Basic HTML Review

HTML Basics

- **Role of HTML:** HTML represents the content and structure of the web page.
- **HTML Elements:** Elements are the building blocks for an HTML document and represent content like headings, paragraphs, links, and images.
  - Most HTML elements consist of an **opening tag** (`<elementName>`) and a **closing tag** (`</elementName>`).
  - Basic syntax: `<elementName>Content goes here</elementName>` .
- **Void Elements:** These elements cannot have any content and only have a start tag.
  - Examples include `img` and `meta` elements.
  - Both `<img>` and `<img/>` are acceptable syntax, as some codebases include a forward slash inside the void element.
- **Attributes:** A value placed inside the opening tag of an HTML element.
  - Attributes provide additional information or specify how the element should behave.
  - Basic syntax: `<element attribute="value"></element>` .
  - A **boolean attribute** is present or absent in a tag; if present, the value is true, otherwise it's false. Examples include `disabled` , `readonly` , and `required` .
- **Comments:** Used to leave notes for yourself and other developers in your code.
  - Syntax: `<!--This is an HTML comment.-->` .

Common HTML Elements

- **Heading Elements** (`h1` through `h6`): Used to signify the **importance** of content below them. The lower the number, the higher the importance (`h2` has less importance than `h1`).
  - `<h1>most important heading element</h1>`
  - `<h6>least important heading element</h6>`
- **Paragraph Elements** (`p`): Used for paragraphs on a web page.
- **Image Elements** (`img`): Used to add images to the web page.

    - The `src` attribute specifies the image's location.
    - The `alt` attribute should also be included as good practice.
    - Example: `<img src="https://cdn.freecodecamp.org/curriculum/cat-photo-app/lasagna.jpg" alt="A slice of lasagna on a plate" />`.
- **Body Element** (`body`): Used to represent the main content for the HTML document.
- **Section Elements** (`section`): Used to divide content into smaller sections.
- **Div Elements** (`div`): A generic HTML element that does not hold any semantic meaning, used as a generic container.
- **Anchor Elements** (`a`): Used to apply links to a web page.
    - The `href` attribute specifies the link destination.
    - Example: `<a href="https://cdn.freecodecamp.org/curriculum/cat-photo-app/running-cats.jpg">cute cats</a>`.
- **Unordered List** (`ul`) **and Ordered List** (`ol`) **Elements:**
    - **Unordered list (`ul`):** Creates a bulleted list with one or more `li` (list item) elements nested inside.
    - **Ordered list (`ol`):** Creates a numbered list of items.
- **Emphasis Element** (`em`): Used to place emphasis on text. Example: `<p>Cats <em>love</em> lasagna.</p>`.
- **Strong Importance Element** (`strong`): Used to place strong emphasis on text, indicating urgency and seriousness.
- **Figure** (`figure`) **and Figcaption** (`figcaption`) **Elements:**
    - `figure` : Used to group content like images and diagrams.
    - `figcaption` : Used to represent a caption for the content inside the `figure` element.
- **Main Element** (`main`): Used to represent the main content for a web page.
- **Footer Element** (`footer`): Placed at the bottom of the document, usually containing copyright information and other important page links.

Identifiers and Grouping

- **IDs:** Unique element identifiers for HTML elements.
    - An ID name should only be used once per HTML document.
    - ID names cannot have spaces; use dashes between words for multiple words (e.g., `id="red-box"`).

- **Classes:** Used to group elements for styling and behavior.
  - Unlike IDs, the same class name can be reused throughout the HTML document.
  - The class value can also have spaces (e.g., `class="box red-box"`).

Special Characters and Linking

- **HTML Entities (Character References):** A set of characters used to represent a reserved character in HTML.
  - Examples: `&` for the ampersand symbol, and `<` for the less than symbol.
- **Link Element** (`link`): Used to link to external resources like stylesheets and site icons.
  - `rel` **attribute:** Specifies the relationship between the linked resource and the HTML document.
  - `href` **attribute:** Specifies the URL location for the external resource.
  - Example for CSS: `<link rel="stylesheet" href="./styles.css" />`.
- **Script Element** (`script`): Used to embed executable code (like JavaScript).
  - Best practice is to link to an external JavaScript file using the `src` attribute.
  - Example for external file: `<script src="path-to-javascript-file.js"></script>`.

