HTML and CSS

Engage and Think

You are part of a web development team working on a new website for a local business. The client wants a well-structured homepage with headings, paragraphs, images, and navigation links. However, the team is debating how to organize the website's files and folders to maintain clarity and efficiency as the project grows. Some team members suggest organizing by type, while others prefer a component-based approach.

What are the advantages and disadvantages of organizing website files by type versus by components, and which approach would you recommend for long-term maintainability?

Learning Objectives

By the end of this lesson, you will be able to:

- Construct web pages using HTML elements to create structured and accessible content
- Implement HTML attributes within elements to enhance content presentation and functionality
- Implement the various attributes, tables, and forms in HTML to enhance the functionality and presentation of web content
- Apply CSS styling to HTML elements to enhance the visual appeal and usability of a web page

Learning Objectives

By the end of this lesson, you will be able to:

- Differentiate between block and inline elements to control content layout effectively
- Utilize CSS selectors to target elements for styling with precision and efficiency
- Integrate plug-ins and multimedia elements into web pages to enrich the user experience

Overview of Hypertext Markup Language (HTML)

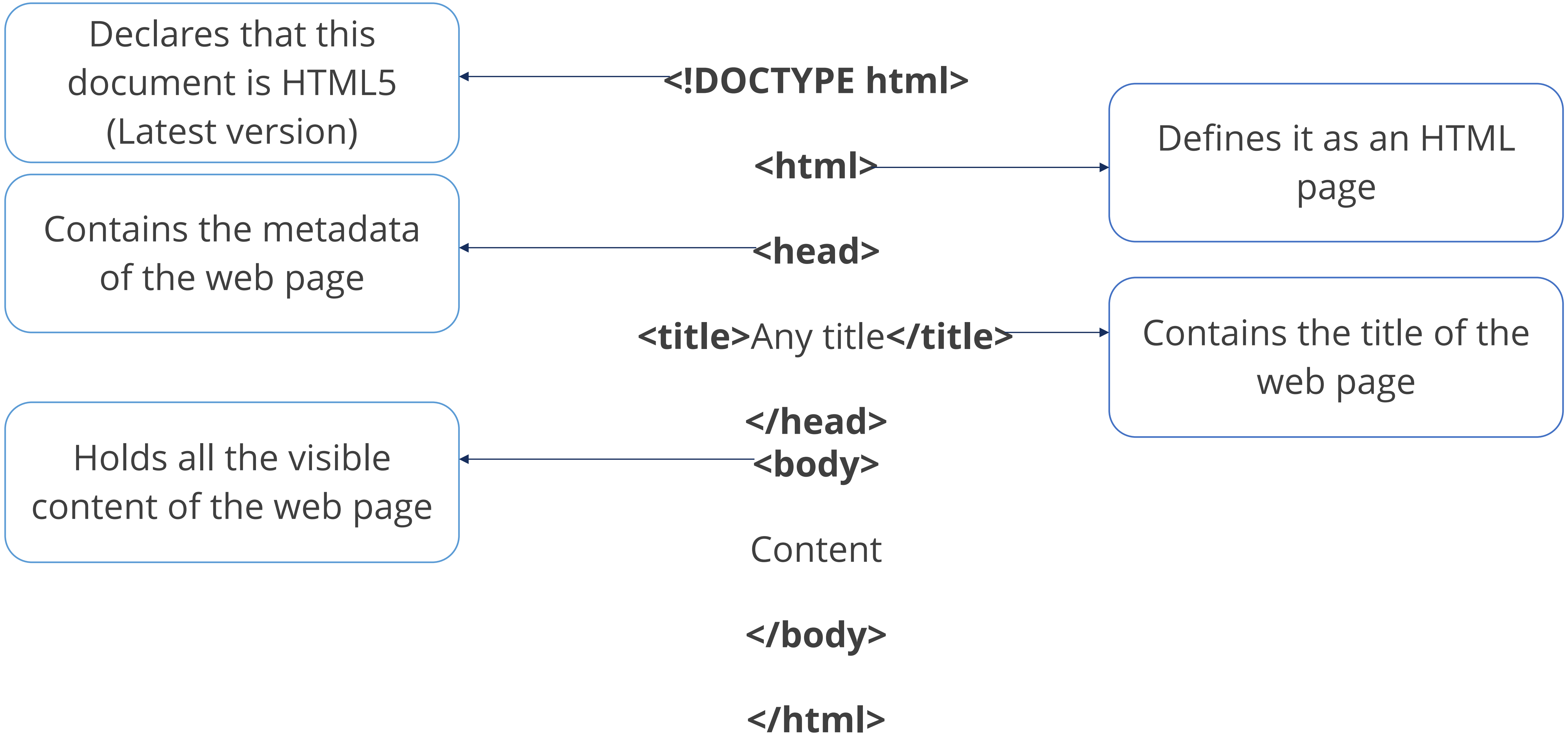
What Is HTML?

It is the standard markup language used for creating and structuring content on the World Wide Web (WWW).

HTML works in conjunction with Cascading Style Sheets (CSS) and JavaScript to create interactive and visually appealing websites.

HTML Structure

The basic structure of an HTML is as follows:



Declares that this document is HTML5 (Latest version)

```
<!DOCTYPE html>
<html>
<head>
<title>Any title</title>
</head>
<body>
Content
</body>
</html>
```

Defines it as an HTML page

Contains the metadata of the web page

Contains the title of the web page

Holds all the visible content of the web page

HTML DOCTYPE

Syntax:

```
<!DOCTYPE html>
<html>
<head>
<title> Home</title>
</head>
<body> The Welcome Page</body>
</html>
```

- In HTML, every document begins with a DOCTYPE declaration.
- This declaration instructs the browser on how to interpret the code and render the web page.
- If the DOCTYPE declaration is missing or incorrect, the browser may struggle to display the page correctly, potentially causing layout and formatting issues.

HTML Title

The **<title>** tag defines the title of the document.

Syntax:

```
<!DOCTYPE html>
<html>
<head>
<title> My first Web page</title>
</head>
<body> The Welcome Page</body>
</html>
```

It provides a clear and concise title that enhances website navigation by helping both users and search engines understand the content of a web page.

HTML Heading

Heading tags <h></h> are used to define the heading.

Syntax:

```
<h1>This is heading 1</h1>

<h2>This is heading 2</h2>

<h3>This is heading 3</h3>

-
-

<h6>This is heading 6</h6>
```

Heading tags <h1> to <h6> are used to define headings, with <h1> being the largest and <h6> the smallest. They are arranged in descending order of text size and importance.

HTML Paragraph

In HTML, paragraphs are written inside the <p></p> tag.

Syntax:

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

The <p> tag defines a paragraph in HTML, automatically adding spacing before and after the text for readability.

HTML <em> Tag

The <em> tag is used to define the emphasized text. The text inside the tag is displayed in italic.

Syntax:

```
<p>Hi,<em>Good Morning</em> how are you</p>
```

HTML <strong> Tag

The <strong> tag is used to give semantic meaning to text.

Syntax:

```
<p> While Driving please <strong> put on your seat belts </strong> </p>
```

It indicates that the enclosed text should be given more importance or emphasis than the surrounding text.

HTML Parent and Child Element

HTML is written in tree format. Head and body are the children of HTML.

Syntax:

```
<html>
<head>
<title>Hi! This is my Web page </title>
</head>
<body> I am a Web page Developer </body>
</html>
```

An element can have many children but has only one parent element. <html> is the root of the tree.

Assisted Practice

Creating the First Web Page

Duration: 10 Min.

Problem statement:

You have been assigned a task to develop a web page using HTML. Your goal is to structure the content using appropriate HTML elements and ensure proper formatting for readability and accessibility.

Outcome:

By the end of this demo, you will be able to create a basic web page using HTML, structure content with appropriate tags, and apply fundamental formatting.

Note: Refer to the demo document for detailed steps:
01_Creating_the_First_Webpage

Assisted Practice: Guidelines

Steps to be followed:

1. Create an HTML file
2. View the result in the local browser

Assisted Practice

Adding a Title, Headings, and Paragraphs to a Web Page

Duration: 10 Min.

Problem statement:

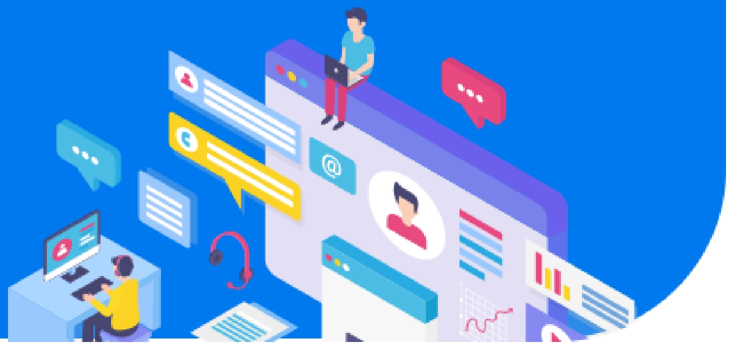
You have been assigned a task to add a title, headings, and paragraphs to a web page using HTML. This will help structure the content effectively and improve readability.

Outcome:

By the end of this demo, you will be able to add a title, headings, and paragraphs to a web page using HTML. You will also structure content for better readability and accessibility.

