HTML 5

HTML 5: Introduction to Web, Overview of Web Technologies, HTML - Introduction, HTML - Need, Case-insensitivity, Platform-independency, DOCTYPE Declaration, Types of Elements, HTML Elements - Attributes, Metadata Element, Sectioning Elements, Paragraph Element, Division and Span Elements, List Element, Link Element, Character Entities, HTML5 Global Attributes, Creating Table Elements, Table Elements - Colspan Rowspan Attributes, border, cellspacing and cellpadding attributes, Creating Form Elements, Input Elements - Attributes, Color and Date Pickers, Select and Datalist Elements, Editing Elements, Media, Iframe, Why HTML Security, HTML Injection, Clickjacking, HTML5 Attributes & Events Vulnerabilities, Local Storage Vulnerabilities, HTML5 - Cross-browser support, Best Practices For HTML Web Pages.

Introduction to Web:

What is Internet

the Internet is a global network comprised of smaller networks that are interconnected using **standardized** communication protocols. The Internet standards describe a framework known as the Internet protocol suite. This model divides methods into a layered system of protocols.

Using the internet we can send emails, photos, videos, messages to our loved ones. Or in other words, the internet is a widespread interconnected network of computers and electronics devices(that support internet). It creates a communication medium to share and get information online. If your device is connected to the Internet then only you will be able to access all the applications, websites, social media apps, and many more services. Internet nowadays is considered as the fastest medium for sending and receiving information.

Origin Of Internet: The internet came in the year 1960 with the creation of the first working model called ARPANET (Advanced Research Projects Agency). It allowed multiple computers to work on a single network that was their biggest achievement at that time. ARPANET use packet switching to communicate multiple computer systems under a single network. In October 1969, using ARPANET first message was transferred from one computer to another. After that technology continues to grow.

What is WWW:

The World Wide Web (WWW) is a network of online content that is formatted in HTML and accessed via HTTP. The term refers to all the interlinked HTML pages that can be accessed over the Internet. The World Wide Web was originally designed in 1991 by Tim Berners-Lee while he was a contractor at CERN.

The World Wide Web is most often referred to simply as "the Web." It is all the Web pages, pictures, videos and other online content that can be accessed via a Web browser. The Internet, in contrast, is the underlying network connection that allows us to send email and access the World Wide Web.

HTML 5

What is DSN :

The Domain Name System (DNS) is called the phonebook of the Internet. When a user types a domain name or website address into the address bar of the browser, the DNS server is responsible for translating the domain name to a specific IP address, driving it to the correct website.

Protocols :

What is protocols

A protocol is a set of rules and guidelines for communicating data. Rules are defined for each step and process during communication between two or more computers. Networks have to follow these rules to successfully transmit data.

Below protocols play the major role in web development

1.HTTP

2.FTP

3. SftTP

1.HTTP:

- HTTP stands for **Hypertext Transfer Protocol**.
- It is a protocol used to access the data on the World Wide Web (www).
- The HTTP protocol can be used to transfer the data in the form of plain text, hypertext, audio, video, and so on.
- This protocol is known as Hypertext Transfer Protocol because of its efficiency that allows us to use in a hypertext environment where there are rapid jumps from one document to another document.

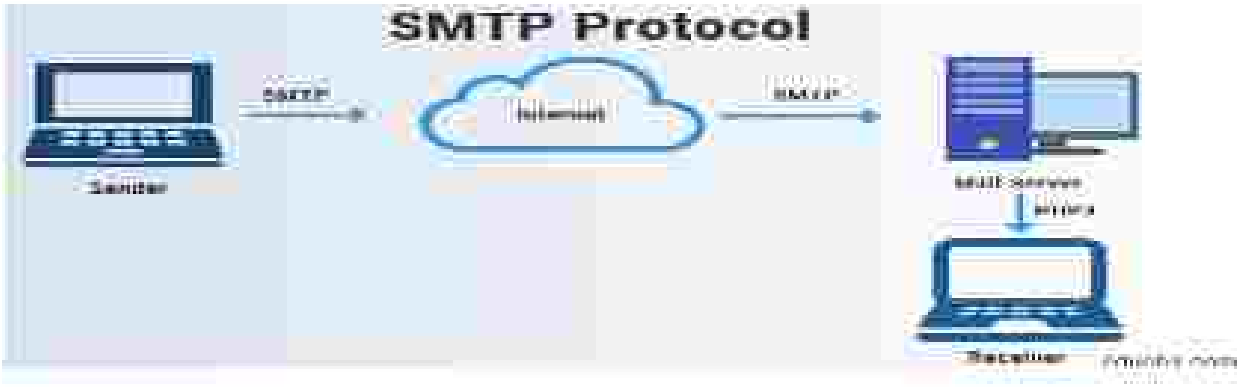
2.FTP

- FTP stands for File transfer protocol.
- FTP is a standard internet protocol provided by TCP/IP used for transmitting the files from one host to another.

