

Full Stack Web Development

1. Understanding Full Stack Web Development

Definition: Full stack web development refers to the practice of developing both the front-end (client-side) and back-end (server-side) parts of a web application.

Components:

- Front-end: HTML, CSS, JavaScript, React, etc.
- Back-end: Node.js, Express, databases (MySQL, MongoDB), etc.

Example:

1. A website that allows users to register and view their profile.
2. An e-commerce site with product listings and a payment system.

2. Web Application

Definition: A web application is software that runs in a web browser and allows user interaction through a web interface.

Examples:

1. Online banking systems.
2. Social media platforms.

3. Full Stack Engineer / Full Stack Developer

Definition: A full stack developer is a professional skilled in both front-end and back-end technologies.

Skills Required:

- Front-end: React, Angular
- Back-end: Node.js, Django

4. Types of Applications

i). Standalone Application

Definition: A software that runs on a single system without the need for network connectivity.

Examples:

1. Media players.
2. Office applications like Microsoft Word.

ii). Web Application

Definition: Applications that run in web browsers and require internet connectivity.

Examples:

1. Google Docs.
2. Facebook.

iii) Mobile Application

Definition: Applications designed to run on mobile devices like smartphones and tablets.

Examples:

1. Instagram.
2. Uber.

4. High-Level Architecture of Web Application

1. Frontend Development

Frontend development focuses on the user interface and user experience aspects of a web application.

2. Backend Development

Backend development manages the server, database, and business logic.

3. Database

Databases store and manage data required by the application.

5. Technologies in Web Development

1. HTML

Markup language used to structure web pages.

2. CSS

Styles web pages to enhance their appearance.

3. JavaScript

Programming language used to add interactivity to web pages.

4. Bootstrap

CSS framework for responsive web design.

5. React JS

JavaScript library for building UI components.

6. Node JS

JavaScript runtime for building backend services.

7. Express JS

Minimalist web framework for Node.js.

8. MongoDB

NoSQL database for storing application data.

9. AWS Basics

Basic cloud computing services for deployment and hosting.

10. Deployment on Vercel

Platform to deploy frontend applications.

11. Git and GitHub

Version control system and platform for code collaboration.

6. Understanding MERN Stack

1. Installation of VS Code

1. Download VS Code from the [official website](#).
2. Run the installer and follow the setup process.

2. Installation of Extensions

1. Open VS Code and navigate to the Extensions tab.
2. Search for desired extensions such as ESLint, Prettier, and React tools.
3. Click "Install" to add them to your workspace.

7. Introduction to HTML

1. Purpose of HTML

Definition: HTML (HyperText Markup Language) is used to structure web pages and their content.

2. Elements in HTML

Definition: Elements are the building blocks of an HTML document, representing content.

3. Tags

Definition: Tags define HTML elements and determine how content is displayed.

4. Types of Tags

i). Paired Tag

- **Opening Tag:** Starts an element, e.g., `<p>`
- **Closing Tag:** Ends an element, e.g., `</p>`

ii). Self Closing Tag

Definition: Tags that do not require a closing tag, e.g., ``, `
`

8. Structure of the Tag / Representation of the Tag

HTML Tag: HTML tags are used to structure content in web pages. They consist of an opening tag, optional content, and a closing tag. The structure is as follows:

```
<tagname>Content</tagname>
```

Example:

```
<h1>Heading</h1>
```

- Opening tag: `<h1>`
- Content: `Heading`
- Closing tag: `</h1>`

9. Button Element

Definition: The `<button>` element is used to create clickable buttons in HTML.

Syntax:

```
<button>Button Text</button>
```

Tag Name: `<button>`

Type of Tag: Paired Tag (has opening and closing tags)

Purpose of the element: Used to trigger actions when clicked, such as submitting forms or executing scripts.

Display Type: Block-level (it behaves like a block element by default, but can be styled to behave as an inline element with CSS)

Examples:

1. Simple button:

```
<button>Click Me</button>
```

2. Button with event:

```
<button onclick="alert('Hello!')">Greet</button>
```

10. Heading Element

Definition: The `<h1>` to `<h6>` elements are used to define headings, where `<h1>` is the most important and `<h6>` the least.

