

WEB TECHNOLOGY
SUBJECT CODE: P3CA51ET

UNIT-I

Introduction to HTML5- Features of HTML5- Basic Structure of HTML5- HTML5 elements: Text Formatting Tags- Headings, Paragraphs, Lists, Tables, Images- Multimedia elements (audio, video)- Advanced HTML5 Tags- Web Forms 2.0 - HTML Canvas - MathML- SVG- Web Storage- Geo Location- HTML Validation.

UNIT-II

Introduction to CSS3- Creating Style Rules- Selectors- Color and backgrounds-Fonts-Text Formatting- Lists-Tables- Padding, Margins, and Borders- Gradients- Text Overflow- Drop Shadows- 2D Transforms- 3D Transforms- Transitions- Animations- Multi- Column Layout- Box Sizing-Filters.

UNIT-III

Introduction to JavaScript- Features of JavaScript- Introduction to ES6- Variables- Template Literals- Expressions- Datatype Conversions- Operators- Conditional Statements- Looping Statements- One Dimensional Arrays- Two dimensional Arrays- Jagged Arrays- Functions- Anonymous Functions Arrow Functions- Recursive function- Default Parameters.

UNIT-IV

JavaScript Classes- Constructors, Prototypes-JavaScript Objects User defined objects - Built in objects Document object Browser object Form object Navigator object Screen object Image Objects- Destructuring - Spread operator - Event Handling and Event Bubbling.

UNIT-V

JavaScript Document Object Model Introduction Advantages and disadvantages of DOM Tree- DOM Interfaces- Working with DOM Tree- Cookies Exception Handling.

TEXT BOOKS

1. DT Editorial Services, HTML5 Black book, Dreamtech Press, 2016.
2. Laura Lemay, Rafe Colburn, Jennifer Kyrnin , MASTERING HTML, CSS & Java Script Web Publishing, BPB Publications.

REFERENCE BOOKS

1. I.Bayross, 2000, Web Enable Commercial Application Development Using HTML, DHTML, Javascript, Perl CGI, BPB Publications.
2. Brian Pfaffenberger, Steven M. Schafer, Charles White, Bill Karow, HTML, XHTML, and CSS Bible, 3rd Edition, Wiley Publishing, Inc
3. JavaScript 2.0 : The Complete Reference, Second Edition by Thomas Powell and Fritz Schneider

WEB RESOURCES

1. HTML Tutorial, <https://www.javatpoint.com/html5-tutorial>
2. Web Technology, https://onlinecourses.swayam2.ac.in/nou24_cs18/preview
3. HTML5 Complete course for Beginners, <https://www.youtube.com/watch?v=ij7FsmZiBIY>
4. JavaScript Tutorial for Beginners in Tamil, <https://www.youtube.com/watch?v=poo0BXryffl>

WEB TECHNOLOGY LAB
SUBJECT CODE: P3CA52MP

PRACTICAL QUESTIONS

1. Develop a web page which implements Canvas and SVG
2. Develop a webpage which maintains the number of visits of the user using Web storage.
3. Design a Student Registration web page using CSS3.
4. Design a web page for Online Book store using CSS3.
5. Construct a JavaScript code to accept array of N Numbers at runtime and find largest and smallest of those numbers.
6. Construct a JavaScript code to find a factorial of a given number.
7. Create a document and add a link to it. When the user moves the mouse over the link, it should load the linked document on its own. (User is not required to click on the link).
8. Create a document, which opens a new window without a toolbar, address bar or a status bar that unloads itself after one minute.
9. Create a document, which modifies the content of webpage using DOM Tree, also summarizes the list of anchors, links, images and input elements in a page.
10. Construct a JavaScript Code to implement Cookies in a web page.

TEXT BOOKS

1. DT Editorial Services, HTML5 Black book, Dreamtech Press, 2016.
2. Laura Lemay, Rafe Colburn, Jennifer Kyrnin , MASTERING HTML, CSS & Java Script Web Publishing, BPB Publications.

REFERENCE BOOKS

1. I.Bayross, 2000, Web Enable Commercial Application Development Using HTML, DHTML, Javascript, Perl CGI, BPB Publications.
2. Brian Pfaffenberger, Steven M. Schafer, Charles White, Bill Karow, HTML, XHTML, and CSS Bible, 3rd Edition, Wiley Publishing, Inc
3. JavaScript 2.0 : The Complete Reference, Second Edition by Thomas Powell and Fritz Schneider

WEB RESOURCES

1. HTML Tutorial, <https://www.javatpoint.com/html5-tutorial>
2. Web Technology, https://onlinecourses.swayam2.ac.in/nou24_cs18/preview
3. HTML5 Complete course for Beginners, <https://www.youtube.com/watch?v=ij7FsmZiBIY>
4. JavaScript Tutorial for Beginners in Tamil, <https://www.youtube.com/watch?v=poo0BXryffl>

WEB TECHNOLOGY
SUBJECT CODE: P3CA51ET

UNIT-I

Introduction to HTML5- Features of HTML5- Basic Structure of HTML5- HTML5 elements: Text Formatting Tags- Headings, Paragraphs, Lists, Tables, Images- Multimedia elements (audio, video)- Advanced HTML5 Tags- Web Forms 2.0 - HTML Canvas - MathML- SVG- Web Storage- GeoLocation- HTML Validation.

INTRODUCTION TO HTML5:

- HTML stands for Hyper Text Markup Language. It is used to design web pages using a markup language. HTML is an abbreviation of Hypertext and Markup language.
- Hypertext defines the link between the web pages. The markup language is used to define the text document within the tag which defines the structure of web pages.
- HTML 5 is the fifth and current version of HTML. It has improved the markup available for documents and has introduced application programming interfaces (API) and Document Object Model (DOM).

Why use HTML5

- It is enriched with advance features which makes it easy and interactive for designer/developer and users.
- It allows you to play a video and audio file.
- It allows you to draw on a canvas.
- It facilitate you to design better forms and build web applications that work offline.
- It provides you advance features for that you would normally have to write JavaScript to do.
- The most important reason to use HTML 5 is, we believe it is not going anywhere.
- It will be here to serve for a long time according to W3C recommendation.

Example

```
<html>
<body>
<h1>Write Your First Heading</h1>
<p>Write Your First Paragraph.</p>
</body>
</html>
```

Features:

- It has introduced new multimedia features which supports both audio and video controls by using <audio> and <video> tags.
- There are new graphics elements including vector graphics and tags.
- Enrich semantic content by including <header> <footer>, <article>, <section> and <figure> are added.

- Drag and Drop- The user can grab an object and drag it further dropping it to a new location.
- Geo-location services- It helps to locate the geographical location of a client.
- Web storage facility which provides web application methods to store data on the web browser.
- Uses SQL database to store data offline.
- Allows drawing various shapes like triangle, rectangle, circle, etc.
- Capable of handling incorrect syntax.
- Easy DOCTYPE declaration i.e., `<!doctype html>`
- Easy character encoding i.e., `<meta charset="UTF-8">`

New Added Elements in HTML 5:

- **<article>:** The `<article>` tag is used to represent an article. More specifically, the content within the `<article>` tag is independent from the other content of the site (even though it can be related).
- **<aside>:** The `<aside>` tag is used to describe the main object of the web page in a shorter way like a highlighter. It basically identifies the content that is related to the primary content of the web page but does not constitute the main intent of the primary page. The `<aside>` tag contains mainly author information, links, related content and so on.
- **<figcaption>:** The `<figcaption>` tag in HTML is used to set a caption to the figure element in a document.
- **<figure>:** The `<figure>` tag in HTML is used to add self-contained content like illustrations, diagrams, photos or codes listing in a document. It is related to main flow, but it can be used in any position of a document and the figure goes with the flow of the document and if it is removed it should not affect the flow of the document.
- **<header>:** It contains the section heading as well as other content, such as a navigation links, table of contents, etc.
- **<footer>:** The `<footer>` tag in HTML is used to define a footer of HTML document. This section contains the footer information (author information, copyright information, carriers etc.). The footer tag is used within body tag. The `<footer>` tag is new in the HTML 5. The footer elements require a start tag as well as an end tag.
- **<main>:** Delineates the main content of the body of a document or web app.
- **<mark>:** The `<mark>` tag in HTML is used to define the marked text. It is used to highlight the part of the text in the paragraph.
- **<nav>:** The `<nav>` tag is used to declaring the navigational section in HTML documents. Websites typically have sections dedicated to navigational links, which enables user to navigate the site. These links can be placed inside a nav tag.
- **<section>:** It demarcates a thematic grouping of content.
- **<details>:** The `<details>` tag is used for the content/information which is initially hidden but could be displayed if the user wishes to see it. This tag is used to create interactive widget which user can open or close it. The content of details tag is visible when open the set attributes.
- **<summary>:** The `<summary>` tag in HTML is used to define a summary for the `<details>` element. The `<summary>` element is used along with the `<details>` element and provides a summary visible to the user. When the summary is clicked by the user, the content placed inside the `<details>` element becomes visible which was previously hidden. The `<summary>` tag was added in HTML 5. The `<summary>` tag requires both starting and ending tag

- **<time>:** The <time> tag is used to display the human-readable data/time. It can also be used to encode dates and times in a machine-readable form. The main advantage for users is that they can offer to add birthday reminders or scheduled events in their calendars and search engines can produce smarter search results.
- **<bdi>:** The <bdi> tag refers to the Bi-Directional Isolation. It differentiates a text from other text that may be formatted in different direction. This tag is used when a user generated text with an unknown direction.
- **<wbr>:** The <wbr> tag in HTML stands for word break opportunity and is used to define the position within the text which is treated as a line break by the browser. It is mostly used when the used word is too long and there are chances that the browser may break lines at the wrong place for fitting the text.
- **<datalist>:** The <datalist> tag is used to provide autocomplete feature in the HTML files. It can be used with input tag, so that users can easily fill the data in the forms using select the data.
- **<keygen>:** The <keygen> tag in HTML is used to specify a key-pair generator field in a form. The purpose of <keygen> element is to provide a secure way to authenticate users. When a form is submitted then two keys are generated, private key and public key. The private key stored locally, and the public key is sent to the server. The public key is used to generate client certificate to authenticate user in future.
- **<output>:** The <output> tag in HTML is used to represent the result of a calculation performed by the client-side script such as JavaScript.
- **<progress>:** It is used to represent the progress of a task. It also defines how much work is done and how much is left to download a task. It is not used to represent the disk space or relevant query.
- **<svg>:** It is the Scalable Vector Graphics.
- **<canvas>:** The <canvas> tag in HTML is used to draw graphics on web page using JavaScript. It can be used to draw paths, boxes, texts, gradient and adding images. By default, it does not contain border and text.
- **<audio>:** It defines the music or audio content.
- **<embed>:** Defines containers for external applications (usually a video player).
- **<source>:** It defines the sources for <video> and <audio>.
- **<track>:** It defines the tracks for <video> and <audio>.
- **<video>:** It defines the video content. **Advantages:**
 - All browsers supported.
 - More device friendly.
 - Easy to use and implement.
 - HTML 5 in integration with CSS, JavaScript, etc. can help build beautiful websites.

Disadvantages:

- Long codes have to be written which is time consuming.
- Only modern browsers support it.

