* **Role of HTML**: HTML represents the content and structure of the web page.
* **HTML Elements**: Elements are the building blocks for an HTML document. They represent headings, paragraphs, links, images and more. Most HTML elements consist of an opening tag (<elementName>) and a closing tag (</elementName>).

Here is the basic syntax:

<elementName>Content goes here</elementName>

* **Void Elements**: Void elements cannot have any content and only have a start tag. Examples include img and meta elements.

<img>

<meta>

It is common to see some codebases that include a forward slash / inside the void element. Both of these are acceptable:

<img>

<img/>

* **Attributes**: An attribute is a value placed inside the opening tag of an HTML element. Attributes provide additional information about the element or specify how the element should behave. Here is the basic syntax for an attribute:

<element attribute="value"></element>

A boolean attribute is an attribute that can either be present or absent in an HTML tag. If present, the value is true otherwise it is false. Examples of boolean attributes include disabled, readonly, and required.

* **Comments**: Comments are used in programming to leave notes for yourself and other developers in your code. Here is the syntax for a comment in HTML:

<!--This is an HTML comment.-->

**Common HTML elements**

* **Heading Elements**: There are six heading elements in HTML. The h1 through h6 heading elements are used to signify the importance of content below them. The lower the number, the higher the importance, so h2 elements have less importance than h1 elements.

<h1>most important heading element</h1>

<h2>second most important heading element</h2>

<h3>third most important heading element</h3>

<h4>fourth most important heading element</h4>

<h5>fifth most important heading element</h5>

<h6>least important heading element</h6>

* **Paragraph Elements**: This is used for paragraphs on a web page.

<p>This is a paragraph element.</p>

* **img Elements**: The img element is used to add images to the web page. The src attribute is used to specify the location for that image. For image elements, it's good practice to include another attribute called the alt attribute. Here's an example of an img element with the src and alt attributes:

<img src="https://cdn.freecodecamp.org/curriculum/cat-photo-app/lasagna.jpg" alt="A slice of lasagna on a plate.">

* **body Element**: This element is used to represent the content for the HTML document.

<body>

<h1>CatPhotoApp</h1>

<p>This is a paragraph element.</p>

</body>

* **section Elements**: This element is used to divide up content into smaller sections.

<section>

<h2>About Me</h2>

<p>Hi, I am Jane Doe and I am a web developer.</p>

</section>

* **div Elements**: This element is a generic HTML element that does not hold any semantic meaning. It is used as a generic container to hold other HTML elements.

<div>

<h1>I am a heading</h1>

<p>I am a paragraph</p>

</div>

* **Anchor (a) Elements**: These elements are used to apply links to a web page. The href attribute is used to specify where the link should go when the user clicks on it.

<a href="https://cdn.freecodecamp.org/curriculum/cat-photo-app/running-cats.jpg">cute cats</a>

* **Unordered (ul) and Ordered (ol) List Elements**: To create a bulleted list of items you should use the ul element with one or more li elements nested inside like this:

<ul>

<li>catnip</li>

<li>laser pointers</li>

<li>lasagna</li>

</ul>

To create an ordered list of items you should use the ol element:

<ol>

<li>flea treatment</li>

<li>thunder</li>

<li>other cats</li>

</ol>

* **Emphasis (em) Element**: This is used to place emphasis on a piece of text.

<p>Cats <em>love</em> lasagna.</p>

* **Strong Importance (strong) Element**: This element is used to place strong emphasis on text to indicate a sense of urgency and seriousness.

<p>

<strong>Important:</strong> Before proceeding, make sure to wear your safety goggles.

</p>

* **figure and figcaption Elements**: The figure element is used to group content like images and diagrams. The figcaption element is used to represent a caption for that content inside the figure element.

<figure>

<img src="https://cdn.freecodecamp.org/curriculum/cat-photo-app/cats.jpg" alt="Five cats looking around a field.">

<figcaption>Cats <strong>hate</strong> other cats.</figcaption>

</figure>

* **main Element**: This element is used to represent the main content for a web page.
* **footer Element**: This element is placed at the bottom of the HTML document and usually contains copyright information and other important page links.

<footer>

<p>

No Copyright - <a href="https://www.freecodecamp.org">freeCodeCamp.org</a>

</p>

</footer>

**Identifiers and Grouping**

* **IDs**: Unique element identifiers for HTML elements. ID names should only be used once per HTML document.