- It is mainly used for transferring the web page files from their creator to the computer that acts as a server for other computers on the internet.
- It is also used for downloading the files to computer from other servers.

3.SMTP

Email is emerging as one of the most valuable services on the internet today. Most internet systems use SMTP as a method to transfer mail from one user to another. SMTP is a push protocol and is used to send the mail whereas POP (post office protocol) or IMAP (internet message access protocol) are used to retrieve those emails at the receiver's side.



HTML 5 CONCEPTS:

Html Case Sensitivity & Platform-independency:

HTML is generally case-insensitive. This means that whether you use uppercase or lowercase letters for HTML tags, attributes, and elements, the browser will usually interpret them correctly.

For example, the following two code snippets will produce the same result:

HTML
<b>This is a paragraph.</B>
<P>This is another paragraph.</P>

O/p: **This is a paragraph.**

HTML 5

This is another paragraph.

HTML Platform-independency: HTML document can be displayed correctly on a wide range of devices and operating systems, including:

- Desktop computers: Windows, macOS, Linux
- Mobile devices: iOS, Android
- Web browsers: Chrome, Firefox, Safari, Edge

Why is HTML Platform-Independent?

1. Text-Based Format: HTML is a text-based language, and text can be interpreted by various devices and software.
2. Browser Interpretation: Web browsers are responsible for rendering HTML. They have been designed to understand and display HTML consistently across different platforms.
3. Standardized Specifications: The World Wide Web Consortium (W3C) sets the standards for HTML, ensuring that browsers adhere to a common interpretation.

HTML5<!DOCTYPE> Declaration :

All HTML documents must start with a <!DOCTYPE> declaration.

The declaration is not an HTML tag. It is an "information" to the browser about what document type to expect.

In HTML 5, the declaration is simple:

<!DOCTYPE html>

Example (Basic Syntax of HTML5 DOCTYPE Declaration)

```
<!DOCTYPE html>

<html>

<head> <title>My first html 5  </title></head>

 <body>

  <data> hello it is my first html5 page  </data>

 <body> </html>
```

HTML 5

Types of elements in HTML

HTML5 introduces a wide range of elements to structure and semantically mark up web content. Here's a categorization of the key element types:

Root Element:

- <html>: Defines the root of an HTML document.

Document Metadata:

- <head>: Contains metadata about the HTML document, such as title, stylesheets, and scripts.
- <title>: Specifies a title for the document, displayed in the browser's title bar or tab.
- <meta>: Provides metadata about the HTML document, such as author, keywords, description, character set, viewport, etc.
- <link>: Links to external resources like stylesheets or other documents.
- <style>: Defines style information for a document, often used to specify CSS styles.
- <script>: Embeds client-side scripts, often JavaScript.

Sectioning Root:

- <body>: Contains the visible page content.

Content Sectioning:

- <header>: Defines a header for a document or section.
- <nav>: Defines navigation links.
- <section>: Defines a generic thematic grouping of content.
- <article>: Defines independent, self-contained content.
- <aside>: Defines content aside from the page content.
- <footer>: Defines a footer for a document or section.

Text Content:

- <p>: Defines a paragraph.
- <h1> to <h6>: Defines heading levels.
- : Defines a generic container for other elements.
- : Defines an inline element, used to group inline elements.

Image and Multimedia:

- <img>: Defines an image.
- <audio>: Defines sound content.
- <video>: Defines video content.

Embedded Content:

- <iframe>: Embeds another document within the current HTML document.
- <embed>: Embeds external content, such as a plugin.

- `<object>`: Embeds external content, such as a plugin.

Scripting:

- `<script>`: Embeds client-side scripts.

Table Content:

- `<table>`: Defines a table.
- `<caption>`: Defines a caption for a table.
- `<thead>`: Defines a header section in a table.
- `<tbody>`: Defines a body section in a table.
- `<tfoot>`: Defines a footer section in a table.
- `<tr>`: Defines a row in a table.
- `<th>`: Defines a header cell in a table.
- `<td>`: Defines a data cell in a table.

Forms:

- `<form>`: Defines an HTML form for user input.
- `<input>`: Defines an input field.
- `<label>`: Defines a label for an input field.
- `<textarea>`: Defines a multi-line text input control.
- `<select>`: Defines a drop-down list.
- `<option>`: Defines an option in a drop-down list.
- `<button>`: Defines a clickable button.
- `<fieldset>`: Groups related form elements.
- `<legend>`: Defines a caption for a `<fieldset>`.

Metadata Element:

The <meta> tag defines metadata about an HTMfL document. Metadata is data (information) about data.

<meta> tags always go inside the <head> element, and are typically used to specify character set, page description, keywords, author of the document, and viewport settings.

Metadata will not be displayed on the page, but is machine parsable.

Metadata is used by browsers (how to display content or reload page), search engines (keywords), and other web services.

