5 folder.
    - ***ADD/EDIT*** the index.html file using the code from the reference file.
    - Save and open it in a browser. You should see the controls and the two links.

THE VIDEO ELEMENT - <VIDEO></VIDEO>

- Plays natively in the browser without the need for plugins.
- This requires an opening and closing tag.
- There are several optional attributes to consider:
    - controls – controls
        - displays the video controls
    - height/width – number value
        - determines the dimensions of the container.
    - loop – loop
        - Tells the browser to continuously play the video
    - poster – file name
        - this is the image to display if the video cannot be played.
    - preload – none, metadata, auto
        - Same as the audio preload
    - title – text description
        - This is a brief text description of the video file.

SOURCE ELEMENT FOR VIDEO

- The same holds true for the video source as the audio source.
- Let us quickly navigate to the sparky2.html file in the Module11 folder and open in your browser.
- Now, look at the source code. It should look pretty much the same as the audio.
- Let us do an example.
    - Create a folder named video in the Module11 folder.

        - Next, copy the lighthouse.m4v, lighthouse.ogv, lighthouse.swf and the lighthouse.jpg files from the starters folder in Module11 to the video folder.
        - Navigate to the index.html file in the 11.4 folder. Right-click and open with Notepad. This is the reference file.
        - Using your editor, open the template.html file from Module2 folder and Save As index.html in the video folder.
        - ***ADD/EDIT*** the index.html file using the code from the reference file.
        - Save and open in your browser. You should see the video zoom out with background music.
    - Again, Let us remind ourselves that we need to be careful about the audio and video files we place on the web.
    - Make sure that if the file is not yours, that you obtain permission or see if it is under the Creative Commons License. For more information on the Creative Commons go to:
        - http://creativecommons.org
    - There are plenty of viable sources for copyrights.
    - Next, Let us dive into CSS3 and Interactivity.

CSS3 AND INTERACTIVITY

- Using CSS3 we can manipulate many elements in a webpage.
- We can:
    - create drop-down menus
        - Menus that expand when the :hover pseudo-class is used.
    - transform elements
        - This allows us to change the element
    - rotate and scale
        - We can rotate the content
        - We can scale the content
    - transition
        - Displays the content in a smoother fashion over a specified time.
- First, Let us look at creating a drop-down menu that cascades with the :hover pseudo-class.
    - First, create a folder called mybistro in the Module11 folder.
    - Next, copy the contents of the bistro folder in Module11 to the mybistro folder.
    - Now, navigate to the index.html file in the 11.5 folder. Right-click and open with Notepad. This is the reference file.
    - Next, open the index.html file from the mybistro folder with your editor.
    - Add the submenu under the cuisine <li><a> tags.
        - You will code in another <ul> and <a> tags for the breakfast.html, lunch.html, and the dinner.html. View the reference file if needed.
        - Save the index.html file.
    - Repeat the previous step for the coffee, cuisine, breakfast, lunch, dinner, directions, and contact .html files. Since visitors be navigating in and out of folders, you can right-click on the .html files in the 11.5 folder to see the reference code for each webpage.
    - Save each file after its been edited.
- Now, we will work on the cascading style sheet.
    - First, locate the bistro.css file in the 11.5 folder. Right-click and open with Notepad. This is the reference file.
    - Open the bistro.css file from the mybistro folder with your editor.
    - ***ADD/EDIT*** the styles using the code from the css reference file.
    - Save the file.

        - Open the index.html file in your browser and click on all the links. Each page should show the same menu look.
        - We can see a two-column version by opening the index.html file in the vertical folder inside the 11.5 folder. This shows the menu in the left nav area of the index.html file.
- This is how CSS3 can make you menus stand out.
- Next, we will look at the Transform property.

CSS3 TRANSFORM PROPERTY

- The transform Property
    - With the transform property we can:
        - rotate
        - scale
        - skew
        - reposition
    - There is a complete list of properties and their definitions is located at:
        - https://www.w3.org/TR/css-transforms/#intro
- For now, we will look at rotate and scale.
    - Rotate() function rotates the element in degree
        - positive value rotates to the right
        - negative value rotates to the left
        - The rotation occurs from the center of the element.
    - Scale() function resizes the element along the x axis, the y axis or both.
        - scale(0) means the element should not display.
        - scale(1) means the element is the original size.
        - scale(2) means the element is twice the original size.
        - scale(3) means the element is three time the original size.
- Let us see an example of using the rotate and scale functions.
    - Create a new folder called transform in the Module11 folder.
    - Next, copy the lighthouseisland.jpg, the lighthousemini.jpg, and the lighthouselogo.jpg from the starters folder to the transform folder.
    - Next, navigate to the index.html file in 11.6. Right-click and open with Notepad. This is the reference file.
    - Open the starter.html in the Module11 folder and Save As index.html in the transform folder.
    - Open the index.html file in your browser and leave it open. The image should look normal.
    - ***ADD/EDIT*** the index.html file using the code from the reference file. Pay close attention to the embedded styles.
    - Save and open the file in your browser. You should notice that the image is slightly rotated clock-wise.

THE CSS3 TRANSITION PROPERTY

- This property allows for smoother transitions over a specified time.
- Each transition will require values.
- The properties are:
    - transition-property
        - The property in use.
    - transition-duration
        - The time it takes to make the transition.
        - Usually in seconds.

