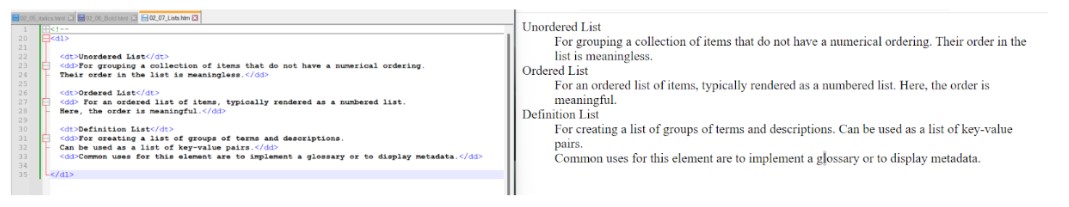
Introduction to HTML and CSS

* HTML stands for HyperText Markup Language and describes the markup of the page.
* Elements tell the browser how to display the content.
* Some HTML elements have no content ( like <br> ) element and these are called empty elements.
* Paragraphs
* Headlines
* Bold and Italics has for different elements namely <em>,<b>, <strong> and <i>
* **Two of them, "<em>" and "<strong>", convey meaning and serve a language-related purpose. The other two, "<i>" and "<b>", do not carry any specific meaning and are used solely for visual styling**
* **There are three types of list namely unordered lists, ordered lists and descriptive/definition list.**



* **<blockquote>** for block-level quotations, accompanied by **<cite>** for source attribution within a **<footer>** element
* the **<time>** element, showcasing its **datetime** attribute for machine-readable dates and times.
* It explains the proper format for conveying timing information to computers and illustrates combining date and time with various format options
* **<code>** is for displaying code, **<br>** is for line breaks, and **<pre>** is for preserving text formatting, often used for code blocks
* Subscripts, superscripts, and small text can be used where you need to mark up certain bits of content as having a different meaning than the rest.
* remember the four most important Global Attributes: "class," "id," "lang," and "dir."
* The "class" attribute allows you to assign a name for styling multiple elements with the same class, while the "id" attribute provides a unique identifier, useful for specific targeting in CSS, JavaScript, or links
* Global attributes like "lang" and "dir" work universally across elements, enabling language specification and text direction indication. Additionally, attributes like "contenteditable" allow users to edit content on a webpage, but back-end development is needed for saving changes
* ARIA Roles are like extra attributes that we can add to HTML elements to make them more meaningful and help browsers understand what they represent.
* ARIA roles supplement HTML elements to enhance accessibility, aiming to use semantic HTML for content meaning
* ARIA Roles are essential for making websites fully accessible to assistive technologies. Developed during the shift from native apps to web interfaces, they play a crucial role in ensuring that individuals with disabilities can effectively use complex interfaces, utilizing tools like screen readers and braille displays.
* The accessibility tree complements the DOM tree, crucial for screen readers. It can treat content as separate text containers, leading to suboptimal reading experiences.
* ARIA (Accessible Rich Internet Applications) can enhance this by adding labels and hiding elements
* ARIA's powerful features, like ARIA Roles, greatly improve web accessibility, making it valuable for teams dealing with complex interfaces or struggling with semantic HTML
* HTML usually ignores spaces, tabs, and line breaks.
* Exceptions include elements like **<pre>**, **<code>**, or **<textarea>**.
* CSS can modify whitespace handling.
* Extra spaces in paragraphs are usually ignored by browsers.
* Example given with extra spaces, but the browser ignores them.
* Comments enhance code readability.
* Inserted using **<!--** and **-->**.
* Browsers disregard comments; code editors may display them differently.
* In early HTML, there was a debate on uppercase or lowercase.
* Current norm is to use lowercase for HTML elements.
* Browser support is not affected by case choice.
* Short elements like **<p>** or **<i>** vs. longer ones like **<article>** or **<video>**.
* Evolution from optimizing file size in the '80s to prioritizing readability.
* Many elements have both opening and closing tags.
* Older elements (e.g., **<img>**) may lack closing tags.
* Slash (/) used to self-close, but now formatting options are flexible.
* Personal preference determines whether to use the slash.
* **<**, **>**, and **&** are crucial in HTML.
* Writing them with spaces around displays them as regular content.
* If written resembling HTML, the browser assumes code and doesn't display it.
* Character entities: ampersand + short code + semicolon.
* Used to represent special characters in HTML.
* Example: Replace **<** and **>** with **&lt;** and **&gt;** respectively.
* W3C provides a reference chart for character entities.
* Useful for typing characters not on the keyboard (e.g., copyright symbol).
* Regular spaces allow line breaks; browser wraps text where space permits.
* Non-breaking space (**&nbsp;**) prevents line breaks between words.
* Example: Keeping "Lebron" and "James" together in a sentence.
* Can also create multiple spaces between words without being ignored by the browser.
* HTML character entities, including non-breaking spaces, ensure specific character display on webpages.
* Web links are common in today's online experience, found in navigation bars, menus, teaser cards, and clickable article titles.
* The concept of linking, including hypertext and hypermedia, was a revolutionary idea in the 1980s, leading to the birth of the web in the early 1990s.
* Creating a link in HTML is done using the "A" element, which stands for anchor. The link's destination is specified using the "href" attribute with a URL enclosed in quotes.
* The term "href" stands for Hypertext Reference. Links can include text or images between the opening and closing "A" tags, making them clickable.
* Links are not limited to text; they can also be wrapped around images or more complex elements like teaser cards.
* URLs can be versatile, including the entire URL when linking to another website or a specific web page. Absolute URLs point to a precise location on the web and must include the HTTP or HTTPS part.
* HTTPS, which stands for Hypertext Transport Protocol Secure, is recommended for enhanced security in web addresses. Modern browsers often automatically add the HTTPS:// part when typing a web address.
* Absolute URLs and relative URLs are options for forming links in web development.
* Absolute URLs point to a precise location on the web and include the full domain. Relative URLs are based on the current file's location and omit the domain.
* Relative URLs are useful when working on a local copy of a website or moving between different servers with different URLs for testing.
* Creating a relative URL involves omitting the domain name and including the initial slash to start from the root level or writing the path relative to the file where the link is written.
* File organization is crucial for understanding how to create URLs. Slashes in a URL indicate navigating the file structure, and filenames have extensions while folder names do not.
* Examples of relative URLs include "/images/logo.gif" (relative to the root) and "../images/logo.gif" (relative to the current file's location).
* Web browsers automatically look for an index.html file when given a URL that points to a folder, enabling clean URL structures.
* Trailing slashes in URLs do not affect the destination; both versions with or without a trailing slash take you to the same place.
* Crafting well-structured URLs is essential for user experience and search engine results, and using relative URLs can be beneficial, especially in projects that move between servers.
* Each link (home, people, prices, contact) is wrapped in an "li" element and enclosed in a "ul" element for an unordered list.
* The entire menu is encompassed in a "nav" element to signify site navigation.
* Apply CSS styling for visual appearance.
* Enhance accessibility by assigning the role "navigation" to the "nav" element and adding an "aria label" for descriptive reading.
* Breadcrumb links are wrapped in a "nav" element and enclosed in an ordered list ("ol") as the order matters.
* Add an "aria label" of "breadcrumb" for context when read aloud.
* Unlike the main menu, do not assign the role "navigation" to this breadcrumb trail.