Note: Refer to the demo document for detailed steps:
02_Adding_a_Title_Headings_and_Paragraphs_to_a_Web_Page

Assisted Practice: Guidelines

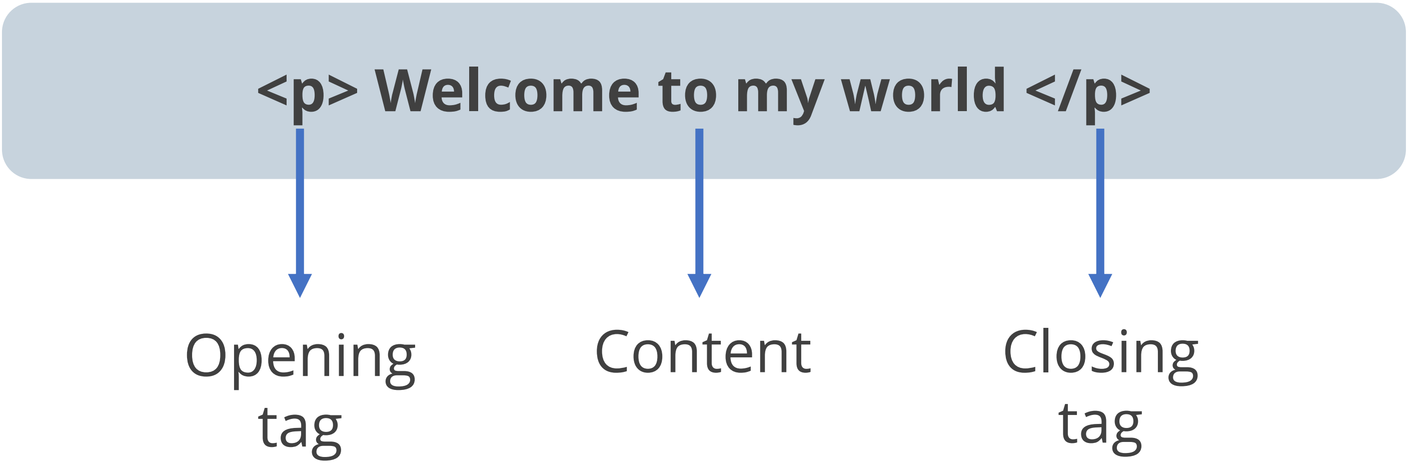


Steps to be followed:

1. Create an HTML file
2. Add a title, headings, and paragraphs to the web page
3. View the result in the local browser

HTML Tags

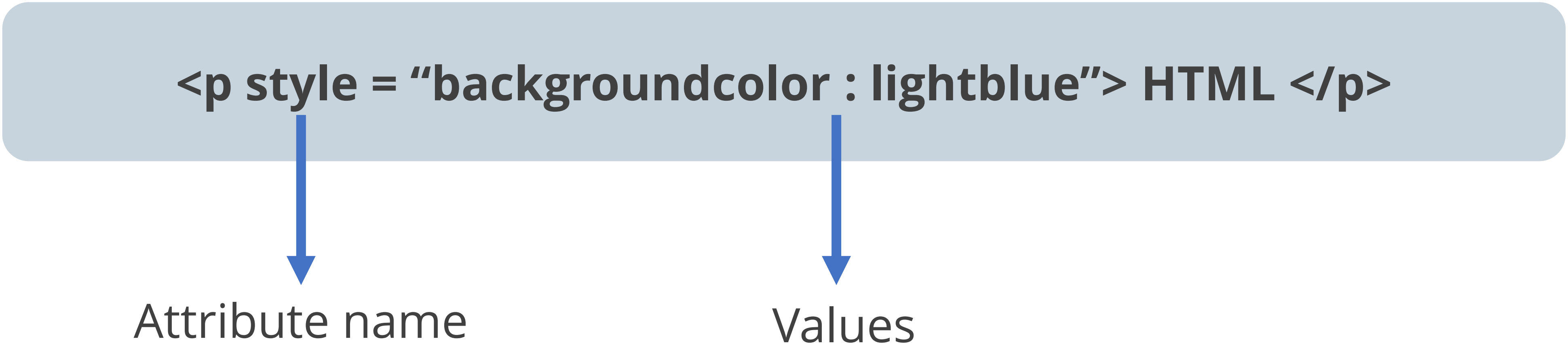
They are similar to keywords, which help differentiate HTML content from other content.



They consist of an opening tag, content, and a closing tag, helping browsers interpret and display information correctly.

HTML Attributes

It offers more details about an HTML element and help in defining its characteristics.

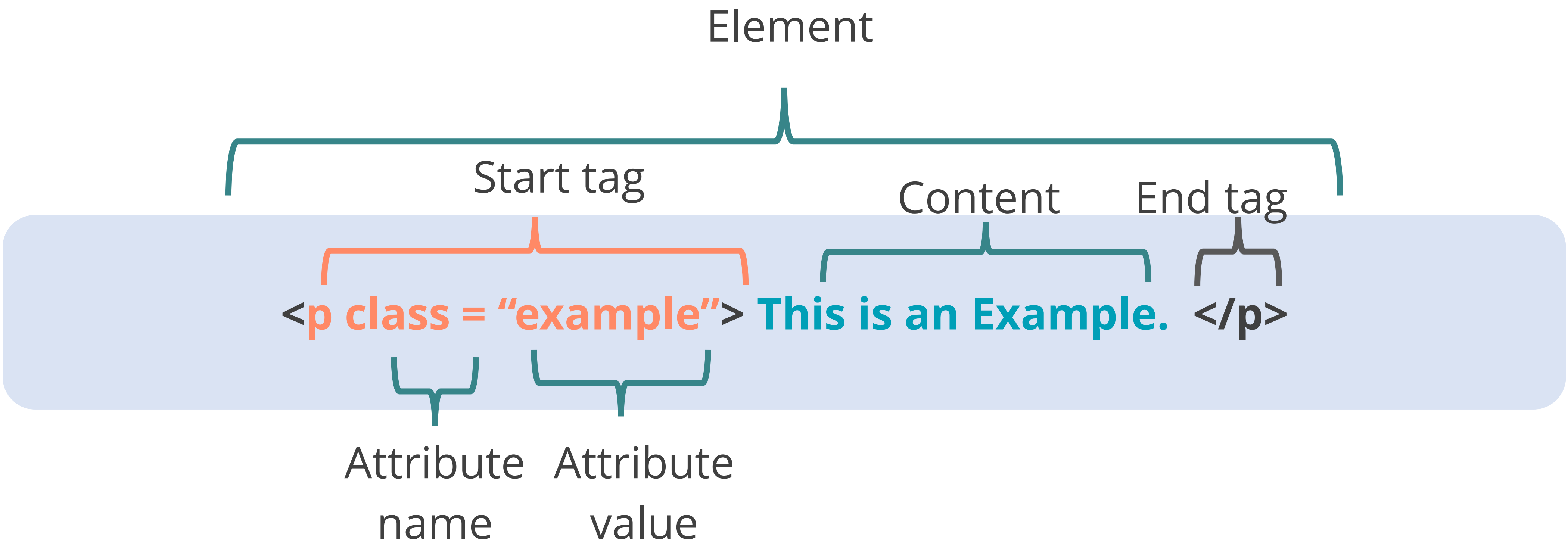


```
<p style = “backgroundcolor : lightblue”> HTML </p>
```

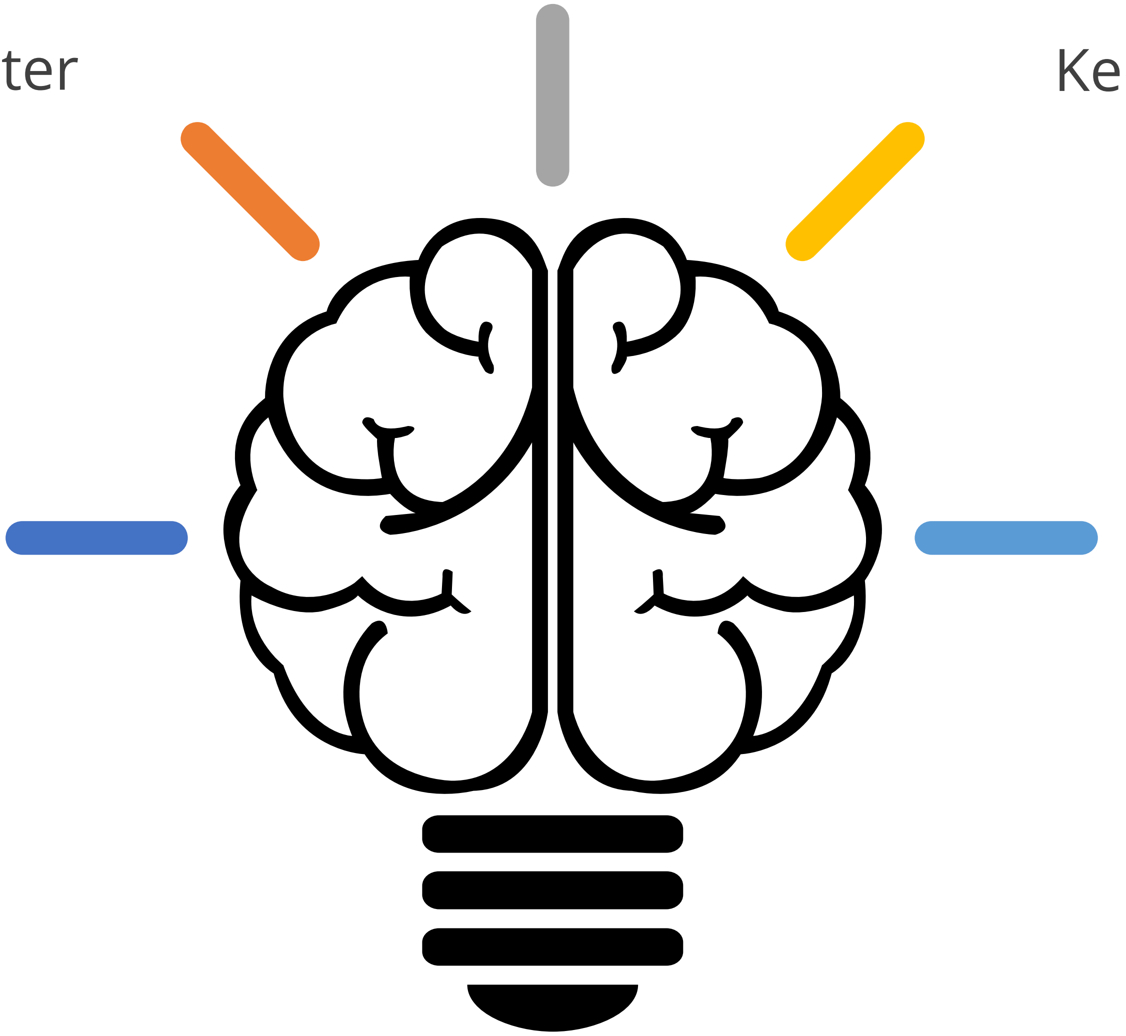
It is the combination of attribute names and values.