There is a method to let web designers take control over the viewport (the user's visible area of a web page)

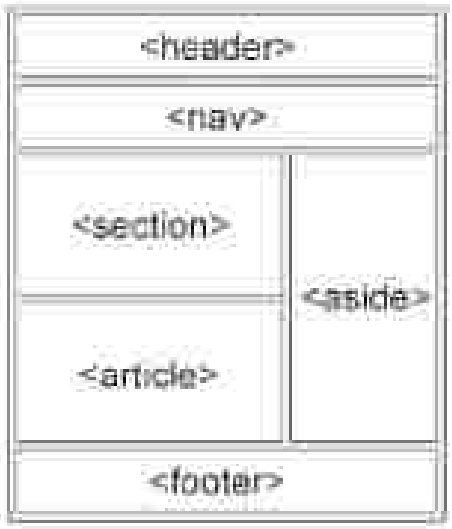
Example :

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

```
 <meta name="description" content="Free Web tutorials">
 <meta name="keywords" content="HTML, CSS, JavaScript">
 <meta name="author" content="John Doe">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
</head>
```

Sectioning Elements:

- <header>: Defines a header for a document or section.
- <nav>: Defines navigation links.
- <section>: Defines a generic thematic grouping of content.
- <article>: Defines independent, self-contained content.
- <aside>: Defines content aside from the page content.
- <footer>: Defines a footer for a document or section.



Example for above tags :

```
<!DOCTYPE html>
<html>
<head>
 <title>Content Sectioning Example</title>
</head>
<body>

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

```
    <ul>
      <li><a href="#">Home</a></li>
      <li><a href="#">About</a></li>
      <li><a href="#">Contact</a></li>
    </ul>
  </nav>
 </header>
 <section>
  <h2>About Me</h2>
  <p>This
is a paragraph about me</p>
  <p>I am a web developer.</p>
 </section>
 <aside>
  <h3>Sidebar</h3>
  <p>This is a sidebar with additional information.</p>
 </aside>
 <article>
  <h2>Latest Blog Post</h2>
  <h3>HTML5: The Future of the Web</h3>
  <p>HTML5 is the latest version of HTML. It introduces many new features and
improvements.</p>
 </article>
 <footer>
  <p>&copy; 2023 My Website</p>
 </footer>
</body>
</html>
```

O/P for above code :

Paragraph Element:

The <p> tag is used to define a paragraph. It's a fundamental element in HTML for structuring content into readable blocks of text.

Example :

```
<!DOCTYPE html>
<html>
<body>
<p>This is the first paragraph. It introduces a topic.</p>
<p>This is the second paragraph. It provides more details about the topic.</p>
</body>
</html>
```

O/P

Division and Span Elements

In HTML, and are two fundamental elements used to structure and style content on a webpage. While they might seem similar at first glance, they serve distinct purposes and have different behaviors.

Element:

- **Block-level Element:** A element is a block-level element, meaning it starts on a new line and takes up the full width available.

HTML 5

- **Purpose:**
  - **Grouping Content:** It's primarily used to group and structure larger sections of content.
  - **Applying Styles:** You can apply CSS styles to entire sections of content by targeting the .
  - **Creating Layout:** By combining elements with CSS, you can create complex layouts and page structures.

Example :

```
<div class="hero">

 <h1>Welcome to My Website</h1>

 <p>This is a brief introduction.</p>

</div>
```

Another example for better understand of the div tag

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Different Border Styles</title>
    <style>
        .solid-border {
            width: 200px;
            height: 150px;
            border: 3px solid blue;
            margin: 20px;
            padding: 10px;
            box-sizing: border-box;
        }
        .dashed-border {
            width: 200px;
            height: 150px;
            border: 5px dashed red;
            margin: 20px;
            padding: 10px;
            box-sizing: border-box;
        }
    </style>
</head>
<body>
<div class="solid-border">
        This div has a solid blue border.
    </div>
<div class="dashed-border">
        This div has a dashed red border.
    </div>
</body>
</html>
```

O/P:

<u> Element:</u>

- **Inline Element:** A element is an inline element, meaning it doesn't start on a new line and only takes up the width of its content.
- **Purpose:**
  - **Styling Text:** It's often used to style specific parts of text within a larger block of text.
  - **Adding Scripts:** You can use it to apply JavaScript to specific parts of text.

<u>Example</u>

```
<p>This is a <span style="color:red"> <u> highlighted </u></span> text.</p>
```

List Elements In HTML5:

We have three types of list

1. Ordered Lists
2. Un Ordered Lists
3. Description Lists

1. Ordered Lists:

- Used for items where the order is significant.
- Each list item is numbered.