            - Zero (0) is immediate with no transition.
        - transition-timing-function
            - This controls the speed of the transition
                - ease – default
                    - begins slow, speeds up, and ends slow.
                - linear
                    - The transition has a constant speed.
                - ease-in
                    - The transition starts slowly then speeds up to a constant speed.
                - ease-out
                    - The transition begins at a constant speed then slows down.
                - ease-in-out
                    - The transition begins slowly, speeds up, then slows down.
    - Let us work a couple of examples.
        - First, create a new folder called transition in the Module11 folder.
        - Now, locate the index.html file in the 11.7 folder. Right-click and open with Notepad. This is the reference file.
        - Next, copy the lighthouseisland.jpg, lighthousemini.jpg, and lighthouselogo.jpg from the starters folder to the transition folder.
        - Open the starter.html file in the Module11 folder and Save As index.html in the transition folder.
        - Open the index.html file in your browser. Hover over the links and see the background-color and the text change color immediately. We will make the background-color action more appealing.
        - Now, look at the embedded styles and locate the 'nav a' selector and descendant.
        - ***ADD/EDIT*** the embedded ccs so the 'nav a' selector has the transition element. View the reference file for the code.
        - Save and display. Now, the background-color fades in slowly.
    - The first one was a rather simple example.
    - Let us look at a more challenging example.
        - First, create a new folder called gallery2 in the Module11 folder.
        - Copy all the images from the starters/gallery folder to the gallery2 folder.
        - Navigate to the index.html file in 11.8 folder. Right-click and open with Notepad. This is the reference file.
        - Next, open the template.html file in Module2 and Save As index.html in the gallery2 folder.
        - ***ADD/EDIT*** the code in the index.html file using the code in the reference file.
        - Study each addition and imagine what these instructions will do.
        - Save and open in a browser.
    - See the difference in how the image gallery looks with transitions rather than immediate displays?
    - Next, we will look at HTML5 Details and Summary Elements

HTML5 DETAILS AND SUMMARY ELEMENTS

    - These two elements are used together (a widget) to show and hide information.
        - Details Element
            - Uses an opening and closing tag <detail></detail>
            - When the visitor clicks on the details element, it expands down and shows the contents of the summary element.

    - Summary Element
        - Uses an opening and closing tag <summary></summary>
        - This is the text is in the expanded element.
- Let us try an example.
    - First, create a folder called ch11details in the Module11 folder.
    - Navigate to the index.html file in the 11.9 folder. Right-click and open with Notepad. This is the reference file.
    - Use the template.html file from Module2 and Save As index.html in the ch11details folder.
    - ***ADD/EDIT*** the index.html file using the code from the reference file.
    - Save and open. It should perform by showing the summary (expanding) when a detail is click.
- In a non-support browser, the details and summaries are listed vertically without the ability to expand.
- Next, we will look at some coding languages. Do not worry, we will not do any extravagant coding.

JAVA, JAVASCRIPT, AJAX, AND JQUERY PROGRAMMING LANGUAGES

- There are several programming languages that developers often use when creating webpages.
- The four we will talk about in this section is
    - Java
    - JavaScript
    - Ajax
    - jQuery.
- Java
    - This is an object-oriented programming language (OOP).
    - It is not the same as JavaScript.
    - It can be used to create applets that can be used inside a webpage.
    - Java applets are platform independent which means they can run on any platform:
        - Windows
        - Mac
        - UNIX
        - Linux
    - Files are saved with the .class extension and is interpreted using the Java Virtual Machine (JVM)
- Java Applets
    - They use the opening and closing <object></object> tags.
    - The attributes for applets are:
        - type – Specifies the MIME type
        - height/width – specifies the dimensions of the applet
        - title – Optional – Brief text description for accessibility.
    - Each applet will require parameters as follows:
        - code – example class
        - message – the displayed text
        - textColor – the hexadecimal color of the text to be displayed
        - backColor – the background of the applet in hexadecimal.
- JavaScript
    - Used to create additional interactivity on a webpage.
    - This is a client-side scripting language
    - It is object-oriented programming

        - It requires opening and closing <script></script> tags.
        - It is an important part of manipulating the DOM (Document Object Mode).
        - We will work with JavaScript in Module12.
    - Ajax
        - It stands for Asynchronous JavaScript and XML
        - Ajax is a combination of different technologies
            - Standards based HTML and CSS
            - The DOM
            - XML
            - uses the XMLHttpRequest or data retrieval
            - JavaScript
        - Ajax is a large part of Web 2.0, which brings in the era of interactivity, social networking, etc.
        - It processes more information on the client side rather than the server side.
        - Ajax lets JavaScript communicate directly with a server to exchange data without having to reload the webpage.
        - You may have seen this in action if you type in a zip code and the city and/or the city and state appear in their appropriate fields in a form.
    - jQuery
        - jQuery uses libraries for JavaScripting applications.
        - The jQuery is an API (Application Programming Interface).
        - It can be used to:
            - enhance slideshows on a webpage
            - animate objects (move, hide, fade, etc.)
            - handle events (mouse movements and clicking)
            - manipulate a document (webpage)
            - Within Ajax
- Next, Let us explore HTML5 APIs

HTML5 APIs

- Let us expand our enhancements by looking at HTML5 APIs
    - Geolocation API
        - This allows a visitor to share their geographic location.
        - This location is derived from:
            - IP addresses
            - Wireless networks (Wi-Fi)
            - Cell towers (triangulation)
            - GPS hardware the is integrated into the device
        - You can visit the following website to see it in action:
            - http://html5demos.com/geo
    - Web Storage
        - Developers often used cookies to store information on the client side.
        - The web storage API allows for two ways to store data:
            - localStorage stores data without an expiration date.
            - sessionStorage stores data for as long as the session remains active.
        - The API lets JavaScript use the values of either storage object.
    - Offline Web Applications
        - Allows users to view documents and web applications when they are not online.