HTML Elements

They are the building blocks of an HTML document. They are used to define the structure and content of a web page.



Best Practices for Naming Files and Folders



Use lowercase letters

Start the filenames with a letter

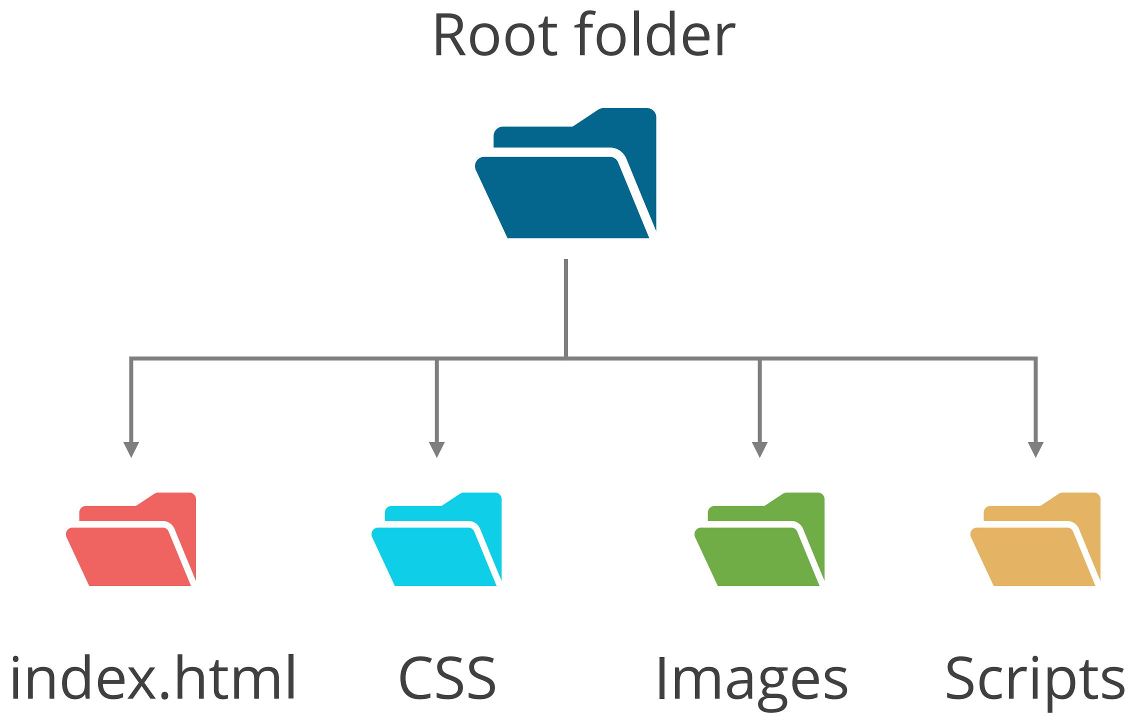
Keep the filenames short

End the filenames with a .html or .htm extension

Avoid spaces and punctuation in file and folder names

Website File and Folder Structure

Users can organize their files by creating separate subfolders. The following is the basic website folder structure:



- **Root folder**: All the additional files and folders are found in this folder.
- **HTML folder**: The files must be saved in the root folder with the **.html** extension and not in the subfolder.
- **CSS**: This folder contains all the Cascading Style Sheets (**.css**) files that define the visual appearance and layout of the website.
- **Images folder**: All the images are in this folder.
- **JavaScript folder**: All the JavaScript files are saved in this folder with the **.js** extension.

Organizing the Website Files and Folders

Users can organize their files and folders using the following methods:

01

Organize by type

02

Organize by component

03

Organize by hybrid approach

Organizing by Type

Group files into separate folders based on their type, such as HTML, CSS, JavaScript, and images.

Example:

```
/js
  /classes
  /libs
  /app.js
/css
  /screen.css
  /print.css
/images
  /icons
  /logo.png
/templates
  /mobile
  /base.html
```

- This approach is particularly beneficial when users anticipate a need for further sub-categorization within each file type.
- It provides a modular and organized foundation that supports developing and maintaining complex web projects.

Organizing by Components

Structure files by components, where each component has its own HTML, CSS, and JavaScript files.

Example:

```
/components
  /navigation
    /navigation.css
    /navigation.html
    /navigation.js
  /section
    /section.css
    /section.html
    /section.js
```

This type of approach is a good choice if the users are organized web developers.

Organizing by Hybrid Approach

Combine both type-based and component-based organizations for better scalability and maintainability.

Example:

```
/base
  /fonts.css
  /analytics.js
/components
  /navigation
    /navigation.css
    /navigation.html
    /navigation.js
  /section
    /section.css
    /section.html
    /section.js
/views
  /products.html
  /product-detail.html
```

- Users can bundle global CSS and JavaScript into a single folder, along with folders for reusable components.
- Users can reuse the bundled folder for new projects and save time while searching for components.

Quick Check

You are developing a new website and need to organize your files efficiently. To keep everything structured, you want to store all your HTML, CSS, images, and JavaScript files in the appropriate locations. Which main directory should contain all these files and subfolders?

A. HTML folder

B. Root folder

C. CSS folder

D. JavaScript folder

Overview of Links, Lists, and Images in HTML

HTML Links

href attribute is used to add links into any HTML elements, followed by the URL inside the anchor tag **<a></a>.**

Example 1:

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

HTML Plug-ins

Plug-ins (Helper applications) extend the functionalities of a web browser. They can be added to web pages with the **<object>** or **<embed>** tags.

Example 1:

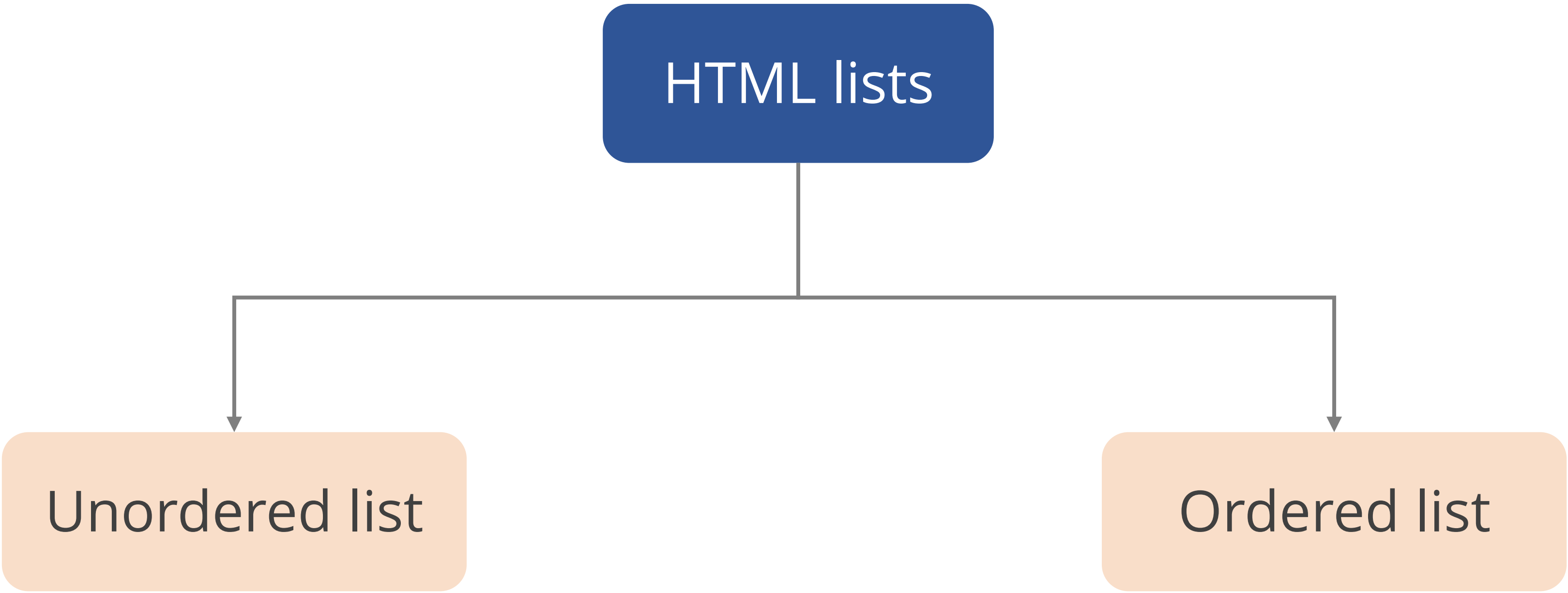
```
<object data=“sample.jpeg"></object>
```

Example 2:

```
<embed src=“sample.jpeg">
```

HTML Lists

HTML lists allow users to group related items in a list.



HTML lists

Unordered list

Ordered list

01 **Unordered list (<ul>):** Displays items with bullet points, making them ideal for lists where order does not matter.

02 **Ordered list (<ol>):** Displays items in a numbered sequence, making them suitable for steps, rankings, or ordered instructions.

Unordered List in HTML

An unordered list begins with **<ul>** tag followed by **<li>** tag for each content of the list.

Example:

```
<ul>
  <li>Apple</li>
  <li>Mango</li>
  <li>Pineapple</li>
</ul>
```

Output:

- Apple
- Mango
- Pineapple

By default, an unordered list starts with a bullet point.

Ordered List in HTML

An ordered list begins with a **<ol>** tag followed by a **<li>** tag for each content of the list.