BASIC STRUCTURE OF HTML5:

Basic structure of an HTML5 document

The basic structure of an HTML (Hypertext Markup Language) document refers to the overall organization and syntax of an HTML file. It consists of several elements that provide information about the web page's content, structure, and formatting.

The basic structure of an HTML document includes the following components:

1. `<!DOCTYPE html>`: This declaration specifies the version of HTML that the document is written in. It is usually the first line of an HTML file.
2. `<html>` element: This is the root element of the document and encloses all other elements.
3. `<head>` element: This element contains metadata about the document, such as the title, author, and character set. It does not contain any visible content on the page.
4. `<title>` element: This element is nested within the head element and specifies the title of the web page, which appears in the browser's title bar.
5. `<body>` element: This element contains all visible content on the page, such as text, images, links, and other HTML elements.
6. Other HTML elements: These are additional elements used to structure and format the content within the body element, such as headings, paragraphs, lists, tables, and more.

By following this basic structure, web developers can create consistent and well-organized HTML documents that are easy to understand and maintain.

`<!DOCTYPE html>`(Document Type Declaration)

Every HTML document begins with a document type declaration, which tells the browser what type of HTML the document is written in.

The `<!DOCTYPE html>` declaration is an instruction that tells the web browser which version of HTML is being used in the document.

The DOCTYPE declaration is required in HTML documents to ensure that the browser renders the page in standards mode. Standards mode ensures that the web page is displayed consistently across different browsers and platforms.

The `<!DOCTYPE html>` declaration is used in HTML5 to indicate that the document is written in the latest version of HTML. It is usually placed at the beginning of an HTML document, before the `<html>` tag.

`<html>` element

The `<html>` element is the root element of an HTML document and represents the top-level container for all other HTML elements in the document. It tells the browser that the document is an HTML document and encloses the entire HTML document, including the `<head>` and `<body>` elements. The HTML tag is opened at the beginning of the document and closed at the end of the document.

The `<html>` element also specifies the language of the document using the `lang` attribute. For example, to specify that the document is written in English, you would add the following attribute to the `<html>` tag:

`<html lang="en">`

`<head>` element :The `<head>` element is a container for metadata about an HTML document, such as the title of the page, links to stylesheets and scripts, and other information that is not directly visible on the page.

Here's an example of how the <head> element is used in an HTML document:

```
<!DOCTYPE html>

<html>

<head>

<title> My Web Page</title>

<meta charset="UTF-8">

<link rel="stylesheet" href="styles.css">

<script src="script.js"></script>

</head>

<body>

<h1>Welcome to my web page!</h1>

<p>This is the content of my web page.</p>

</body>

</html>
```

In this example, the <head> element contains several elements that provide metadata about the document:

- <title>: This element specifies the title of the web page, which appears in the browser's title bar and is used by search engines to index the page.
- <meta>: This element provides information about the character set used in the document. In this example, the charset attribute specifies that the document is encoded using UTF-8.
- <link>: This element specifies the location of an external stylesheet that is used to define the visual style of the web page.
- <script>: This element specifies the location of an external JavaScript file that is used to add interactivity to the web page.

Overall, the <head> element is an important part of an HTML document as it provides information about the document that is used by the browser and other software to render the page correctly.

<body> element

In HTML, the <body> element represents the content of an HTML document that is visible in a web browser. It is the main container for all the content that is displayed on a webpage, such as text, images, videos, and other HTML elements.

The `<body>` element typically comes after the `<head>` element in an HTML document, which contains information about the document that is not displayed in the browser, such as metadata and scripts. The content of the `<body>` element is what is rendered in the browser window, and it can include other HTML elements such as headings, paragraphs, lists, links, images, forms, and more.

Here is an example of a basic HTML document structure that includes a `<body>` element:

```
<!DOCTYPE html>

<html>

<head>

<title>My Webpage</title>

</head>

<body>

<h1>Welcome to My Webpage</h1>

<p>This is some sample text that will be displayed in the browser window.</p>



</body>

</html>
```

In this example, the `<body>` element contains a heading (`<h1>`), a paragraph (`<p>`), and an image (``). When this HTML document is loaded in a web browser, these elements will be displayed in the main content area of the webpage.

The Opening and the Closing Tags

In HTML, elements are usually written as opening tags, content, and closing tags. The opening tag is denoted by the angle bracket "`<`", followed by the name of the element, and then another angle bracket "`>`". The closing tag has the same name as the opening tag, but with a forward slash "/" before the name, and then another angle bracket "`>`".

For example, consider the following HTML code:

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

In this code, `<p>` is the opening tag, and `</p>` is the closing tag. The content of the paragraph, "This is a paragraph.", is located between the opening and closing tags.

It is important to include both the opening and closing tags in your HTML code to ensure that the web browser can properly interpret the content and render it correctly on the webpage. If a closing tag is omitted, the web browser may not be able to properly interpret the HTML code, leading to display issues on the webpage.

HTML Text Formatting

HTML text formatting tags enhance web content’s visual presentation and semantic meaning. By using tags like ``, `<i>`, ``, and ``, you can style text as bold, italic, emphasized, or important, improving readability and user experience on webpages.

Table of Content

- Logical Tags
- Physical Tags
- HTML Formatting Elements
- Example of Text Formatting
- Conclusion

HTML text formatting can be categorized into two techniques:

1.Logical Tags

These tags denote the logical or semantic value of the text. They convey meaning without directly affecting the visual appearance.

- ``: Emphasizes text (commonly styled as italic).
- ``: Indicates important content (often styled as bold).

2.Physical Tags

These tags directly impact the visual appearance of the text, altering font styles, sizes, and other visual properties.

- ``: Displays text in a bold format.
- `<i>`: Italicizes text.

HTML Formatting Elements

Tags	Descriptions
<code><i></code>	Showcases italicized text.
<code><small></code>	Renders text in a smaller font size.
<code><ins></code>	Highlights added or inserted text.
<code><sub></code>	Creates subscript text.
<code></code>	Emphasizes text with importance, often in bold.
<code></code>	Displays text in a bold format.
<code><mark></code>	Accentuates text with a background highlight.
<code></code>	Strikes through text to signify deletion.
<code></code>	Adds emphasis to text, commonly styled as italic.
<code><sup></code>	Formats text as superscript.

Example of Text Formatting

Example 1: Basic Text Formatting

In this example we demonstrates various text formatting tags: `` for important and bold text, `` for emphasized and italic text, `` for bold text, `<i>` for italic text, and `<mark>` for highlighted text.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Text Formatting Example</title>
</head>
<body>
  <p>
    <strong>Strong:</strong>
    This text is important and bold.
  </p>
  <p>
    <em>Emphasized:</em>
    This text is emphasized and italic.
  </p>
  <p>
    <b>Bold:</b>
    This text is bold.
  </p>
  <p>
    <i>Italic:</i>
    This text is italic.
  </p>
  <p>
    <mark>Marked:</mark>
    This text is highlighted.
  </p>
</body>

</html>
```

Output:

Strong: This text is important and bold.

Emphasized: This text is emphasized and italic.

Bold: This text is bold.

Italic: This text is italic.

Marked: This text is highlighted.

HTML Headings

HTML headings, from `<h1>` to `<h6>`, indicate content hierarchy and importance. Search engines use them for indexing. Users navigate by headings. Employ `<h1>` for main titles, `<h2>` for subsections, and progressively lower levels for less important content to maintain structure and readability.

HTML provides six levels of heading elements, ranging from `<h1>` (highest level) to `<h6>` (lowest level).

Purpose and Usage of Heading Tags:

1. `<h1>` – Main Heading (Largest):

- Represents the primary topic of the page.
- Use it sparingly to maintain clarity.

2. `<h2>` – Subheadings:

- Ideal for dividing content into sections.
- If further subsections exist, employ `<h3>` elements.

3. `<h3>` to `<h6>` – Smaller Headings:

- Gradually decrease in size.
- Use them for finer subdivisions.

This is heading h1

This is heading h2

This is heading h3

This is heading h4

This is heading h5

This is heading h6

Syntax:

// the 'h' inside the tag should be in small case only.

<h1>Heading1</h1>

<h2>Heading2</h2>

.

.

.

<h6>Heading6</h6>

Why Do Headings are Important?**1. SEO Impact:**

- Search engines index headings to understand content structure.
- Well-organized headings enhance visibility and ranking.

2. Highlighting Key Topics:

- Headings emphasize essential information.
- Readers quickly grasp the document's structure.

Example: In this example illustrates the **HTML heading tags**.

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
  <h1>This is heading h1</h1>
```

```
  <h2>This is heading h2</h2>
```

```
  <h3>This is heading h3</h3>
```

```
  <h4>This is heading h4</h4>
```

```
  <h5>This is heading h5</h5>
```

```
  <h6>This is heading h6</h6>
```

```
</body>
```

```
</html>
```

Output:

This is heading h1

This is heading h2

This is heading h3

This is heading h4

This is heading h5

This is heading h6

Customization in HTML Heading Tags

The default size of HTML headings can be changed, using the style attribute.

Example: In this example explains the HTML heading tags by specifying the size of the font.

```
<!DOCTYPE html>
<html>
<body>
<h1>H1 Heading</h1>
<!-- With the help of Style attribute you can customize the size of the heading, As done below-->
<h1 style="font-size: 50px">H1 with new size.</h1>
<!-- Here font-size is the property by which we can modify the heading. Here we kept it 50px
i.e. 50 pixels -->
</body>
</html>
```

Output:

H1 Heading

H1 with new size.

HTML Paragraphs

The <p> tag in HTML signifies a paragraph. Enclosed within the opening <p> and closing </p> tags, any content is recognized as a paragraph. As a block-level element, a new paragraph inherently starts on a fresh line, with browsers intuitively adding space before and after a paragraph for enhanced readability.

Syntax:

<p> Content </p>

Properties of the paragraph Tag

- The browser reduces multiple spaces added by users to a single space.
- If a user adds multiple lines, the browser compresses them into a single line.
- By default, the display of the paragraph element is set to “block,” meaning each new paragraph is placed on a new line. This behavior can be modified using CSS.

Example 1: The example explains the HTML <p> tag.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">   <meta name="viewport"
    content="width=device-width, initial-scale=1.0">

  <title>The p tag</title>
</head>
<body>
  <p>A Computer Science portal for geeks.</p>
  <p>It contains well written, well thought articles.</p>
</body>
</html>
```

Output:

A Computer Science portal for geeks.
It contains well written, well thought articles.

Multiple p Tags

Example 2: This example explains the HTML <p> tag having multiple lines.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">   <meta name="viewport"
    content="width=device-width, initial-scale=1.0">
  <title>HTML p tag</title>
</head>
<body>
  <p>
    This paragraph has multiple lines. But HTML reduces them to a single line, omitting the
    carriage return we have used.
  </p>
  <p>
    This paragraph has multiple spaces. But HTML reduces them all to a single space, omitting
    the extra spaces and line we have used.
  </p>
</body>
</html>
```

Output:

This paragraph has multiple lines. But HTML reduces them to a single line, omitting the carriage return we have used.

This paragraph has multiple spaces. But HTML reduces them all to a single space, omitting the extra spaces and line we have used.

p tag with multiple lines

The
 tag

The HTML [
](#) tag element creates a line break, giving you a new line without starting a new paragraph. Use
 when you want to move to the next line without beginning a whole new paragraph.