Syntax:

```
<h1>Heading Text</h1>
```

Tag Name: `<h1>` (or `<h2>` to `<h6>`)

Type of Tag: Paired Tag (has opening and closing tags)

Purpose of the element: Used to define headings and subheadings in web pages to organize content.

Display Type: Block-level (headings are typically displayed as block elements, taking up the full width of their container)

Examples:

1. Main heading:

```
<h1>Main Heading</h1>
```

2. Subheading:

```
<h2>Subheading</h2>
```

Heading Elements:

1. `<h1>` - Main Heading

- Most important heading, typically used for the page or section title.
- **Display Type:** Block-level

Example:

```
<h1>Main Heading</h1>
```

2. `<h2>` - Subheading

- Used for subsections under the main heading.
- **Display Type:** Block-level

Example:

```
<h2>Subheading</h2>
```

3. **<h3>** - Sub-subheading

- Used for further division under an **<h2>** element.
- **Display Type:** Block-level

Example:

```
<h3>Sub-subheading</h3>
```

4. **<h4>** - Fourth-level Heading

- Used for detailed sections within an **<h3>**.
- **Display Type:** Block-level

Example:

```
<h4>Fourth-level Heading</h4>
```

5. **<h5>** - Fifth-level Heading

- Used for even more granular sections within an **<h4>**.
- **Display Type:** Block-level

Example:

```
<h5>Fifth-level Heading</h5>
```

6. **<h6>** - Sixth-level Heading

- The least important heading, used for the most granular divisions.
- **Display Type:** Block-level
- **Example:**

```
<h6>Sixth-level Heading</h6>
```

11. Paragraph Element

Definition:

The `<p>` element is used to define a paragraph in HTML. It is a block-level element that automatically adds space before and after the paragraph.

Syntax:

```
<p>Paragraph Text</p>
```

Tag Name:

`<p>`

Type of Tag:

Paired Tag (has opening and closing tags)

Purpose of the element:

Used to group text into a paragraph, making it easier to read and understand content. The browser automatically adds margins before and after the paragraph to visually separate it from other content.

Display Type:

Block-level (typically displayed with some space above and below)

Examples:

1. Simple paragraph:

```
<p>This is a paragraph of text.</p>
```

12. Structure of HTML Document

The structure of an HTML document outlines the essential components of a webpage. It includes a declaration, document scope, and key sections for content.

1. Document Declaration

The **Document Declaration** (`<!DOCTYPE html>`) is the very first line of an HTML document. It specifies the document type and version of HTML that is being used, ensuring the browser renders the content correctly.

Syntax:

```
<!DOCTYPE html>
```

Purpose:

- Declares the document type (HTML5 in modern web development) and tells the browser which version of HTML is being used.
- Ensures consistent rendering across different browsers.

2. Document Scope

The **Document Scope** consists of the basic structure of the HTML document, including the `<html>`, `<head>`, and `<body>` elements.

1. `<html>` Element

- Contains all the content of the HTML document.

Syntax:

```
<html lang="en">  
</html>
```

Purpose:

- Wraps the entire document.
- `lang="en"` specifies the document's language as English.

2. `<head>` Element

- Contains metadata about the HTML document, such as title, character encoding, styles, and links to external files like CSS and JavaScript.

Syntax:

```
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Document Title</title>
</head>
```

Purpose:

- Provides information that isn't visible on the webpage but is necessary for proper functioning.
- Specifies the character set and viewport settings.

3. <body> Element

- Contains the content that is displayed on the webpage, including text, images, links, and other media.

Syntax:

```
<body>
  <h1>Main Heading</h1>
  <p>Content goes here.</p>
</body>
```

Purpose:

- Holds the visible part of the HTML document.
- Everything inside <body> is rendered to the browser.

Example of Full HTML Document Structure:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Sample Webpage</title>
  </head>
  <body>
    <h1>Welcome to My Webpage</h1>
    <p>This is a simple paragraph of text.</p>
  </body>
</html>
```

13. List Element

The list elements are used to group a set of related items, making it easier to organize content in HTML. There are three main types of lists in HTML: Ordered List, Unordered List, and Description List.

1. Ordered List

An **Ordered List** is a list where each item is numbered or ordered based on a specific sequence.

Definition:

The `` element is used to create an ordered (numbered) list in HTML.

Syntax:

```
<ol>
  <li>First item</li>
  <li>Second item</li>
  <li>Third item</li>
</ol>
```

Tag Name:

``

Type of Tag:

Paired Tag (has opening and closing tags)

Purpose of the element:

Used to represent an ordered list of items, where the order of the list matters. Each list item is marked with the `` tag. By default, items are numbered (1, 2, 3, etc.), but other numbering styles can be customized with CSS.

i). List Item (Inside Ordered List)**Definition:**

The `` element is used to define a list item inside an ordered or unordered list.

Syntax:

```
<li>List item text</li>
```

Tag Name:

``

Type of Tag:

Paired Tag (has opening and closing tags)

Purpose of the element:

Represents an individual item within a list. It can be used in ``, ``, and `<menu>` elements.

2. Unordered List

An **Unordered List** is a list where each item is marked with bullet points instead of numbers.

Definition:

The `` element is used to create an unordered (bulleted) list in HTML.

Syntax:

```
<ul>
  <li>First item</li>
  <li>Second item</li>
  <li>Third item</li>
</ul>
```

Tag Name:

``

Type of Tag:

Paired Tag (has opening and closing tags)

Purpose of the element:

Used to represent a list of items where the order doesn't matter. Items are typically displayed with bullet points by default, but this can be customized using CSS.

i). List Item (Inside Unordered List)**Definition:**

The `` element is used to define a list item inside an unordered list.

Syntax:

```
<li>List item text</li>
```

Tag Name:

``

Type of Tag:

Paired Tag (has opening and closing tags)

Purpose of the element:

Represents an individual item in a list. It can be used in ``, ``, and `<menu>` elements.

3. Description List

A **Description List** is used to define a list of terms and their associated descriptions, making it suitable for glossaries or any content where you need to pair terms with definitions.

Definition:

The `<dl>` element is used to define a description list, which consists of one or more pairs of terms and their descriptions.

Syntax:

```
<dl>
  <dt>Term 1</dt>
  <dd>Definition of Term 1</dd>
  <dt>Term 2</dt>
  <dd>Definition of Term 2</dd>
</dl>
```

Tag Name:

<dl>

Type of Tag:

Paired Tag (has opening and closing tags)

Purpose of the element:

Used to create a list of terms and their corresponding definitions. The <dt> tag represents a term, and the <dd> tag represents the description of the term.

i). Data Term (Inside Description List)**Definition:**

The <dt> element is used to define a term within a description list.

Syntax:

```
<dt>Term</dt>
```

Tag Name:

<dt>

Type of Tag:

Paired Tag (has opening and closing tags)

Purpose of the element:

Used to represent a term that is being described in a <dl> (description list).

ii). Data Definition (Inside Description List)**Definition:**

The <dd> element is used to define the description or definition of a term in a description list.

Syntax:

```
<dd>Definition or description</dd>
```

Tag Name:

```
<dd>
```

Type of Tag:

Paired Tag (has opening and closing tags)

Purpose of the element:

Used to provide the description or definition of a term in a `<dl>` (description list).

Example of All List Types:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>HTML List Types</title>
</head>
<body>
  <h1>Ordered List</h1>
  <ol>
    <li>First item</li>
    <li>Second item</li>
    <li>Third item</li>
  </ol>

  <h1>Unordered List</h1>
  <ul>
    <li>First item</li>
    <li>Second item</li>
    <li>Third item</li>
  </ul>

  <h1>Description List</h1>
  <dl>
    <dt>HTML</dt>
    <dd>A markup language for creating web pages.</dd>
    <dt>CSS</dt>
    <dd>A stylesheet language used for describing the presentation of a
document written in HTML.</dd>
  </dl>
```

```
</body>  
</html>
```

14. Attributes in HTML

Purpose:

HTML attributes provide additional information about elements, helping to control their behavior and appearance.

Syntax:

```
<tagname attribute="value">Content</tagname>
```

Rules:

1. Attributes should be included inside the opening tag.
2. Attribute names are case-insensitive (use lowercase for best practice).
3. Attribute values should be enclosed in quotes.
4. Some attributes do not require values (e.g., `checked`, `disabled`).

Examples:

Example 1: Anchor Element with Attributes

```
<a href="https://google.com" target="_blank" title="Visit Example">Click  
Here</a>
```

- `href` → Specifies the link's URL.
- `target="_blank"` → Opens the link in a new tab.
- `title` → Displays a tooltip when hovered.

Example 2: Input Element with Attributes

```
<input type="email" placeholder="Enter your email" required>
```

- `type="email"` → Specifies input type.
- `placeholder` → Displays a hint inside the input box.
- `required` → Ensures input must be filled.

15. Attributes of Ordered List ()

1. **type** Attribute

Defines the numbering style.

Example 1: Uppercase Letters

```
<ol type="A">
  <li>Apple</li>
  <li>Banana</li>
</ol>
```

Output: A. Apple, B. Banana

Example 2: Roman Numerals

```
<ol type="i">
  <li>First</li>
  <li>Second</li>
</ol>
```

Output: i. First, ii. Second

2. **start** Attribute

Specifies the starting number of the list.

Example 1: Start from 5

```
<ol start="5">
  <li>Item One</li>
  <li>Item Two</li>
</ol>
```

Output: 5. Item One, 6. Item Two

Example 2: Start from 10

```
<ol start="10">
  <li>Chapter One</li>
  <li>Chapter Two</li>
</ol>
```

Output: 10. Chapter One, 11. Chapter Two

3. reversed Attribute

Displays list items in reverse order.

Example 1: Simple Reversed List

```
<ol reversed>
  <li>Step One</li>
  <li>Step Two</li>
</ol>
```

Output: 2. Step One, 1. Step Two

Example 2: Reversed with Type

```
<ol type="I" reversed>
  <li>Point A</li>
  <li>Point B</li>
</ol>
```

Output: II. Point A, I. Point B

16. Attributes of Unordered List ()

1. type Attribute (Deprecated in HTML5, but still works in some browsers)

Example 1: Disc Marker (default)

```
<ul type="disc">
  <li>Milk</li>
  <li>Bread</li>
</ul>
```

Output:

- Milk
- Bread

Example 2: Square Marker

```
<ul type="square">
  <li>Notebook</li>
  <li>Pen</li>
</ul>
```

Output:

- Notebook
- Pen

17. Markers of Ordered and Unordered Lists

1. List of Markers:

Ordered List Markers:

- Numbers (1, 2, 3)
- Uppercase letters (A, B, C)
- Lowercase letters (a, b, c)
- Roman numerals (I, II, III)

Unordered List Markers:

- Disc (•)
- Circle (○)
- Square (▪)

2. Changing the Marker (Using CSS)

Example 1: Custom Marker for Unordered List

```
<ul style="list-style-type: circle;">
  <li>First Item</li>
  <li>Second Item</li>
</ul>
```

Output:

- First Item
- Second Item

Example 2: Custom Marker for Ordered List

```
<ol style="list-style-type: lower-alpha;">
  <li>Task One</li>
  <li>Task Two</li>
</ol>
```

Output:

- a. Task One
- b. Task Two

18. Image Element ()

Purpose:

The `` element is used to embed images in a web page to enhance visual content.

Syntax:

```

```

Examples:

Example 1: Basic Image with Dimensions

```

```

Displays an image of a mountain with defined width and height.

Example 2: Image with Tooltip and Lazy Loading

```

```

- `src="beach.jpg"` → Image source file.
- `alt="Sunny Beach"` → Descriptive text if the image doesn't load.
- `title="Relaxing Beach View"` → Tooltip on hover.
- `loading="lazy"` → Improves performance by loading the image when needed.

19. Attributes of Image Element ()

The element supports several attributes to define its appearance and functionality.

1. width Attribute

Purpose: Specifies the width of the image in pixels or percentages.

Example 1:

```

```

Example 2:

```

```

2. height Attribute

Purpose: Specifies the height of the image in pixels or percentages.

Example 1:

```

```

Example 2:

```

```

3. **title** Attribute

Purpose: Provides additional information about the image (appears as a tooltip on hover).

Example 1:

```

```

Example 2:

```

```

4. **alt** Attribute

Purpose: Provides alternative text if the image cannot be displayed.

Example 1:

```

```

Example 2:

```

```

5. **src** Attribute

Purpose: Specifies the URL or path of the image.

Example 1:

```

```

Example 2:

```

```

6. **id** Attribute

Purpose: Uniquely identifies the image element.

Example 1:

```

```

Example 2:

```

```

7. **class** Attribute

Purpose: Assigns a class to style the image via CSS.

Example 1:

```

```

Example 2:

```

```

8. **name** Attribute

Purpose: Assigns a name to the image element (useful in forms).

Example 1:

```

```

Example 2:

```

```

20. Division Element (<div>)

Purpose:

The `<div>` element is a container used to group HTML elements and apply CSS styles or JavaScript functions.

Syntax:

```
<div>Content here</div>
```

Examples:

Example 1: Basic `div` Container

```
<div style="background-color: lightgrey; padding: 20px;">
  <h2>Welcome</h2>
  <p>This is a content section.</p>
</div>
```

Example 2: `div` for Layout

```
<div class="header">Header Section</div>
<div class="content">Main Content</div>
<div class="footer">Footer Section</div>
```

21. Table Element (<table>)

The `<table>` element is used to create tabular data structures in HTML.

1. Table Header (<thead>)

Defines the header section of a table.

Example:

```
<table border="1">
  <thead>
    <tr>
      <th>Name</th>
      <th>Age</th>
    </tr>
  </thead>
</table>
```

2. Table Body (<tbody>)

Contains the main content of the table.

Example:

```
<tbody>
  <tr>
    <td>John</td>
    <td>25</td>
  </tr>
</tbody>
```

3. Table Footer (<tfoot>)

Provides a summary or footer content.

Example:

```
<tfoot>
  <tr>
    <td colspan="2">Total: 2 Rows</td>
  </tr>
</tfoot>
```

4. Table Row (<tr>)

Defines a row inside a table.

Example:

```
<tr>
  <td>Item 1</td>
  <td>$10</td>
</tr>
```

5. Table Cell (<td> and <th>)

<td> for data cells, <th> for header cells.

Example:

```
<tr>
  <th>Product</th>
  <td>Phone</td>
</tr>
```

6. Caption (<caption>)

Provides a title for the table.

Example:

```
<table>
  <caption>Student Grades</caption>
  <tr>
    <th>Name</th>
    <th>Grade</th>
  </tr>
</table>
```

22. Attributes of Table Element

1. **frame** Attribute

Defines the border style around the table.

Example:

```
<table frame="box">
  <tr><td>Content</td></tr>
</table>
```

2. **rules** Attribute

Specifies the internal borders (none, rows, cols, all).

Example:

```
<table rules="rows">
  <tr><td>Item</td></tr>
</table>
```

3. **width** and **height** Attributes

Define the size of the table.

Example:

```
<table width="500" height="200">
  <tr><td>Data</td></tr>
</table>
```

4. **align** and **valign** Attributes

Align content horizontally and vertically.

Example:

```
<td align="center" valign="middle">Centered Text</td>
```

5. **colspan** Attribute

Merges multiple columns.

Example:

```
<tr>
  <td colspan="2">Merged Columns</td>
</tr>
```

23. Anchor Element (**<a>**)

The **<a>** element is used to create hyperlinks in HTML.

1. Creating Links

Purpose: Link to an external or internal webpage.

Example 1: External Link

```
<a href="https://www.google.com" target="_blank">Visit Google</a>
```

- **href** → Specifies the URL of the page.
- **target="_blank"** → Opens link in a new tab.

Example 2: Internal Link

```
<a href="about.html">About Us</a>
```

Links to another page within the same website.

2. Inter-navigation (between different pages)

Purpose: Navigate between multiple pages of a website.

Example:

```
<a href="contact.html">Contact Us</a>
```

Clicking the link will take the user to the "contact.html" page.

3. Intra-navigation (within the same page)

Purpose: Navigate to a section within the same page using anchors.

Example:

```
<a href="#section1">Go to Section 1</a>
<h2 id="section1">Section 1</h2>
<p>Welcome to this section.</p>
```

24. Form Elements (<form>)

Forms allow users to input data and submit it to a server.

1. Input Element (<input>)

Purpose: Allows users to enter data.

Example:

```
<input type="text" placeholder="Enter your name">
```

2. Select Element (<select>)

Purpose: Creates a dropdown menu.

Example:

```
<select name="country">
  <option value="us">USA</option>
  <option value="uk">UK</option>
</select>
```

3. Textarea Element (<textarea>)

Purpose: Provides a multi-line input field.

Example:

```
<textarea rows="4" cols="30" placeholder="Write your comments"></textarea>
```

4. Label Element (<label>)

Purpose: Associates a label with form elements.

Example:

```
<label for="email">Email:</label>  
<input type="email" id="email">
```

5. Optgroup Element (<optgroup>)

Purpose: Groups related options in a dropdown list.

Example:

```
<select>  
  <optgroup label="Fruits">  
    <option>Apple</option>  
    <option>Banana</option>  
  </optgroup>  
</select>
```

6. Option Element (<option>)

Purpose: Represents individual options within a <select> dropdown.

Example:

```
<option value="car">Car</option>
```

7. Datalist Element (<datalist>)

Purpose: Provides predefined suggestions for input fields.

Example:

```
<input list="browsers" name="browser">
<datalist id="browsers">
  <option value="Chrome">
  <option value="Firefox">
</datalist>
```

25. Attributes of Input Element

1. **type** Attribute

Specifies the type of input (text, email, password, etc.).

Example:

```
<input type="email">
```

2. **placeholder** Attribute

Provides a hint inside the input field.

Example:

```
<input type="text" placeholder="Enter your name">
```

3. **required** Attribute

Ensures the field must be filled before form submission.

Example:

```
<input type="text" required>
```

4. **name** Attribute

Assigns a name to the input field for form submission.

Example:

```
<input type="text" name="username">
```

5. **value** Attribute

Sets a default value for the input.

Example:

```
<input type="text" value="John Doe">
```

6. **minLength** and **maxLength** Attributes

Specifies the minimum and maximum number of characters.

Example:

```
<input type="text" minlength="5" maxlength="10">
```

7. **multiple** Attribute

Allows multiple values (useful for file uploads).

Example:

```
<input type="file" multiple>
```

8. **checked** Attribute

Specifies whether a checkbox or radio button is selected by default.

Example:

```
<input type="checkbox" checked>
```

9. **disabled** Attribute

Disables the input field.

Example:

```
<input type="text" disabled>
```

10. **readonly** Attribute

Makes the input field non-editable.

Example:

```
<input type="text" value="Read Only" readonly>
```

11. **pattern** Attribute

Specifies a regex pattern for input validation.

Example:

```
<input type="text" pattern="[A-Za-z]{3,}" title="Only letters allowed, min 3 characters">
```

12. **size** Attribute

- Defines the visible width of the input field.

Example:

```
<input type="text" size="40">
```

26. Media Elements

a) Audio Element

Definition:

The `<audio>` element is used to embed audio content in a web page. It supports multiple formats such as MP3, WAV, and OGG.

Syntax:

```
<audio controls>
  <source src="audio-file.mp3" type="audio/mpeg">;
</audio>
```

Examples:

1. Basic usage with MP3 file:

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

2. Adding multiple formats for better browser support:

```
<audio controls>
  <source src="song.mp3" type="audio/mpeg">
  <source src="song.ogg" type="audio/ogg">
  Your browser does not support the audio tag.
</audio>
```

b) Video Element

Definition:

The `<video>` element is used to embed video content into a web page. It supports formats like MP4, WebM, and OGG.

Syntax:

```
<video width="320" height="240" controls>
  <source src="video-file.mp4" type="video/mp4">
  Your browser does not support the video tag.
</video>
```

Examples:

1. Basic video with controls:

```
<video controls>
  <source src="movie.mp4" type="video/mp4">
</video>
```

2. Adding poster image and autoplay:

```
<video width="400" controls autoplay poster="poster.jpg">
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
  Your browser does not support the video tag.
</video>
```

c) Source Element

Definition:

The `<source>` element is used to specify multiple media files for the `<audio>` or `<video>` elements, allowing the browser to choose the best option.

Syntax:

```
<video controls>
  <source src="example.mp4" type="video/mp4">
  <source src="example.ogg" type="video/ogg">
</video>
```

Examples:

1. Using multiple audio formats:

```
<audio controls>
  <source src="sound.mp3" type="audio/mpeg">
  <source src="sound.ogg" type="audio/ogg">
</audio>
```

2. Using multiple video formats:

```
<video controls>
  <source src="clip.mp4" type="video/mp4">
  <source src="clip.webm" type="video/webm">
</video>
```

27. Semantic Elements

a) Header Element

Definition:

The `<header>` element represents introductory content at the beginning of a page or section.

Syntax:

```
<header>
  <h1>Welcome to My Website</h1>
</header>
```

Examples:

1. Basic header:

```
<header>
  <h1>My Blog</h1>
  <p>A place to share thoughts</p>
</header>
```

2. Header with navigation:

```
<header>
  <h1>Company Name</h1>
  <nav>
    <a href="#">Home</a>
    <a href="#">About</a>
  </nav>
</header>
```

b) Main Element

Definition:

The `<main>` element represents the dominant content of the document. It should not contain repeated content like sidebars or footers.

Syntax:

```
<main>
  <p>Main content goes here.</p>
</main>
```

Examples:

1. Simple main section:

```
<main>
  <h2>Article Title</h2>
  <p>Article content...</p>
</main>
```

2. Main with article sections:

```
<main>
  <article>
    <h2>News</h2>
    <p>Latest updates...</p>
  </article>
</main>
```

c) Footer Element

Definition:

The `<footer>` element contains footer information such as copyright, links, and contact details.

Syntax:

```
<footer>
  <p>(c) 2024 My Website</p>
</footer>
```

Examples:

1. Simple footer:

```
<footer>
  <p>Contact us at info@example.com</p>
</footer>
```

2. Footer with links:

```
<footer>
  <p>&copy; 2024 My Website</p>
  <a href="privacy.html">Privacy Policy</a>
</footer>
```

d) Nav Element

Definition:

The `<nav>` element is used to define a set of navigation links.

Syntax:

```
<nav>
  <ul>
    <li><a href="#">Home</a></li>
    <li><a href="#">About</a></li>
  </ul>
</nav>
```

Examples:

1. Basic navigation:

```
<nav>
  <a href="#">Home</a>
  <a href="#">Services</a>
</nav>
```

2. Navigation with unordered list:

```
<nav>
  <ul>
    <li><a href="#">Portfolio</a></li>
    <li><a href="#">Contact</a></li>
  </ul>
</nav>
```

e) Menu Element

Definition:

The `<menu>` element represents a list of commands or options.

Syntax:

```
<menu>
  <li><a href="#">Item 1</a></li>
</menu>
```

Examples:

1. Simple menu:

```
<menu>
  <li>Option 1</li>
  <li>Option 2</li>
</menu>
```

2. Menu with links:

```
<menu>
  <li><a href="#">Settings</a></li>
  <li><a href="#">Profile</a></li>
</menu>
```

f) Section Element

Definition:

The `<section>` element groups related content together.

Syntax:

```
<section>
  <h2>Title</h2>
  <p>Content here...</p>
</section>
```

Examples:

1. Section for articles:

```
<section>
  <h2>About Us</h2>
  <p>Company history...</p>
</section>
```

2. Section with multiple subsections:

```
<section>
  <h2>Our Services</h2>
  <p>Web Development</p>
</section>
```

g) Aside Element

Definition:

The `<aside>` element contains content related to the main content, such as sidebars.

Syntax:

```
<aside>
  <p>Related links here.</p>
</aside>
```

Examples:

1. Sidebar with links:

```
<aside>
  <h3>Related Articles</h3>
</aside>
```


2. Ad section:

```
<aside>
  <p>Advertisement</p>
</aside>
```

h) Details & Summary Elements

Definition:

The `<details>` element creates a collapsible section, and `<summary>` defines the visible heading.

Syntax:

```
<details>
  <summary>Click to expand</summary>
  <p>Hidden content here...</p>
</details>
```

Examples:

1. FAQ section:

```
<details>
  <summary>What is HTML?</summary>
  <p>HTML is a markup language.</p>
</details>
```

2. Expanding product details:

```
<details>
  <summary>Product Features</summary>
  <ul><li>Feature 1</li></ul>
</details>
```