Example:

```
<ol>
  <li>Apple</li>
  <li>Mango</li>
  <li>Grapes</li>
</ol>
```

Output:

1. Apple
2. Mango
3. Grapes

By default, the ordered list starts with numbers. Ordered lists are suitable when the order of items is meaningful, and you want to display them in a specific sequence.

HTML Images

The **<img>** tag is used to include an image on the web page.

Example:

```
<body>

<img src = "img1.jpg" alt = "HTML Demo" height =
"150" width = "140" />

</body>
```

Assisted Practice

Adding Hyperlinks Lists and Images to a Web Page

Duration: 15 Min.

Problem statement:

You have been assigned a task to create a web page using HTML that incorporates hyperlinks, ordered and unordered lists, and an image, and preview it in a local browser to verify if all elements are correctly structured, formatted, and functioning as intended.

Outcome:

By the end of this demo, you will be able to create a web page using HTML that includes hyperlinks, ordered and unordered lists, and an image while ensuring proper structure, formatting, and functionality when viewed in a local browser.

Note: Refer to the demo document for detailed steps:
03_Adding_Hyperlinks_Lists_and_Images_to_a_Web_Page

Assisted Practice: Guidelines

Steps to be followed:

1. Create an HTML file and add hyperlinks, lists, and an image
2. View the web page in a local browser

Assisted Practice

Adding HTML Plug-Ins

Duration: 10 Min.

Problem statement:

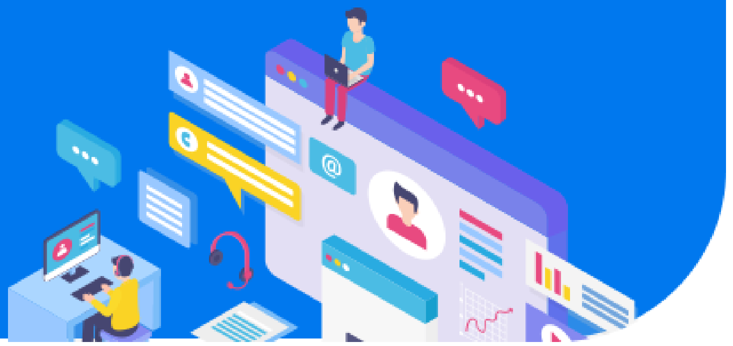
You have been assigned a task to embed plug-ins into a web page using HTML and display them in a local browser to enhance functionality and user experience.

Outcome:

By the end of this demo, you will be able to embed plug-ins into a web page using HTML and display them in a local browser to enhance functionality and user experience.

Note: Refer to the demo document for detailed steps:
04_Adding_HTML_Plug-Ins

Assisted Practice: Guidelines



Steps to be followed:

1. Create an HTML file
2. View the web page in a local browser

Quick Check

You are creating a web page that lists the features of a product. The order of the features does not matter, and you want them to be displayed with bullet points for easy readability. Which type of HTML list should you use?

A. Ordered list (<ol>)

B. Numbered list (<nl>)

C. Unordered list (<ul>)

D. Definition list (<dl>)

Structuring Data with Tables and Forms

Tables in HTML

HTML tables are used to arrange information like images, text, and links into rows and columns.

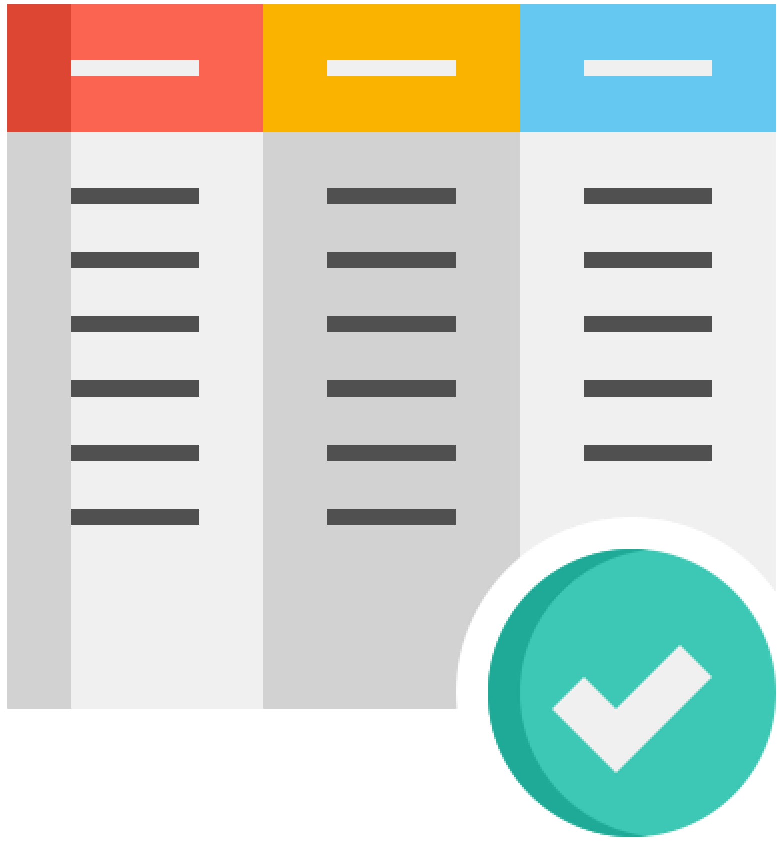


Table Structure

A table consists of cells inside rows and columns.

Example:

```
<table>
  <tr>
    <th></th>
    <th></th>
    <th></th>
  </tr>
  <tr>
    <td></td>
    <td></td>
    <td></td>
  </tr>
</table>
```

- A table is defined by **<table>** and **</table>** tags.
- Each table cell is defined by **<td>** and **</td>** tags.
- Each table row is defined by **<tr>** and **</tr>** tags.
- The table header cells are defined by **<th>** and **</th>** tags.

Table Data

The table data tags **<td>** and **</td>** are used to create data cells in a table.

Example:

```
<table>
  <tr>
    <td>Emily</td>
    <td>Tobias</td>
    <td>Linus</td>
  </tr>
</table>
```

Table Rows

The table row tags **<tr>** and **</tr>** are used to create rows in a table.

Example:

```
<table>
  <tr>
    <td>Emily</td>
    <td>Tobias</td>
    <td>Linus</td>
  </tr>
</table>
```

Table Headers

The table header tags **<th>** and **</th>** are used to create header data cells in a table.

Example:

```
<table>
  <tr>
    <th>Person 1</th>
    <th>Person 2</th>
    <th>Person 3</th>
  </tr>
  <tr>
    <td>Emily</td>
    <td>Tobias</td>
    <td>Linus</td>
  </tr>
</table>
```

Assisted Practice

Adding Tables to a Web Page

Duration: 10 Min.

Problem statement:

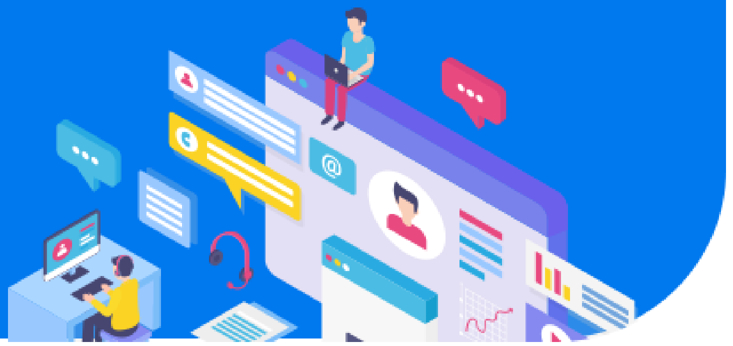
You have been assigned a task to create and display tables using HTML on a web page and view the output in a local browser.

Outcome:

By the end of this demo, you will be able to create and display tables using HTML, structure data using rows and columns, and preview the output in a local browser to ensure proper formatting and alignment.

Note: Refer to the demo document for detailed steps:
05_Adding_Tables_to_a_Web_Page

Assisted Practice: Guidelines



Steps to be followed:

1. Create an HTML file
2. View the web page in a local browser

Form in HTML

An HTML form is used to collect user input. The **<form>** tag is used to create a form.

Example:

```
<form>
.
form elements
.
</form>
```

It contains various types of elements, such as:

- <input>
- <label>
- <select>
- <textarea>
- <button>
- <fieldset>
- <legend>
- <datalist>
- <output>
- <option>
- <optgroup>

Form Method

Method attribute states how the details of a form would be processed.

There are two types of form methods:

- Get
- Post

Form Method: Get

Get is a default method in which the form data gets attached to the URL with name-value pairs.

Example:

```
<form method="get">

  <label>First name:</label>
  <input type="text"><br>
  <label>Last name:</label>
  <input type="text"><br>
  <input type="submit" value="Submit">

</form>
```

Form Method: Post

In the **post** method, the form data is carried in a message body like an HTTP post transaction.