Syntax:

Example: This example explains the
 tag inside the <p> tag to add the line-break.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">   <meta name="viewport"
    content="width=device-width, initial-scale=1.0">
  <title>HTML paragraph
</title>
</head>
<body>
  <p>
    This paragraph has multiple
    <br />lines. But HTML reduces them
    <br />to a single line, omitting
    <br />the carriage return we have used.
  </p>
</body>
</html>
```

Output:

This paragraph has multiple
lines. But HTML reduces them
to a single line, omitting
the carriage return we have used.

The <hr> tag

The HTML [<hr>](#) tag is used to create a horizontal rule or line, visually separating content on a webpage. Use <hr> when you want to insert a horizontal line to signify a division between sections or elements, providing a clear visual break in the page.

Syntax: <hr>

Example: In this example we will use the <hr> tag with an example by using <p> tag also.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">   <meta name="viewport"
    content="width=device-width, initial-scale=1.0">
  <title>HTML Horizontal Rule Example</title>
</head>
<body>
  <h1>
    Welcome to My Website
  </h1>
  <p>
    GeeksforGeeks is a leading platform that provides computer science resources and coding
challenges
  </p>
  <hr>
  <p>
    GeeksforGeeks is a leading platform that provides computer science resources and coding
challenges
  </p>
</body>

</html>
```

Output:

Welcome to My Website

GeeksforGeeks is a leading platform that provides computer science resources and coding challenges

GeeksforGeeks is a leading platform that provides computer science resources and coding challenges

Align attribute

The <p> tag specifically supports the [alignment attribute](#) and allows us to align our paragraphs in left, right, or center alignment.

Syntax:

```
<p align="value">
```

Example: This example explains the align attribute to align the content in the <p> tag.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">   <meta name="viewport"
    content="width=device-width, initial-scale=1.0">
  <title>HTML paragraph </title>
</head>
<body>
  <p align="center">
    Welcome Geeks
  </p>
  <p align="left">
    A Computer Science portal for geeks.
  </p>
  <p align="right">
    It contains well written, well thought articles.
  </p>
</body>
</html>
```

Output:

```

Welcome Geeks

A Computer Science portal for geeks.

It contains well written, well thought articles.
```

The <pre> tag

The [HTML <pre> tag](#) is used for pre-formatted text. It keeps the original spaces and line breaks exactly as they are in the code. When you use <pre>, the text appears in a fixed-width font, preserving the formatting and layout just as it looks in the HTML code.

Syntax:

<pre> Content </pre>

Example: This example explains the use of the <pre> tag in the <p> tag.

```
<!DOCTYPE html>
```

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

```
<head>
```

```
  <meta charset="UTF-8">  <meta name="viewport"
    content="width=device-width, initial-scale=1.0">
```

```
  <title>HTML paragraph </title>
```

```
</head>
```

```
<body>
```

```
  <pre>
```

```
    This paragraph has multiple    lines. But it is displayed    as it is unlike the paragraph    tag.
```

```
  </pre>
```

```
  <pre>
```

```
    This    paragraph has multiple    spaces. But    it is displayed    as it is    unlike the paragraph
tag.
```

```
  </pre>
```

```
</body>
```

```
</html>
```

Output:

```
This paragraph has multiple
lines. But it is displayed
as it is unlike the paragraph
tag.
```

```
This    paragraph has multiple
spaces. But    it is displayed
as it is    unlike the paragraph
tag.
```

HTML Lists

An **HTML list** is a record of related information used to display the data or any information on web pages in the **ordered** or **unordered form**.

HTML List Example

Basic Implementation of HTML lists.

```
<!DOCTYPE html>
<html>
<head>
  <title>GeeksforGeeks</title> </head>

<body>
  <h2>Welcome To GeeksforGeeks Learning</h2>
  <h5>List of available courses</h5>
  <ul>
    <li>Data Structures & Algorithm</li>
    <li>Web Technology</li>
    <li>Apptitude & Logical Reasoning</li>
    <li>Programming Languages</li>
  </ul>
  <h5>Data Structures topics</h5>
  <ol>
    <li>Array</li>
    <li>Linked List</li>
    <li>Stacks</li>
    <li>Queues</li>
    <li>Trees</li>
    <li>Graphs</li>
  </ol>
</body>
</html>
```

Output:

Welcome To GeeksforGeeks Learning

List of available courses

- Data Structures & Algorithm
- Web Technology
- Apptitude & Logical Reasoning
- Programming Languages

Data Structures topics

1. Array
2. Linked List
3. Stacks
4. Queues
5. Trees
6. Graphs

HTML List

There are **three types of lists in HTML**:

- HTML Unordered List or Bulleted List
- HTML Ordered List
- HTML Description List

HTML List Tags

Here is the list of all **lists tags HTML**:

Tag	Description
	Defines an unordered list
	Defines an ordered list
	Defines a list item
<dl>	Defines a description list
<dt>	Defines a term in a description list
<dd>	Details the term in a description list

The HTML Unordered List or Bulleted List

The unordered list items are marked with bullets. It is also known as bulleted lists. An unordered list starts with the ** tag**. Each list item starts with the ** tag**.

Syntax:

```
<ul> list of items </ul>
```

Attribute: This tag contains two attributes which are listed below:

- **compact:** It will render the list smaller.
- **type:** It specifies which kind of marker is used in the list.

Example: This example describes the unordered list.

```
<!DOCTYPE html>
<html>
<body>
  <h2>Grocery list</h2>
  <ul>
    <li>Bread</li>
    <li>Eggs</li>
    <li>Milk</li>
    <li>Coffee</li>
  </ul>
```

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

Output:

Grocery list

- Bread
- Eggs
- Milk
- Coffee

Unordered List

HTML Ordered List

In an ordered list, all list items are marked with **numbers** by default. An ordered list starts with the ** tag**. Each list item starts with the “li” tag.

Syntax:

```
<ol>
  <li>Item1</li>
  <li>Item2</li>
  <li>Item3</li> </ol>
```

Attributes:

- **compact:** It defines the list should be compacted (compact attribute is not supported in HTML5. Use CSS instead.).
- **reversed:** It defines that the order will be descending.
- **start:** It defines from which number or alphabet the order will start.
- **type:** It defines which type(1, A, a, I, and i) of the order you want in your list of numeric, alphabetic, or roman numbers.

Example: This example illustrates the use of the reverse attribute, control list counting & type attribute.

```
<!DOCTYPE html>
<html>
<head>
  <title>HTML ol tag</title>
</head>
<body>
  <h1 style="color: green">GeeksforGeeks</h1>
  <h3>HTML ol tag</h3>
  <p>reversed attribute</p>
  <ol reversed>
    <li>HTML</li>
    <li>CSS</li>
```

```

        <li>JS</li>
    </ol>
<p>start attribute</p>
    <ol start="5">
        <li>HTML</li>
        <li>CSS</li>
        <li>JS</li>
    </ol>
<p>type attribute</p>
    <ol type="i">
        <li>HTML</li>
        <li>CSS</li>
        <li>JS</li>
    </ol>
</body>
</html>

```

Output:

GeeksforGeeks

HTML tag

reversed attribute

```

3. HTML
2. CSS
1. JS

```

start attribute

```

5. HTML
6. CSS
7. JS

```

type attribute

```

i. HTML
ii. CSS
iii. JS

```

Ordered List with different list style

HTML Description List

An description list is a list of terms, with a description of each term. The <dl> tag defines the description list, the <dt> tag defines the term name, and the <dd> tag describes each term.

Syntax:

```
<dl> Contents... </dl>
```

Example: This example describes the HTML Description List.

```
<!DOCTYPE html>
<html>
<body>
  <h2>A Description List</h2>
  <dl>
    <dt>Coffee</dt>
    <dd>- 500 gms</dd>
    <dt>Milk</dt>
    <dd>- 1 ltr Tetra Pack</dd>
  </dl>
</body>
</html>
Output:
```

A Description List

Coffee
- 500 gms
Milk
- 1 ltr Tetra Pack

What is HTML Tables?

An HTML Table is an arrangement of data in *rows* and *columns* in tabular format. Tables are useful for various tasks, such as presenting text information and numerical data. A table is a useful tool for quickly and easily finding connections between different types of data. Tables are also used to create databases.

HTML Table Code Example

```
<!-- index.html -->
<!DOCTYPE html>
<html>
<body>
  <table>
    <tr>
      <th>Firstname</th>
      <th>Lastname</th>
      <th>Age</th>
    </tr>
    <tr>
      <td>Priya</td>
      <td>Sharma</td>
      <td>24</td>
    </tr>
```

```

        <tr>
            <td>Arun</td>
            <td>Singh</td>
            <td>32</td>
        </tr>
    <tr>
        <td>Sam</td>
        <td>Watson</td>
        <td>41</td>
    </tr>
</table>
</body>

</html>

```

Output:

Firstname	Lastname	Age
Priya	Sharma	24
Arun	Singh	32
Sam	Watson	41

Simple HTML Table

Tags used in HTML Tables

HTML Tags

<table>

<tr>

<th>

<td>

<caption>

<thead>

<tbody>

Descriptions

Defines the structure for organizing data in rows and columns within a web page.

Represents a **row** within an HTML table, containing individual cells.

Shows a table **header** cell that typically holds titles or headings.

Represents a standard **data** cell, holding content or data.

Provides a title or description for the entire table.

Defines the header section of a table, often containing column labels.

Represents the main content area of a table, separating it from the header or footer.

<code><tfoot></code>	Specifies the footer section of a table, typically holding summaries or totals.
<code><col></code>	Defines attributes for table columns that can be applied to multiple columns at once.
<code><colgroup></code>	Groups together a set of columns in a table to which you can apply formatting or properties collectively.

Defining Tables in HTML

An HTML table is defined with the “table” tag. Each table row is defined with the “tr” tag. A table header is defined with the “th” tag. By default, table headings are bold and centered. A table data/cell is defined with the “td” tag.

Table Cells

Table Cell are the building blocks for defining the Table. It is denoted with `<td>` as a start tag & `</td>` as an end tag.

Syntax

`</td> Content...</td>`

Table Rows

The rows can be formed with the help of combination of Table Cells. It is denoted by `<tr>` and `</tr>` tag as a start & end tags.

Syntax

`</tr> Content...</tr>`

Table Headers

The Headers are generally used to provide the Heading. The Table Headers can also be used to add the heading to the Table. This contains the `<th>` & `</th>` tags.

Syntax

`</th> Content...</th>`

Example 1: Creating a simple table in HTML using a table tag.

```
<!-- index.html -->
<!DOCTYPE html>
<html>
<body>
  <table>
```

```

<tr>
  <th>Book Name</th>
  <th>Author Name</th>
  <th>Genre</th>
</tr>
<tr>
  <td>The Book Thief</td>
  <td>Markus Zusak</td>
  <td>Historical Fiction</td>
</tr>
<tr>
  <td>The Cruel Prince</td>
  <td>Holly Black</td>
  <td>Fantasy</td>
</tr>
<tr>
  <td>The Silent Patient</td>
  <td>Alex Michaelides</td>
  <td>Psychological Fiction</td>
</tr>
</table>
</body>
</html>

```

Output:

Book Name	Author Name	Genre
The Book Thief	Markus Zusak	Historical Fiction
The Cruel Prince	Holly Black	Fantasy
The Silent Patient	Alex Michaelides	Psychological Fiction

HTML Table

Adding a border to an HTML Table

A border is set using the CSS border property. If you do not specify a border for the table, it will be displayed without borders.

Syntax

```

table, th, td {
  border: 1px solid black; }

```

Example: Addition of the border to the HTML Table.