<h1 id="title">Movie Review Page</h1>

ID names cannot have spaces. If your ID name contains multiple words then you can use dashes between the words like this:

<div id="red-box"></div>

* **Classes**: Classes are used to group elements for styling and behavior.

<div class="box"></div>

Unlike IDs, you can reuse the same class name throughout the HTML document. The class value can also have spaces like this:

<div class="box red-box"></div>

<div class="box blue-box"></div>

**Special Characters and Linking**

* **HTML entities**: An HTML entity, or character reference, is a set of characters used to represent a reserved character in HTML. Examples include the ampersand (&amp;) symbol and the less than symbol (&lt;).

<p>This is an &lt;img /&gt; element</p>

* **link Element**: This element is used to link to external resources like stylesheets and site icons. Here is the basic syntax for using the link element for an external CSS file:

<link rel="stylesheet" href="./styles.css" />

The rel attribute is used to specify the relationship between the linked resource and the HTML document. The href attribute is used to specify the location of the URL for the external resource.

* **script Element**: This element is used to embed executable code.

<body>

<script>

alert("Welcome to freeCodeCamp");

</script>

</body>

While you can technically write all of your JavaScript code inside the script tags, it is considered best practice to link to an external JavaScript file instead. Here is an example of using the script element to link to an external JavaScript file:

<script src="path-to-javascript-file.js"></script>

The src attribute is used here to specify the location for that external JavaScript file.

**Boilerplate and Encoding**

* **HTML boilerplate**: This is a boilerplate that includes the basic structure and essential elements every HTML document needs.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8" />

<title>freeCodeCamp</title>

<link rel="stylesheet" href="./styles.css" />

</head>

<body>

<!--Headings, paragraphs, images, etc. go inside here-->

</body>

</html>

* **DOCTYPE**: This is used to tell browsers which version of HTML you're using.
* **html Element**: This represents the top level element or the root of an HTML document. To specify the language for the document, you should use the lang attribute.
* **head Element**: The head section contains important meta data which is behind-the-scenes information needed for browsers and search engines.
* **meta Elements**: These elements represent your site's metadata. These element have details about things like character encoding, and how websites like Twitter should preview your page's link and more.
* **title Element**: This element is used to set the text that appears in the browser tab or window.
* **UTF-8 character encoding**: UTF-8, or UCS Transformation Format 8, is a standardized character encoding widely used on the web. Character encoding is the method computers use to store characters as data. The charset attribute is used inside of a meta element to set the character encoding to UTF-8.

**SEO and Social Sharing**

* **SEO**: Search Engine Optimization is a practice that optimizes web pages so they become more visible and rank higher on search engines.
* **Meta (description) Element**: This is used to provide a short description for the web page and impacting SEO.

<meta

name="description"

content="Discover expert tips and techniques for gardening in small spaces, choosing the right plants, and maintaining a thriving garden."

/>

* **Open Graph tags**: The open graph protocol enables you to control how your website's content appears across various social media platforms, such as Facebook, LinkedIn, and many more.

By setting these open graph properties, you can entice users to want to click and engage with your content. You can set these properties through a collection of meta elements inside your HTML head section.

* **og:title Property**: This is used to set the title that displays for social media posts.

<meta content="freeCodeCamp.org" property="og:title" />

* **og:type Property**: The type property is used to represent the type of content being shared on social media. Examples of this content include articles, websites, videos, or music.

<meta property="og:type" content="website" />

* **og:image Property**: This is used to set the image shown for social media posts.

<meta

content="https://cdn.freecodecamp.org/platform/universal/fcc\_meta\_1920X1080-indigo.png"

property="og:image"

/>

* **og:url Property**: This is used to set the URL that users will click on for the social media posts.

<meta property="og:url" content="https://www.freecodecamp.org" />

**Media Elements and Optimization**

* **Replaced elements**: A replaced element is an element whose content is determined by an external resource rather than by CSS itself. An example would be an iframe element. iframe stands for inline frame. It's an inline element used to embed other HTML content directly within the HTML page.

<iframe src="https://www.example.com" title="Example Site"></iframe>

You can include the allowfullscreen attribute which allows the user to display the iframe in full screen mode.

