HTML Review

HTML Basics

- **Role of HTML**: HTML (Hypertext Markup Language) is the foundation of web structure, defining the elements of a webpage.
- **HTML Elements**: Used to represent content on the page. Most of them are made by an opening and a closing tag (e.g., <h1></h1>,).
- **HTML Structure**: HTML consists of a head and body, where metadata, styles, and content are structured.
- **Common HTML Elements**: Headings (<h1> - <h6>), paragraphs ( ), and div containers ( ).
- **div elements**: The div 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.
- **Void Elements**: Do not have a closing tag (e.g., <img>).
- **Attributes**: Adding metadata and behavior to elements.

Identifiers and Grouping

- **IDs**: Unique element identifiers.
- **Classes**: Grouping elements for styling and behavior.

Special Characters and Linking

- **HTML entities**: Using special characters like & and <.
- **link element**: Linking to external stylesheets.
- **script element**: Embedding external JavaScript files.

Boilerplate and Encoding

- **HTML boilerplate**: Basic structure of a webpage, which includes the DOCTYPE, html, head, and body elements. It should be used as the starting point for an HTML document.
- **UTF-8 character encoding**: Ensuring universal character display.

SEO and Social Sharing

- **Meta tags (description)**: Providing a short description for the web page and impacting SEO.
- **Open Graph tags**: Enhancing social media sharing.

Media Elements and Optimization

- **Replaced elements**: Embedded content (e.g., images, iframes).
- **Optimizing media**: Techniques to improve media performance.
- **Image formats and licenses**: Understanding usage rights and types.
- **SVGs**: Scalable vector graphics for sharp visuals.

Multimedia Integration

- **HTML audio and video elements**: Embedding multimedia.
- **Embedding with <iframe>**: Integrating external video content.

Paths and Link Behavior

- **Target attribute types**: Controlling link behavior.
- **Absolute vs. relative paths**: Navigating directories.
- **Path syntax**: Understanding /, ./, ../ for file navigation.
- **Link states**: Managing different link interactions (hover, active).

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**: These elements provide meaning to the structure of the content. Examples include:
    - <header>: Represents introductory content.
    - <nav>: Contains navigation links.
    - <article>: Represents self-contained content.

- <aside>: Used for sidebars or related content.
- <section>: Groups related content within a document.
- <footer>: Defines the footer for a section or document.

Semantic HTML Elements

- **Emphasis (em) element**: Marks text that has stress emphasis.
- **Idiomatic Text (i) element**: Used for highlighting alternative voice or mood, idiomatic terms from another language, technical terms, and thoughts.
- **Strong Importance (strong) element**: Marks text that has strong importance.
- **Bring Attention To (b) element**: Used to bring attention to text that is not important for the meaning of the content.
- **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.
- **Block Quotation (blockquote) element**: Used to represent a section that is quoted from another source.
- **Inline Quotation (q) element**: Used to represent a short inline quotation.
- **Abbreviation (abbr) element**: Used to represent an abbreviation or acronym.
- **Contact Address (address) element**: Used to represent the contact information.
- **(Date) Time (time) element**: Used to represent a date and/or time.
- **Superscript (sup) element**: Used to represent superscript text.
- **Subscript (sub) element**: Used to represent subscript text.
- **Inline Code (code) element**: Used to represent a fragment of computer code.
- **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.
- **Ruby Annotation (ruby) element**: Used to represent the text of a ruby annotation.

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

HTML Form Elements and Attributes

Forms

- **form element**: Used to create an HTML form for user input.
- **action attribute**: Defines where to send form data.
- **method attribute**: Determines how form data is sent (GET or POST).
- **Common Input Types**:
    - text, email, password, radio, checkbox, number, date.
- **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.

```
<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.
- **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.

```
<!-- 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.

```
<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.

```
<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`.

```
<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.

```
<!-- 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" />

  <label for="no-option">No</label>
  <input id="no-option" type="radio" name="hotel-stay" />
</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.

```
<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>
```

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 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 assistive technologies understand the structure of the table.
- **Importance for inputs to have an associated label**: You should use the label element to associate labels with form inputs. This helps 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 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 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.
- **accesskey attribute**: Used to define a keyboard shortcut for an element. This can help users with mobility impairments navigate the website more easily.

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 assistive technologies understand the content of the website. Examples include role="tab", role="menu", and role="alert".
- **aria-label and aria-labelledby attributes**: These attributes are used to give an element a programmatic (or accessible) name, which helps 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.

- **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.
- **aria-expanded attribute**: Used to convey the state of a toggle (or disclosure) feature to screen reader users.
- **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.
- **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 assistive technologies understand the content of the website.
- **aria-controls attribute**: Used to associate an element with another element that it controls. This helps assistive technologies understand the relationship between the elements.
- **aria-describedby attribute**: Used to provide additional information about an element by associating it with another element that contains the information. This helps assistive technologies understand the purpose of the element.