Example:

```
<form method=“post">

  <label>First name:</label>
  <input type="text"><br>
  <label>Last name:</label>
  <input type="text"><br>
  <input type="submit" value="Submit">

</form>
```

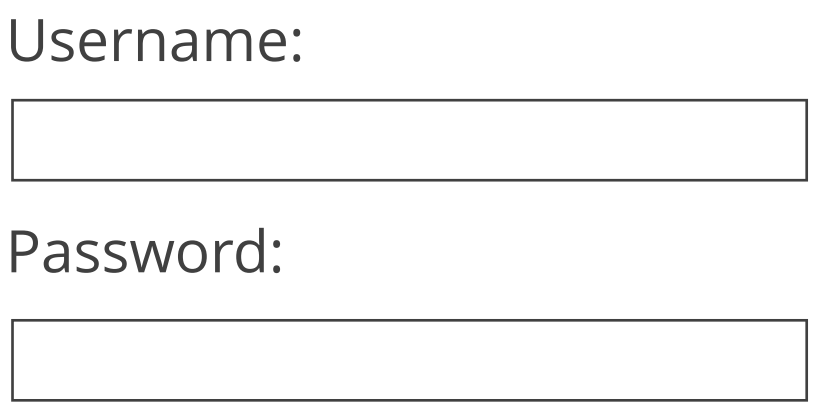
Form in HTML

The following is an example of a simple form that takes username and password as user inputs:

Example:

```
<form>
  <label for="username">Username:</label><br>
  <input type="text" id="username" name="username"><br>
  <label for="pwd">Password:</label><br>
  <input type="password" id="pwd" name="pwd">
</form>
```

Output:

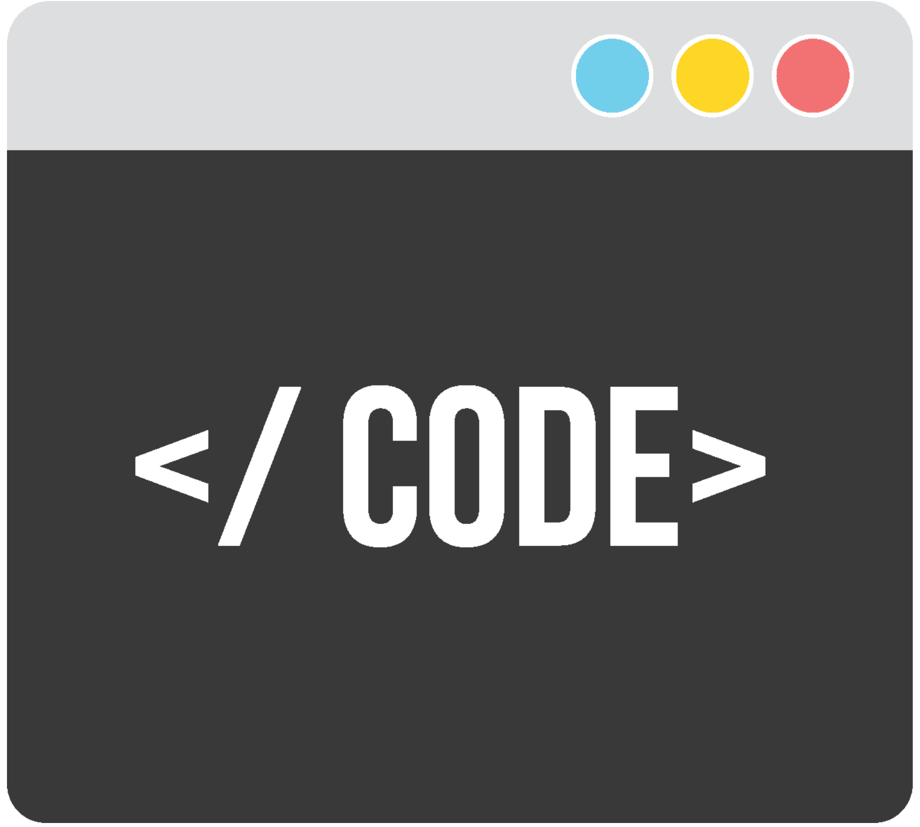


Username:

Password:

HTML Special Characters

In HTML, several characters are reserved and have meaning when used in HTML documents.



HTML Special Characters

A few examples of HTML special characters are as follows:

Symbol	Description	Entity name	Number code
“	Quotation mark	&quot;	&#34;
‘	Apostrophe	&apos;	&#39;
&	Ampersand	&amp;	&#38;
<	Less-than	&lt;	&#60;
>	Greater-than	&gt;	&#62;

Assisted Practice

Creating Forms Using HTML

Duration: 10 Min.

Problem statement:

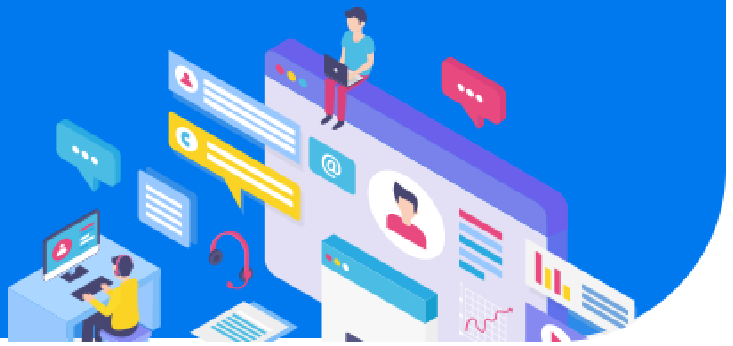
You have been assigned a task to develop a basic web form using HTML and display it in a local browser.

Outcome:

By the end of this demo, you will be able to create a basic web form using HTML, incorporate various form elements such as text fields, buttons, and dropdowns, and display the form in a local browser for validation and testing.

Note: Refer to the demo document for detailed steps: 06_Creating_Forms_Using_HTML

Assisted Practice: Guidelines



Steps to be followed:

1. Create an HTML file
2. View the web page in a local browser

ASSISTED PRACTICE

Quick Check

You are creating a login form for a website where users enter their username and password. Since security is a priority, you need to ensure that the form data is not visible in the URL when submitted. Which form method should you use?

A. POST

B. SEND

C. FETCH

D. GET

HTML Styling and Layout: ID, Class, Header, and Footer

HTML ID

This attribute is used to specify a unique identifier for an HTML element.

The ID attribute points to a specific style declaration in a style sheet.

The syntax for ID is **#id_name** followed by CSS properties within curly braces.

HTML ID

The following is an example of an ID attribute named **#myid** followed by CSS properties to change the color and padding:

Example:

```
<!DOCTYPE html>
<html>
<head>
<style>
#myid {
Backgroundcolor: lightblue;
color: black;
Padding: 40px;
}
</style>
</head>
<body>
<h1 id=“my”> My filename</h>
</body>
</html>
```

HTML Class

The class attribute is used to specify a **class** for an HTML element. The syntax for class is **.class_name**, followed by CSS properties inside curly braces.

- Users can access and manipulate elements with the specific class name, such as **class = ".myclass"**.
- Users can specify one or more class names for an element.
- The same page can have multiple elements with the same class name, but the ID should be unique for each element.

HTML Class

The following is an example of a class attribute named **.my** followed by CSS properties to change the color and padding:

Example:

```
<!DOCTYPE html>
<html>
<head>
 <style>
 .myclass {
    Backgroundcolor: lightblue;
    color: black;
    Padding: 40px;
}
</style>
</head>
  <body>
  <h1 class=“myclass”> My filename</h>
</body>
</html>
```

HTML Span

The **<span>** tag is used as a generic container for inline elements.

Syntax:

```
<span class="">Some Text</span>
```

Example:

```
<!DOCTYPE html>
<html>
<head>
</head>
  <body>
  <h1 class=“my”> My filename</h>
<p> Here is an <span> example of Span
element.</span></p>
</body>
</html>
```

HTML Div

The **<div>** tag is used to group the large section of HTML elements.

Example:

```
<!DOCTYPE html>
<html>
<head>
</head>
<div>
  <body>
  <h1 class=“my”> My filename</h>
<p> Here is an example of Span element. HTML
is used to develop web pages.</p>
</div>
</body>
</html>
```

- The <**div**> tag is styled by using a class or id attribute.
- The <**div**> tag is used to wrap large sections of elements.
- The <**div**> tag is similar to the <**span**> tag, but <**div**> is a block-level element and <**span**> is an inline element.

HTML Header

The **<header>** tag in the HTML element is used to define the header for a web page or a section.

Example:

```
<!DOCTYPE html>
<html>
<head>
</head>
<header>
HEADER TAG
</header>
  <body>
  <h1 class="my"> My filename</h>
<p>HTML is used to develop web pages.</p>
</body>
</html>
```

A <**header**> element can contain:

- Heading elements (<**h1**> to <**h6**>)
- Logo or icon
- Authorship information

HTML Footer

The **<footer>** tag defines the footer of a web page or a section.

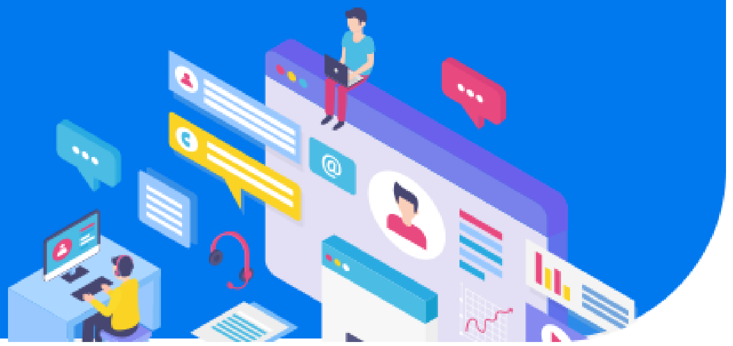
Example:

```
<!DOCTYPE html>
<html>
<head>
<title>Footer</title>
</head>
<body>
  <h1 class=“my”> My filename</h>
<p>HTML is used to develop web pages.</p>
<footer>Contact no:34538568
About us</footer>
</body>
</html>
```

A <**footer**> element can contain:

- Authorship information
- Copyright information
- Contact information
- Sitemap
- Back to top links
- Related documents

Assisted Practice



Using ID, Class, Header, and Footer in HTML

Duration: 10 Min.

Problem statement:

You have been assigned a task to create an HTML file that includes ID and class selectors, along with a header and footer, to structure a web page and enhance its styling.

Outcome:

By the end of this demo, you will be able to create an HTML file that incorporates ID and class selectors to apply to style, use the <header> and <footer> elements to structure a web page and enhance its overall design and organization.

Note: Refer to the demo document for detailed steps:
07_Using_ID_Class_Header_and_Footer_in_HTML

Assisted Practice: Guidelines

Steps to be followed:

1. Create an HTML file and add the ID, Class, Header, and Footer
2. Execute the HTML file

HTML Nav

The **<nav>** tag defines a set of navigation links that help show users and search engines some type of navigation menu (primary, pagination, breadcrumbs, and others)

Example:

```
<nav>
<a href=“/English”>ENGLISH</a> |
<a href=“/Science”>SCIENCE</a> |
<a href=“/hindi”>HINDI</a> |
<a href=“/maths”>MATHEMATICS</a>
</nav>
```

HTML Section

The HTML **<section>** is used to create standalone sections within a web page.

Example:

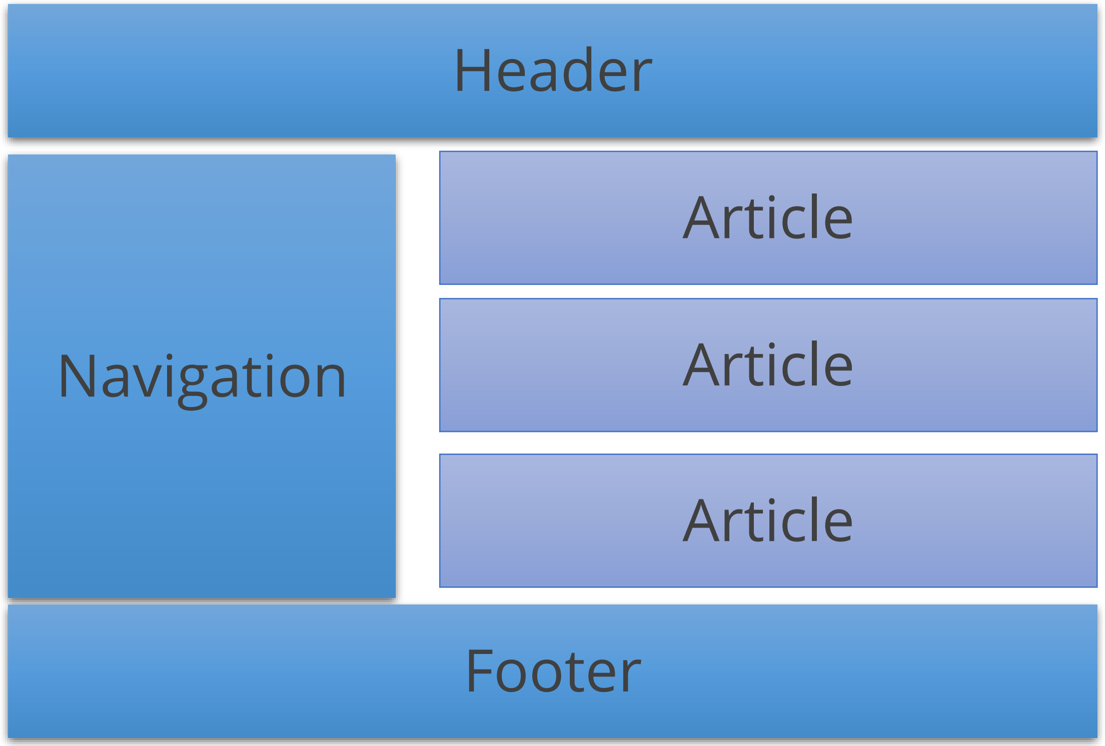
```
<section>
<h3>Welcome to the HTML World</h3>
<p>HTML is the hypertext markup language for
creating the web page.</p>
</section>
```

HTML Article

The **<article>** element represents a standalone composition in a document, page, application, or site that is intended to be distributed or reused independently.

Example:

```
<article class="browser">
    <h2>Google</h2>
    <p>Google Chrome is a web browser developed by
Google and released in 2008. </p>
  </article>
```



Header

Navigation

Article

Article

Article

Footer

HTML Time

The HTML <time> tag is used to define a specific time or date on a web page. It can be used in conjunction with the datetime attribute to specify the exact date and time.

Example:

```
<html>
<head>
<title>Time</title>
</head>
<body>
<h3>Welcome to the HTML World</h3>
<p>We are going to meet up at <time>10:00 am to
12:00pm</time><p>
</body>
</html>
```

HTML Abbreviations

HTML <abbr> tag define the abbreviation of the acronym.

Example:

```
<html>
<head>
<title>Time</title>
</head>
<body><h1>CSS <abbr title = Cascading Styling
Sheet></abbr>CSS is an language used to style the
sheet.</h1></body>
</html>
```

Assisted Practice

Using Nav Section Time and Abbreviations in HTML

Duration: 15 Min.

Problem statement:

You have been assigned a task to create an HTML file that includes navigation links, sections, time formatting, and abbreviations to structure and enhance a web page.

Outcome:

By the end of this demo, you will be able to create an HTML file that incorporates navigation links using the <nav> element, structure content with the <section> tag, format dates and times using the <time> element, and improve readability with the <abbr> tag for abbreviations, enhancing both accessibility and user experience.

Note: Refer to the demo document for detailed steps:
08_Using_Nav_Section_Time_and_Abbreviations_in_HTML

Assisted Practice: Guidelines

Steps to be followed:

1. Create an HTML file and add the required elements
2. Execute the HTML file

Quick Check

You are designing a web page with multiple sections, such as a navigation menu, a sidebar, and a main content area. You want to apply the same styling to multiple elements without repeating CSS code. Which HTML attribute should you use to group these elements under a common style?

A. id

B. style

C. group

D. class

Introduction to CSS

What Is CSS?

It is a stylesheet language used to describe the presentation and formatting of a document written in a markup language such as HTML.

CSS enhances the visual appearance and user experience of web pages and supports responsive design to ensure compatibility across different devices and screen sizes.

Types of CSS

CSS is used to set the style of web pages that contain HTML elements. It sets the background color, font size, font family, and color of a web page.

There are three types of CSS, which are listed below:

- Inline
- Internal
- External

Inline Style

It is used to apply a unique style to a single HTML document.

Example:

```
<body>
    <p style = "color:#2bafc7; font-size:50px;
            font-style:italic; text-align:center;">
        Simplilearn
    </p>

</body>
```

Internal Style

It is used to define a style for a single HTML page.

Example:

```
<head>
<style>
body {background-color:lightyellow;}
h1   {color: blue;}
p    {color: black;}
</style>
</head>
```

The internal CSS is placed inside a <style> element within the <head> section of an HTML page.

External Style

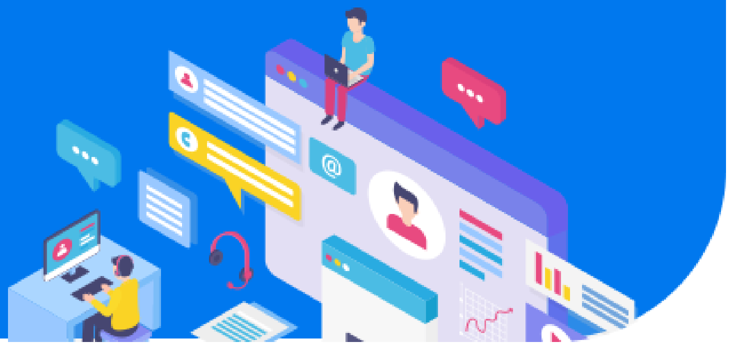
The external style sheet is used to specify the style for many HTML pages or to maintain stylistic information in a separate file from the content for better management.

Example:

```
body {
  background-color: lightblue;
}
h1  {
  color: blue;
}
p {
  color: red;
}
```

- External CSS includes a separate CSS file with only style properties using tag attributes.
- CSS properties are written in separate files with a **.css** extension which should be linked to the HTML content using the link tag.

Assisted Practice



Using Inline Styles in HTML

Duration: 10 Min.

Problem statement:

You have been assigned a task to apply inline styles in an HTML file and customize the appearance of web page elements.

Outcome:

By the end of this demo, you will be able to apply inline styles in an HTML file using the style attribute, modify the appearance of web page elements such as text, colors, and spacing, and understand when to use inline styles for quick styling adjustments.

Note: Refer to the demo document for detailed steps:
09_Using_Inline_Styles_in_HTML

Assisted Practice: Guidelines

Steps to be followed:

1. Create an HTML file and add inline styling code
2. View the expected output

Assisted Practice

Using External Styling in HTML

Duration: 15 Min.

Problem statement:

You have been assigned a task to create an HTML file that uses an external CSS file for styling to separate content from design.

Outcome:

By the end of this demo, you will be able to create an HTML file that links to an external CSS file, apply styles to various HTML elements, and understand the benefits of separating content from design for better maintainability and scalability.

Note: Refer to the demo document for detailed steps:
10_Using_External_Styling_in_HTML

Assisted Practice: Guidelines

Steps to be followed:

1. Create and add external styling code
2. View the expected output in a browser

Quick Check

You are designing a website with multiple pages, and you want to ensure a consistent look across all pages. Instead of adding styles individually to each HTML file, you prefer to maintain all styling rules in a separate file to make future updates easier. Which type of CSS should you use?