    - The local device stores the HTML, CSS, and JavaScript so the information can be accessed when offline.
    - You may sometimes see a prompt asking if you want the information available offline.
    - This is accomplished by the manifest file with the extension of .appcache.
        - Cache – list all the web resources for the app
        - Fallback – These are the files that are displayed when the device is offline.
        - Network – These are the files that are available only with an Internet connection.
- Drawing with the Canvas Element
    - This element is used to dynamically draw graphics.
        - lines
        - strokes
        - arcs
        - fills
        - gradients
        - images
        - text
    - It requires the opening and closing <canvas></canvas> tags.
    - Let us open the canvas.html file in the Module11 folder.
    - View the source code. The JavaScript is little complicated but if you take some time, you will be able to follow along the code.
- Accessibility and Multimedia and Interactivity
    - Each of the programming techniques need to be carefully scrutinized.
    - Each element should have its text equivalent when available for accessibility.
    - Research WCGA 2.0 guidelines for more information
    - Research the Rehabilitation Act, Section 508 for more information.

MODULE 12 – JAVASCRIPT AND JQUERY

- Welcome back to HTML5 and CSS3 Video Series
- This module is intended to demonstrate JavaScript and jQuery. But first, Let us jump into JavaScript.

WHAT IS JAVASCRIPT

- What is JavaScript? JavaScript is a client-side scripting language. We talked in an earlier module regarding client-side and server-side scripting.
- JavaScript allows developers a means of using a scripting language outside of a server.
- You have seen JavaScript in action many times, and maybe without knowing it.

USES OF JAVASCRIPT

- Alerts
  - JavaScript is also known to generate an alert until the user clicks something or the alert times out, again using JavaScript.
- Popup Windows
  - When a popup window appears because of certain actions within the webpage take place.
    - Could be on a timer, could be an alert if something is missing in a form, etc.
- Using Jump Menus
  - Jump menus are similar to a select list with the exception that when a choice is made, the page “jumps” to the link rather than having to click a Submit button.
- Mouse Actions
  - When you see a cascading menu, when you hover your mouse over a link and it expands and then expands again when you hover over another link.

USING JAVASCRIPT IN A WEBPAGE

- The <script></script> element is used to place scripting instructions in a webpage.
- Keep in mind that JavaScript operates within the Normal Flow discussed earlier.
- If the JavaScript is embedded in the webpage, such as an alert or popup window, the action takes place when the browser reaches the script. There is an exception in that all JavaScript that is placed in the <head> will initiate when it is called from actions within the <body>.
- Let us do an example.
  - Locate the file alert1.html in the Module12 folder. Right-click and open with Notepad. This is the reference file.
  - Using your editor, open the template.html file in Module2 and Save As alert2.html in the Module12 folder.
  - ***ADD/EDIT*** the code in the index.html file using the code from the reference file.
  - Save the file and navigate to the file and open it in a browser.
    - I used Google Chrome and the popup showed before the Coding an Alert.
    - Let us try Internet Explorer. IE should have loaded the header, then the alert, then the <h2> text.
- As with any language there is a need to learn how to debug the language.

DEBUGGING

- Let us look at debugging JavaScript.
- When an error occurs with JavaScript there may not be any visible errors except that the script did not perform.
- One way to troubleshoot scripting errors is with the built-in console that most browsers have within.
  - Google – Right-click in the webpage and select “Inspect.”
  - Internet Explorer – Right-click and select “Inspect element.”
  - FireFox – From the webpage menu select “Browser Console.”
- Another method is to create errors and then see what happens.
- However, nothing beats the line by line search for syntax errors.
- Now we are going create an error and see what it looks like.
  - Open the file alert2.html if it is not already open.
  - Next, Let us place an extra “a” in alert… aalert and save.
  - I am going to open it with Chrome first.
  - The alert does not execute. Let us open the console. FireFox will show a similar console as Chrome.
    - ***Right-click in the window and select Inspect.
    - Now click the Console tab.
    - It shows where the error occurred and the reason, aalert is not defined. It also shows the error is on line 11.
    - Using Internet Explorer, we see a similar result. However, IE lets us step through the code.
    - Keep in mind that the console may state the error is one line above or below the actual error. Take time to examine all the relevant code.
    - ****Play with this a bit.
- Now, go ahead and remove the additional “a” and save the file and close.
- In the next video, we will discuss the Document Object Model.