<iframe

src="video-url"

width="width-value"

height="height-value"

allowfullscreen

></iframe>

To embed a video within an iframe you can copy it directly from popular video services like YouTube and Vimeo, or define it yourself with the src attribute pointing to the URL of that video. Here's an example of embedding a popular freeCodeCamp course from YouTube:

<h1>A freeCodeCamp YouTube Video Embedded with the iframe Element</h1>

<iframe

width="560"

height="315"

src="https://www.youtube.com/embed/PkZNo7MFNFg?si=-UBVIUNM3csdeiWF"

title="YouTube video player"

allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share"

referrerpolicy="strict-origin-when-cross-origin"

allowfullscreen

></iframe>

There are some other replaced elements, such as video, and embed. And some elements behave as replaced elements under specific circumstances. Here's an example of an input element with the type attribute set to image:

<input type="image" alt="Descriptive text goes here" src="example-img-url">

* **Optimizing media**: There are three tools to consider when using media, such as images, on your web pages: the size, the format, and the compression. A compression algorithm is used to reduce the size for files or data.
* **Image formats**: Two of the most common file formats are PNG and JPG, but these are no longer the most ideal formats for serving images. Unless you need support for older browsers, you should consider using a more optimized format, like WEBP or AVIF.
* **Image licenses**: An image under the public domain has no copyright attached to it and is free to be used without any restrictions. Images licensed specifically under the Creative Commons 0 license are considered public domain. Some images might be released under a permissive license, like a Creative Commons license, or the BSD license that freeCodeCamp uses.
* **SVGs**: Scalable Vector Graphics track data based on paths and equations to plot points, lines, and curves. What this really means is that a vector graphic, like an SVG, can be scaled to any size without impacting the quality.

**Multimedia Integration**

* **audio and video Elements**: The audio and video elements allow you to add sound and video content to your HTML documents. The audio element supports popular audio formats like mp3, wav, and ogg. The video element supports mp4, ogg, and webm formats.

<audio src="CrystalizeThatInnerChild.mp3"></audio>

If you want to see the audio player on the page, then you can add the audio element with the controls attribute:

<audio src="CrystalizeThatInnerChild.mp3" controls></audio>

The controls attribute enables users to manage audio playback, including adjusting volume, and pausing, or resuming playback. The controls attribute is a boolean attribute that can be added to an element to enable built-in playback controls. If omitted, no controls will be shown.

* **loop Attribute**: The loop attribute is a boolean attribute that makes the audio replay continuously.

<audio

src="https://cdn.freecodecamp.org/curriculum/js-music-player/can't-stay-down.mp3"

loop

controls

></audio>

* **muted Attribute**: When present in the audio element, the muted boolean attribute will start the audio in a muted state.

<audio

src="https://cdn.freecodecamp.org/curriculum/js-music-player/can't-stay-down.mp3"

loop

controls

muted

></audio>

* **source Element**: When it comes to audio file types, there are differences in which browsers support which type. To accommodate this, you can use source elements inside the audio element and the browser will select the first source that it understands. Here's an example of using multiple source elements for an audio element:

<audio controls>

<source src="audio.ogg" type="audio/ogg" />

<source src="audio.wav" type="audio/wav" />

<source src="audio.mp3" type="audio/mpeg" />

</audio>

All the attributes we have learned so far are also supported in the video element. Here's an example of using a video element with the loop, controls, and muted attributes:

<video

src="https://archive.org/download/BigBuckBunny\_124/Content/big\_buck\_bunny\_720p\_surround.mp4"

loop

controls

muted

></video>

* **poster Attribute**: If you wanted to display an image while the video is downloading, you can use the poster attribute. This attribute is not available for audio elements and is unique to the video element.

<video

src="https://archive.org/download/BigBuckBunny\_124/Content/big\_buck\_bunny\_720p\_surround.mp4"

loop

controls

muted

poster="https://peach.blender.org/wp-content/uploads/title\_anouncement.jpg?x11217"

width="620"

></video>

**Target attribute types**

* **target Attribute**: This attribute tells the browser where to open the URL for the anchor element. There are four important possible values for this attribute: \_self, \_blank, \_parent and \_top. There is a fifth value, called \_unfencedTop, which is currently used for the experimental FencedFrame API. You probably won't have a reason to use this one yet.
* **\_self Value**: This is the default value for the target attribute. This opens the link in the current browsing context. In most cases, this will be the current tab or window.