Sample Code and Out put for Ordered Lists :

  tag for create ordered list
 : list items it is common tag in ordered and unordered list for define individual list items

```
<!DOCTYPE html>
<html lang="en">
<ol>
<li> cse-1 </li>
<li> cse-2 </li>
</ol>
</html>
```

Out put

1. cse-1
2. cse-2

2. Unordered Lists

- Used for items where the order doesn't matter.
- Each list item is marked with a bullet point.

Sample Code and Output for Unordered lists:

```
<!DOCTYPE html>
<html lang="en">
<ul>
<li> cse-1 </li>
<li> cse-2 </li>
</ul>
</html>
```

Output:

- cse-1
- cse-2

The Link Elements:

The <link> element in HTML5 is used to define the relationship between the current document and an external resource. This is primarily used to link to stylesheets and favicons, but it can also be used for other purposes.

Example :

<link rel="relationship" href="resource_url" >

Key Attributes:

- **rel:** Specifies the relationship between the current document and the linked resource.
  - **stylesheet:** Links to a CSS stylesheet.

- icon: Links to a favicon.
- **href**: Specifies the URL of the linked resource.

Example: Linking to a CSS Stylesheet

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

Character Entities in HTML5

Character entities are used to represent special characters that might be difficult to type directly into HTML code or that have special meaning in HTML. They are represented by a specific code sequence, typically starting with an ampersand (&) and ending with a semicolon (;).

Common character Entities :

Character	Entity Name	Entity Number
&	&	&
<	<	<
>	>	>
"	"	"
'	'	'
©	©	©
®	®	®
™	™	™
§	§	§
π	π	π
€	€	€

Example of Character set in html

```
<p>this is a less than symbol: &lt;</p>
<p>this is a copyright symbol: &copy;</p>
```

Output

This is a less than symbol: <

This is a copyright symbol: ©

HTML5 Global Attributes

What is the attributes in html:

HTML attributes are special words used within the opening tag of an HTML element. They provide additional information about the element and its behaviour, appearance, or functionality.

Structure of attribute:

```
<tag_name attribute_name="value">Content</tag_name>
```

Example :

Now we will take image tag with **src** and **alt** attributes

src : Source address. Specifies the path to the image

alt: Specifies an alternate text for the image

```
<!DOCTYPE html>
<html>
  <head>
    <title>Example image </title>

  </head>
  <body>
    <img src="https://assets.onecompiler.app/43253dexc/43gp5fug/logo.png" alt="logo"/>
  </body>
</html>
```

Output:

Global Attributes

Global attributes are a set of attributes that can be applied to any HTML element, providing a consistent way to add additional information or functionality. They are designed to enhance the structure, behavior, and presentation of elements across the HTML document.

Here some of the most commonly used global attributes :

HTML 5

Core Attributes:

- **id:** Assigns a unique identifier to an element. Used for styling with CSS and scripting with JavaScript.
- **class:** Assigns one or more class names to an element. Used for styling with CSS.
- **title:** Provides a tooltip-like text for an element.
- **style:** Specifies inline styles for an element.
- **lang:** Specifies the language of the element's content.
- **tabindex:** Sets the tabbing order of elements.

Event Attributes:

- **onclick:** Executes a script when the element is clicked.
- **onmouseover:** Executes a script when the mouse pointer moves over the element.
- **onmouseout:** Executes a script when the mouse pointer moves away from the element.
- **onkeydown:** Executes a script when a key is pressed.
- **onkeyup:** Executes a script when a key is released.
- **onkeypress:** Executes a script when a key is pressed and released.
- **onfocus:** Executes a script when the element receives focus.
- **onblur:** Executes a script when the element loses focus.
- **onchange:** Executes a script when the element's value changes.
- **onsubmit:** Executes a script when a form is submitted.
- **onreset:** Executes a script when a form is reset.

Example with title and class :

```
<!DOCTYPE html>
<html>
  <head>
    <title>Example Image </title>
    <style>
  .tooltip {
    position: relative;
    display: inline-block;
  }

  .tooltip:hover::after {
    content: attr(data-tooltip);
    position: absolute;
    top: 100%;
    left: 50%;
    transform: translateX(-50%);
    background-color: #333;
    color: white;
    padding: 5px 10px;
    border-radius: 5px;
    z-index: 1;
  }
</style>

  </head>
  <body>


    <!-- <img src="https://assets.w3schools.com/w3images/firework.jpg" alt="Image"> -->
    <p> <span class="tooltip" title="This is a tooltip text">Hover over me</p>

  </body>
</html>
```

Output :

Creating Table Elements in HTML5:

In HTML5, tables are created using a combination of elements to structure the data into rows and columns. Here's a basic breakdown of the key elements involved:

1. `<table>` **Element:**

- This is the root element that defines the entire table.

2. `<tr>` **Element:**

- Represents a table row. Each table must contain at least one row.

3. `<th>` **Element:**

- Defines a table header cell. These cells typically contain headings for columns or rows.

4. `<td>` **Element:**

- Defines a table data cell. These cells contain the actual data within the table.