```

<!-- index.html -->
<!DOCTYPE html>
<html>
<head>

```

```

<style>
table, th, td {
    border: 1px solid black;
}
</style>
</head>
<body>
<table style="width:100%">
<tr>
<th>Firstname</th>
<th>Lastname</th>
<th>Age</th>
</tr>
<tr>
<td>Priya</td>
<td>Sharma</td>
<td>24</td>
</tr>
<tr>
<td>Arun</td>
<td>Singh</td>
<td>32</td>
</tr>
<tr>
<td>Sam</td>
<td>Watson</td>
<td>41</td>
</tr>
</table>
</body>
</html>

```

Output:

Firstname	Lastname	Age
Priya	Sharma	24
Arun	Singh	32
Sam	Watson	41

HTML Table with border

Adding Collapsed Borders in an HTML Table

For borders to collapse into one border, add the CSS border-collapse property.

Syntax

```

table, th, td { border: 1px solid black;
    border-collapse: collapse; }

```

Example: Addition of Collapsed Borders in HTML.

```
<!-- index.html -->
<!DOCTYPE html>
<html>
<head>
  <style>
table, th, td {
  border: 1px solid black;
  border-collapse: collapse;
}
  </style>
</head>
<body>
  <table style="width:100%">
    <tr>
      <th>Firstname</th>
      <th>Lastname</th>
      <th>Age</th>
    </tr>

    <tr>
      <td>Priya</td>
      <td>Sharma</td>
      <td>24</td>
    </tr>
    <tr>
      <td>Arun</td>
      <td>Singh</td>
      <td>32</td>
    </tr>
    <tr>
      <td>Sam</td>
      <td>Watson</td>
      <td>41</td>
    </tr>
  </table>
</body>
</html>
```

Output:

Firstname	Lastname	Age
Priya	Sharma	24
Arun	Singh	32
Sam	Watson	41

Adding Cell Padding in an HTML Table

Cell padding specifies the space between the cell content and its borders. If we do not specify a padding, the table cells will be displayed without padding.

Syntax

```
th, td {  
    padding: 20px; }
```

Example: Addition of Table cell padding in HTML.

```
<!-- index.html -->  
<!DOCTYPE html>  
<html>  
<head>  
    <style>  
table, th, td {  
    border: 1px solid black;  
    border-collapse: collapse;  
}  
th, td {  
    padding: 20px;  
}  
    </style>  
</head>  
<body>  
    <table style="width:100%">  
        <tr>  
            <th>Firstname</th>  
            <th>Lastname</th>  
            <th>Age</th>  
        </tr>  
        <tr>  
            <td>Priya</td>  
            <td>Sharma</td>  
            <td>24</td>  
        </tr>  
        <tr>  
            <td>Arun</td>  
            <td>Singh</td>  
            <td>32</td>  
        </tr>  
  
        <tr>  
            <td>Sam</td>  
            <td>Watson</td>  
            <td>41</td>
```

```

    </tr>
</table>
</body>
</html>

```

Output:

Firstname	Lastname	Age
Priya	Sharma	24
Arun	Singh	32
Sam	Watson	41

Adding Table cell padding

Adding Left Align Headings in an HTML Table

By default, the table headings are bold and centered. To left-align the table headings, we must use the CSS text-align property.

Syntax

```

th {
    text-align: left; }

```

Example: Explains the [text-align property](#) where the text is aligned to the left.

```

<!-- index.html -->
<!DOCTYPE html>
<html>

<head>
  <style>
    table, th, td {
      border: 1px solid black;
      border-collapse: collapse;
    }
    th, td {
      padding: 20px;
    }
    th {
      text-align: left;
    }
  </style>
</head>
<body>
  <table style="width:100%">

```

```

<tr>
  <th>Firstname</th>
  <th>Lastname</th>
  <th>Age</th>
</tr>
<tr>
  <td>Priya</td>
  <td>Sharma</td>
  <td>24</td>
</tr>
<tr>
  <td>Arun</td>
  <td>Singh</td>
  <td>32</td>
</tr>
<tr>
  <td>Sam</td>
  <td>Watson</td>
  <td>41</td>
</tr>
</table>
</body>
</html>

```

Output:

Firstname	Lastname	Age
Priya	Sharma	24
Arun	Singh	32
Sam	Watson	41

text-align Property

Adding Border Spacing in an HTML Table

Border spacing specifies the space between the cells. To set the border-spacing for a table, we must use the CSS border-spacing property.

Syntax table {
 border-spacing: 5px; }

Example: Explains the [border space](#) property to make the space between the Table cells.

```
<!-- index.html -->
<!DOCTYPE html>
<html>
<head>
  <style>
    table, th, td {
      border: 1px solid black;
    }
    table {
      border-spacing: 5px;
    }
  </style>
</head>
<body>
  <table style="width:100%">
    <tr>
      <th>Firstname</th>
      <th>Lastname</th>
      <th>Age</th>
    </tr>
    <tr>
      <td>Priya</td>
      <td>Sharma</td>
      <td>24</td>
    </tr>
    <tr>
      <td>Arun</td>
      <td>Singh</td>
      <td>32</td>
    </tr>
    <tr>
      <td>Sam</td>
      <td>Watson</td>
      <td>41</td>
    </tr>
  </table>
</body>
</html>
```

Output:

Firstname	Lastname	Age
Priya	Sharma	24
Arun	Singh	32
Sam	Watson	41

Border Spacing Property

Adding Cells that Span Many Columns in HTML Tables

To make a cell span more than one column, we must use the colspan attribute.

Example: Use of [colspan attribute](#) in HTML.

```
<!-- index.html -->
<!DOCTYPE html>
<html>

<head>
  <style>
    table, th, td {
      border: 1px solid black;
      border-collapse: collapse;
    }
    th, td {
      padding: 5px;
      text-align: left;
    }
  </style>
</head>
<body>
  <h2>Cell that spans two columns:</h2>
  <table style="width:100%">
    <tr>
      <th>Name</th>
      <th colspan="2">Telephone</th>
    </tr>
    <tr>
      <td>Vikas Rawat</td>
      <td>9125577854</td>
      <td>8565557785</td>
    </tr>
  </table>
</body>
</html>
```

Output:

Cell that spans two columns:

Name	Telephone	
Vikas Rawat	9125577854	8565557785

Adding Cells that span many rows in HTML Tables

To make a cell span more than one row, we must use the `rowspan` attribute.

Example: Use of the [rowspan attribute](#) in HTML.

```
<!-- index.html -->
<!DOCTYPE html>
<html>
<head>
  <style>
    table, th, td {
      border: 1px solid black;
      border-collapse: collapse;
    }
    th, td {
      padding: 5px;
      text-align: left;
    }
  </style>
</head>
<body>
  <h2>Cell that spans two rows:</h2>
  <table style="width:100%">
    <tr>
      <th>Name:</th>
      <td>Vikas Rawat</td>
    </tr>
    <tr>
      <th rowspan="2">Telephone:</th>
      <td>9125577854</td>
    </tr>
    <tr>
      <td>8565557785</td>
    </tr>
  </table>
</body>
</html>
```

Output:

Cell that spans two rows:

Name:	Vikas Rawat
Telephone:	9125577854
	8565557785

Adding a Caption in an HTML Table

To add a caption to a table, we must use the “caption” tag.

Syntax

```
<table style="width:100%">
```

```
<caption>DETAILS</caption>
```

Example: HTML Table caption by specifying the CSS properties for setting its width.

```
<!-- index.html -->
```

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
  <style>
```

```
    table, th, td {
```

```
      border: 1px solid black;
```

```
      border-collapse: collapse;
```

```
    }
```

```
    th, td {
```

```
      padding: 20px;
```

```
    }
```

```
    th {
```

```
      text-align: left;
```

```
    }
```

```
  </style>
```

```
</head>
```

```
<body>
```

```
  <table style="width:100%">
```

```
    <caption>DETAILS</caption>
```

```
    <tr>
```

```
      <th>Firstname</th>
```

```
      <th>Lastname</th>
```

```
      <th>Age</th>
```

```
    </tr>
```

```
    <tr>
```

```
      <td>Priya</td>
```

```
      <td>Sharma</td>
```

```
      <td>24</td>
```

```
    </tr>
```

```
  </table>
```

```

        <td>Arun</td>
        <td>Singh</td>
        <td>32</td>
    </tr>
<tr>
    <td>Sam</td>
    <td>Watson</td>
    <td>41</td>
</tr>
</table>
</body>
</html>

```

Output:

DETAILS		
Firstname	Lastname	Age
Priya	Sharma	24
Arun	Singh	32
Sam	Watson	41

Adding a Background Colour to the Table

A color can be added as a background in an HTML table using the “*background-color*” option.

Syntax table#t01 { width: 100%;
background-color: #f2f2d1;}

Example: Addition of the Table background color in HTML.

```

<!-- index.html -->
<!DOCTYPE html>
<html>
<head>
    <style>
        table, th, td {
            border: 1px solid black;
            border-collapse: collapse;
        }
        th, td {
            padding: 5px;          text-align: left;
        }
        table#t01 {
            width: 100%;
            background-color: #f2f2d1;
        }
    </style>

```

```
</head>
<body>
  <table style="width:100%">
    <tr>
      <th>Firstname</th>
      <th>Lastname</th>
      <th>Age</th>
    </tr>
    <tr>
      <td>Priya</td>
      <td>Sharma</td>
      <td>24</td>
    </tr>
    <tr>
      <td>Arun</td>
      <td>Singh</td>
      <td>32</td>
    </tr>
    <tr>
      <td>Sam</td>
      <td>Watson</td>
      <td>41</td>
    </tr>
  </table>
  <br />
  <br />
  <table id="t01">
    <tr>
      <th>Firstname</th>
      <th>Lastname</th>
      <th>Age</th>
    </tr>
    <tr>
      <td>Priya</td>
      <td>Sharma</td>
      <td>24</td>
    </tr>
    <tr>
      <td>Arun</td>
      <td>Singh</td>
      <td>32</td>
    </tr>
    <tr>
      <td>Sam</td>
      <td>Watson</td>
      <td>41</td>
    </tr>
  </table>
```

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

Output:

Firstname	Lastname	Age
Priya	Sharma	24
Arun	Singh	32
Sam	Watson	41

Firstname	Lastname	Age
Priya	Sharma	24
Arun	Singh	32
Sam	Watson	41

Creating Nested Tables

Nesting tables simply means making a Table inside another Table. Nesting tables can lead to complex tables layouts, which are visually interesting and have the potential of introducing errors.

Example: Nesting of HTML Table.

```
<!-- index.html -->
<!DOCTYPE html>
<html>
<body>
  <table border=5 bordercolor=black>
    <tr>
      <td> First Column of Outer Table </td>
      <td>
        <table border=5 bordercolor=grey>
          <tr>
            <td> First row of Inner Table </td>
          </tr>
          <tr>
            <td> Second row of Inner Table </td>
          </tr>
        </table>
      </td>
    </tr>
  </table>
</body>
</html>
```

Output:

Fisrt Column of Outer Table	First row of Inner Table
	Second row of Inner Table

HTML Images

The **HTML tag** is used to embed an image in web pages by linking them. It creates a placeholder for the image, defined by attributes like src, width, height, and alt, and does not require a closing tag.