DOCUMENT OBJECT MODEL

- Let us discuss the Document Object Model, or DOM
- JavaScript is an object-oriented language meaning that each element within a script is an object. Which means that images, form fields, alerts and popups are all objects. These are just a few stated as examples.
- This gives us a more natural way to code based on how the object would react if it were “real.”
- Let us say your object is a car. A car can have a color, a model, a manufacturer, two or four doors, etc.
  - We could have commands like:
    - car.color
    - car.model
    - car.manufacture
    - etc.
  - Each of these are attributes of the object car. The more attributes can make the object more defined.
- When creating an alert, we are using the window object and the method is the alert() method.
  - window.alert(“Click to continue”);
- We can also write to the document using the write() method.
- An example may be:

    - document.write("What is JavaScript and Methods");
    - The document is the webpage, the object.
  - Let us do an example
    - Navigate to the write.html file in the 12.2 folder. Right-click and open with Notepad. This is the reference file.
    - Next, using your editor, navigate to the 12.1 folder and open alert.html and Save As write.html in the Module12 folder.
    - Remove the alert between the script tags.
    - ***ADD/EDIT*** the code in the write.html file using the code from the reference file.
    - Save, navigate to the file, and open with your browser.
    - It should look like the write.html file in the 12.2 folder when opened in a browser.
    - If not, use the debugging technique or do a line by line scan to see if there is a tag missing or punctuation missing.
    - Remember that all script lines will end with a semi-colon.
    - Double-check your spelling and case-sensitive words like LastModified.
  - Next, we will explore events and event handlers

EVENTS AND EVENT HANDLERS

  - An event in an action. This could be user initiated or could be initiated by a script.
  - The event is executed by an event handler, which is the action that takes place.
  - Some of the more popular events and their handlers are:
    - click – onclick – An action takes place with the visitor clicks on an object. Remember that an object can be a link, an image, a button, etc.
    - load – onload – The browser displays a document, which could be another webpage, a document in another tab, etc.
    - mouseover – onmouseover – The event takes place with the cursor is hovering over an object.
    - mouseout – onmouseout – This event takes place with the cursor is removed from the object.
    - submit – onsubmit – This event takes place with the visitor click on a button, perhaps the submit button on a form.
    - unload – onunload – The event will close a webpage before opening another.
  - Let us do an example using the onmouseover and onmouseout event handlers.
    - Navigate to the mouse.html file in the 12.3 folder. Right-click and open with Notepad. This is the reference file.
    - Next, open a new document and Save As mouse.html in the Module12 folder.
    - ***ADD/EDIT*** the mouse.html code using the code from the reference file.
    - Save the file and open it your browser.
    - Hover over the links. See how the alerts are activated depending on which link you hover to?
  - Using these two methods are quite useful to add a bit of flare.
  - Next, we will add variables to the mix.

VARIABLES

  - Variables are placeholders for values.
  - Strings are the alphanumeric characters that make the variable.
  - Strings are always enclosed in double quotes " ".
  - There is a limit to the length a literal string (phrase, paragraph, full text) of up to 65,536 characters (including spaces and line breaks) but this truly depends on the memory needed to parse such large strings.
  - There are numerous keywords or reserved words that cannot be used as variables.

- Keywords or reserved words can be found at https://www.w3schools.com/js/js_reserved.asp
- We must initiate a variable before a script can use it.
- Its syntax is relatively simple.
    - var = someWord;
    - someWord = “This Word”;
        - The phrase
- Let us do an example of using a variable.
    - First, navigate to the var.html file in the 12.4 folder, right-click and open with Notepad. This is our reference file.
    - Next, open a new document and Save As var.html in the Module12 folder.
    - ***ADD/EDIT*** the var.html file using the code from the reference file.
    - Be sure to get the syntax correct, double-check your work.
    - Save and open the file in your browser. It should look like the var.html file in the 12.4 folder when opened in a browser.
    - You see that Karen is inserted in the document as a var called userName and was written to the webpage with the document.write() method.
    - Also note that we had to place all of this in between the <h2> tags. This is not the best way to add a header. We can use the <h2> header inside the write() method by concatenation.
    - Concatenation is the combining of one or more characters using a plus + sign, not to be mistaken for an arithmetic function.
    - Close the reference var.html file.
    - Navigate to the var2.html file in the 12.4 folder. Right-click and open with Notepad. This is our reference file for the next example.
    - Let us go back to our var.html file. It should still be open.
    - Save As var2.html in the Module12 folder.
    - ***ADD/EDIT*** the var2.html file using the code from the reference file.
    - Notice the + signs and the quotes.
        - The + signs are combining, concatenating, the strings in quotes.
        - One thing to keep in mind is that spacing before and after the plus sign needs to be inside the quotes if spacing is needed. If not, the output will run together.
    - Save the file and open it in your browser.
    - It should look like the var2.html file in the 12.4 folder when opened in a browser.
    - Next, we will create a prompt for you to enter your name and display it on the webpage.
    - First, navigate to the var3.html file in the 12.5 folder. Right-click and open with Notepad. This is the reference file.
    - Next, open (if it is not already open) the var2.html file you just made in your editor. Save As var3.html in the Module12 folder.
    - ***ADD/EDIT*** the var3.html file using the code from the reference file.
    - Save and open it in your browser. You should get the prompt to enter your name. Type in your name and click Enter. You should see your name in the webpage.
- Let us have the user change the background color for the webpage.
    - First, navigate to the var4.html file in the 12.5 folder. Right-click and open with Notepad. This is the reference file.
    - Using the var3.html file you just made, Save As var4.html in the Module12 folder.
    - ***ADD/EDIT*** the code in the var4.html file using the code from the reference file.