<a href="https://freecodecamp.org" target="\_self">Visit freeCodeCamp</a>

* **\_blank Value**: This opens the link in a new browsing context. Typically, this will open in a new tab. But some users might configure their browsers to open a new window instead.

<a href="https://freecodecamp.org" target="\_blank">Visit freeCodeCamp</a>

* **\_parent Value**: This opens the link in the parent of the current context. For example, if your website has an iframe, a \_parent value in that iframe would open in your website's tab/window, not in the embedded frame.

<a href="https://freecodecamp.org" target="\_parent">Visit freeCodeCamp</a>

* **\_top Value**: This opens the link in the top-most browsing context - think "the parent of the parent". This is similar to \_parent, but the link will always open in the full browser tab/window, even for nested embedded frames.

<a href="https://freecodecamp.org" target="\_top">Visit freeCodeCamp</a>

**Absolute vs. Relative Paths**

* **Path definition**: A path is a string that specifies the location of a file or directory in a file system. In web development, paths let developers link to resources like images, stylesheets, scripts, and other web pages.
* **Path Syntax**: There are three key syntaxes to know. First is the slash, which can be a backslash (\) or forward slash (/) depending on your operating system. The second is the single dot (.). And finally, we have the double dot (..). The slash is known as the "path separator". It is used to indicate a break in the text between folder or file names. A single dot points to the current directory, and two dots point to the parent directory.

public/index.html

./favicon.ico

../src/index.css

* **Absolute Path**: An absolute path is a complete link to a resource. It starts from the root directory, includes every other directory, and finally the filename and extension. The "root directory" refers to the top-level directory or folder in a hierarchy. An absolute path also includes the protocol - which could be http, https, and file and the domain name if the resource is on the web. Here's an example of an absolute path that links to the freeCodeCamp logo:

<a href="https://design-style-guide.freecodecamp.org/img/fcc\_secondary\_small.svg">

View fCC Logo

</a>

* **Relative Path**: A relative path specifies the location of a file relative to the directory of the current file. It does not include the protocol or the domain name, making it shorter and more flexible for internal links within the same website. Here's an example of linking to the about.html page from the contact.html page, both of which are in the same folder:

<p>

Read more on the

<a href="about.html">About Page</a>

</p>

**Link states**

* **:link**: This is the default state. This state represents a link which the user has not visited, clicked, or interacted with yet. You can think of this state as providing the base styles for all links on your page. The other states build on top of it.
* **:visited**: This applies when a user has already visited the page being linked to. By default, this turns the link purple - but you can leverage CSS to provide a different visual indication to the user.
* **:hover**: This state applies when a user is hovering their cursor over a link. This state is helpful for providing extra attention to a link, to ensure a user actually intends to click it.
* **:focus**: This state applies when we focus on a link.
* **:active**: This state applies to links that are being activated by the user. This typically means clicking on the link with the primary mouse button by left clicking, in most cases.

## **Importance of Semantic HTML**

* ****Structural hierarchy for heading elements****: It is important to use the correct heading element to maintain the structural hierarchy of the content. The h1 element is the highest level of heading and the h6 element is the lowest level of heading.
* ****Presentational HTML elements****: Elements that define the appearance of content. Ex. the deprecated center, big and font elements.
* ****Semantic HTML elements****: Elements that hold meaning and structure. Ex. header, nav, figure.

## **Semantic HTML Elements**

* ****Header element****: used to define the header of a document or section.
* ****Main element****: used to contain the main content of the web page.
* ****Section element****: used to divide up content into smaller sections.
* ****Navigation Section (**nav**) element****: represents a section with navigation links.
* ****Figure element****: used to contain illustrations and diagrams.
* ****Emphasis (**em**) element****: marks text that has stress emphasis.

<p>

Never give up on <em>your</em> dreams.</p>

* ****Idiomatic Text (**i**) element****: used for highlighting alternative voice or mood, idiomatic terms from another language, technical terms, and thoughts.

<p>

There is a certain <i lang="fr">je ne sais quoi</i> in the air.</p>

The lang attribute inside the open i tag is used to specify the language of the content. In this case, the language would be French. The i element does not indicate if the text is important or not, it only shows that it's somehow different from the surrounding text.

* ****Strong Importance (**strong**) element****: marks text that has strong importance.

<p>

<strong>Warning:</strong> This product may cause allergic reactions.</p>

* ****Bring Attention To (**b**) element****: used to bring attention to text that is not important for the meaning of the content. It's commonly used to highlight keywords in summaries or product names in reviews.