There are 2 ways to insert the images into a webpage:

- By providing a full path or address (URL) to access an internet file.
- By providing the file path relative to the location of the current web page file.

We will first discuss inserting the image into the webpage & simultaneously, we will understand both the above approaches.

Adding images on a webpage:

The tag embeds images in web pages, requiring only attributes without a closing tag. Images enhance design and content quality, linked via the tag to preserve space and optimize webpage performance.

Syntax:

 Attribute: The ** tag has following attributes:

Attribute	Description
src	Specifies the path to the image file.
alt	Provides alternate text for the image, useful for accessibility and when the image cannot be displayed.
crossorigin	Allows importing images from third-party sites with cross-origin access, typically used with canvas.
height	Specifies the height of the image.
width	Specifies the width of the image.
ismap	Specifies an image as a server-side image map.
loading	Specifies whether the browser should defer image loading or load it immediately.
longdesc	Specifies a URL to a detailed description of the image.

referrer policy Specifies which referrer information to use when fetching the image. sizes Specifies image sizes for different page layouts.

src set Specifies a list of image files to use in different situations, allowing for responsive images. use map Specifies an image as a client-side image map.

src attribute:

The src attribute in tag specifies the image's location. If the URL is correct, the browser retrieves and displays the image; otherwise, it shows a broken link icon.

Example : In this example we displays an image using the tag with src attribute pointing to a GeeksforGeeks logo. It enhances content by visually representing the website's identity.

```
<!DOCTYPE html>
<html>
<head>
  <title>Welcome To GFG</title>
</head>
<body>
  <h2>GeeksforGeeks</h2>
  <p>This is the demo of <img /> tag.</p>
  <img src=
"https://media.geeksforgeeks.org/wp-content/uploads/20210915115837/gfg3-300x300.png"
alt="GFG image" />
</body>
</html>
```

Output:



HTML Images Example Output alt attribute :

The **alt attribute** in tag provides a text alternative if the image fails to load. It aids accessibility for users unable to view images due to slow internet, src errors, or screen reader usage.

Example : The example illustrates the use of the alt attribute in the tag.

```
<!DOCTYPE html>
<html>
<head>
  <title>Alt Attribute Example</title>
</head>
<body>
  <p>Using Alt Attribute</p>
  
</body>
</html>
```

Output:



Setting width and height of Image:

The width and height attributes are used to specify the height and width of an image. The attribute values are specified in pixels by default. The width and height attributes are always declared in pixels.

Example: In this example we set the width and height of an image using the tag's width and height attributes. It showcases the GeeksforGeeks logo with dimensions of 300×300 pixels.

```
<!DOCTYPE html>
<html>
<head>
  <title>
    Setting width and height of image
  </title>
</head>
<body>
  <p> Setting width and height of image</p>
  <img src=
"https://media.geeksforgeeks.org/wp-content/cdn-uploads/20190710102234/download3.png"
alt="GeeksforGeeks logo" width="300" height="300" />
</body>
</html>
```

Output:

Setting width and height of image



Adding titles to Image:

To add a title to an image, include the title attribute in the `` tag, providing descriptive text for enhanced user interaction. Please refer to the [HTML | title Attribute](#) article for further details.

Example: In this example we are showing an image with width, height, and **title attributes** using the `` tag. The title attribute provides descriptive text for user interaction.

```
<!DOCTYPE html>
<html>
<head>
  <title>
    Inserting an image using "img" tag
  </title>
</head>
<body>
  <p>Inserted image using image tag</p>
  <img src=
"https://media.geeksforgeeks.org/wp-content/cdn-
uploads/20190710102234/download3.png"alt="GeeksforGeekslogo"width="200"
height="200"
title="Logo of GeeksforGeeks" />
</body>
</html>
```

Setting style to the Image:

In this example, we are using the border property to decorate the image. By default, every picture has a border around it. By using the border attribute, the thickness of the border can be changed. A thickness of "0" means that there will be no border around the picture.

Example: This example illustrates the use of style property inside the tag in HTML.

```
<!DOCTYPE html>
<html>
<head>
  <title>Setting border to image</title>
</head>
<body>
  <p>Setting style to the Image</p>
  <img src=
"https://media.geeksforgeeks.org/wp-content/cdn-uploads/20190710102234/download3.png"
alt="GeeksforGeeks logo" width="200" height="200" border="5" />
</body>

</html>
```

Output:

Setting style to the Image



Aligning an Image:

Aligning an image in HTML involves using the align attribute within the tag to position it horizontally. Options include left, right, or center, enhancing page layout and visual appeal.

Example: In this example we aligns an image to the right using the “align” attribute within the tag. It positions the GeeksforGeeks logo for visual enhancement.

```
<!DOCTYPE html>
<html>
<head>
  <title>Aligning an image</title>
</head>
<body>
  <p>Aligning an Image</p>
  
```

```
</body>
```

```
</html>
```

Output:

Aligning an Image



Adding Image as a Link:

To add an image as a link, enclose the `` tag within an `<a>` tag, setting the image's source with the `href` attribute. This creates a clickable image linking to external content, such as images, videos, or other web pages.

File paths are of two types:

- **Absolute File Paths:** It always contains the root element along with the complete directory list required to locate the file.
- **Relative File Paths:** It is the hierarchical path representation that locates the file or folder on a file system beginning from the current directory.

Example: In this example we embeds the GeeksforGeeks logo as a clickable link using the `<a>` tag, wrapping the `` tag with the link destination specified in the `href` attribute.

```
<!DOCTYPE html>
```

```
<html>
```

```
  <head>
```

```
    <title>Adding image as link</title>
```

```
  </head>
```

```
  <body>
```

```
    <h3>GeekforGeeks</h3>
```

```
    <p>Adding image as link</p>
```

```
    <a href=
```

```
"https://ide.geeksforgeeks.org/tryit.php" >
```

```
  <img src=
```

```
"https://media.geeksforgeeks.org/wp-content/cdn-uploads/20190710102234/download3.png"
```

```
alt="GeeksforGeeks logo" />
```

```
  </a>
```

```
  </body>
```

```
</html>
```

Output:

GeekforGeeks

Adding image as link



Adding animated Image:

To add an animated image in HTML, use the `` tag with the `src` attribute pointing to a GIF file, providing engaging motion to enhance webpage content.

Example: This example explains the addition of an image in the GIF format in HTML.

```
<!DOCTYPE html>
<html>
<body>
  <h3>Adding a gif file on a webpage</h3>
  
</body>

</html>
```

Output:

Adding a gif file on a webpage



Common Image Format: Here is the commonly used image file format that is supported by all the browsers.

S.No.	Abbreviation	File Type	Extension
1.	PNG	Portable Network Graphics.	.png
2.	JPEG	Joint Photographic Expert Group image.	.jpg,.jpeg,.jfif, .jpeg, .jpp
3.	SVG	Scalable Vector Graphics.	.svg.
4.	GIF	Graphics Interchange Format.	.gif
5.	ICO	Microsoft Icon.	.ico, .cur
6.	APNG	Animated Portable Network Graphics.	.apng

Video and Audio Elements in HTML5 (Live Playground)

Introduction

Incorporating multimedia content such as videos and audio files can greatly enhance your website's interactivity and user experience. HTML5 introduced the `<video>` and `<audio>` elements, which enable developers to easily embed multimedia content into web pages. In this tutorial, we will explore these elements and their usage, along with best practices for accessibility.

The `<video>` Element

The `<video>` element allows you to embed video files into your web pages. It provides several attributes to control video playback, such as controls, autoplay, loop, and poster.

Example:

HTML

```
<video width="320" height="180" controls>
  <source src="example-video.mp4" type="video/mp4" />
  <source src="example-video.webm" type="video/webm" />
  Your browser does not support the video element. </video>
```

In this example, the `<video>` element specifies the `width`, `height`, and `controls` attribute to display video controls. The `<source>` elements define different video file formats to ensure compatibility with various browsers. The text inside the `<video>` element is displayed if the browser does not support the element.

HTML

```
<video width="320" height="180" controls>
<source src="example-video.mp4" type="video/mp4" />
<source src="example-video.webm" type="video/webm" />
Your browser does not support the video element.
</video>
```

The <audio> Element

The <audio> element enables you to embed audio files into your web pages. Similar to the <video> element, it provides attributes like `controls`, `autoplay`, and `loop`.

Example:

HTML

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

What is Web Forms 2.0 ?

An HTML form is a special part of a web page that contains certain controls, including labels, text fields, password fields, hidden fields (used by the software), radio buttons, checkboxes, fieldsets, legends, and submit buttons. During the interaction with these forms, users provide all the necessary information for the server to process. Currently, client-side scripting is used for effects and simple validation (usually using Javascript), but HTML5 will reduce the need for this materially in the near future.

WHAT IS WEB FORMS 2.0?



What Is Web Forms 2.0

HTML5 form elements are supported by many browsers, although some of them differ in their compatibility and display features. This article explores both the main components of an HTML5 website form and those new features designed to make HTML more user-friendly.

HTML5 Input Element (Text):


E-mail: It will only accept email values. Input fields that need to contain an email address should use this type. When you try to submit a simple text, it only asks for an email address in the format email@example.com.

Example:

```
<form>  
  <label for="myemail">Enter Email Address:</label>  
  <input type="email" id="myemail" required> </form>
```

Output:

Enter Email Address:

 Please include an '@' in the email address. 'abc' is missing an '@'.


Number: This field accepts only numerical values. The step attribute specifies the precision, which defaults to 1.

Example:

```
<form>  
  <label for="mynumber">Enter a Number:</label>  
  <input type="number" min="1" max="10" step="0.5" id="mynumber"> </form>
```

Output:

Enter a Number:

 Value must be less than or equal to 10.

Time: The time (hour, minute, second, fractional second) is encoded according to ISO 8601.

Example:

```
<form>  
  <label for="mytime">Select Time:</label>  
  <input type="time" id="mytime"> </form>
```

Output:

Select Time:

06	18	AM
07	19	PM
08	20	
09	21	
10	22	
11	23	
12	24	

Week: A date that is composed of a weekday and a year is encoded according to ISO 8061.

Example:

```
<form>  
  <label for="myweek">Select Week:</label>  
  <input type="week" id="myweek">  
</form>
```

Output:

Select Week:

November, 2021								↑	↓
Week	Mo	Tu	We	Th	Fr	Sa	Su		
44	1	2	3	4	5	6	7		
45	8	9	10	11	12	13	14		
46	15	16	17	18	19	20	21		
47	22	23	24	25	26	27	28		
48	29	30	1	2	3	4	5		
49	6	7	8	9	10	11	12		
Clear								This week	

Month: According to ISO 8061, dates consisting of a year and a month are encoded.

Example:

```
<form>  
  <label for="mymonth">Select Month:</label>  
  <input type="month" id="mymonth"> </form>
```

Output:

Select Month:

2021			
Jan	Feb	Mar	Apr
May	Jun	Jul	Aug
Sep	Oct	Nov	Dec
Clear			
This month			

Date: A date (year, month, day) is encoded using the ISO 8601 standard.