  - Save the file and open it in your browser. You should be prompted to type in red or blue. You can try a different color by name, aqua, brown, tan, yellow, etc.
  - This little script is a keeper because you can use hexadecimal and rgb(xxx.xxx.xxx) color codes.
  - Try #44DD11 and see what you get. Just refresh the page to get the prompt back.
  - This would be good to use when creating color schemes for foreground against background colors.
- This is just a minor use of JavaScript. It can get much more involved.
- Next, we will talk about Programming concepts.

PROGRAMMING CONCEPTS

- Let us explore an introduction to Programming Concepts.
- Programming using JavaScript is a very powerful tool and we will work a few examples.
- The three main aspects of programming are:
  - Arithmetic
  - Decision Making
  - Comparison
  - Functions
- We first need to understand how each works.
  - Arithmetic
    - = equal sign – is used to assign a value.
    - + plus sign – is used to add two or more values.
    - – minus sign – is used to subtract two or more values.
    - x multiplication sign – is used to multiply two or more values.
    - / division sign – is used to divide two or more values.
  - Decision Making
    - Decisions are based on the “if” statement.
    - If a condition is true, do this action, if the condition is false, do this alternate action.
    - It is written as if (condition) {true, do this}
    - else
    - {do this alternate};
  - Comparison
    - == double equal sign – is exactly equal to.
    - > greater than sign – is one value greater than another value.
    - >= great than or equal signs – is one value greater than or equal to another value.
    - < less than sign – is one value less than another value.
    - <= less than or equal signs – is one value less than or equal to another value.
  - Function
    - Simply stated, a function is a set of actions that take place when a user interacts with the page.
    - This could be initiating a prompt, an alert, or insert data base on a form (we will talk about this later).
  - Each of these programming aspects are pretty much universal for all languages, although the syntax of how each is used may differ slightly.
- Let us do an example.

    - First, navigate to the if.html file in the 14.6 folder. Right-click and open with Notepad. This is the reference file.
    - Next, open a new document in your editor and Save As quantityif.html in the Module12 folder.
    - ***ADD/EDIT*** the quantityif.html code using the code from the reference file.
    - Save and open it in a browser and enter 0, 5, -1 and see the response. You will need to refresh the browser to get the prompt back.
- Now that we have a sense of arithmetic, decision making, and comparisons, Let us explore Functions.
    - Functions are generally placed in the <head> tags because we do not want the function to be part of the normal flow.
    - The function needs to be loaded before any actions can take place where the function is called.
    - We want the function to be called or invoked when the visitor performs some action.
    - Functions generally are called when a button is clicked. The three types are:
        - Submit
        - Reset
        - Button – remember these types of buttons do require an event handler.
    - Functions are coded as:
        - function function_name() {
            - JavaScript statements;
            - }
        - They do require the use of braces.
- Let us do an example. I think we can cut down on the typing for the remainder of this module.
    - Navigate to the if2.html file in the 12.7 folder. Right-click and open with Notepad. This is the reference file.
    - Using your editor open the template.html file in the Module2 folder and Save As if2.html in the Module12 folder.
    - ***ADD/EDIT*** the if2.html code using the code in the reference file.
    - Save the file and open it in a browser.
    - It should perform with a button to click, a prompt, and then an alert.
    - Notice the button is of type=”button” which means it needs an event handler to initiate an action. In this case, to call the promptQuantity() function.
    - Also take notice of the //end if and // end function promptQuantity. These are comments so developers know when this function has ended.
- Are you starting to see the versatility of functions?
- Next, we will have a look at form handling using JavaScript.

FORM HANDLING

- In this section we will discuss Form Handling.
- Form handling has its own set of rules. However, using JavaScript can save time by validating the data in the form on the client-side rather than having to do the check on the server-side.
    - Earlier in this series, we created a couple of form examples and used a submit button. The form sent the data to the action=”” which is sent to the server.
    - One other item. Functions can return values back to the source if needed by using the “return;” keyword. We will see this in use shortly.
    - However, now we want to validate the data before it is sent.

- We are going to do both so you can see how the validation works before and after the action.
    - First, navigate to the form.html file in the 14.8 folder. Right-click and open with Notepad. This is the reference file.
    - Next, open a new document in your editor and Save As form.html in the Module12 folder.
    - ***ADD/EDIT*** the form.html code using the code from the reference file.
    - Save the file and open it in a browser. Since there is nothing to perform, the server will send back an acknowledgement, but display no data.
    - Now, we will add some <script>.
    - Since there can be multiple forms on a webpage, we need to distinguish which form is the one being used.
    - We do this by numbering the forms in the script.
    - document.forms[0], which indicates it is the first form.
    - document.forms[1] is the second form and so on.
    - Just remember that in coding, objects that need numbered start with 0.
- Let us validate the userAge first.
    - Enter the userAge “if” statement first. Place this between the <script> tags in the head tags as seen in the reference file. It should be inserted below the </style> and above the </head> tags.
    - Save the file and open it in a browser.
    - We see the same form, but now when we input the age, it will validate the age. Either greater than or equal to 18 or less than 18.
    - If you look at this “if” statement, you will see the statement.
    - The document is the webpage object, the forms[0] is the object of the document, the userAge is the attribute of the form, and value is the value from the input field userAge.
- Now, Let us validate the userName.
    - Now, Let us add the other “if” statement to validate the userName. This validation is to make sure there is something in the field (true) or empty (false).
    - ***ADD/EDIT*** the code using the code from the reference file.
    - Save and open it in a browser.
    - Now we can play with this and see what happens.
- That is how we can validate the data before it is sent to the database. This frees up resources on the server, which is always a good thing.
- One thing to keep in mind is Accessibility.
- When using JavaScript, you need to also code plain text navigation for text-only and text-speech browsers.
- Now that we have a basic understanding of JavaScript, Let us check out jQuery in the next section.