Example :

```
<!DOCTYPE html>
<html>
    <head>
        <title>Car Information</title>
    </head>
    <body>
        <table border="1">
            <tr>
                <th>Brand Name</th>
                <th>Car Model</th>
                <th>Vehicle</th>
            </tr>
            <tr>
                <td>Hyundai</td>
                <td>i20</td>
                <td>Hatchback</td>
            </tr>
            <tr>
                <td>Tata</td>
                <td>Nexon</td>
                <td>SUV</td>
            </tr>
        </table>
    </body>
</html>
```

Output:

Brand Name	Car Model	Vehicle
Hyundai	i20	Hatchback
Tata	Nexon	SUV

Explanation:

- The `<table>` element encloses the entire table.
- The first `<tr>` element defines the header row with three `<th>` elements for the column headers.
- The subsequent `<tr>` elements define the data rows, each containing three `<td>` elements for the data.

Additional Table Elements:

- `<caption>`: Defines a caption for the table.
- `<thead>`: Groups the header rows.
- `<tbody>`: Groups the body rows.
- `<tfoot>`: Groups the footer rows.

Colspan/ Row span Attributes, border, cells pacing and cell padding attributes

Colspan and Rowspan Attributes

In HTML, the `colspan` and `rowspan` attributes are used to manipulate the layout of table cells. They allow you to merge cells horizontally or vertically, respectively.

Colspan

- **Purpose:** Merges multiple cells horizontally.
- **Syntax:** `<td colspan="number">`

Example:

```
<table border="1">
  <tr>
    <td colspan="2">Header 1 and 2</td>
  </tr>
  <tr>
    <td>Cell 1</td>
    <td>Cell 2</td>
  </tr>
</table>
```

Output:

Header 1 and 2	
Cell 1	Cell 2

In this example, the first row's cell spans across two columns.

Rowspan

- **Purpose:** Merges multiple cells vertically.
- **Syntax:** <td rowspan="number">

Example:

```
<table>
  <tr>
    <td>Header 1</td>
    <td rowspan="2">Header 2 and 3</td>
  </tr>
  <tr>
    <td>Cell 1</td>
  </tr>
</table>
```

Out Put:

Header 1	Header 2 and 3
Cell 1	

Here, the second column's cell spans across two rows.

Border, Cellspacing, and Cellpadding Attributes

Border

- **Purpose:** Sets the width of the border around each cell.
- **Syntax:** <table border="number">

```
<table border="1">
  <tr>
    <td>Cell 1</td>
    <td>Cell 2</td>
  </tr>
</table>
```

This will create a table with a 1-pixel border around each cell.

Cellspacing

- **Purpose:** Sets the space between cells.
- **Syntax:** <table cellspacing="number">

```
HTML

<table cellspacing="10">
  <tr>
    <td>Cell 1</td>
    <td>Cell 2</td>
  </tr>
</table>
```

This will create a table with a 10-pixel space between cells.

Output:

Cell 1	Cell 2

Cellpadding

- **Purpose:** Sets the space between cell content and the cell border.
- **Syntax:** <table cellpadding="number">

Example :

```
<table cellpadding="10">
  <tr>
    <td>Cell 1</td>
    <td>Cell 2</td>
  </tr>
</table>
```

This will create a table with a 10-pixel space between the cell content and the cell border.

Output :

Cell 1	Cell 2

HTML 5

Creating Form Elements

HTML forms are essential for collecting user input on websites. They consist of various elements that allow users to input text, select options, and submit information. In HTML5, we have a rich set of form elements to create interactive and user-friendly forms.

Basic Form Structure:

```
<form action="your_script.php" method="post">
  <input type="text" name="name" placeholder="Enter your name">
  <button type="submit">Submit</button>
</form>
```

Key Form Elements:

1. **<input> Element:**
    - **Type Attributes:**
        - text: Single-line text input.
        - password: Password input (masked characters).
        - email: Email input with validation.
        - url: URL input with validation.
        - tel: Telephone number input.
        - number: Numeric input.
        - date: Date input.
        - month: Month input.
        - week: Week input.
        - time: Time input.
        - datetime-local: Date and time input.
        - search: Search input.
        - color: Color picker.
        - checkbox: Checkbox input.
        - radio: Radio button input.
        - file: File upload input.
        - submit: Submit button.
        - reset: Reset button.
        - button: Generic button.
        - image: Image button.
        - hidden: Hidden input field.
2. **<textarea> Element:**
    - For multi-line text input.
3. **<select> Element:**
    - For dropdown lists.
    - Use <option> elements to define options.
4. **<label> Element:**
    - To associate a label with a form element.
5. **<fieldset> and <legend> Elements:**
    - To group related form elements.