Example:

```
<form>
  <label for="mydate">Select Date:</label>
  <input type="date" value="2019-04-15" id="mydate"> </form>
```

Output:

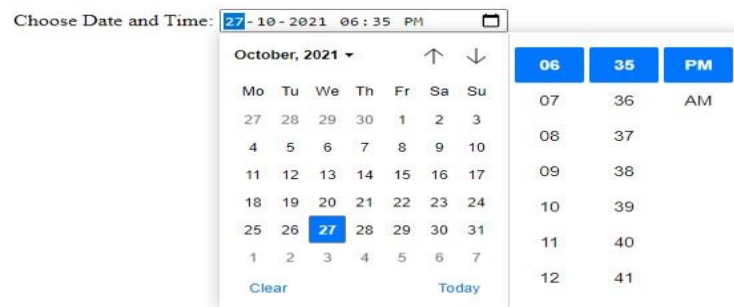
A screenshot of a web browser's date picker. The input field shows '15-10-2021'. Below it, a calendar for October 2021 is displayed. The 15th is highlighted in blue. The calendar shows days of the week (Mo to Su) and dates from 1 to 31. There are 'Clear' and 'Today' buttons at the bottom.

datetime-local: Date and time encoded in ISO 8601 (year, month, day, hour, minute, second, fraction of a second) without time zone information.

Example:

```
<form>
  <label for="mydatetime">Choose Date and Time:</label>
  <input type="datetime-local" id="mydatetime"> </form>
```

Output:

A screenshot of a web browser's datetime-local picker. The input field shows '27-10-2021 06:35 PM'. Below it, a calendar for October 2021 is displayed with the 27th highlighted in blue. To the right of the calendar, there are buttons for hour (06, 07, 08, 09, 10, 11, 12), minute (35, 36, 37, 38, 39, 40, 41), and AM/PM (PM, AM). There are 'Clear' and 'Today' buttons at the bottom.

Week: A drop-down calendar lets the user choose a week and year from the week input type.

Example:

```
<form>
  <label for="myweek">Select Week:</label>
  <input type="week" id="myweek"> </form>
```

Output:



range: For input fields, the range type is used to represent a range of values.

Example:

`<form>`

`<label for="mynumber">Select a Number:</label>`

`<input type="range" min="1" max="10" step="0.5" id="mynumber"> </form>`

Output:

Select a Number:

URL: It can only accept URL values. In this type of field, URL addresses should be entered. Those who submit simple text entries must specify the URL, either `http://www.example.com` or <http://example.com>

Example:

`<form>`

`<label for="myurl">Enter Website URL:</label>`

`<input type="url" id="myurl" required> </form>`

Output:

Enter Website URL:

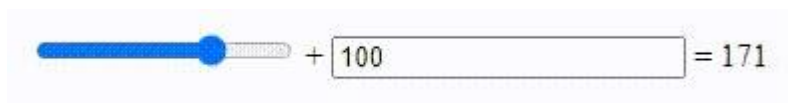
The `<output>` element:

- HTML5 introduced a new element called `<output>` to represent the output of different types of scripts, like the ones written by a script.
- An output element's `for` attribute is used to specify a relationship between that element and other elements in the document that influence the calculation.
- A space-separated list of IDs of other elements is the value for this attribute.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Example of HTML output Tag</title>
</head>
<body>
  <form oninput=
"result.value=parseInt(a.value)+parseInt(b.value)">
    <input type="range" id="a" value="50"> +
    <input type="number" id="b" value="100"> =
    <output name="result" for="a b"></output>
  </form>
</body>
</html>
```

Output:



Web Form2.0 Attributes:

1. placeholder attribute

- HTML5 introduced a new attribute called placeholder.
- With placeholder attributes on <input> and <textarea> elements, users are able to know what they can enter in the field.
- The placeholder text cannot contain line-feeds or carriage returns.

Example:

```
<input type="text" name="search" placeholder="search the internet"/>
```

2. autofocus attribute

- This is a simple one-step pattern that can be easily programmed in JavaScript as soon as the document loads. When the form loads, it automatically focuses on a particular field in the document.

Example:

```
<input type="text" name="search" autofocus/>
```

3. required attribute:

- The required attribute is used in place of Javascript validations.

- Due to this attribute, Javascript is now only required for client-side validations where an empty text box cannot be submitted.

Example:

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

HTML <canvas> Tag

The **<canvas>** tag in HTML is used to draw graphics on a web page using [JavaScript](#). It can be used to draw paths, boxes, texts, [gradients](#), and adding images. By default, it does not contain borders and text.

Syntax

`<canvas id = "script"> Contents... </canvas>` **Note:**

- The `<canvas>` tag is new in HTML5.
- The following are the default values for `<canvas>` element that can be displayed by most of the Browsers:

```
canvas { height: 150px; width: 300px; }
```

Attributes

The `<canvas>` tag accepts two attributes which are described below:

Attributes

height

width

Descriptions

This attribute is used to set the height of the canvas by taking the value in pixels and its default value is 150.

This attribute is used to set the width of the canvas by taking the value in pixels and its default value is 300.

The `<canvas>` tag supports both **Global Attributes** & **Event Attributes** in HTML.

Example 1: In this example, we will create a canvas element with a width of 200 pixels and a height of 100 pixels, displayed with a black border.

```
<!DOCTYPE html>
<html>
<body>
  <canvas id="GeeksforGeeks"width="200" height="100" style="border:1px solid black">
  </canvas>
</body>
</html>
```

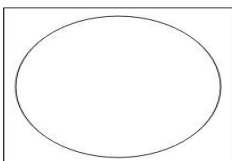
Output:



Example 2: In this example, a 200×200 canvas with ID “geeks” displays a black-bordered circle at coordinates (100,100) using HTML5 Canvas.

```
<!DOCTYPE html>
<html>
<body>
  <canvas id="geeks"height="200" width="200"style="border:1px solid black">
  </canvas>
  <script>
    var c = document.getElementById("geeks");
    var cx = c.getContext("2d");
    cx.beginPath();
    cx.arc(100, 100, 90, 0, 2 * Math.PI);
    cx.stroke();
  </script>
</body>
</html>
```

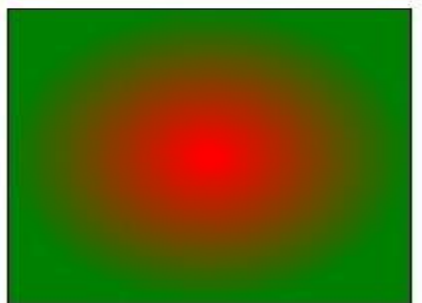
Output:



Example 3: In this example, a 200×200 canvas with ID “geeks” displays a radial gradient that transitions from red to green, filling the entire canvas using HTML5 Canvas.

```
<!DOCTYPE html>
<html>
<body>
  <canvas id="geeks"width="200" height="200" style="border:1px solid black">
  </canvas>
  <script>
    var c = document.getElementById("geeks");
    var cx = c.getContext("2d");
    var grd = cx.createRadialGradient (100, 100, 5, 100, 100, 100);
    grd.addColorStop(0, "red");
    grd.addColorStop(1, "green");
    cx.fillStyle = grd;
    cx.fillRect(0, 0, 200, 200);
  </script>
</body>
</html>
```

Output:



HTML5 MathML

The **MathML** comes in HTML5 the current MathML version is 3 it was introduced in the year of 2015. MathML stands for Mathematics Markup Language. It is used to represent mathematical equations or expressions in web browsers like other HTML elements. The 1st version of MathML was released in the year of 1998 and after that the 2nd version was released.

Basically, MathML is a complex mathematical formula or equation visual representation made easy. The MathML is supported in HTML5, all the MathML tags must be used inside the **$** and **$** tags. The MathML is used to describe mathematics as a basis for the machine to machine communication, it is intended to be handled by specialized authoring tools such as equation editors and it is meaningful to other applications also.

MathML is not-capable:

- The MathML is not a calculator to calculate or solve complex equations it is just a way to show the equation.
- It is not a programming language it is a Mathematics Markup Language.

MathML is capable:

- The MathML can be used to represent the Matrix form.
- The MathML can be used to represent the Partial Differential Equation.
- The MathML can be used to represent the Chemical Reaction Equation.

The below example gives you a brief idea about **MathML**:

Example:

```
<!DOCTYPE html>
<html>
<head>
  <title>HTML5 MathML</title>
</head>
<body style="text-align:center;">
  <h1 style="color:green">
    GeeksforGeeks
  </h1>
  <h3>HTML5 MathML</h3>
  <!--start tag of the whole representation-->
  <math>
    <!-- Creating Matrix -->
    <mrow>
      <mi>A</mi>
      <mo>=</mo>
      <mfenced open="[" close="]">
        <mtable>
          <mtr>
            <mtd>
              <mi>a</mi>
            </mtd>
            <mtd>
              <mi>b</mi>
            </mtd></mtr>
          <mtr>
            <mtd>
              <mi>x</mi>
            </mtd>
            <mtd>
```