A. Inline CSS

B. Internal CSS

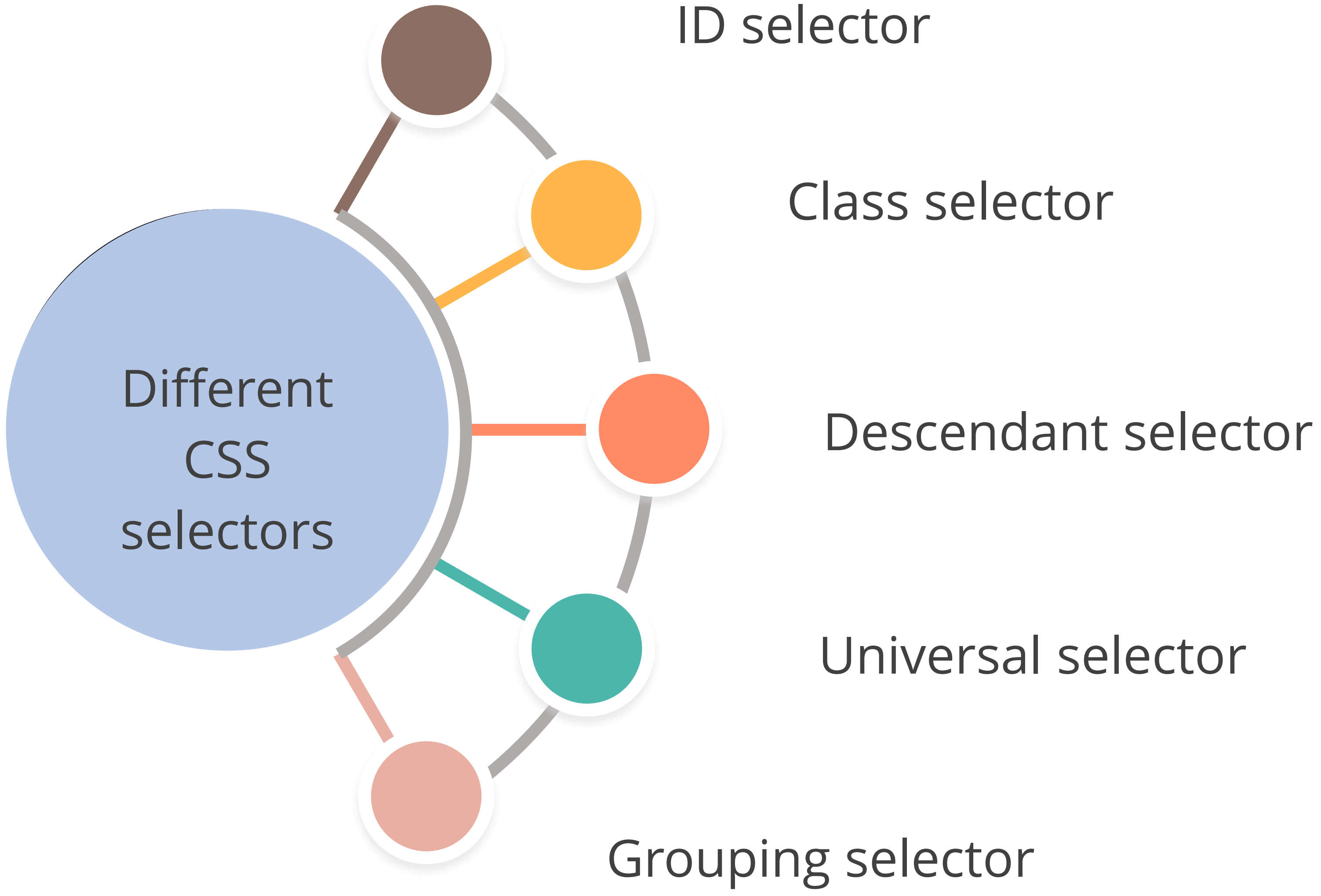
C. External CSS

D. Embedded CSS

Essential CSS Selectors for Styling

CSS Selectors

CSS selectors are used for styling the content, and it is a part of the CSS rule set.



Different CSS selectors

- ID selector
- Class selector
- Descendant selector
- Universal selector
- Grouping selector

ID Selector

Example:

```
<style>
#para1 {
  text-align: center;
  color: red;
}
</style>
```

- The ID selector selects a specific HTML element based on its ID attribute.
- The ID attribute's value is used by the CSS ID selector to match an element.
- The element’s ID attribute must exactly match the value specified in the selector for it to be selected.

Class Selector

Example:

```
.center {
  text-align: center;
  color: red;
}
```

- The class selector is used to select HTML items that have a specified class attribute.
- A period (.) character followed by the class name is used to select the element of the specific class.

Descendant Selectors

The descendant selectors match all elements that are descendants of the specified element.

Example:

```
 div p {
  text-align: center;
  color: red;
}
```

Assisted Practice

Using ID Class and Descendant Selectors in HTML

Duration: 10 Min.

Problem statement:

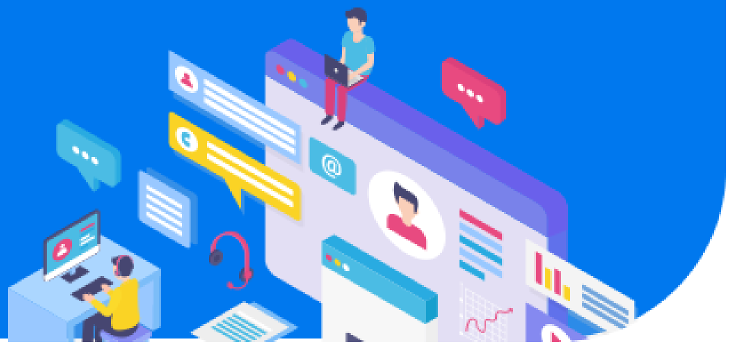
You have been assigned a task to create an HTML file that utilizes ID, class, and descendant selectors in CSS to style web page elements effectively.

Outcome:

By the end of this demo, you will be able to apply ID, class, and descendant selectors in CSS to target and style specific HTML elements, enhance webpage design, and improve styling efficiency using structured CSS rules.

Note: Refer to the demo document for detailed steps:
11_Using_ID_Class_and_Descendant_Selectors_in_HTML

Assisted Practice: Guidelines



Steps to be followed:

1. Create and add code for ID, class, and descendant selectors
2. View the output in a browser

CSS Universal Selector

The universal selector is a unique type of selector.

Example:

```
/* Selects all elements */
* {
  color: green;
}
```

- It can be a namespace when using @namespace.
- This helps with documents containing several namespaces, such as **HTML** with **embedded SVG** or **MathML**, or **XML** that combines different vocabularies.

CSS Grouping Selector

Multiple items can be chosen and styled together using the CSS grouping selector.

Example:

```
<head>
<style>
article, p, img {
   display: block;
   margin: auto;
   text-align: center;
   border-bottom: double orange;
}
</style>
</head>
```

- Declaring common styles for each element saves time and decreases the amount of code that must be written.
- Each selector is separated from the others by a space to form groups.

Assisted Practice

Using Grouping Selectors in HTML

Duration: 10 Min.

Problem statement:

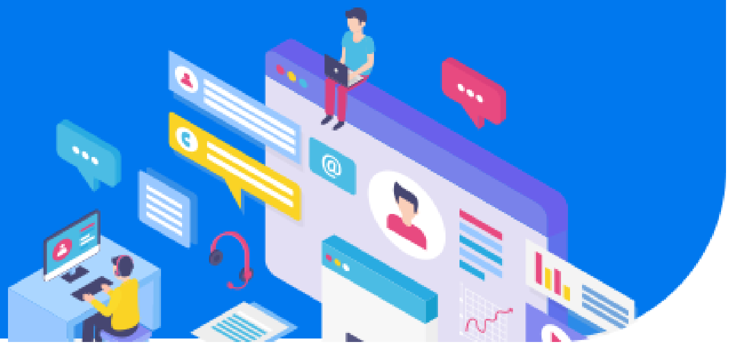
You have been assigned a task to create an HTML file that applies CSS grouping selectors to efficiently style multiple elements at once.

Outcome:

By the end of this demo, you will be able to apply CSS grouping selectors to style multiple elements simultaneously, reduce code redundancy, and improve the efficiency and maintainability of your stylesheets.

Note: Refer to the demo document for detailed steps:
12_Using_Grouping_Selectors_in_HTML

Assisted Practice: Guidelines



Steps to be followed:

1. Create and add grouping selectors in HTML and CSS
2. View the output in a browser

Quick Check

You are designing a web page with multiple elements, such as headings, paragraphs, and images. You want to apply the same styling, like text alignment and margin settings, to all these elements without writing separate CSS rules for each one. Which CSS technique should you use to achieve this efficiently?

A. Universal selector

B. Grouping selector

C. ID selector

D. Class selector

Enhancing Web Design with Text, Color, and Form Styles

CSS Color Property

The color property in CSS is used to add color to the text, web page's background, and borders.

Example:

```
<style>
body {
  color: red;
}

h1 {
  color: #00f099;
}

p.ex {
  color: rgb(0,221,255);
}
</style>
</head>
```

CSS Color

Color in CSS can be added by using the hex, RGB, or HSL value of the required color.

The hexadecimal code is #RRGGBB, where RR (red), GG (green), and BB (blue).

CSS color using HEX value:

```
style="background-color:#3cb371;"
```

The following formula can be used to specify an RGB value for a color:
RGB (red, green, blue)