Example:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Hello, World!</title>
    <link rel="stylesheet" href="styles.css" />
  </head>
  <body>
     <form action="process_form.php" method="post">
  <label for="name">Name:</label>
  <input type="text" id="name" name="name" required>
<br/>
  <label for="email">Email:</label>
  <input type="email" id="email" name="email" required>
<br/>
<br/>
  <label for="message">Message:</label>
  <textarea id="message" name="message" rows="5" cols="20"></textarea>
<br/>
  <input type="submit" value="Submit">
</form>
  </body>
</html>
```

Output :

Another example for form elements : Registration Page

This registration page contain Name, Email, Password, Confirm password, Date of Birth, Gender, Country, Terms and condition

HTML 5

```
<!DOCTYPE html>
<html>
<head>
 <title>Registration Form</title>
</head>
<body>
 <h2>Registration Form</h2>
 <form action="register.php" method="post">
  <label for="name">Name:</label>
  <input type="text" id="name" name="name" required>
  <label for="email">Email:</label>
  <input type="email" id="email" name="email" required>
  <label for="password">Password:</label>
  <input type="password" id="password" name="password" required pattern=".{8,}"
title="Password must be at least 8 characters">
  <label for="confirm_password">Confirm Password:</label>
  <input type="password" id="confirm_password" name="confirm_password" required
pattern=".{8,}" title="Password must be at least 8 characters">
  <label for="dob">Date of Birth:</label>
  <input type="date" id="dob" name="dob" required>
  <label for="gender">Gender:</label>
  <input type="radio" id="male" name="gender" value="male">
  <label for="male">Male</label>
  <input type="radio" id="female" name="gender" value="female">
  <label for="female">Female</label>
  <input type="radio" id="other" name="gender" value="other">
  <label for="other">Other</label>
  <label for="country">Country:</label>
  <select id="country" name="country" required>
   <option value="">Select Country</option>
   <option value="india">India</option>
```

```
    <option value="usa">USA</option>
    </select>
   <label for="terms">Terms and Conditions:</label>
   <input type="checkbox" id="terms" name="terms" required>
   <label for="terms">I agree to the terms and conditions</label>
   <button type="submit">Register</button>
  </form>
</body>
</html>
```

In above program

- **Required Attribute:** Ensures that the field must be filled.
- **Pattern Attribute:** Specifies a regular expression for input validation.
- **Title Attribute:** Provides a tooltip-like message for the input field.
- **Radio Buttons:** Allow users to select only one option from a group.
- **Select Element:** Creates a dropdown list.
- **Checkbox:** Allows users to check or uncheck an option.

Input Elements:

Input elements are fundamental building blocks of HTML forms, allowing users to input various types of data. HTML5 has introduced a rich set of input types, enhancing form functionality and user experience.

Attributes:

Input elements in HTML5 are essential for creating interactive forms. They have various attributes that enhance their functionality and customization. Here are some common attributes used with `<input>` elements in HTML5:

1. **type**: Specifies the type of input (e.g., text, password, email, number, etc.).
2. **name**: Defines the name of the input element, which is used to reference the form data after it is submitted.
3. **value**: Sets the initial value of the input element.
4. **placeholder**: Provides a hint to the user of what to enter in the input field.
5. **required**: Indicates that the input field must be filled out before submitting the form.
6. **disabled**: Disables the input element, making it uneditable and un-submittable.
7. **readonly**: Makes the input field uneditable but still submit its value.
8. **maxlength**: Sets the maximum number of characters allowed in the input field.
9. **min** and **max**: Define the minimum and maximum values for input types like number and date.
10. **pattern**: Specifies a regular expression that the input's value must match for validation.

11. **autofocus**: Automatically focuses on the input field when the page loads.
12. **multiple**: Allows multiple values (e.g., for file or email inputs).
13. **autocomplete**: Controls whether the browser should provide autocomplete suggestions.

Example Program:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Hello, World!</title>
    <link rel="stylesheet" href="styles.css">
  </head>
  <body>
    <form action="/submit" method="post">
    <label for="username">Username:</label>
    <input type="text" id="username" name="username" placeholder="Enter your username" required minlength="5">
    <label for="email">Email:</label>
    <input type="email" id="email" name="email" placeholder="Enter your email" required>
    <label for="age">Age:</label> <input type="number" id="age" name="age" min="1" max="120" required>
    <label for="dob">Date of Birth:</label>
    <input type="date" id="dob" name="dob" required>
    <input type="submit" value="Submit"></form>

  </body>
</html>
```

OutPut:

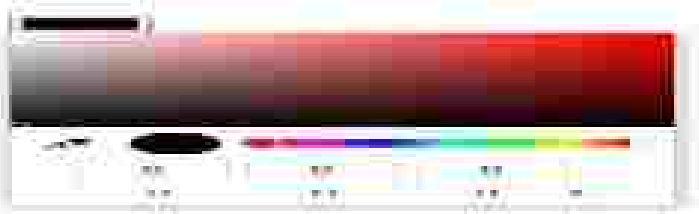
Color and Date Pickers in HTML5

HTML5 offers a variety of input types to enhance user interaction and data input. Among these are the color and date pickers, which provide intuitive ways for users to select colors and dates.

Color Picker: The color input type allows users to select a color from a color picker.

<input type="color" name="favoriteColor" id="favoriteColor">

Output :



When a user clicks on this input, a color picker dialog will appear, allowing them to choose a color. The selected color will be represented as a hexadecimal color code.

Date Picker:

The date input type enables users to select a date from a calendar.

Example with output

```
<!DOCTYPE html>
<html>
  <head>
    <title>Hello, World!</title>
    <link rel="stylesheet" href="style.css" />
  </head>
  <body>

 <input type="date" name="birthday" id="birthday">

  </body>
</html>
```

Select and Datalist Elements

The <select> element creates a drop-down list of options for the user to choose from.

```
<!DOCTYPE html>
<html>
  <head>
    <title>Hello, World!</title>
    <link rel="stylesheet" href="style.css" />
  </head>
  <body>
 Select Courses
<select name="cars">
  <option value="mern">MERN STACK</option>
  <option value="web">COMPILER DESIGN</option>
  <option value="mercedes">CNS</option>
  <option value="audi">MACHINE LEARNING</option>
</select>

  </body>
</html>
```

Datalist Element

HTML 5

The <datalist> element provides a list of suggestions for an <input> element. It offers an autocomplete-like functionality.

Select vs DataList

- **Select:** Use when you want to restrict the user's choices to a specific set of options.
- **Datalist:** Use when you want to provide suggestions to the user but allow them to enter their own values.

Editable Elements:

HTML5 introduces the contenteditable attribute, which allows you to make any element editable directly within the browser. This can be applied to elements such as , <p>, and .

Example:

```
<div contenteditable="true">
  This is an editable div. You can change this text.
</div>
```

The contenteditable attribute specifies whether the content of an element is editable or not.

Media, Iframe

The <iframe> element is used to embed another web page within the current page. It's often used to embed content from other websites, such as YouTube videos, Google Maps, or social media feeds.

YouTube and Google Maps provide embed codes to display content on other websites.

```
<iframe src="https://www.example.com" width="600" height="400"></iframe>
```

Common Attributes for <iframe> Element:

- **src:** Specifies the URL of the content to be embedded.
- **width:** Sets the width of the iframe.
- **height:** Sets the height of the iframe.
- **frameborder:** Specifies whether or not to display a border around the iframe.
- **scrolling:** Specifies whether or not to display scrollbars.

Example

Media Elements:

1. <audio> Element:

This element is used to embed audio content into a web page. It supports various audio formats like MP3, WAV, OGG, etc.

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

Output :

2. <video> Element:

This element is used to embed video content into a web page. It supports various video formats like MP4, WebM, OGG, etc.

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

Output :

Common Attributes for <audio> and <video> Elements:

- **controls:** Displays playback controls.
- **autoplay:** Automatically starts playback.
- **loop:** Loops the media.
- **muted:** Mutes the media.
- **poster:** Sets a poster image to display before playback starts.

HTML SECURITY

HTML, the backbone of the web, is often overlooked when it comes to security. However, it's a critical component in safeguarding your website and protecting user data. Here's why HTML security is essential:

1. Cross-Site Scripting (XSS):

- **How it works:** Malicious code is injected into a website, allowing an attacker to steal sensitive information, hijack user sessions, or redirect users to malicious sites.
- **How to prevent:**
  - Input validation and sanitization: Ensure that user-provided input is properly cleaned and filtered.
  - Output encoding: Encode output to prevent malicious scripts from executing.
  - Content Security Policy (CSP): Restrict the sources of content that can be loaded on your website.

HTML 5

2. HTML Injection:

- **How it works:** Attackers inject malicious HTML code into a website's structure, often leading to unauthorized access or data breaches.
- **How to prevent:**
  - Input validation and sanitization: As mentioned above, this is crucial to prevent injection attacks.
  - Output encoding: Properly encode output to render HTML as text instead of executable code.

3. Clickjacking:

- **How it works:** Attackers trick users into clicking on malicious links or buttons hidden beneath legitimate content.
- **How to prevent:**
  - Use `X-Frame-Options` header: This header can be set to prevent your website from being embedded in an iframe, making it harder for attackers to use clickjacking techniques.

4. Other Vulnerabilities:

- **Malicious Script Embeddings:** Attackers can embed malicious scripts in your HTML, often through third-party libraries or advertisements.
- **Session Hijacking:** Vulnerable HTML forms can expose session tokens, allowing attackers to hijack user sessions.

Best Practices for HTML Security:

- **Keep Software Updated:** Regularly update your web server, frameworks, and libraries to address known vulnerabilities.
- **Validate and Sanitize Input:** Always validate and sanitize user input to prevent injection attacks.
- **Use a Content Security Policy (CSP):** A CSP defines a set of security policies that help mitigate a range of web application attacks, including XSS and data injection attacks.
- **Be Careful with Third-Party Libraries and Scripts:** Only use trusted and well-maintained libraries.
- **Regularly Audit Your Website:** Conduct regular security audits to identify and fix potential vulnerabilities.
- **Stay Informed:** Keep up-to-date with the latest security best practices and threats.

By following these guidelines, you can significantly improve the security of your HTML-based websites and protect your users from potential attacks.

HTML5 Cross-Browser Compatibility

HTML5 is widely supported across modern web browsers, ensuring a consistent user experience. However, older browsers or specific browser versions might have varying levels of support for certain features.

Key Considerations for Cross-Browser Compatibility:

1. **Doctype Declaration:**
   - Ensure a valid DOctype declaration at the beginning of your HTML document:

     HTML

     ```
     <!DOCTYPE html>
     ```

   - This tells the browser to interpret the document according to the latest HTML standard.
2. **HTML5 Semantic Elements:**
   - While most modern browsers support HTML5 semantic elements like `<header>`, `<footer>`, `<nav>`, `<section>`, and `<article>`, older browsers might not.
   - Consider using CSS to style these elements to ensure consistent appearance across different browsers.
3. **HTML5 Forms:**
   - Newer browsers support advanced form elements like `<input type="email">`, `<input type="tel">`, `<input type="url">`, and `<input type="date">`.
   - For older browsers, use JavaScript-based polyfills or fallback solutions to provide similar functionality.
4. **HTML5 Media:**
   - The `<audio>` and `<video>` elements are widely supported, but older browsers might require specific codecs or additional JavaScript libraries for playback.
   - Consider providing fallback options like Flash or embedded media players for older browsers.
5. **CSS3:**
   - Modern browsers support a wide range of CSS3 features, but older browsers might have limited support.
   - Use CSS3 features judiciously and provide fallback styles for older browsers.
   - Consider using CSS preprocessors like Sass or Less to write more efficient and maintainable CSS.

Testing and Debugging:

- **Browser Compatibility Testing Tools:** Use tools like BrowserStack or LambdaTest to test your website on different browsers and devices.
- **Browser Developer Tools:** Use the built-in developer tools in your browser to inspect the rendered HTML and CSS, and debug any issues.

- **Feature Detection:** Use JavaScript to detect the capabilities of the user's browser and dynamically adjust the behavior of your website.

By following these guidelines and using testing tools, you can ensure that your HTML5 websites and web applications are compatible with a wide range of browsers, providing a consistent and optimal user experience.

Best Practices for HTML Web Pages

Here are some best practices to create well-structured, efficient, and maintainable HTML web pages:

1. Semantic HTML:

- Use appropriate HTML elements for their intended purpose.
- For example, use <header>, <nav>, <section>, <article>, <aside>, <footer>, and <main> elements to structure your content semantically.
- This improves accessibility, SEO, and readability for both humans and machines.

2. Clean and Valid HTML:

- Write clean, well-formatted HTML code.
- Use a linter or validator to check for errors and inconsistencies.
- Follow HTML standards and guidelines.

3. Mobile-First Design:

- Design your website with mobile devices in mind.
- Use responsive design techniques or a mobile-first framework to ensure your website looks good and functions well on all screen sizes.

4. Optimize Image Files:

- Compress images to reduce file size without compromising quality.
- Use appropriate formats (e.g., JPEG, PNG, WebP) for different image types.
- Consider using responsive images to serve different image sizes based on the device's screen size.

5. Optimize Loading Speed:

- Minimize HTTP requests by combining and minifying CSS and JavaScript files.
- Leverage browser caching to reduce server load and improve performance.
- Use a Content Delivery Network (CDN) to distribute your website's assets across multiple servers, improving load times.

6. Accessibility:

- Use semantic HTML to improve accessibility for screen readers and other assistive technologies.
- Provide alternative text for images (alt attribute).

- Use clear and concise language.
- Test your website with assistive technologies.

7. Security:

- Validate and sanitize user input to prevent XSS attacks.
- Use HTTPS to encrypt data transmission between the server and the client.
- Keep your website and server software up-to-date with the latest security patches.

8. Cross-Browser Compatibility:

- Test your website on different browsers and devices to ensure consistent performance and appearance.
- Use CSS techniques like feature detection and progressive enhancement to support older browsers.

9. SEO Best Practices:

- Use descriptive and keyword-rich titles and meta descriptions.
- Optimize your website's URL structure.
- Create high-quality, relevant content.
- Build backlinks to your website from other reputable websites.

10. Regular Maintenance:

- Keep your website up-to-date with the latest technologies and security practices.
- Monitor your website's performance and make necessary optimizations.
- Regularly update your content to keep it fresh and relevant.

By following these best practices, you can create high-quality, performant, and secure HTML web pages that provide a great user experience.