JQUERY

- Welcome back. Let us jump into jQuery.
- jQuery is a form of JavaScript and is used in conjunction.
- Sites like Amazon use jQuery to provide an enhanced and more functional experience for the visitor.
- jQuery is an open-source library with many, many scripts. Developers use jQuery libraries rather than code and test their own scripts. This is time saving.
- Also, developers can customize these library scripts for their individual purposes.
- The library is stored in a .js file. Before we proceed, you will need to download the jQuery library at:

- http://jquery.com/download and right-click on the "Download the compressed, production jQuery 3.2.1 and save the file as jquery-3.2.1.min.js in the Module12 folder.
- jQuery is available locally by using this script in the <head> tag
- <script src="jquery-3.2.1.min.js"></script>
- You can also use jQuery from a repository such as Google, Microsoft, Amazon, to name a few.
- This is called a CDN, Content Delivery Network.
- To access Google's repository the code is:
- <script src=https://ajax.googleapis.com/libs/jquery/3.2.1/jquery.min.js> </script>
- jQuery can be coded as jQuery (selector).method() or using the $ sign like $ (selector).method().
- The (selector) is the DOM element to work with.
- jQuery needs to have the webpage loaded before it can operate.
- A ready() method is used to tell jQuery the page is loaded and ready of interaction.
- JavaScript and other jQuery statements are inside the ready() method.
- An example might be:
  - $ (document).ready(function() (alert("Ready for jQuery");));
- Notice the number of parenthesis in this statement. Syntax is extremely crucial, so pay close attention to opening and closing parenthesis.
- Let us try an example.
  - First, navigate to the ready.html file in 14.9. Right-click and open with Notepad. This is the reference file.
  - Next, open a new document and Save As ready.html in the Module12 folder.
  - ***ADD/EDIT*** the ready.html code using the code from the reference file.
  - ***Notice that we are using the CDN from Google's API library.
  - Save the file and open it in a browser. You will see an alert stating Ready of jQuery. This means the page is fully loaded and ready for jQuery.
  - Comment out the script and see what happens:
    - <!-- <script> …. </script> -->
  - You will see that the alert did not execute because the jQuery library was not accessed and therefore the ready() method cannot be initiated to state that the page is loaded.
  - Remove the comment tags and save the file and refresh.
- Next, we will talk about jQuery Selectors and jQuery Methods.

JQUERY SELECTORS AND JQUERY METHODS

- Let us see how selectors and methods operate in jQuery.
- jQuery selectors are the same elements used in HTML and CSS.
- We will want to use single quotes when working with selectors rather than double-quotes.
- All selectors will be enclosed in parenthesis
  - Selectors are coded the same as CSS statements with the exception of the $ and the parenthesis.
    - CSS - .className {} – for the class named className.
    - jQuery - $('.className')
    - CSS - * {} - Wildcard for all elements.
    - jQuery - $('*')
    - CSS – nav a {}
    - jQuery - $('nav a')
- A list of jQuery selectors can be found at:
  - http://api.jquery.com/category/selectors
- jQuery methods are used to act upon a DOM element(s) that are selected.

- Methods work with CSS, Effects, Events, Forms, etc. There are too many to list, but additional methods can be found at:
    - https://www.w3schools.com/jquery/jquery_events.asp
- Some of the more popular methods are:
    - attr() – gets or sets attributes for the selected elements.
    - click() – Binds a jQuery event to the JavaScript event handler.
    - css() – sets CSS properties for the selected element(s).
    - html() – Gets or sets content for the selected element(s).
- Let us refer to the color.html file in Module12 folder.
    - Open it in your editor and see the jQuery inside the <script></script> tags.
    - When the page is loaded, jQuery is notified the document will make all lists use a css for the text color of #FF0000.
- Now, Let us refer to the file changeme.html in the Module12 folder.
    - Open it in your editor and see the jQuery this time.
    - even one… all start with zero as zero is even in coding practices.
- Let us work an example. We will use the toggle() and click() methods.
    - First, navigate to the toggle.html file in the 12.10 folder. Right-click and open with Notepad. This is our reference file.
    - Next, open a new document in your editor and Save As toggle.html in the Module12 folder.
    - ***ADD/EDIT*** the toggle.html code with the code from the reference file.
    - Save the file and open it in your browser.
    - When you click on ‘More”, you will see additional content.
- We can also change how the “more” content is displayed.
    - Locate the fadetoogle.html file in the 12.10 folder. Right-click and open with Notepad. This is the reference file for this next exercise.
    - Edit the toggle.html file by a Save As fadeToggle.html.
        - Replace the $(‘details’) jQuery selector with the code in the reference file.
    - Save the file and open in a browser.
    - Locate the slidetoggle.html file in the 12.10 folder. Right-click and open with Notepad. This is the reference file.
    - Edit the fadeToggle.html and Save As slideToggle.html.
        - Replace the $(‘details’) jQuery selector with the code in the reference file.
    - Save the file each time and open it in a browser.
- jQuery is quite the powerful tool and offers a lot of flexibility for creating dynamic webpages.
- Next, we will look at how jQuery can work images.