<p>

We tested several products, including the <b>SuperSound 3000</b> for audio quality, the <b>QuickCharge Pro</b> for fast charging, and the <b>Ecoclean Vacuum</b> for cleaning. The first two performed well, but the <b>Ecoclean Vacuum</b> did not meet expectations.</p>

* ****Description List (**dl**) element****: used to represent a list of term-description groupings.
* ****Description Term (**dt**) element****: used to represent the term being defined.
* ****Description Details (**dd**) element****: used to represent the description of the term.

<dl>

<dt>HTML</dt>

<dd>HyperText Markup Language</dd>

<dt>CSS</dt>

<dd>Cascading Style Sheets</dd></dl>

* ****Block Quotation (**blockquote**) element****: used to represent a section that is quoted from another source. This element has a cite attribute. The value of the cite attribute is the URL of the source.

<blockquote cite="https://www.freecodecamp.org/news/learn-to-code-book/">

"Can you imagine what it would be like to be a successful developer? To have built software systems that people rely upon?"</blockquote>

* ****Citation (**cite**) element****: used to attribute the source of the referenced work visually. Marks up the title of the reference.

<div>

<blockquote cite="https://www.freecodecamp.org/news/learn-to-code-book/">

"Can you imagine what it would be like to be a successful developer? To have built software systems that people rely upon?"

</blockquote>

<p>

-Quincy Larson, <cite>How to learn to Code and Get a Developer Job [Full Book].</cite>

</p></div>

* ****Inline Quotation (**q**) element****: used to represent a short inline quotation.

<p>

As Quincy Larson said,

<q cite="https://www.freecodecamp.org/news/learn-to-code-book/">

Momentum is everything.

</q></p>

* ****Abbreviation (**abbr**) element****: used to represent an abbreviation or acronym. To help users understand what the abbreviation or acronym is, you can show its full form, a human readable description, with the title attribute.

<p>

<abbr title="HyperText Markup Language">HTML</abbr> is the foundation of the web.</p>

* ****Contact Address (**address**) element****: used to represent the contact information.
* ****(Date) Time (**time**) element****: used to represent a date and/or time. The datetime attribute is used to translate dates and times into a machine-readable format.

<p>

The reservations are for the <time datetime="20:00">20:00 </time></p>

* ****ISO 8601 Date (**datetime**) attribute****: used to represent dates and times in a machine-readable format. The standard format is YYYY-MM-DDThh:mm:ss, with the capital T being a separator between the date and time.
* ****Superscript (**sup**) element****: used to represent superscript text. Common use cases for the sup element would include exponents, superior lettering and ordinal numbers.

<p>

2<sup>2</sup> (2 squared) is 4.</p>

* ****Subscript (**sub**) element****: used to represent subscript text. Common uses cases for the subscript element include chemical formulas, foot notes, and variable subscripts.

<p>

CO<sub>2</sub></p>

* ****Inline Code (**code**) element****: used to represent a fragment of computer code.
* ****Preformatted Text (**pre**) element****: represents preformatted text

<pre>

<code>

body {

color: red;

}

</code></pre>

* ****Unarticulated Annotation (**u**) element****: used to represent a span of inline text which should be rendered in a way that indicates that it has a non-textual annotation.

<p>

You can use the unarticulated annotation element to highlight

<u>inccccort</u> <u>spling</u> <u>issses</u>.</p>

* ****Ruby Annotation (**ruby**) element****: used for annotating text with pronunciation or meaning explanations. An example usage is for East Asian typography.
* ****Ruby fallback parenthesis (**rp**) element****: used as a fallback for browsers lacking support for displaying ruby annotations.
* ****Ruby text (**rt**) element****: used to indicate text for the ruby annotation. Usually used for pronunciation, or translation details in East Asian typography.

<ruby>

明日 <rp>(</rp><rt>Ashita</rt><rp>)</rp></ruby>

* ****Strikethrough (**s**) element****: used to represent content that is no longer accurate or relevant.

<p>

<s>Tomorrow's hike will be meeting at noon.</s></p><p>

Due to unforeseen weather conditions, the hike has been cancelled.</p>

## **HTML Form Elements and Attributes**

* **form**element****: used to create an HTML form for user input.
* **action**attribute****: used to specify the URL where the form data should be sent.
* **method**attribute****: used to specify the HTTP method to use when sending the form data. The most common methods are GET and POST.