Boilerplate and Encoding

- **HTML Boilerplate:** Includes the basic structure and essential elements every HTML document needs.
  - `<!DOCTYPE html>`: Tells browsers which version of HTML you're using.
  - `<html>`: The top-level or root element of an HTML document. The `lang` attribute specifies the document's language.
  - `<head>`: Contains important meta data for browsers and search engines.
  - `<meta>` elements: Represent site metadata, including character encoding.
    - The `charset` attribute is used inside a `meta` element to set the character encoding.
  - `<title>` element: Sets the text that appears in the browser tab or window.
  - `<body>`: Content goes inside here.

- **UTF-8 Character Encoding:** A standardized character encoding widely used on the web, where computers store characters as data.

SEO and Social Sharing

- **Meta (`description`) Element:** Provides a short description for the web page, impacting SEO (Search Engine Optimization).
- **Open Graph Tags:** Properties set through `meta` elements in the `<head>` section to control how website content appears across social media platforms (like Facebook and LinkedIn).
  - `og:title` : Sets the title for social media posts.
  - `og:type` : Represents the type of content being shared (e.g., articles, websites, videos).
  - `og:image` : Sets the image shown for social media posts.
  - `og:url` : Sets the URL users will click on.

Media Elements and Optimization

- **Replaced Elements:** Elements whose content is determined by an external resource rather than CSS itself.
  - **`iframe` (inline frame) element:** Used to embed other HTML content directly within the page.
    - The `allowfullscreen` attribute enables full-screen display.
  - Other replaced elements include `video` and `embed` .
  - An `input` element with `type="image"` can also behave as a replaced element.
- **Optimizing Media (Images):** Consider the **size**, **format**, and **compression**.
  - **Compression algorithms** are used to reduce file size.
  - **Image formats:** While PNG and JPG are common, optimized formats like **WEBP** or **AVIF** are now more ideal unless support for older browsers is required.
  - **Image licenses:** Images in the **public domain** (e.g., Creative Commons 0 license) have no copyright and can be used without restrictions.
- **SVGs (Scalable Vector Graphics):** Track data based on paths and equations, allowing them to be scaled to any size without impacting quality.

Multimedia Integration

- **Audio** (`audio`) **and Video** (`video`) **Elements:** Allow you to add sound and video content.
  - **Audio formats:** Supports `mp3` , `wav` , and `ogg` .
  - **Video formats:** Supports `mp4` , `ogg` , and `webm` .
  - `controls` **attribute:** A boolean attribute that enables built-in playback controls (volume, pausing, resuming); if omitted, no controls are shown.
  - `loop` **attribute:** A boolean attribute that makes the audio/video replay continuously.
  - `muted` **attribute:** A boolean attribute that starts the audio/video in a muted state.
  - `source` **element:** Can be nested inside `audio` or `video` to offer multiple file types; the browser selects the first source it understands to accommodate browser support differences.
  - `poster` **attribute:** Unique to the `video` element, it displays an image while the video is downloading.

Target Attribute Types

The `target` **attribute** on an anchor (`a`) element tells the browser where to open the URL. Possible values are:

- `_self` **:** The default value. Opens the link in the **current browsing context** (usually the current tab or window).
- `_blank` **:** Opens the link in a **new browsing context** (typically a new tab or window).
- `_parent` **:** Opens the link in the immediate **parent browsing context**.
- `_top` **:** Opens the link in the **top-most browsing context** (the full browser tab/window, even for nested frames).

Absolute vs. Relative Paths

A **path** is a string that specifies the location of a file or directory.

- **Path Syntax:**
  - **Slash (** `/` **or** `\` **):** The "path separator," indicating a break between folder or file names.
  - **Single dot (** `.` **):** Points to the current directory.
  - **Double dot (** `..` **):** Points to the parent directory.

- **Absolute Path:** A **complete link** to a resource.
  - Starts from the **root directory**.
  - Includes the **protocol** (`http` , `https` , or `file`) and the **domain name** if the resource is on the web.
- **Relative Path:** Specifies the location of a file **relative to the directory of the current file**.
  - Does not include the protocol or domain name, making it shorter and more flexible for internal links within the same website.

Link States

These states provide different styling for links based on user interaction (typically leveraged with CSS):

- `:link` **(Default State):** Represents a link the user has not visited, clicked, or interacted with yet, providing the base styles.
- `:visited` **:** Applies when a user has already visited the page being linked to (defaults to turning the link purple).
- `:hover` **:** Applies when a user is hovering their cursor over a link.
- `:focus` **:** Applies when we focus on a link (e.g., by using the tab key).
- `:active` **:** Applies to links that are currently being activated by the user (typically by left-clicking the primary mouse button).