JQUERY IMAGE GALLERY

- We have seen how useful jQuery is with DOM elements and CSS.
- Let us have a look when we use jQuery with images.
- There are several methods that can enhance how image galleries are displayed on the webpage.
- We can hover:
    - Hovering over an image will initiate an action. Some actions may be:
    - Full-sized image in another window or tab.
    - Replace an image with the hovered image.
- We can toggle:
    - Toggling the image by clicking on a thumbnail.
    - Toggling allows us to change the speed and effect of the displayed image.
- We can click:

    - Click on an image to open a new window or tab.
    - Click will load the image from a thumbnail to its larger version.
- In this example we will use the hover(), html(), and attr() methods.
- mouseenter and mouseleave – these work with the elements to perform an action with the hover() method.
- This is a long example so grab a snack and a refreshment.
    - First, navigate to the hover.html file in the 12.11 folder. Right-click and open with Notepad. This is the reference file.
    - Create a folder called gallery12 in the Module12 folder.
    - Next, copy all the images from Module11/starters/gallery folder to the gallery12 folder.
    - Now, navigate to the template.html file in Module2 folder. Save As hover.html in the gallery12 folder. This file has to be in the gallery12 folder to use the images.
    - ***ADD/EDIT*** the hover.html code using the code from the reference file.
    - Save the file and open it in a browser.
    - The large image changes when hovering over the thumbnails.
    - Next, Let us navigate to the click.html file in 12.11. Right-click and open with Notepad. This is the reference file.
    - Save As the hover.html to click.html.
    - ***ADD/EDIT*** the click.html code using the code from the reference file.
    - Save the file and open it in a browser. The large image changes when a thumbnail is clicked.
    - Next Let us make the images fade in slowly using the fadeToggle() method.
    - Navigate to the toggle.html file in 12.11. Right-click and open with Notepad. This is the reference file.
    - Save As the click.html to toggle.html.
    - This time we will leave the click() method and add a css and a fadeToggle for 1000 milliseconds.
    - ***ADD/EDIT*** the toggle.html code using the code from the reference file.
    - Save the file and open it in a browser. When we click on a thumbnail image, the large image fades into view.
- Using jQuery for an image gallery is quite cool. And, there are numerous options available.
- Next, we will explore using jQuery plugins.

JQUERY PLUGINS

- Let us see with jQuery plugins can do for us.
- Plugins extend the functionality of jQuery.
- They are configured to work with elements and objects outside the normal jQuery methods.
- Let us look at a plugin for creating a slideshow.
- We will create a slideshow using the fotorama plugin for jQuery.
    - First, create a folder called slideshow in the Module12 folder.
    - Next, copy/paste all the non-thumbnail images from the 12.12 folder to the slideshow folder.
    - Next, navigate to the index.html file in the 12.12 folder. Right-click and open with Notepad. This is the reference file.
    - Open the template.html from the Module2 folder in your editor and Save As index.html in the slideshow folder.
    - ***ADD/EDIT*** the index.html code using the code from the reference file.
    - Save the file and open it in your browser.

    - The slideshow is set for about 5 seconds, but you can hover inside the image and click the arrows that appear.
- Next we will use a validation plugin to validate the data in a simple form.
    - Create a folder called jform in the Module12 folder.
    - Next, open the file index.html from the 12.13 folder. This is our reference file.
    - Next, from your editor open the file formstarter.html from the Module12 folder and Save As index.html in the jform folder.
    - ***ADD/EDIT*** the index.html code using the code from the reference file.
    - Save the file and open it in a browser.
    - You can see that if the form was coded correctly, you should be able to enter the information and when the ‘Sign Up’ button is clicked, you should see a response from the server.
- JavaScript and jQuery are powerful tools for the developer and even more powerful if combined.
- There are dozens of resources for JavaScript and jQuery.

REFERENCES

Felke-Morris, D. T. (2019). *Web Development and Design Foundations with HTML5 - 9th Edition.* New York: Pearson.

Mozilla. (2020, n.d. n.d.). *HTML elements reference*. Retrieved from MDN web docs: https://developer.mozilla.org/en-US/docs/Web/HTML/Element

W3C. (2020, n.d. n.d.). *CSS Tutorial*. Retrieved from w3schools.com: https://www.w3schools.com/css/default.asp

W3C. (2020, n.d. n.d.). *HTML Semantic Elements*. Retrieved from w3schools.com: https://www.w3schools.com/html/html5_semantic_elements.asp

W3C. (2020, n.d. n.d.). *HTML Symbols*. Retrieved from w3schools.com: https://www.w3schools.com/html/html_symbols.asp

W3C. (2020, n.d. n.d.). *HTML Tutorial*. Retrieved from w3schools.com: https://www.w3schools.com/html/