Example Code

<form method="value-goes-here" action="url-goes-here">

<!-- inputs go inside here --></form>

* **input**element****: used to create an input field for user input.
* **type**attribute****: used to specify the type of input field. Ex. text, email, number, radio, checkbox, etc.
* **placeholder**attribute****: used to show a hint to the user to show them what to enter in the input field.
* **value**attribute****: used to specify the value of the input. If the input has a button type, the value attribute can be used to set the button text.
* **name**attribute****: used to assign a name to an input field, which serves as the key when form data is submitted. For radio buttons, giving them the same name groups them together, so only one option in the group can be selected at a time.
* **size**attribute****: used to define the number of characters that should be visible as the user types into the input.
* **min**attribute****: can be used with input types such as number to specify the minimum value allowed in the input field.
* **max**attribute****: can be used with input types such as number to specify the maximum value allowed in the input field.
* **minlength**attribute****: used to specify the minimum number of characters required in an input field.
* **maxlength**attribute****: used to specify the maximum number of characters allowed in an input field.
* **required**attribute****: used to specify that an input field must be filled out before submitting the form.
* **disabled**attribute****: used to specify that an input field should be disabled.
* **readonly**attribute****: used to specify that an input field is read-only.

Example Code

<!-- Text input --><input

type="text"

id="name"

name="name"

placeholder="e.g. Quincy Larson"

size="20"

minlength="5"

maxlength="30"

required/>

<!-- Number input --><input

type="number"

id="quantity"

name="quantity"

min="2"

max="10"

disabled/>

<!-- Button --><input type="button" value="Show Alert" />

* **label**element****: used to create a label for an input field.
* **for**attribute****: used to specify which input field the label is for.
* ****Implicit form association****: inputs can be associated with labels by wrapping the input field inside the label element.

Example Code

<form action="">

<label>

Full Name:

<input type="text" />

</label></form>

* ****Explicit form association****: inputs can be associated with labels by using the for attribute on the label element.

Example Code

<form action="">

<label for="email">Email Address: </label>

<input type="email" id="email" /></form>

* **button**element****: used to create a clickable button. A button can also have a type attribute, which is used to control the behavior of the button when it is activated. Ex. submit, reset, button.

Example Code

<button type="button">Show Form</button><button type="submit">Submit Form</button><button type="reset">Reset Form</button>

* **fieldset**element****: used to group related inputs together.
* **legend**element****: used to add a caption to describe the group of inputs.

Example Code

<!-- Radio group --><fieldset>

<legend>Was this your first time at our hotel?</legend>

<label for="yes-option">Yes</label>

<input id="yes-option" type="radio" name="hotel-stay" value="yes" />

<label for="no-option">No</label>

<input id="no-option" type="radio" name="hotel-stay" value="no" /></fieldset>

<!-- Checkbox group --><fieldset>

<legend>

Why did you choose to stay at our hotel? (Check all that apply)

</legend>

<label for="location">Location</label>

<input type="checkbox" id="location" name="location" value="location" />

<label for="price">Price</label>

<input type="checkbox" id="price" name="price" value="price" /></fieldset>

* ****Focused state****: this is the state of an input field when it is selected by the user.

## **Working with HTML Table Elements and Attributes**

* ****Table element****: used to create an HTML table.
* ****Table Head (**thead**) element****: used to group the header content in an HTML table.
* ****Table Row (**tr**) element****: used to create a row in an HTML table.
* ****Table Header (**th**) element****: used to create a header cell in an HTML table.
* ****Table body (**tbody**) element****: used to group the body content in an HTML table.
* ****Table Data Cell (**td**) element****: used to create a data cell in an HTML table.
* ****Table Foot (**tfoot**) element****: used to group the footer content in an HTML table.
* **caption**element****: used to add a title of an HTML table.
* **colspan**attribute****: used to specify the number of columns a table cell should span.