```

        <mi>y</mi>
      </mtd> </mtr>
    </mtable>
  </mfenced> </mrow>
<!-- Creating equation -->
<br><br>
<msub>
  <mi>Geeks</mi>
  <mn>4</mn>
</msub>
<mo>+</mo>
<mn>Geeks</mn>
<mo>=</mo>
<msub>
  <mi>G</mi>
</msub>
<mo>→</mo>
<msub>
  <mi>e</mi>
  <mn>2</mn>
</msub>
<mo>→</mo>
<mi>k</mi>
<mi>s</mi>
<mn>4
</mn>
<msub>
  <mi>G</mi>
</msub>
<mo>→</mo>
<msub>
  <mi>e</mi>
  <mn>2</mn>
</msub>
<mo>→</mo>
<mi>k</mi>
<mi>s</mi>
</math> </body>
</html>

```

Output:

GeeksforGeeks

HTML5 MathML

$$A = \begin{bmatrix} a & b \\ x & y \end{bmatrix}$$

Geeks 4 + Geeks = G → e 2 → k s 4 G → e 2 → k s

Tag	Description
-----	-------------

MathML <math>	tag MathML element you want to use they should wrapped inside of the <math> tag.
---------------	--

MathML <maction>	tag This tag is used to show the bind action of any expression.
------------------	---

MathML <menclose>	This tag is used to renders the contents which is inside of an enclosing notation specified
-------------------	---

Tag	by the notation attribute.
-----	----------------------------

MathML <merror>	Tag This tag is used to wrap the expression in a box, makes that expression eye-catching.
-----------------	---

MathML <mfenced>	
------------------	--

	This tag is used to add custom open and closing parentheses. tag
--	--

MathML <mfrac>	tag This tag is use to add fraction symbol between two digits or equations.
----------------	---

	This tag is used only for those characters or symbols that are not available on Unicode
--	---

MathML <mglyph>	Tag characters.
-----------------	-----------------

MathML <mi>	This tag is used as an identifier such as any kind of symbol or function. You can put any statement inside this tag.
-------------	--

MathML <mlabeledtr>	This tag is used to represent a label in a row, either on the left or on the right side inside of the <mtable> element.
---------------------	---

tag	
-----	--

MathML <mmultiscripts>	This tag is used to create multi-dimensional matrices. The degree depends on the conditionality of a representative array.
------------------------	--

Tag	
-----	--

MathML <mn>	THis tag is used to display a numeric character which is normally a sequence of digits with a possible separator.
-------------	---

MathML <mo>	This tag is used print operator between elements. Any kind of mathematical operator can be used by this tag.
-------------	--

MathML <mover>	This tag is used to attach an accent or a limit over an expression.
----------------	---

Tag	
-----	--

MathML <mpadded>	This tag is used to add extra padding and to set the general adjustment of position and size of enclosed contents.
------------------	--

Tag	
-----	--

MathML
`<mphantom>` Tag It is used to rendered invisibly but the dimensional are still kept.

MathML
`<mroot>` Tag This tag is use to display the power of the root like root squire.

MathML
`<mrow>` Tag This tag is used to create a row that contains some mathematical expression or any random text.

MathML `<ms>` Tag This tag is used to represent string that will represent the mathematical expression by programming languages and computer algebra systems.

This tag is used to print blank space. The size of the blank space has to be mentioned in MathML `<mSPACE>` Tag the attributes.

MathML `<msqrt>` Tag It is use to display the root squire of the element content. It is an inbuilt tag in HTML5. This tag is used to change the styles of the children MathML `<style>` Tag elements.

MathML `<msub>` Tag It is used to print the base power on any expression.

MathML `<msubsup>` Tag It is used to print base power and power on any expression. Tag

Tag	Description
MathML <code><msup></code> Tag	It is used to print power on any expression.
MathML <code><mtable></code> Tag	This tag is similar to the normal HTML <code><table></code> tag.
MathML <code><mtd></code> Tag	This tag is used to create table data of a table or the matrices in HTML5
MathML <code><mtext></code> Tag	This tag is used to print any text before after any expression.
MathML <code><mtr></code> Tag	This tag is used to create row of a table or the matrices in HTML5.
MathML <code><mth></code> Tag	This tag is used to create header of a table or the matrices in HTML5.
MathML <code><munder></code> Tag	This tag is used attach any accent or limit under the expression.
MathML <code><munderover></code> Tag	This tag is used attach any accent or limit under the expression plus over the expression.
MathML <code><semantics></code> Tag	This tag is used to markup the mathematics there are two possible ways to markup mathematics

HTML SVG Basics

SVG stands for **Scalable Vector Graphics** . It defines vector-based graphics in XML format. SVG graphics do not lose any quality when zoomed or resized, and every element and attribute in SVG files can be animated.

Advantages of SVG:

The advantages of using SVG over other image formats (like JPEG and GIF) are:

- SVG images scale without losing quality, making them perfect for responsive designs.
- SVG files are often smaller than bitmap images, reducing load times.
- SVG images can be edited easily with text editors, allowing for quick changes.
- SVG content is accessible to screen readers, enhancing web accessibility.
- SVG supports CSS and JavaScript animations, allowing for interactive and dynamic graphics.

Differences between SVG and Canvas

SVG	Canvas
SVG is a language for describing 2D graphics in XML	Canvas draws 2D graphics, on the fly with JavaScript
If attributes of an SVG object are changed, the browser can automatically re-render the shape	Canvas is rendered pixel by pixel. In Canvas, once the graphic is drawn, it is forgotten by the browser.
SVG is resolution-independent	CANVAS is resolution-dependent.
SVG supports event handlers	CANVAS doesn't have support for event handlers.

Example 1: Drawing a Line

In this example we displays an SVG line on a webpage. It includes an <h2> header and a blue line drawn diagonally within an SVG element.

```
<!DOCTYPE html>
<html>
  <head>
    <title>HTML SVG</title>
  </head>
```

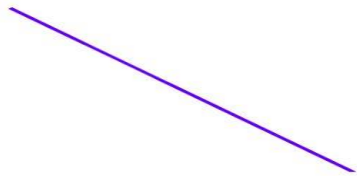
```

<body>
  <h2>Welcome To GeeksforGeeks</h2>
  <svg height="250" width="600"> <line x1="10" y1="10"x2="400" y2="400"
    style="stroke:rgb(0,0,255);stroke-width:3" />
  </svg>
</body>
</html>

```

Output:

Welcome To GeeksforGeeks



Example 2: Drawing a Circle

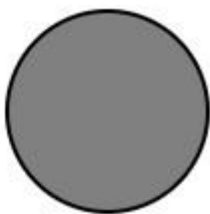
In this example we uses the <svg> tag to draw a grey circle with a black border. The circle has a center at (80, 80) and a radius of 50.

```

<!DOCTYPE html>
<html>
  <head>
    <title>HTML SVG</title>
  </head>
<body>
  <!-- html svg tag is used here -->
  <svg width="200" height="200">
    <circle cx="80" cy="80" r="50" stroke="black" stroke-width="2" fill="grey" />
  </svg>
</body>
</html>

```

Output:



What are the different types of storage in HTML5 ?

In this article, you will get to know about different types of Web storage in [HTML5](#). The web storage is more secure and large amounts of data can be stored locally on the client-side web browser. All the data is stored in key-value pairs.

In [HTML5](#) there are two types of web storage API.

- **local Storage**
- **Session Storage**

local Storage: It is used to store data on the client side. It has no expiration time, so the data in the Local Storage exists always till the user manually deletes it.

Syntax:

- **For storing data in web storage:** The key and value both should be string or number;

```
LocalStorage.setItem("key", "value");
```

- **For getting data from web storage:** We will pass the key and it will return value.

```
LocalStorage.getItem("key");
```

Example: In this example, we will use local storage.

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
color: green;
text-align: center;
font-size: 30px;
margin-top: 30px;
font-style: italic;
}
#data {
text-align: center;
}
</style>
</head>
<body>
<input id="name" type="text" placeholder="enter your name" />
<button type="submit" onClick="handleClick()">click </button>
<br />
```

```

<div id="data"></div>
<script>
    function handleClick() {
        if (typeof Storage !== "undefined") {
            let name = document.getElementById("name").value;
            localStorage.setItem("name", name);
            document.getElementById("data").innerHTML =
                "Welcome To GeeksforGeeks" + " " + localStorage.name;
        }
    }
    else {
        alert("Sorry! your browser doesn't support Web Storage");
    }
}
</script>
</body>

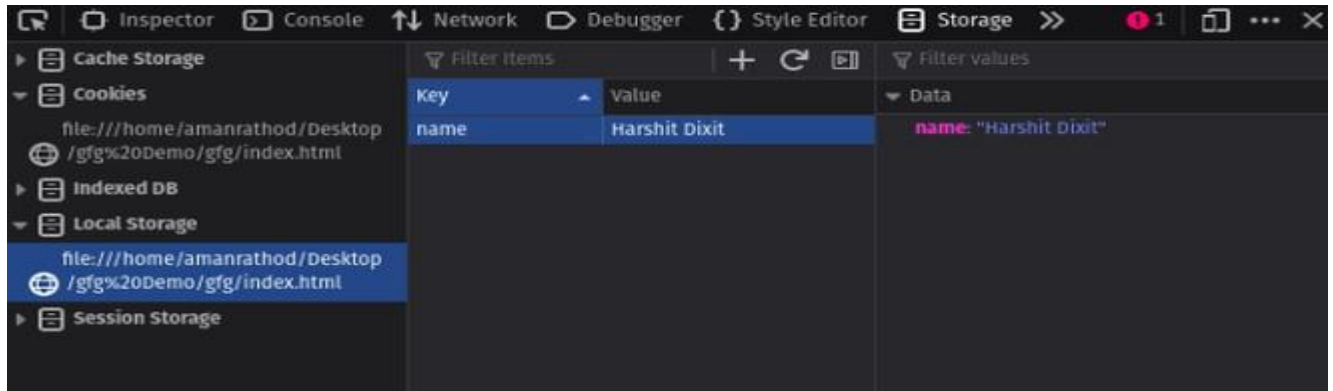
</html>

```

Output:



Stored data on Local Storage :



We can clearly see that local storage items are stored in the form of key/value pair and you can check by inspecting elements on the web page and then go to the Application option where you will find the local storage.

As the local Storage object stores the data with no expiration date, you can cross-check this by closing the current tab and visiting the same page again, you will find the same data is present in the local Storage of that tab or window.

Session Storage: It is used to store data on the client-side. Data in the Session Storage exist till the current tab is open, if we close the current tab then our data will also erase automatically from the Session Storage.

Syntax:

- **For storing data in web storage:**

```
SessionStorage.setItem("key", "value");
```

- **For getting data from web storage:**

```
SessionStorage.getItem("key");
```

Example: In this example, we will use session storage

```
<!DOCTYPE html>
<html>
<head>
<style>
  body {
    color: green;
    text-align: center;
    font-size: 30px;
    margin-top: 30px;
    font-style: italic;
  }
  #data {
    text-align: center;
  }
</style>
</head>
<body>
  <input id="name" type="text" placeholder="enter your name">
  <button type="submit" onClick="handleClick()">click </button>
  <br>
  <div id="data"></div>
  <script>
    function handleClick() {
      if (typeof (Storage) !== "undefined") {
        let name = document.getElementById("name").value;
        sessionStorage.setItem("name", name);
        document.getElementById("data").innerHTML =
          ("Welcome To GeeksforGeeks" + " " + sessionStorage.name);
      }
    }
  </script>
</body>
</html>
```

```

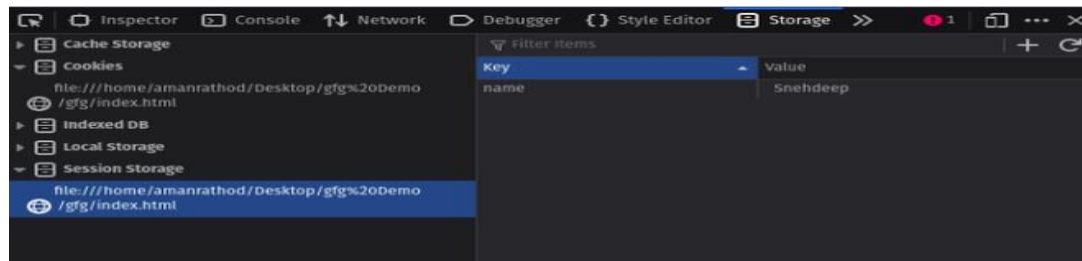
else {
alert("Sorry! your browser is not supporting the browser")
}
}
</script>
</body>
</html>

```

Output:

enter your name

Stored data on Session Storage :



HTML Geolocation

The HTML Geolocation is used to get the geographical position of a user. Due to privacy concerns, this position requires user approval. **Geo-location** in HTML5 is used to share the location with some websites and be aware of the exact location. It is mainly used for local businesses, and restaurants, or showing locations on the map. It uses JavaScript to give latitude and longitude to the backend server.

Most browsers support Geolocation API. Geo-location API uses a global navigator object. In this article, we will know **HTML Geolocation**, various properties, methods & their implementation through examples.

Syntax:

```
var loc = navigator.geolocation
```

Location Properties:

The following table determines properties used in `getCurrentPosition()` and their returning values.

Location Properties**coords.latitude****coords.accuracy****coords.longitude****coords.altitude****coords.altitudeAccuracy****coords.heading****Description**

Always returns latitude as a decimal number.

Always returns the accuracy of position.

Always returns longitude as a decimal number.

Returns the altitude in meters above sea level if available.

Returns altitude accuracy of position if available.

Returns heading in degree

clockwise from North if available

Location Properties	Description
coords.speed	Returns speed in mps, if available.
Timestamp	Returns date or time of response if timestamp available

Geolocation Methods:

The Geolocation has following methods which make it interesting and easier to work.