W3C. (2020, n.d. n.d.). *JavaScript Tutorial*. Retrieved from w3schools.com: https://www.w3schools.com/js/default.asp

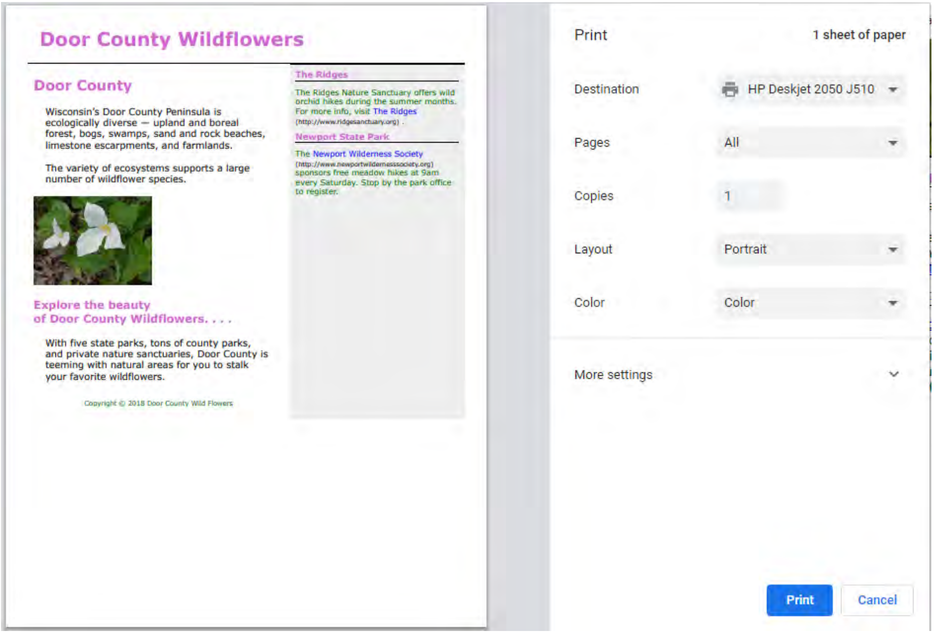
W3C. (2020, n.d. n.d.). *jQuery Tutorial*. Retrieved from w3schools.com: https://www.w3schools.com/jquery/

ADDENDUM

This addendum accounts for discrepancies between the video tutorials and this manual.

MODULE 7:

- Three-column CSS Page Layout Pt 1
  - Approximately @ 11:40 minutes
    - Save As index.html in the wildflowers3 folder, not the wildflowers folder
- Three-column CSS Page Layout Pt 2
  - Approximately @ 10:10 minutes
    - Open wildflowers3 folder, not the wildflowers folder
- CSS Styling for Print
  - Approximately @ 2:15 minutes
    - Use the index.html file from the 7.5 folder, not from the wildflowers or wildflowers3 folder.
  - Approximately @ 7:00 minutes
    - The video shows the <link> as wildflower.css. It should be wildflowers.css, the media=”screen”, not media=”sreen” - <link rel=”stylesheet” href=”wildflowers.css” media=”screen”>
  - Approximately @ 8:00 minutes
    - The video shows the <link> as wildflowerprint.css. It should be wildflowersprint.css - <link rel=”stylesheet” href=”wildflowersprint.css” media=”print”>
  - Approximately @ 8:40 minutes
    - The video states to save the external css as wildflower.css. Save the file as – wildflowers.css.
  - Approximately @ 16:10 minutes
    - .news class – The font-size should be 10pt, not 10.
  - Approximately @ 20:15 minutes
    - Instructor states “background color” for h2 and h3.
      - Instructor types in color: #FFFFFF. It should be { background-color: #FFFFFF; }
  - Approximately @ 22 minutes
    - There was a correction needed in the reference files in the 7.6 folder.
      - The reference files (index.html, wildflowers.css, and wildflowersprint.css) have since been corrected.
      - The video shows the print preview for the index.html file in the wildflowersPrint folder. This is due to the error(s) in coding for the reference files noted above.
      - The following image of the index.html file should look similar to this image for the print preview

- o 

Door County Wildflowers

Door County

Wisconsin's Door County Peninsula is ecologically diverse — upland and boreal forest, bogs, swamps, sand and rock beaches, limestone escarpments, and farmlands.

The variety of ecosystems supports a large number of wildflower species.

Explore the beauty of Door County Wildflowers. . . .

With five state parks, tons of county parks, and private nature sanctuaries, Door County is teeming with natural areas for you to stalk your favorite wildflowers.

Copyright © 2018 Door County Wild Flowers

The Ridges

The Ridges Nature Sanctuary offers wild orchid hikes during the summer months. For more info, visit The Ridges (http://www.ridgesanctuary.org) .

Newport State Park

The Newport Wilderness Society (http://www.newportwildernesssociety.org) sponsors free meadow hikes at 9am every Saturday. Stop by the park office to register.

Print 1 sheet of paper

Destination HP Deskjet 2050 J510

Pages All

Copies 1

Layout Portrait

Color Color

- Responsive Images
  - o The instructions in the manual are condensed to avoid a lengthy video. Add/edit the reference code to the index.html file(s) stated in the video.
    - This video section combines:
      - Responsive Images
      - HTML5.1 Picture Element
      - HTML5.1 Responsive IMG Element Attributes