Example Code

<table>

<caption>Exam Grades</caption>

<thead>

<tr>

<th>Last Name</th>

<th>First Name</th>

<th>Grade</th>

</tr>

</thead>

<tbody>

<tr>

<td>Davis</td>

<td>Alex</td>

<td>54</td>

</tr>

<tr>

<td>Doe</td>

<td>Samantha</td>

<td>92</td>

</tr>

<tr>

<td>Rodriguez</td>

<td>Marcus</td>

<td>88</td>

</tr>

</tbody>

<tfoot>

<tr>

<td colspan="2">Average Grade</td>

<td>78</td>

</tr>

</tfoot></table>

## **Working with HTML Tools**

* ****HTML validator****: a tool that checks the syntax of HTML code to ensure it is valid.
* ****DOM inspector****: a tool that allows you to inspect and modify the HTML structure of a web page.
* ****Devtools****: a set of web developer tools built directly into the browser that helps you debug, profile, and analyze web pages.

## **Introduction to Accessibility**

* ****Web Content Accessibility Guidelines****: The Web Content Accessibility Guidelines (WCAG) are a set of guidelines for making web content more accessible to people with disabilities. The four principles of WCAG are POUR which stands for Perceivable, Operable, Understandable, and Robust.

## **Assistive Technology for Accessibility**

* ****Screen readers****: Software programs that read the content of a computer screen out loud. They are used by people who are blind or visually impaired to access the web.
* ****Large text or braille keyboards****: Used by people with visual impairments to access the web.
* ****Screen magnifiers****: Software programs that enlarge the content of a computer screen. They are used by people with low vision to access the web.
* ****Alternative pointing devices****: Used by people with mobility impairments to access the web. This includes devices such as joysticks, trackballs, and touchpads.
* ****Voice recognition****: Used by people with mobility impairments to access the web. It allows users to control a computer using their voice.

## **Accessibility Auditing Tools**

* ****Common Accessibility Tools****: Google Lighthouse, Wave, IBM Equal Accessibility Checker, and axe DevTools are some of the common accessibility tools used to audit the accessibility of a website.

## **Accessibility Best Practices**

* ****Proper heading level structure****: You should use proper heading levels to create a logical structure for your content. This helps people using assistive technologies understand the content of your website.
* ****Accessibility and Tables****: When using tables, you should use the th element to define header cells and the td element to define data cells. This helps people using assistive technologies understand the structure of the table. With the caption element, you can write the caption (or title) of a table, so users, especially those who use assistive technologies, can quickly understand the table's purpose and content. You should place the caption element immediately after the opening tag of the table element. This way, screen readers and other assistive technologies can provide more context by announcing the caption before reading the content.
* ****Importance for inputs to have an associated label****: You should use the label element to associate labels with form inputs. This helps people using assistive technologies understand the purpose of the input.
* ****Importance of good**alt**text****: You should use the alt attribute to provide a text alternative for images. This helps people using assistive technologies understand the content of the image.
* ****Importance of good link text****: You should use descriptive link text to help users understand the purpose of the link. This helps people using assistive technologies understand the purpose of the link.
* ****Best practices for making audio and video content accessible****: You should provide captions and transcripts for audio and video content to make it accessible to people with hearing impairments. You should also provide audio descriptions for video content to make it accessible to people with visual impairments.
* **tabindex**attribute****: Used to make elements focusable and define the relative order in which they should be navigated using the keyboard. It is important to never use the tabindex attribute with a value greater than 0. Instead, you should either use a value of 0 or -1. For more information, review the prior lecture video on keyboard accessibility.

<p tabindex="-1">Sorry, there was an error with your submission.</p>

* **accesskey**attribute****: Used to define a keyboard shortcut for an element. This can help users with mobility impairments navigate the website more easily.

<button accesskey="s">Save</button><button accesskey="c">Cancel</button><a href="index.html" accesskey="h">Home</a>

## **WAI-ARIA, Roles, and Attributes**

* ****WAI-ARIA****: It stands for Web Accessibility Initiative - Accessible Rich Internet Applications. It is a set of attributes that can be added to HTML elements to improve accessibility. It provides additional information to assistive technologies about the purpose and structure of the content.
* ****ARIA roles****: A set of predefined roles that can be added to HTML elements to define their purpose. This helps people using assistive technologies understand the content of the website. Examples include role="tab", role="menu", and role="alert".

There are six main categories of ARIA roles:

****Document structure roles****: These roles define the overall structure of the web page. With these roles, assistive technologies can understand the relationships between different sections and help users navigate the content.

****Widget roles****: These roles define the purpose and functionality of interactive elements, like scrollbars.

****Landmark roles****: These roles classify and label the primary sections of a web page. Screen readers use them to provide convenient navigation to important sections of a page.