Geolocation Methods	Description
getCurrentPosition()	It fetches periodic updates of the user's current location.
watchPosition()	It fetches periodic updates of the user's current location.
clearWatch()	It cancels a watchPosition call currently in execution.

Example: This example explains returning the user's current location using the `getCurrentPosition()` method.

```
var loc = navigator.geolocation; function getLoc() {  
    loc.getCurrentPosition(showLoc, errHand); }
```

The above function can also be written without creating a navigator object as shown below:

```
function getlocation() {  
    navigator.geolocation.getCurrentPosition(showLoc, errHand); }
```

Example: In this example, we simply display the current location with the help of latitude and longitude via HTML Geolocation.

```
<!DOCTYPE html>  
<html>  
<body>  
    <p>Displaying location using Latitude and Longitude</p>  
    <button class="geeks" onclick="getlocation()">  
        Click Me  
    </button>  
    <p id="demo1"></p>  
    <script>  
        let variable1 = document.getElementById("demo1");  
        function getlocation() {  
            navigator.geolocation.getCurrentPosition(showLoc);  
        }  
        function showLoc(pos) {  
            variable1.innerHTML = "Latitude: " + pos.coords.latitude + "<br>Longitude: " +  
pos.coords.longitude;    }  
    </script>  
</body>  
</html>
```

Output:

Displaying location using Latitude and Longitude

Click Me

Latitude: 28.502412399999997
Longitude: 77.40561009999999

Error and Rejection Handling:

It is very important to handle the errors generated in Geolocation and show a required message when an error occurs. Functions like `getCurrentPosition()` make use of an error handler to handle the error generated (function `errHand` as used in the above example). The `PositionError` object used by the error handler has two properties that let the function handle error efficiently.

- Code
- Message

Generated errors in geolocation:

Error	Error Description
UNKNOWN_ERROR	An unknown error
PERMISSION_DENIED	The application doesn't have permission to make use of location services
POSITION_UNAVAILABLE	The location of the device is uncertain
TIMEOUT	The time to fetch location information exceeded the maximum timeout interval

Example: In this example we will see the implementation of geo location in HTML document.

```
<!DOCTYPE html>
<html>
<head>
  <title>Errors in geolocation</title>
  <style>
    .gfg {
      font-size: 40px;
      font-weight: bold;
      color: #009900;
      margin-left: 20px;
    }
    .geeks {
      margin-left: 150px;
    }
  </style>
</head>
<body>
  <div class="gfg">
    <h1>Geolocation</h1>
  </div>
  <div class="geeks">
    <h2>Geolocation</h2>
  </div>
</body>
</html>
```

```

p {
    font-size: 20px;
    margin-left: 20px;
}
</style> </head>
<body>
    <div class="gfg">GeeksforGeeks</div>
    <p>Error handling in geolocation</p>
    <button class="geeks" onclick="getLocation()"> Click </button>
    <p id="demo1"></p>
    <script>
        var variable1 = document.getElementById("demo1");
        function getLocation() {
            navigator.geolocation.getCurrentPosition(showLoc, errHand);
        }
        function showLoc(pos) {
            variable1.innerHTML = "Latitude: " + pos.coords.latitude + "<br>Longitude: " +
pos.coords.longitude;
        }
        function errHand(err) {
switch (err.code) {
    case err.PERMISSION_DENIED:
variable1.innerHTML = "The application doesn't have the" +"permission to make use of
location services";
break;
    case err.POSITION_UNAVAILABLE:
variable1.innerHTML = "The location of the device is uncertain";
break;
    case err.TIMEOUT:
variable1.innerHTML = "The request to get user location timed out";
break;
    case err.UNKNOWN_ERROR:
variable1.innerHTML = "Time to fetch location information exceeded" + "the maximum timeout
interval";
break;
        }
    }
    </script>
</body>
</html>

```

Output:

GeeksforGeeks

Error handling in geolocation

Click

Latitude: 28.5011974

Longitude: 77.4099066

Displaying result in MAP

Displaying location on a map is a very interesting task. These services are used to provide the exact location on the map.

Example: This example describes getting the current position in Google Maps with the help of latitude and longitude via HTML Geolocation.

```
<!DOCTYPE html>
<html>
<head>
  <title>Display location in map</title>
  <style>
    .gfg {
      font-size: 40px;
      font-weight: bold;
      color: #009900;
      margin-left: 20px;
    }
    .geeks {
      margin-left: 150px;
    }
    p {
      font-size: 20px;
      margin-left: 20px;
    }
  </style> </head>
<body>
  <div class="gfg">GeeksforGeeks</div>
  <p>Display location in map</p>
  <button class="geeks" type="button" onclick="getlocation();">
    Current Position
```

```

</button>

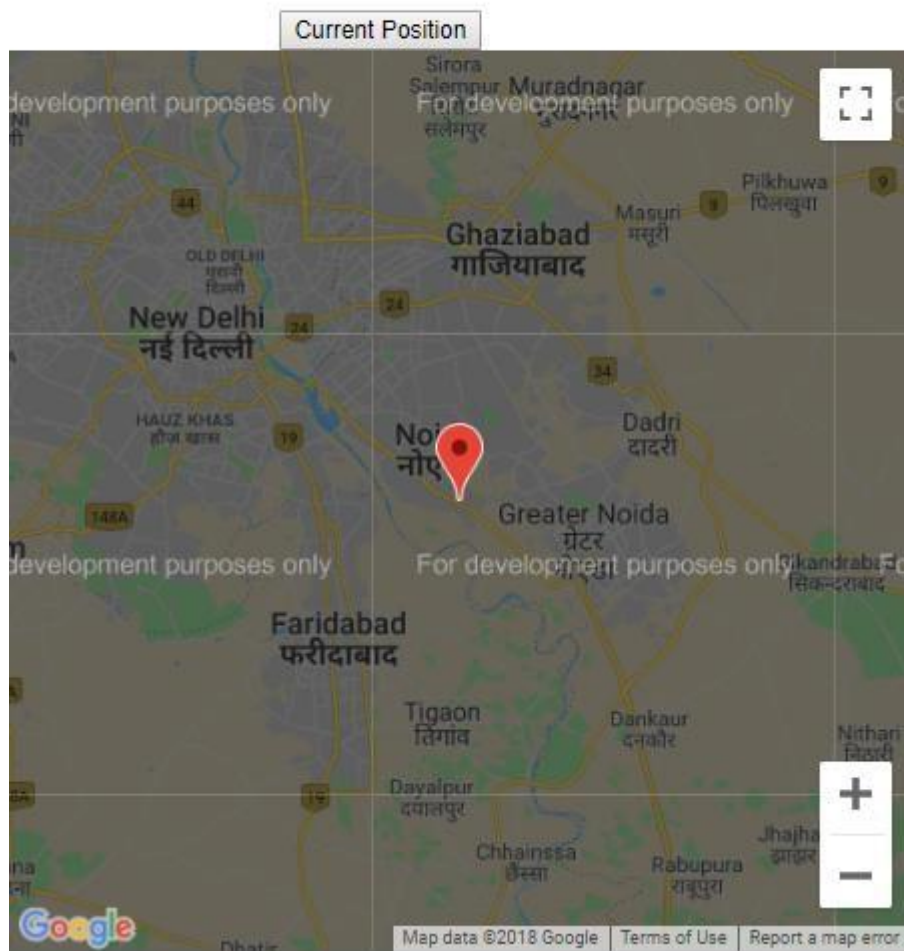
<div id="demo2" style="width: 500px; height: 500px"></div>
<script src="https://maps.google.com/maps/api/js?sensor=false"> </script>
<script type="text/javascript">
function getLocation() {
    if (navigator.geolocation) {
        navigator.geolocation.getCurrentPosition(showLoc, errHand);
    }
}
function showLoc(pos) {
    latt = pos.coords.latitude;
    long = pos.coords.longitude;
    var lattlong = new google.maps.LatLng(latt, long);
    var OPTions = {center: lattlong, zoom: 10, mapTypeControl: true,
navigationControlOptions:
{
    style: google.maps.NavigationControlStyle.SMALL,
    }
};
var mapg = new google.maps.Map(document.getElementById("demo2"), OPTions );
    var markerg = new google.maps.Marker({position: lattlong, map: mapg,
    title: "You are here!", }); }
    function errHand(err) {
switch (err.code) {
case err.PERMISSION_DENIED:
    result.innerHTML = "The application doesn't have the permission" +
        "to make use of location services";
break;
case err.POSITION_UNAVAILABLE:
    result.innerHTML = "The location of the device is uncertain";
    break;
case err.TIMEOUT:
    result.innerHTML = "The request to get user location timed out";
    break;
case err.UNKNOWN_ERROR:
    result.innerHTML = "Time to fetch location information exceeded" +
        "the maximum timeout interval";
    break;
    }
}
</script>
</body>
</html>

```


Output:

GeeksforGeeks

Display location in map



Getting the current position in Google Map

HTML Validation.

How to create form validation by using only HTML ?

In Web Development, we often use JavaScript with HTML to validate the form, but we can also do the same via HTML in the following ways.

- **HTML <input> required Attribute**
- **HTML <input> type Attribute**
- **HTML <input> pattern Attribute**

HTML <input> required Attribute: In input tag of HTML, we can specify via “**required attribute**”. It informs the browser (HTML5 supported) that the field can’t be left blank. Browsers vary in terms of this implementation, some browsers cast a shadow to the box or some show a warning.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Form</title>
</head>
<body>
  <p>This is a form</p>
  <form>
    <p>Name:</p>
    <input type="text" required>
    <p>Email:</p>
    <input type="email" required>
    <p>Address:</p>
    <input type="text" required>
    <br>
    <button style="margin-top: 5px;">
      Submit
    </button>
  </form>
</body>
</html>
```

Output:



The screenshot shows a web form with the following elements:

- A heading: "This is a form"
- A label: "Name:"
- An empty text input field.
- A validation error message box with a yellow warning icon and the text: "Please fill out this field."
- A text input field containing the email address: "asd1@gmail.com"
- A label: "Address:"
- A text input field containing the address: "Delhi"
- A "Submit" button.

HTML `<input>` type Attribute: In input tag, if we require for the user to input their email-id we can set the type attribute to email, the same is applicable to number, date or URL. Similar to the required attribute, different browsers have different implementations.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Form</title>
</head>
<body>
  <p>This is a form</p>
  <form>
    <p>URL:</p>
    <input type="url">
    <p>Email:</p>
    <input type="email">
    <p>Phone Number:</p>
    <input type="number">
    <br>
    <button style="margin-top: 5px;">
      Submit
    </button>
  </form>
</body>
</html>
```

Output:

This is a form

URL:

 Please enter a URL.

Phone Number:

HTML `<input>` pattern Attribute: We already know, apart from using default rules we can also set our rules as for the pattern of URL, date or price, etc.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Form</title>
</head>
<body>
  <p>This is a form</p>
  <form>
    <p>URL:</p>
    <input type="url" pattern="https?:/.+>
    <p>Date1:</p>
    <input type="date"
      pattern="\d{2,1}/\d{2,1}/\d{4}">
    <br>
    <button style="margin-top: 5px;">
      Submit
    </button>
  </form>
</body>
</html>
```

This is a form

URL:

Date1:

