

# 1. Introduction to HyperText Markup Language (HTML)

---

Hyper Text Markup Language, commonly known as HTML, serves as the fundamental markup language utilized in the creation of web pages. It plays a pivotal role in establishing the structural framework for web documents, making it an indispensable component in the realm of web development. Acquiring a comprehensive understanding of HTML is imperative for individuals aspiring to construct websites or pursue a career in web design.

When delving into the realm of HTML, it is essential to consider several key aspects:

- **Structural Components:** HTML employs a diverse array of elements or tags to delineate the layout of a web page. Each element comprises an opening tag, content, and a corresponding closing tag.
- **Elementary Components:** Noteworthy HTML elements encompass headings (ranging from `<h1>` to `<h6>`), paragraphs (`<p>`), hyperlinks (`<a>`), images (`<img>`), and lists (`<ul>` for unordered lists and `<ol>` for ordered lists).
- **Attribute Integration:** HTML elements have the capacity to incorporate attributes that furnish supplementary details about the element. For instance, the `href` attribute within a link element specifies the URL to which the link directs.
- **Document Organization:** A rudimentary HTML document encompasses a `<!DOCTYPE>` declaration, succeeded by the `<html>`, `<head>`, and `<body>` segments. The `<head>` section houses meta-information, while the `<body>` segment contains the content exhibited on the web page.
- **Semantic HTML:** Leveraging semantic HTML elements (such as `<header>`, `<footer>`, `<article>`, and `<section>`) contributes to enhancing accessibility and search engine optimization (SEO) by conferring significance to the content.

HTML is frequently utilized in conjunction with Cascading Style Sheets (CSS) and JavaScript to craft visually captivating and interactive web pages. While HTML establishes the structure, CSS is employed for styling, and JavaScript imparts functionality.

To encapsulate, HTML stands as a foundational technology in the realm of web development. Proficiency in HTML serves as the initial stride towards fabricating compelling and captivating web experiences.

## 2. History of HTML

---

HTML, or HyperText Markup Language, is the standard markup language used to create web pages. Its history dates back to the early 1990s and has evolved significantly over the years. Below is a comprehensive overview of the key milestones in the development of HTML.

- **1989 - The Concept:** Tim Berners-Lee, a British computer scientist, proposed a system for sharing information over the internet, which included the idea of using hypertext to link documents.
- **1991 - HTML 1.0:** The first version of HTML was released, which included basic elements such as headings, paragraphs, links, and lists. This version laid the groundwork for web development.

- **1995 - HTML 2.0:** The Internet Engineering Task Force (IETF) published HTML 2.0, which standardized the language and included support for forms, tables, and other features that enhanced web interactivity.
- **1997 - HTML 3.2:** The World Wide Web Consortium (W3C) released HTML 3.2, which introduced new elements like applets, tables, and more sophisticated layout options.
- **1998 - HTML 4.0:** This version brought significant improvements, including support for scripting languages, style sheets, and internationalization. HTML 4.01 was released shortly after, providing minor updates and bug fixes.
- **2004 - XHTML 1.0:** XHTML (Extensible Hypertext Markup Language) was introduced as a reformulation of HTML 4.01 in XML. It aimed to improve the structure and consistency of web documents.
- **2014 - HTML5:** HTML5 was officially released, marking a major leap forward in web technology. It introduced new semantic elements, multimedia support (audio and video), and APIs for enhanced web applications.
- **2020 - HTML Living Standard:** The W3C transitioned to a living standard model, meaning HTML is continuously updated rather than released in distinct versions. This allows for more rapid incorporation of new features and improvements.

Throughout its history, HTML has played a crucial role in the development of the web, enabling the creation of rich, interactive, and accessible content. As web technologies continue to evolve, HTML remains a foundational element of web

### 3. HTML vs. Other Markup Languages

HTML, or HyperText Markup Language, is the standard markup language used to create web pages. It is essential for structuring content on the internet. However, there are several other markup languages that serve different purposes and have unique features. This note will explore the differences between HTML and other markup languages.

#### What is HTML?

HTML is the backbone of web development. It allows developers to create structured documents by using a series of elements and tags. Some key features of HTML include:

- Defines the structure of web pages.
- Uses tags to create elements like headings, paragraphs, links, images, and lists.
- Supports multimedia elements such as audio and video.
- Is widely supported by all web browsers.

#### Other Markup Languages

While HTML is primarily used for web development, there are other markup languages that serve different functions. Here are a few notable examples:

- **XML (eXtensible Markup Language):**
  - Designed to store and transport data.
  - Allows users to define their own tags.
  - Focuses on data structure rather than presentation.
- **Markdown:**

- Lightweight markup language for formatting plain text.
- Commonly used for writing documentation and readme files.
- Easy to convert to HTML and other formats.
- **LaTeX:**
  - Used primarily for typesetting documents, especially in academia.
  - Excellent for creating complex mathematical equations and scientific documents.
  - Not designed for web pages but can be converted to HTML.
- **YAML (YAML Ain't Markup Language):**
  - Used for data serialization, often in configuration files.
  - Emphasizes human readability.
  - Not a markup language in the traditional sense, but often compared to XML.

## Key Differences

Here are some of the main differences between HTML and other markup languages:

- **Purpose:** HTML is primarily for web content, while other markup languages may focus on data representation or document formatting.
- **Tag Definition:** HTML has predefined tags, whereas languages like XML allow users to create custom tags.
- **Complexity:** HTML is relatively simple to learn, while languages like LaTeX can have a steeper learning curve due to their complexity.
- **Output:** HTML is rendered by web browsers, while other markup languages may require conversion to be displayed in a web format.

## 4. HTML Basics

---

HTML, or HyperText Markup Language, is the standard language used to create web pages. Understanding the basics of HTML is essential for anyone looking to build or design websites. This note covers the fundamental aspects of HTML, including syntax, document structure, tags, elements, attributes, and commonly used tags.

### HTML Syntax

HTML syntax consists of a series of elements that are used to structure content on the web. The basic syntax includes:

```
<html>
<head>
  <title>Title of the document</title>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
</head>
<body>
  <h1>This is a heading</h1>
  <p>This is a paragraph.</p>
</body>
</html>
```

- Elements are defined by tags, which are enclosed in angle brackets (e.g., ).
- Most elements have an opening tag and a closing tag (e.g., content ).
- Some elements are self-closing and do not require a closing tag (e.g., ).
- Attributes provide additional information about an element and are included in the opening tag (e.g., ).

## Structure of an HTML Document

An HTML document has a specific structure that must be followed for it to be valid. The basic structure includes:

1. **DOCTYPE Declaration:** This declaration defines the document type and version of HTML being used (e.g., ).
2. **Element:** This is the root element that contains all other elements in the document.
3. **Section:** This section contains meta-information about the document, such as the title, character set, and links to stylesheets.
4. **Section:** This section contains the content of the document that is displayed in the web browser.

## Tags, Elements, and Attributes

In HTML, tags, elements, and attributes are the building blocks of web pages:

- **Tags:** The basic units of HTML that define elements. Tags are used to create elements.
- **Elements:** Combinations of tags and content. An element can be a single tag (self-closing) or a pair of opening and closing tags with content in between.
- **Attributes:** Additional information provided within a tag that modifies the behavior or appearance of an element. Attributes are always specified in the opening tag.

## Commonly Used HTML Tags

Here are some of the most commonly used HTML tags:

- **<html>:** The root element of an HTML page.
- **<head>:** Contains meta-information about the document.
- **<title>:** Sets the title of the document, which appears in the browser tab.
- **<body>:** Contains the content of the document, such as text, images, and links.
- **<p>:** Defines a paragraph of text.
- **<a>:** Creates a hyperlink to another page or resource.
- **<img>:** Embeds an image in the document.
- **<div>:** A container for grouping elements and applying styles.
- **<span>:** An inline container for text or other elements.

# 5. HTML Elements with Examples

---

# Headings

## HTML Headings

HTML headings are used to define the headings of a web page. Headings are important for SEO (Search Engine Optimization) and accessibility, as they help search engines and screen readers understand the structure and content of a web page.

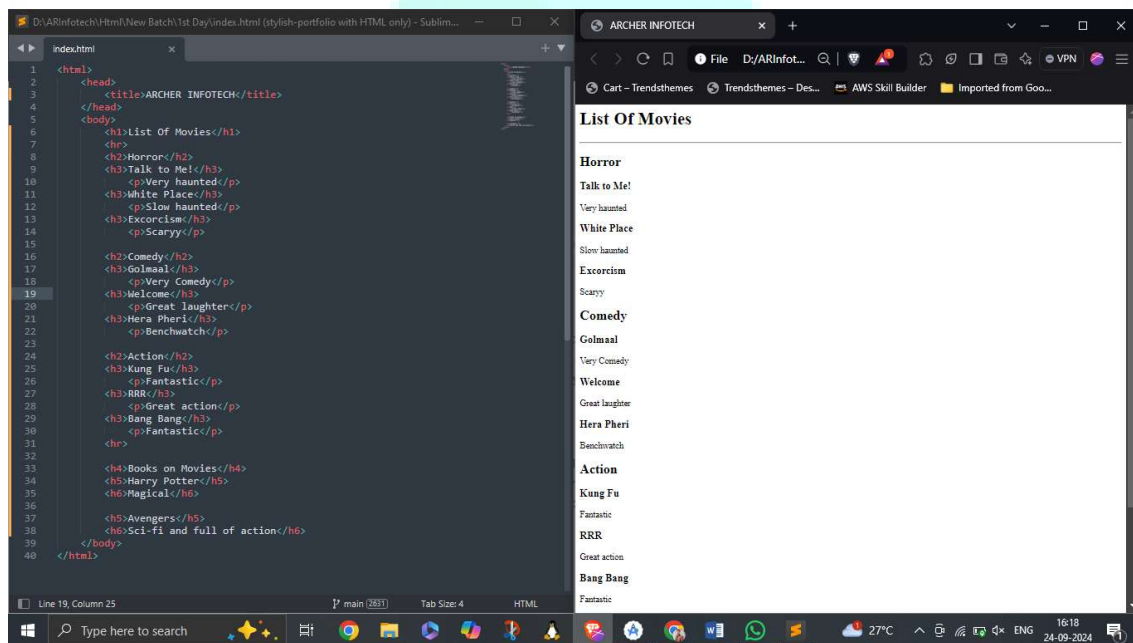
## Heading Tags

HTML provides six levels of headings, from `<h1>` to `<h6>`, with `<h1>` being the highest (most important) level and `<h6>` being the lowest (least important) level.

- Define headings from `<h1>` to `<h6>`.

Example:

- `<h1>Main Title</h1>`
- `<h2>Subheading</h2>`
- `<h3>Section Heading</h3>`
- `<h4>Subsection Heading</h4>`
- `<h5>Detail Heading</h5>`
- `<h6>Minor Detail Heading</h6>`



# Paragraphs

## HTML paragraph:

The `<p>` tag in HTML is used to define paragraphs in a web page. While its primary function is straightforward—wrapping blocks of text to create paragraphs—it can be used in various ways to enhance and structure content.

- Use `<p>` for paragraphs.

Example:

- `<p>This is a paragraph of text that provides information to the reader.</p>`

## Hyperlinks

These are build using `<a>` tag. Formally known as anchor tag, hyperlink defines a link `<a href="">` in html document. A hyperlink connect a webpage with other webpages or external pages. Hyperlink can be used as Internal Link, External Link, Email Link or telephone link .

In HTML5, an hyperlink can contain both block level and inline level elements. That means, we can write text, image or a div inside hyperlink.

By default, links will appear as follow in most of the browsers:

- An unvisited link is underlined and blue.
- A visited link is underlined and purple.
- An active link is underlined and red.

**Anchor Tag:** Anchor Tag was the previous name of hyperlinks till html4/xhtml. But HTML5 renamed anchor tag to hyperlink.

syntax: `<a>Anchor Tag</a>`

**Hyperlink:** Hyperlink means an a tag with href attribute. Hyperlink is used to link webpages.

Hyperlink Tag: `<a href="">Hyperlink Tag</a>`

**Hyperlink Attributes:** Hyperlinks included both compulsory and recommended attributes. Here is a attributes used in hyperlinks.

- href: http path hyperlink
- target: target of hyperlink, by-default its `_self`.
  - `_self`: Default. Opens the link in the same window/tab.
  - `_blank`: Opens the link in a new tab/window.
  - `_parent`: Opens the link in the parent frame.
  - `_top`: Opens the link in the full body of the window.
- download: to download hyperlink path instead of opening
- rel: `rel="noreferrer"` or `rel="noopener"` for external links  
(Improves security by preventing the new page from being able to access the window.opener property.)
- tabindex: change tabindex of hyperlinks

**Type of Hyperlinks:** There are six types of hyperlinks. Types of Hyperlinks are defined on the basis of their path.

Here are six types of HTML hyperlinks with example and usage.

1. **Basic Hyperlink:** Directs users to another webpage.

```
<a href="https://archerinfotech.in/">Visit Example</a>
```

2. **Open Link in a New Tab:** Opens the link in a new browser tab.

```
<a href="https://archerinfotech.in/" target="_blank">Visit  
Example</a>
```

3. **Email Link:** Opens the user's email client with a new email to the specified address.

```
<a href="mailto:someone@example.com">Send Email</a>
```

4. **Telephone Link:** Initiates a phone call on devices that support calling.

```
<a href="tel:+1234567890">Call Us</a>
```

5. **Download Link:** Allows users to download a file directly.

```
<a href="path/to/file.zip" download>Download File</a>
```

6. **Anchor Link (Jump to Section):** Jumps to a specific section within the same page.

```
<!-- Link -->  
<a href="#section1">Go to Section 1</a>
```

```
<!-- Target Section -->  
<h2 id="section1">Section 1</h2>  
<p>Content for section 1.</p>
```

7. **Button Styled Link:** Styles the link to look like a button.

```
<a href="https://archerinfotech.in/" class="button">Click Me</a>
```

```
<!-- CSS for Button -->  
<style>  
  .button {  
    display: inline-block;  
    padding: 10px 20px;  
    background-color: #4CAF50;  
    color: white;  
    text-align: center;  
    text-decoration: none;  
    border-radius: 4px;  
  }  
</style>
```



**8. Image Link:** Uses an image as a clickable link.

```
<a href="https://archerinfotech.in/">
  
</a>
```

**9. Link to a File or Document:** Provides a link to open a PDF or other document.

```
<a href="path/to/document.pdf">Read the Document</a>
```

**10. Links with Rel Attributes:** Improves security by preventing the new page from being able to access the window.opener property.

```
<a href="https://archerinfotech.in/" rel="noopener noreferrer">Visit Example</a>
```

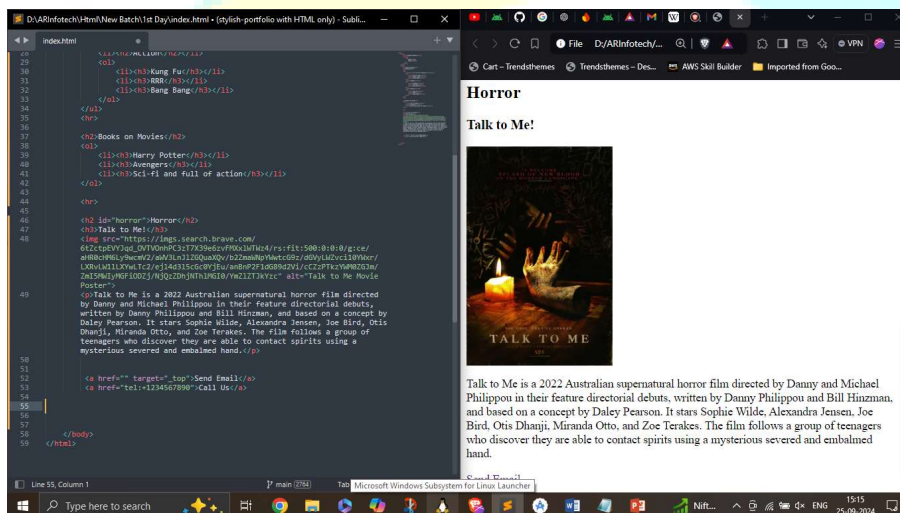
- Create clickable links with <a>.

Example:

- `<a href="https://www.example.com">Visit Example.com</a>`

## Images

- Embed images using <img>.
- Example:
- ``





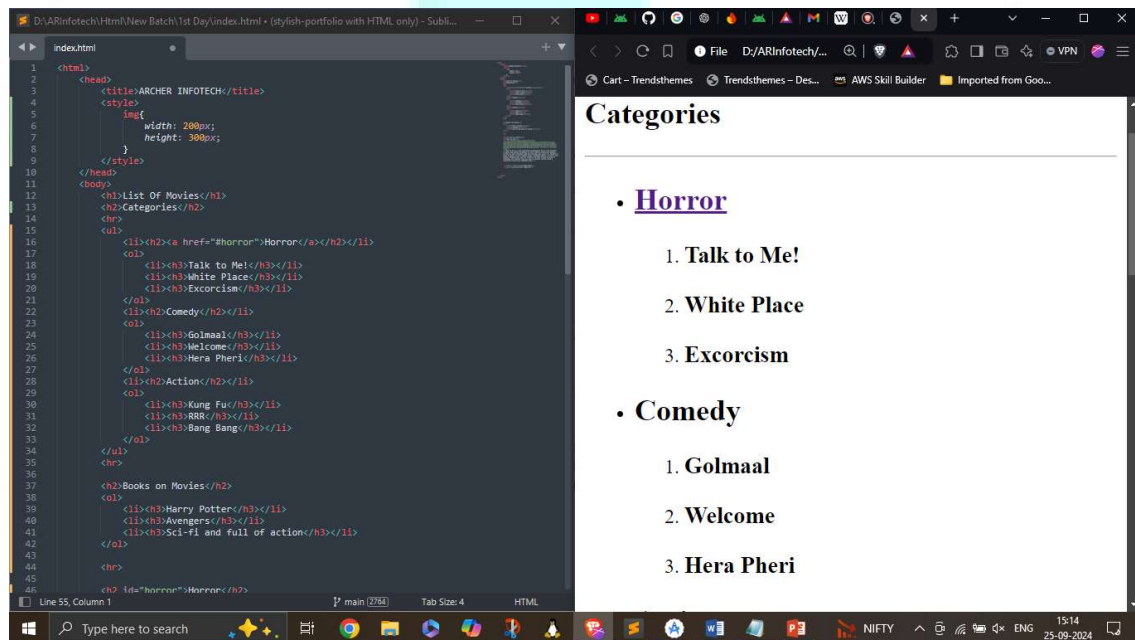
# Lists

- Ordered lists (<ol>) and unordered lists (<ul>).

Example:

```
<!-- Unordered List -->
<ul>
  <li>Item One</li>
  <li>Item Two</li>
  <li>Item Three</li>
</ul>

<!-- Ordered List -->
<ol>
  <li>First Item</li>
  <li>Second Item</li>
  <li>Third Item</li>
</ol>
```



## Unordered Lists with Types

- The type attribute in <ul> changes the bullet style.
- Types include: disc, circle, and square.

Example:

```
<ul type="square">
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

## Ordered Lists with Types and Start Value

- The type attribute in `<ol>` specifies the kind of marker.
- Types include: 1, A, a, I, i.
- The start attribute changes the starting number or letter.

Example:

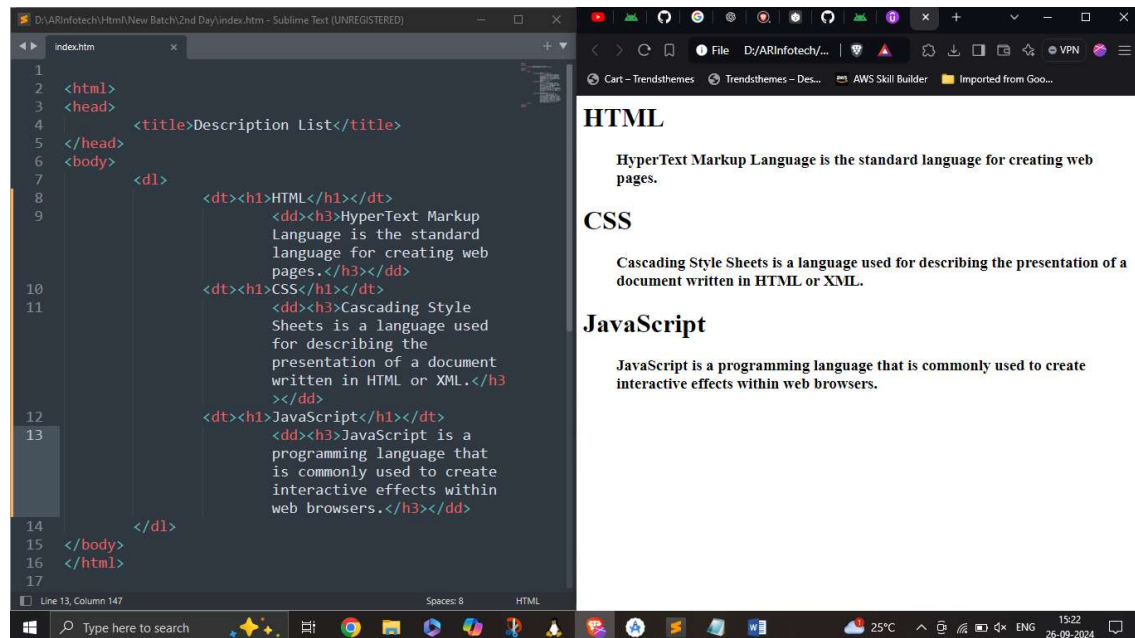
```
<ol type="A" start="3">
  <li>Third item</li>
  <li>Fourth item</li>
  <li>Fifth item</li>
</ol>
```

## Description Lists

- `<dl>` is used for a list of terms and descriptions.
- `<dt>` for the term.
- `<dd>` for the description.

Example:

```
<dl>
  <dt>HTML</dt>
  <dd>Hypertext Markup Language</dd>
  <dt>CSS</dt>
  <dd>Cascading Style Sheets</dd>
</dl>
```



Use these elements to structure your content effectively for both users and search engines.

## 6. HTML Text Formatting

---

### Bold

- `<b>` and `<strong>` make text bold.
- `<strong>` indicates importance, while `<b>` is just for visual effect.

Example:

```
<b>Bold text</b>  
<strong>Important text</strong>
```

### Italic

- `<i>` and `<em>` italicize text.
- `<em>` implies emphasis, `<i>` is for style.

Example:

```
<i>Italic text</i>  
<em>Emphasized text</em>
```

### Underline

- `<u>` underlines text.
- Use sparingly as it can be confused with links.

Example:

```
<u>Underlined text</u>
```

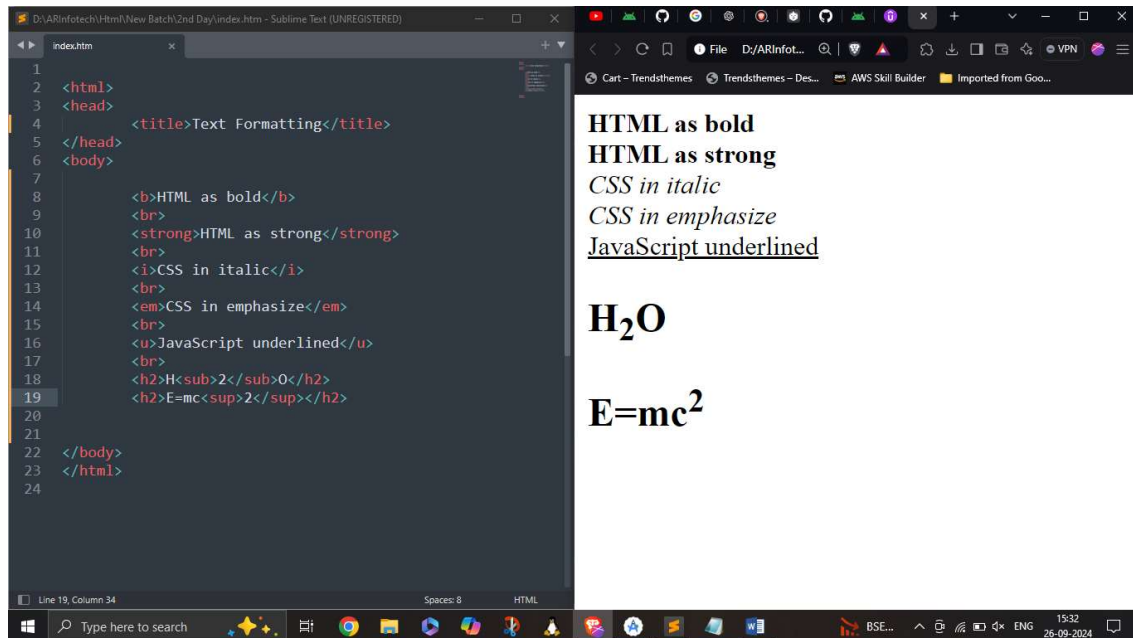
### Superscript and Subscript

- `<sup>` raises text above the baseline.
- `<sub>` lowers text below the baseline.
- Commonly used in mathematical or chemical formulas.

Example:

```
H<sub>2</sub>O (Water)  
E=mc<sup>2</sup> (Einstein's Theory of Relativity)
```

Use these tags to add meaning and emphasis to your text, but remember not to overdo it to maintain readability.



```
1 <html>
2 <head>
3   <title>Text Formatting</title>
4 </head>
5 <body>
6
7   <b>HTML as bold</b>
8   <br>
9   <strong>HTML as strong</strong>
10  <br>
11  <i>CSS in italic</i>
12  <br>
13  <em>CSS in emphasize</em>
14  <br>
15  <u>JavaScript underlined</u>
16  <br>
17  <h2>H<sub>2</sub></h2>
18  <h2>E=mc<sup>2</sup></h2>
19
20
21
22 </body>
23 </html>
24
```

HTML as bold  
HTML as strong  
CSS in italic  
CSS in emphasize  
JavaScript underlined  
H<sub>2</sub>O  
E=mc<sup>2</sup>

## 7. HTML Forms

### Form Element

- `<form>` wraps all the form elements.
- Attributes like `action` and `method` define form submission behavior.

Example:

```
<form action="/submit-form" method="post">
  <!-- Form inputs go here -->
</form>
```

### Input Types

- `<input>` varies based on the `type` attribute.
- Common types: `text`, `password`, `submit`, `radio`, `checkbox`.

Example:

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

```
<input type="submit" value="Login">
```

## Form Labels

- `<label>` defines labels for input elements.
- Improves accessibility when connected to inputs with `for` attribute.

Example:

```
<label for="username">Username:</label>  
<input type="text" id="username" name="username">
```

### 1. Text Input (`<input type="text">`):

- Used for single-line text input fields.
- Example: `<input type="text" id="username" name="username">`

### 2. Password Input (`<input type="password">`):

- Similar to text input but hides the entered characters (typically as asterisks or dots) for security reasons.
- Example: `<input type="password" id="password" name="password">`

### 3. Checkbox (`<input type="checkbox">`):

- Allows users to select one or more options from a list of choices.
- Example:  
html  
`<input type="checkbox" id="option1" name="option1" value="value1">`  
`<label for="option1">Option 1</label>`

### 4. Radio Button (`<input type="radio">`):

- Allows users to select only one option from a list of choices.
- Example:  
html  
`<input type="radio" id="option1" name="options" value="option1">`  
`<label for="option1">Option 1</label><br>`  
`<input type="radio" id="option2" name="options" value="option2">`  
`<label for="option2">Option 2</label><br>`

### 5. Number Input (`<input type="number">`):

- Allows users to enter a number. Typically includes controls for incrementing and decrementing the value.
- Example: `<input type="number" id="quantity" name="quantity" min="1" max="10">`

### 6. Date Input (`<input type="date">`):

- Allows users to select a date from a calendar picker.
- Example: `<input type="date" id="birthdate" name="birthdate">`

### 7. Time Input (`<input type="time">`):

- Allows users to select a time (hour and minute) from a time picker.
- Example: `<input type="time" id="meeting-time" name="meeting-time">`

### 8. Email Input (`<input type="email">`):

- Validates that the entered text is in an email format.
- Example: `<input type="email" id="email" name="email">`

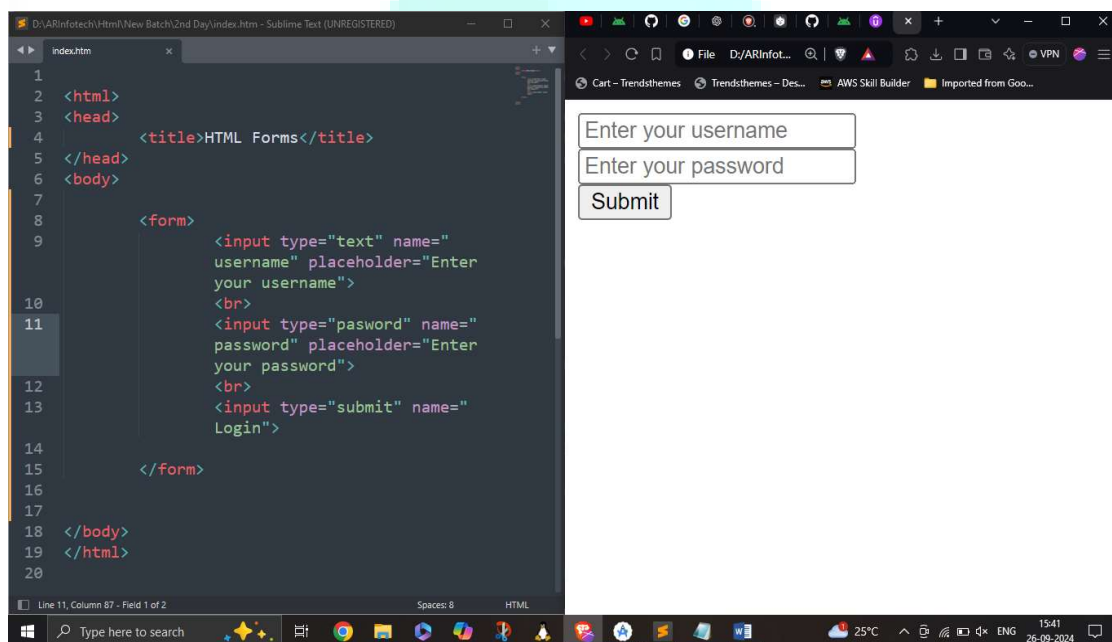
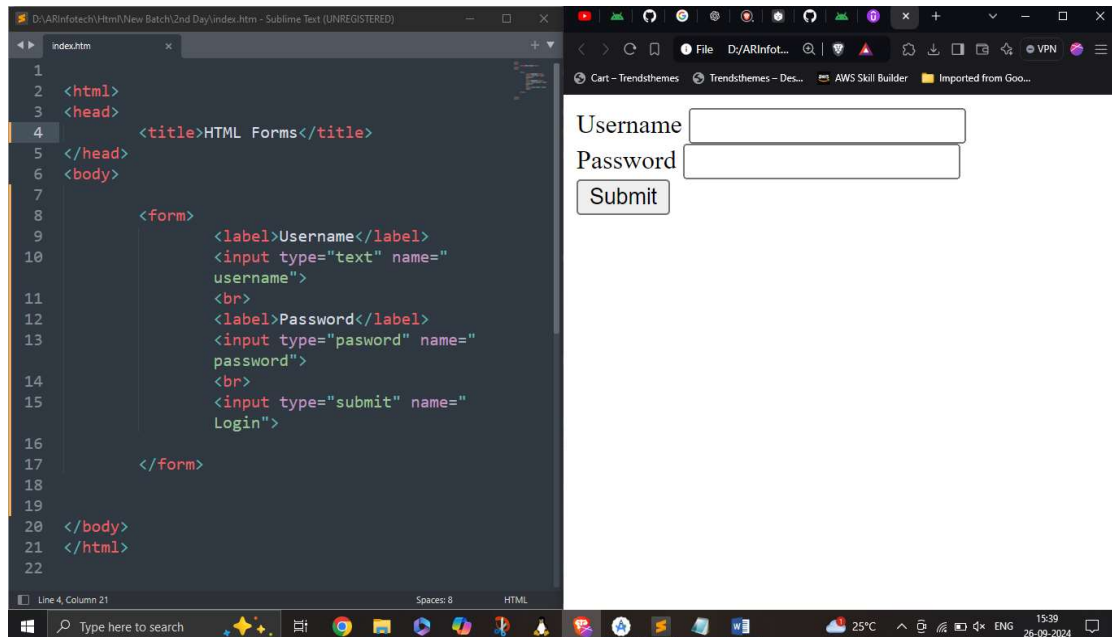
### 9. File Input (`<input type="file">`):

- Allows users to select one or more files to upload to the server.

- Example: `<input type="file" id="fileUpload" name="fileUpload">`

#### 10. Hidden Input (`<input type="hidden">`):

- Used to store a hidden value that is not displayed to the user but is sent with the form submission.
- Example: `<input type="hidden" id="secretToken" name="secretToken" value="abc123">`

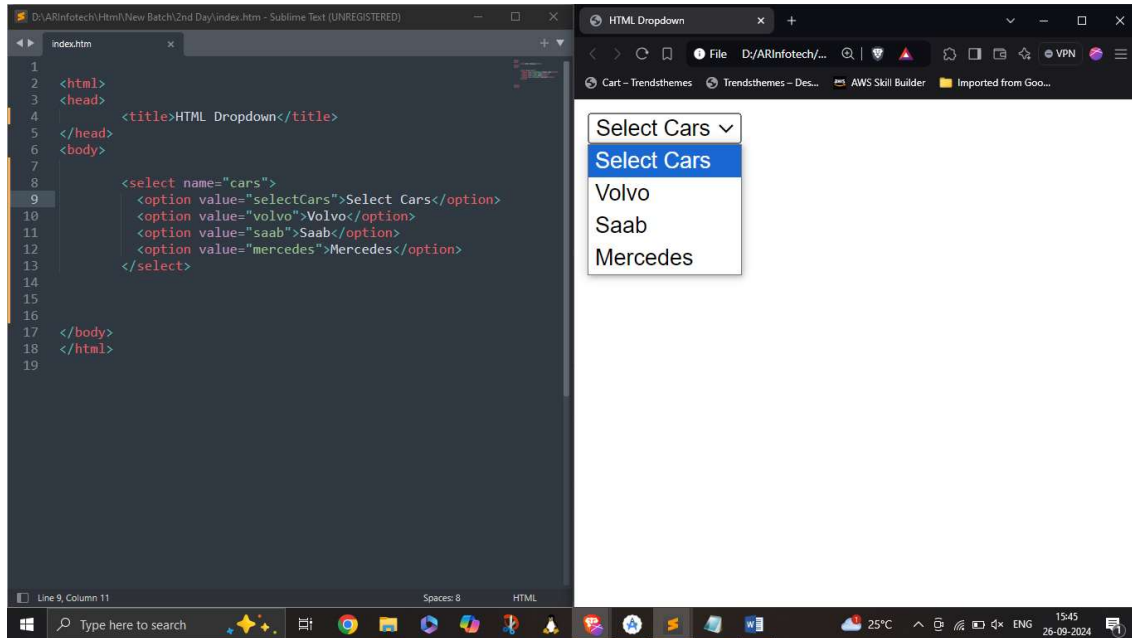


## Dropdowns

- `<select>` creates a dropdown list.
- `<option>` tags define the available options.

Example:

```
<select name="cars">
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="mercedes">Mercedes</option>
</select>
```



## Buttons

- `<button>` represents a clickable button.
- Can be used for form submission or other actions.

Example:

```
<button type="submit">Submit</button>
```

## Form Validation

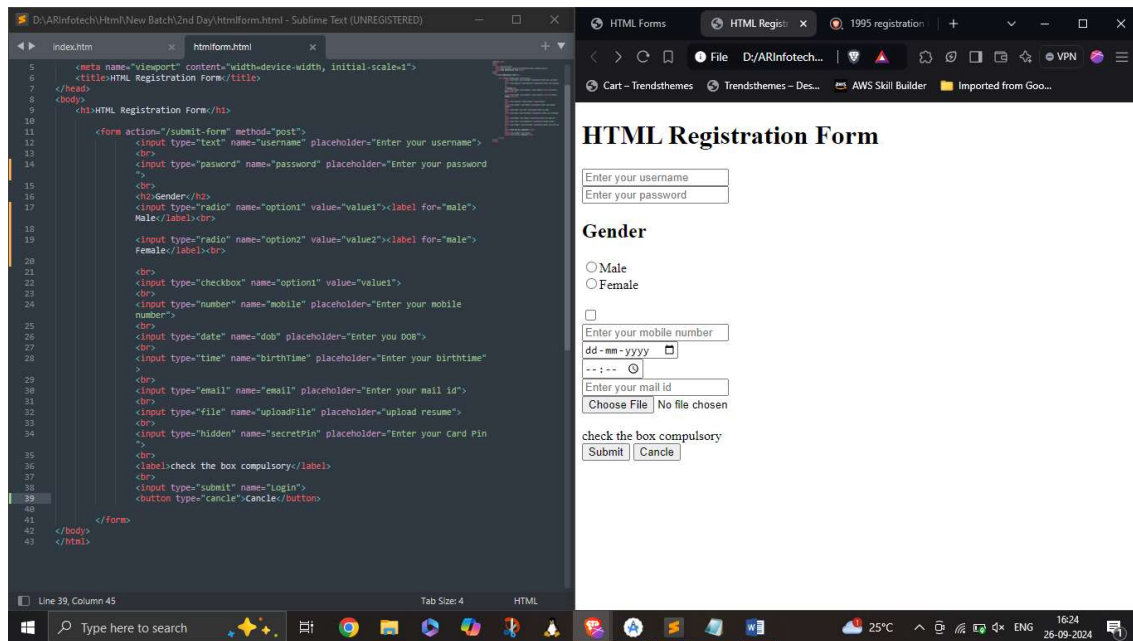
- Built-in HTML5 validation uses attributes like `required`, `minlength`, and `pattern`.
- Ensures users fill out forms correctly before submitting.

Example:

```
<input type="email" name="email" required>
<input type="text" name="phone" pattern="[0-9]{3}-[0-9]{2}-[0-9]{3}">
```

Forms are essential for user interaction, so ensure they are user-friendly and accessible.





## 8. HTML Tables

### Creating Tables

- `<table>` creates the table structure.
- `<tr>` defines a table row.
- `<td>` is used for table data cells.
- `<th>` indicates table header cells.

Example:

```
<table>
  <tr>
    <th>Header 1</th>
    <th>Header 2</th>
  </tr>
  <tr>
    <td>Data Cell 1</td>
    <td>Data Cell 2</td>
  </tr>
</table>
```

### Table Headers, Rows, and Cells

- Use `<thead>`, `<tbody>`, and `<tfoot>` to group table content.
- `<thead>` contains header rows.
- `<tbody>` wraps the main table content.
- `<tfoot>` holds the footer rows.

Example:

```
<table>
  <thead>
    <tr>
      <th>Header 1</th>
      <th>Header 2</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Row 1, Cell 1</td>
      <td>Row 1, Cell 2</td>
    </tr>
    <tr>
      <td>Row 2, Cell 1</td>
      <td>Row 2, Cell 2</td>
    </tr>
  </tbody>
</table>
```

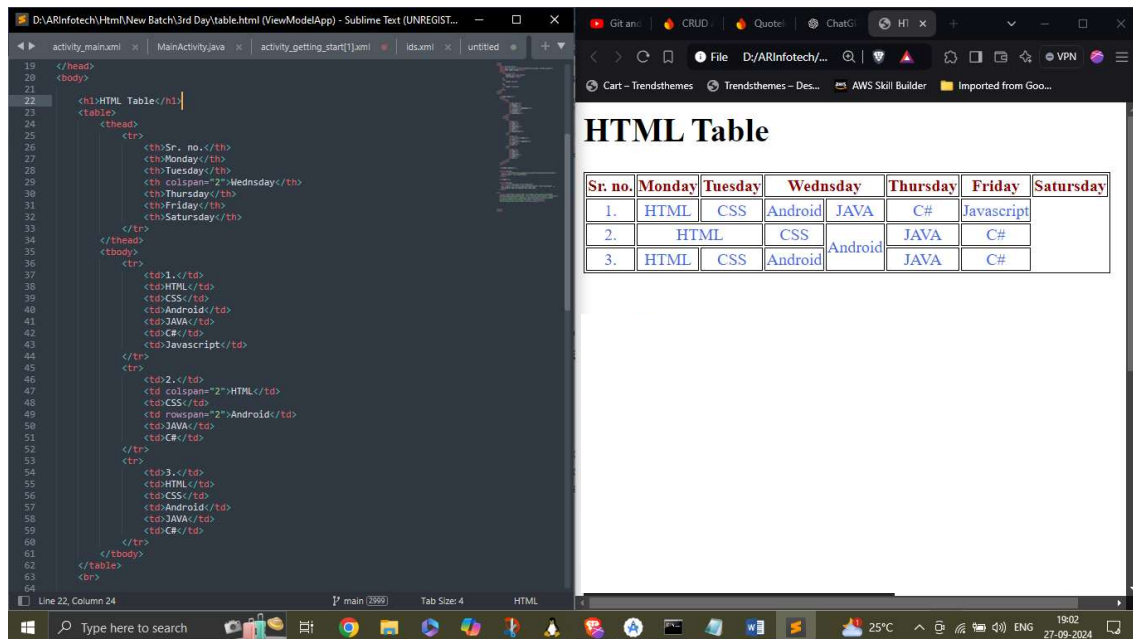
## Merging Cells

- `colspan` merges cells across columns.
- `rowspan` merges cells across rows.

Example:

```
<table>
  <tr>
    <th>Header 1</th>
    <th>Header 2</th>
  </tr>
  <tr>
    <td colspan="2">Merged Cell across two columns</td>
  </tr>
  <tr>
    <td>Regular Cell</td>
    <td rowspan="2">Merged Cell across two rows</td>
  </tr>
  <tr>
    <td>Another Regular Cell</td>
  </tr>
</table>
```

Tables are powerful for organizing data, but keep accessibility in mind by using proper markup and providing clear headers for your table data.



## 9. HTML Media

### Embedding Images

- Use `<img>` to embed images.
- `src` attribute specifies the image URL.
- `alt` attribute provides alternative text.

Example:

```

```

### Audio

- `<audio>` embeds sound content.
- `controls` attribute adds audio controls like play, pause, and volume.

Example:

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

## Video

- `<video>` embeds video files.
- `controls` attribute for playback controls.
- Can specify multiple sources for different formats.

Example:

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

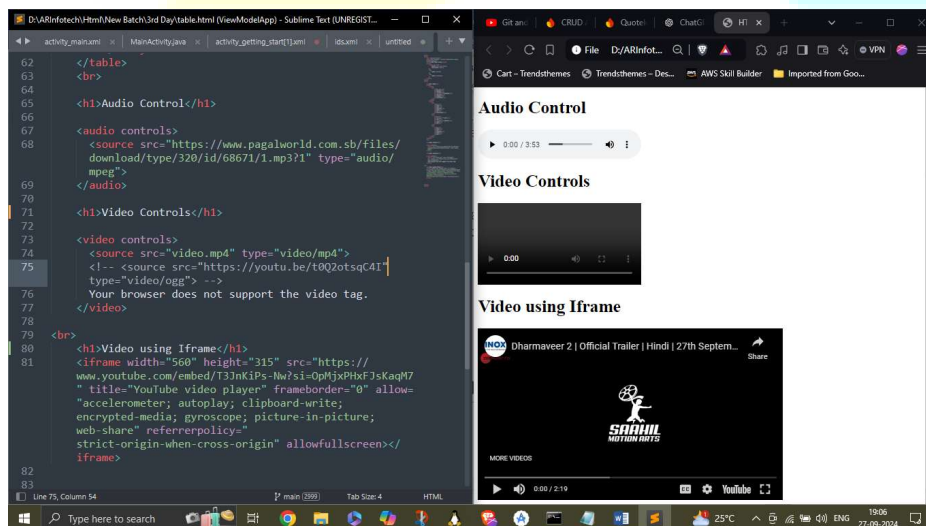
## Embedding External Media

- `<iframe>` embeds external web pages, videos, maps, etc.
- Set the `src` to the URL of the media or page.

Example:

```
<iframe width="560" height="315"
src="https://www.youtube.com/embed/dQw4w9WgXcQ" frameborder="0"
allowfullscreen></iframe>
```

When embedding media, ensure that it enhances the user experience and is accessible to all users. Always provide fallback content for browsers that do not support certain media types.



## 10. HTML Links and Navigation

---

### Creating Links

- `<a>` defines a hyperlink.
- `href` attribute specifies the URL or anchor point.

Example:

```
<a href="https://www.example.com">Visit Example.com</a>
```

### Target Attribute

- `target` attribute specifies where to open the linked document.
- `_blank` opens the link in a new window or tab.

Example:

```
<a href="https://www.example.com" target="_blank">Open in new tab</a>
```

### Link Types

- **Internal Links:** Navigate within the same webpage or website.

Example:

```
<a href="#section1">Jump to Section 1</a>
```

- **External Links:** Lead to different websites.

Example:

```
<a href="https://www.externalwebsite.com">Visit External Website</a>
```

- **Email Links:** Open the user's email client with a pre-filled recipient.

Example:

```
<a href="mailto:someone@example.com">Send Email</a>
```

# Navigation Bars

- Use a list of links for creating navigation bars.
- Often styled with CSS for better presentation.

Example:

```
<nav>
  <ul>
    <li><a href="#home">Home</a></li>
    <li><a href="#news">News</a></li>
    <li><a href="#contact">Contact</a></li>
    <li><a href="#about">About</a></li>
  </ul>
</nav>
```

Links are crucial for web navigation, so ensure they are clear and accessible. Use meaningful text for link descriptions to improve SEO and user experience.

## • Difference between HTML4 and HTML5

Feature	HTML4	HTML5
Doctype Declaration	<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">	<!DOCTYPE html> (simpler and shorter)
Multimedia Support	Limited support, requires plugins like Flash for audio and video.	Native support for audio (<audio>) and video (<video>) tags.
New Structural Elements	No dedicated tags for structure (e.g., no <article>, <section>).	Introduced new semantic tags like <article>, <section>, <header>, <footer>.
Forms Enhancements	Basic form controls with limited types.	New input types (e.g., email, date, number), form validation features.
Graphics and Media APIs	No built-in support for 2D/3D graphics.	Introduced the <canvas> element for 2D drawing and WebGL for 3D graphics.
Support for Offline Browsing	Not supported in HTML4.	HTML5 includes offline capabilities via the cache manifest and later the Service Workers API.
Mobile and Responsive Design	No native support; requires custom methods.	Designed with mobile support and responsive design in mind, with elements like <meta viewport>.
Global Attributes	Limited use of global attributes.	Introduced several new global attributes, such as contenteditable, draggable, and hidden.
JavaScript APIs	Limited API capabilities (e.g., DOM level 2).	Extended JavaScript APIs, including Geolocation, Web Storage (localStorage, sessionStorage), and Web Workers.
Character Encoding	Requires specifying charset in the meta tag, e.g., <meta http-	Simpler encoding declaration with <meta charset="UTF-8">.

## 11. HTML Semantic Elements

---

### Header

- `<header>` represents the introductory content or a set of navigational links.

Example:

```
<header>
  <h1>Website Title</h1>
  <p>Welcome to my website!</p>
</header>
```

### Footer

- `<footer>` defines the footer for a document or section.
- Contains information like authorship, copyright, and contact links.

Example:

```
<footer>
  <p>Copyright © 2023. All rights reserved.</p>
</footer>
```

### Main Content

- `<main>` specifies the main content of the document.
- Should be unique to the document and not repeated across pages.

Example:

```
<main>
  <article>
    <h2>Article Heading</h2>
    <p>This is the main article content.</p>
  </article>
</main>
```



## Sections

- `<section>` groups thematically related content.
- `<article>` holds content that stands independently, like a blog post.

Example:

```
<section>
  <h2>Section Title</h2>
  <p>Content belonging to this section.</p>
</section>

<article>
  <h2>Blog Post Title</h2>
  <p>The content of the blog post.</p>
</article>
```

## Navigation

- `<nav>` is used for major navigation blocks.
- Often contains a list of links to different pages or sections within the site.

Example:

```
<nav>
  <ul>
    <li><a href="#home">Home</a></li>
    <li><a href="#about">About</a></li>
    <li><a href="#services">Services</a></li>
    <li><a href="#contact">Contact</a></li>
  </ul>
</nav>
```

## Aside

- `<aside>` marks content that is tangentially related to the content around it.
- Used for sidebars, call-out boxes, or related links.

Example:

```
<aside>
  <h2>Related Topics</h2>
  <ul>
    <li><a href="#">Topic One</a></li>
    <li><a href="#">Topic Two</a></li>
    <li><a href="#">Topic Three</a></li>
  </ul>
</aside>
```

Semantic elements help structure the content clearly and meaningfully, improving accessibility and SEO. Use them to define the role of different parts of your web page.

## 12. HTML Meta Tags

---

### Defining Meta Tags

- `<meta>` tags provide metadata about the HTML document.
- Not displayed on the page but used by browsers and search engines.
- Placed inside the `<head>` section of the page.

Example:

```
<head>
  <meta name="description" content="A brief description of the page">
</head>
```

### Character Encoding

- Specifies the character set for the HTML document.
- UTF-8 is a common encoding that includes most characters from all languages.

Example:

```
<head>
  <meta charset="UTF-8">
</head>
```

### Viewport Settings

- Controls how a webpage is displayed on mobile devices.
- `width=device-width` sets the page width to follow the screen's width.
- `initial-scale=1.0` sets the initial zoom level.

Example:

```
<head>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
```

## SEO-Related Meta Tags

- Help improve the website's SEO.
- `name="description"` gives search engines a summary of the page content.
- `name="keywords"` lists keywords relevant to the page content (less important nowadays).

Example:

```
<head>
  <meta name="description" content="Learn about our organic gardening
services and products.">
  <meta name="keywords" content="organic gardening, eco-friendly lawn
care, natural pesticides">
</head>
```

Meta tags are essential for defining your document's metadata, which can affect how your site is represented in search results and how it behaves on different devices.

## 13. HTML Graphics

---

### Using SVG in HTML

- SVG (Scalable Vector Graphics) is used for vector-based graphics in HTML.
- Can be embedded directly into HTML using the `<svg>` element.
- Retains quality if scaled or zoomed.

Example:

```
<svg width="100" height="100">
  <circle cx="50" cy="50" r="40" stroke="green" stroke-width="4"
fill="yellow" />
</svg>
```

### The `<canvas>` Element

- Used to draw graphics via scripting (usually JavaScript).
- Defines a drawing area in the HTML document.
- Does not have drawing capabilities on its own.

Example:

```
<canvas id="myCanvas" width="200" height="100" style="border:1px solid
#000000;">
  Your browser does not support the HTML canvas tag.
</canvas>
```

**You would then use JavaScript to draw on the canvas:**

```
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.fillStyle = "#FF0000";
ctx.fillRect(0, 0, 150, 75);
```

SVG is best for complex, scalable illustrations that need to stay sharp at various sizes, while `<canvas>` is ideal for graphic-intensive games or dynamic visualizations where the image changes over time.

## 14. HTML5 New Features

---

### New Semantic Elements

- `<section>`: Groups related content.
- `<article>`: Contains self-contained content, like blog posts.
- `<aside>`: For tangential content, like sidebars.
- `<nav>`: For navigation links.

Example:

```
<article>
  <h2>Blog Post Title</h2>
  <p>The content of the blog post.</p>
</article>
```

### HTML5 Form Enhancements

- New input types: email, date, time, url, search, etc.
- Attributes like placeholder, required, autofocus, and autocomplete.

Example:

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

## HTML5 Audio and Video

- `<audio>` and `<video>` tags for embedding media without third-party plugins.
- Support for multiple source files to ensure cross-browser compatibility.

Example:

```
<video controls>
  <source src="movie.mp4" type="video/mp4">
  Your browser does not support the video tag.
</video>
```

## Local Storage and Web Storage

- Allows websites to store data on the client's computer persistently.
- Data is stored in key/value pairs and can be accessed using JavaScript.

Example:

```
localStorage.setItem("username", "JohnDoe");
var user = localStorage.getItem("username");
```

## Geolocation API

- Lets web applications retrieve the geographical location of a user.
- Requires user's permission to provide location data.

Example:

```
if ("geolocation" in navigator) {
  navigator.geolocation.getCurrentPosition(function(position) {
    console.log("Latitude is :", position.coords.latitude);
    console.log("Longitude is :", position.coords.longitude);
  });
}
```

HTML5 introduced many features that enhanced the functionality of web pages, making them more interactive and user-friendly.

## 15. HTML Best Practices

---

## Accessibility in HTML

- Use semantic elements to convey meaning and structure.
- Ensure all images have descriptive `alt` text.
- Provide transcripts for audio and captions for videos.
- Use `<label>` for form inputs for better screen reader support.

Example:

```

```

## Semantic HTML

- Choose elements that accurately describe their purpose.
- Use `<header>`, `<footer>`, `<article>`, and `<section>` to structure content.
- Avoid using non-semantic elements like `<div>` or `<span>` where a semantic element is appropriate.

Example:

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

## Proper Use of IDs and Classes

- Use `id` for unique elements on the page, as they must be unique within the document.
- Use `class` for elements that share styling or behavior.
- Keep names descriptive and consistent.

Example:

```
<div id="navigation-bar" class="menu">
  <!-- Menu items -->
</div>
```

Following these best practices will help ensure your HTML documents are accessible, maintainable, and semantically correct.

## 16. HTML and CSS Integration

---

## Adding CSS Styles to HTML

- `<style>` for internal styles within the HTML document.
- `<link>` to attach external CSS files.

Example:

```
<head>
  <!-- Internal CSS -->
  <style>
    body { background-color: #f0f0f0; }
  </style>

  <!-- External CSS -->
  <link rel="stylesheet" href="styles.css">
</head>
```

## Inline, Internal, and External CSS

- **Inline CSS:** Directly in the `style` attribute of HTML elements. Limited scope and not recommended for styling multiple elements.

Example:

```
<p style="color: blue;">This is an inline styled paragraph.</p>
```

- **Internal CSS:** Within `<style>` tags in the `<head>` section. Affects only the current page.
- **External CSS:** Separate `.css` file linked via `<link>`. Best for styling multiple pages consistently.

## Basic CSS Selectors

- **Element Selector:** Targets HTML elements directly.

Example: `p { color: red; }`

- **Class Selector:** Targets elements with a specific class attribute.

Example: `.highlight { background-color: yellow; }`

- **ID Selector:** Targets a unique element with a specific id.



Example: `#navbar { border: 1px solid black; }`

- **Attribute Selector:** Targets elements with a specific attribute or attribute value.

Example: `input[type="text"] { border-color: grey; }`

Integrating CSS with HTML is essential for creating visually appealing websites. Use the appropriate method of adding styles depending on the scope and scale of your project.

## 17. HTML Templates and Boilerplate

---

### HTML Boilerplate Code

- A standard set of HTML code that forms the basis of any web page.
- Includes doctype, head, title, meta tags, and body structure.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document Title</title>
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <header></header>
  <nav></nav>
  <main></main>
  <footer></footer>
  <script src="scripts.js"></script>
</body>
</html>
```

### HTML Template Tags

- `<template>` holds HTML code that is not rendered until called upon by JavaScript.
- Useful for reusable components or when injecting content dynamically.

Example:

```
<template id="my-template">
  <div class="user-card">
```

```
<img src="" alt="User Avatar">
<p class="username"></p>
</div>
</template>
```

You can then use JavaScript to clone and insert the template content:

```
const template = document.getElementById('my-template').content;
const userCard = template.cloneNode(true);
userCard.querySelector('.username').innerText = 'John Doe';
document.body.appendChild(userCard);
```

Using boilerplate code helps in setting up the basic structure of a webpage quickly, while `<template>` tags are great for managing and cloning sections of code that you plan to use multiple times.

## Assignment Questions:

### 1. Basic Structure of an HTML5 Document

- **Assignment:** Create a basic HTML5 webpage that includes the following sections: a 'header', a 'footer', a 'nav', a 'section', and an 'article'. Use the correct semantic tags for each section. Add a relevant title for your page.
- **Goal:** Understand the structure and use of HTML5 semantic elements.

### 2. HTML5 Forms

- **Assignment:** Create a registration form with input fields for name, email, password, date of birth, gender (radio buttons), hobbies (checkboxes), and a "Submit" button. Use HTML5 form validation features such as 'required', 'minlength', and 'email' type.
- **Goal:** Learn to create and validate forms using HTML5 attributes.

### 3. Multimedia in HTML5 (Audio and Video)

- **Assignment:** Embed an audio file and a video file into a webpage. Use controls such as play, pause, volume, and display custom text for browsers that do not support these tags.
- **Goal:** Understand how to embed multimedia files using the 'audio' and 'video' elements in HTML5.

### 4. HTML5 Canvas

- **Assignment:** Create a simple drawing using the HTML5 'canvas' element. Draw shapes like rectangles, circles, and lines using JavaScript. Add text to the canvas and change its font style.
- **Goal:** Learn how to draw basic graphics using the 'canvas' API.

### 5. HTML5 SVG (Scalable Vector Graphics)

- **Assignment:** Create an SVG-based graphic on a webpage that includes a rectangle, circle, and line. Make the graphic responsive by adjusting its size when the viewport is resized.
- **Goal:** Practice embedding and manipulating SVG elements in HTML5.