****Live region roles****: These roles define elements with content that will change dynamically. This way, screen readers and other assistive technologies can announce changes to users with visual disabilities.

****Window roles****: These roles define sub-windows, like pop up modal dialogues. These roles include alertdialog and dialog.

****Abstract roles****: These roles help organize the document. They're only meant to be used internally by the browser, not by developers, so you should know that they exist but you shouldn't use them on your websites or web applications.

**aria-label**and**aria-labelledby**attributes****: These attributes are used to give an element a programmatic (or accessible) name, which helps people using assistive technology (such as screen readers) understand the purpose of the element. They are often used when the visual label for an element is an image or symbol rather than text. aria-label allows you to define the name directly in the attribute while aria-labelledby allows you to reference existing text on the page.

<button aria-label="Search">

<i class="fas fa-search"></i></button>

<input type="text" aria-labelledby="search-btn"><button type="button" id="search-btn">Search</button>

* **aria-hidden**attribute****: Used to hide an element from assistive technologies such as screen readers. For example, this can be used to hide decorative images that do not provide any meaningful content.

<button>

<i class="fa-solid fa-gear" aria-hidden="true"></i>

<span class="label">Settings</span></button>

* **aria-expanded**attribute****: Used to convey the state of a toggle (or disclosure) feature to screen reader users.

<button aria-expanded="true">Menu</button>

* **aria-live**attribute****: Used to indicate that an element's content is important enough to require that any changes to the content should be announced immediately to screen reader users. This can include status messages like updating the number of results returned from a search, or an error message displayed after an unsuccessful form submission.

<div aria-live="assertive">

<p>Your session will expire in 30 seconds.</p></div>

<div aria-live="polite">

<p>File successfully uploaded</p></div>

* ****Common ARIA states****: Common ARIA states include aria-haspopup, aria-checked, aria-disabled, and aria-selected. These attributes can be used to indicate the state of an element and help people using assistive technologies understand the content of the website.
* **aria-haspopup**attribute****: This state is used to indicate that an interactive element will trigger a pop-up element when activated. You can only use the aria-haspopup attribute when the pop-up has one of the following roles: menu, listbox, tree, grid, or dialog. The value of aria-haspopup must be either one of these roles or true, which is the same as menu.

<button id="menubutton" aria-haspopup="menu" aria-controls="filemenu" aria-expanded="false">File</button><ul id="filemenu" role="menu" aria-labelledby="menubutton" hidden>

<li role="menuitem" tabindex="-1">Open</li>

<li role="menuitem" tabindex="-1">New</li>

<li role="menuitem" tabindex="-1">Save</li>

<li role="menuitem" tabindex="-1">Delete</li></ul>

* **aria-checked**attribute****: This attribute is used to indicate whether an element is in the checked state. It is most commonly used when creating custom checkboxes, radio buttons, switches, and listboxes.

<div role="checkbox" aria-checked="true" tabindex="0">Checkbox</div>

* **aria-disabled**attribute****: This state is used to indicate that an element is disabled only to people using assistive technologies, such as screen readers.

<div role="button" tabindex="-1" aria-disabled="true">Edit</div>

* **aria-selected**attribute****: This state is used to indicate that an element is selected. You can use this state on custom controls like a tabbed interface, a listbox, or a grid.

<div role="tablist">

<button role="tab" aria-selected="true">Tab 1</button>

<button role="tab" aria-selected="false">Tab 2</button>

<button role="tab" aria-selected="false">Tab 3</button></div>

* **aria-controls**attribute****: Used to associate an element with another element that it controls. This helps people using assistive technologies understand the relationship between the elements.

<div role="tablist">

<button role="tab" id="tab1" aria-controls="section1" aria-selected="true">

Tab 1

</button>

<button role="tab" id="tab2" aria-controls="section2" aria-selected="false">

Tab 2

</button>

<button role="tab" id="tab3" aria-controls="section3" aria-selected="false">

Tab 3

</button></div>

* **aria-describedby**attribute****: Used to provide additional information about an element by associating it with another element that contains the information. This gives people using screen readers immediate access to the additional information when they navigate to the element. Common usage would include associating formatting instructions to a text input or an error message to an input after an invalid form submission.

<form>

<label for="password">Password:</label>

<input type="password" id="password" aria-describedby="password-help" />

<p id="password-help">Your password must be at least 8 characters long.</p></form>