CSS color using RGB value:

```
style="background-color:rgb(255, 165, 0);"
```

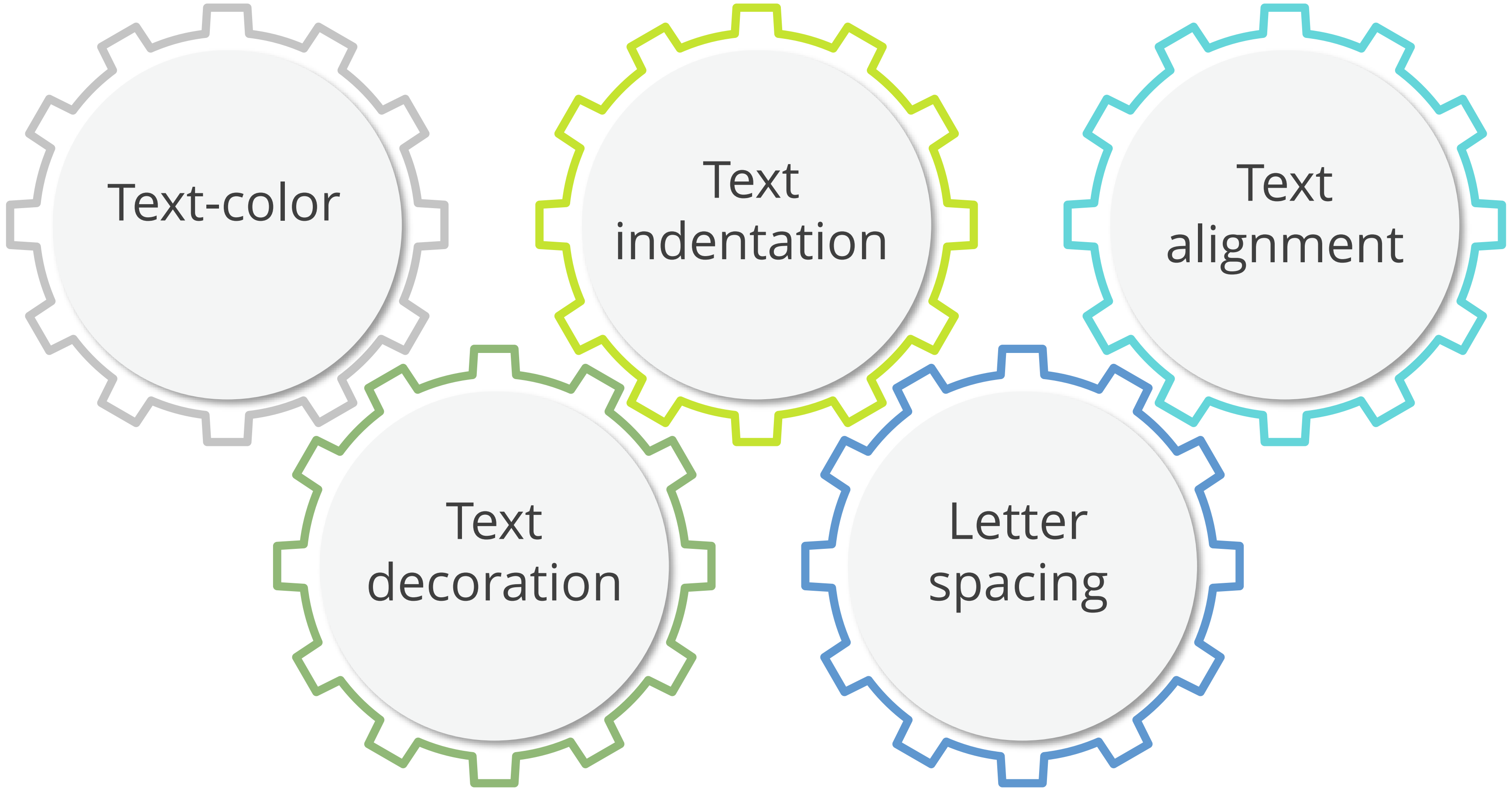
A color can be specified using hue (HSL) using the following formula:
HSL (hue, saturation, lightness)

CSS color using HSL value:

```
style="background-color:hsl(0, 20%, 50%);"
```

Text Styling and Formatting

Text can be styled and formatted using CSS text formatting attributes. The properties are as follows:



Text-color

Text indentation

Text alignment

Text decoration

Letter spacing

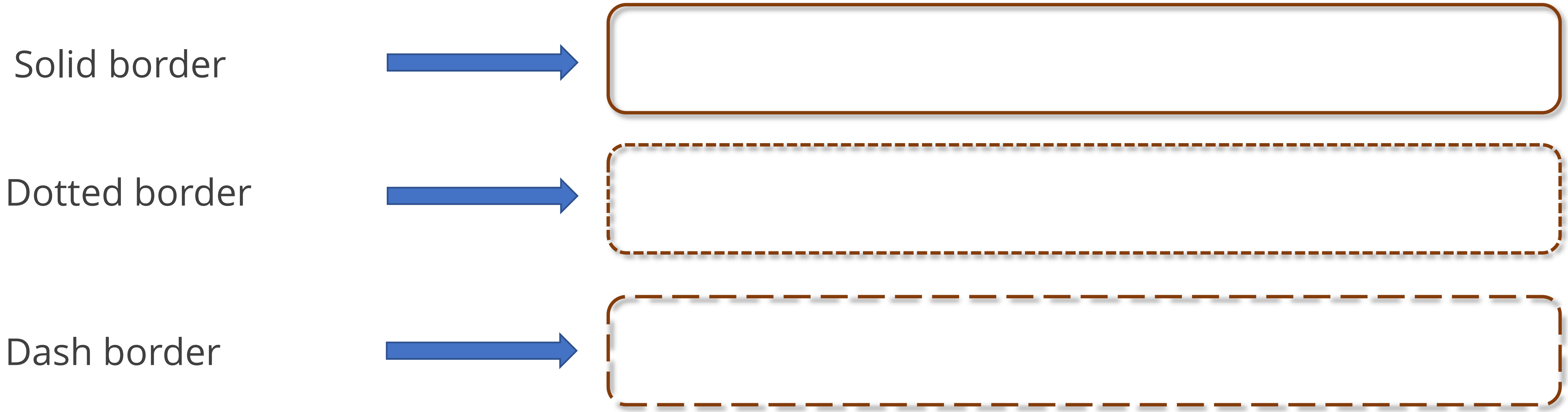
Typography

It is the art of arranging letters and phrases to enhance a website's readability and even navigation.

The two groups of CSS properties that can control typography are **font** and **text**.

CSS Border

The style, width, and color of an element's border may be customized using the CSS border properties. A few examples are as follows:



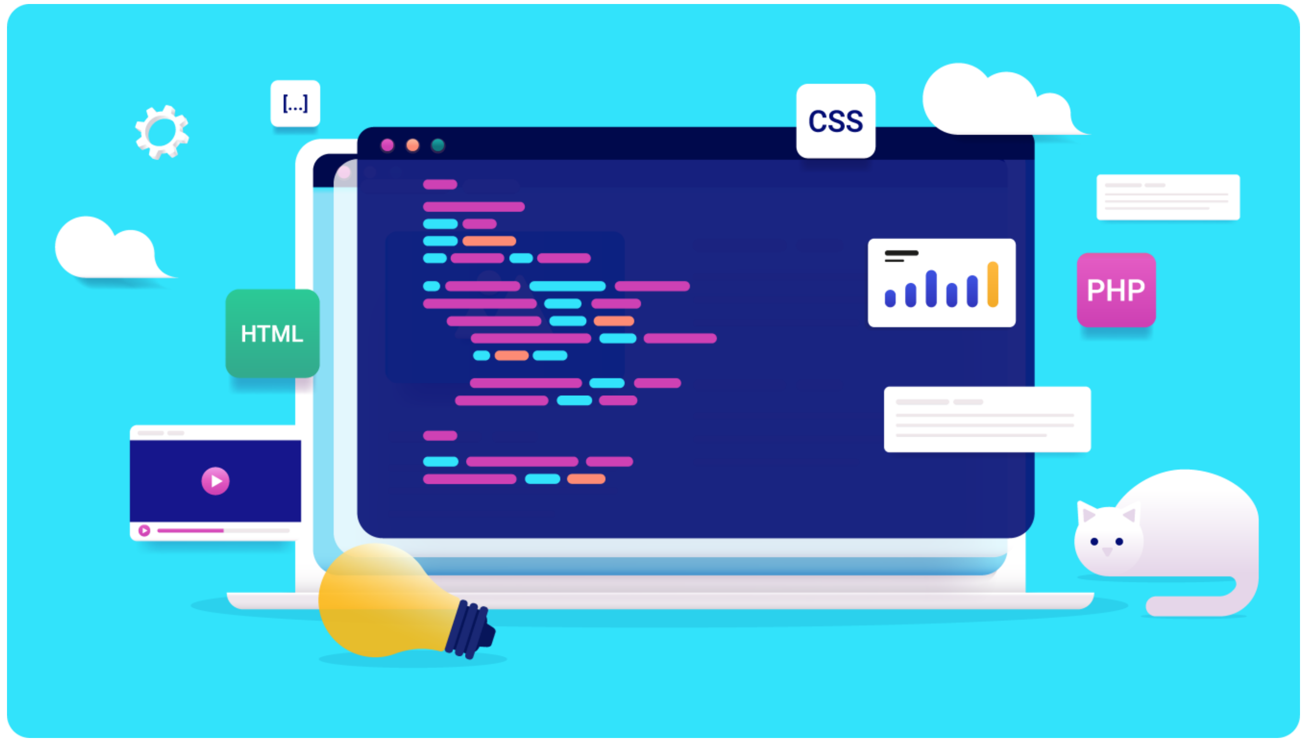
Solid border

Dotted border

Dash border

Background Images

The background picture of an HTML page or table can be specified using the **<background>** attribute in the HTML document.



To add a background image, use the HTML **style** attribute and the CSS **background-image** property

Assisted Practice

Applying Text Styling Borders and Background Images in HTML

Duration: 15 Min.

Problem statement:

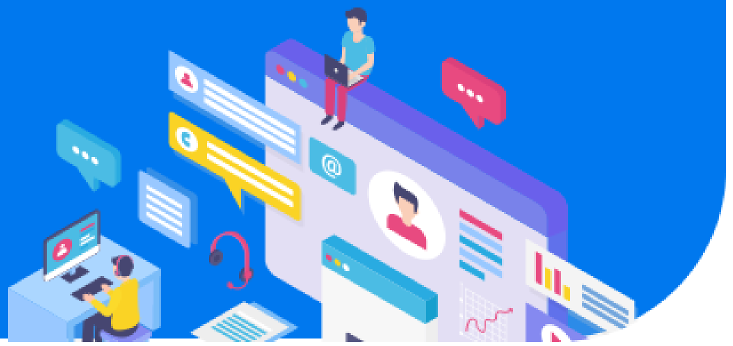
You have been assigned a task to create an HTML file that incorporates text styling, borders, and background images to enhance web page design.

Outcome:

By the end of this demo, you will be able to apply text styling using CSS properties, customize borders for various elements, and incorporate background images to enhance the visual design and readability of a web page.

Note: Refer to the demo document for detailed steps:
13_Applying_Text_Styling_Borders_and_Background_Images_in_HTML

Assisted Practice: Guidelines



Steps to be followed:

1. Create and add code for text styling, borders, and background images
2. View the output in a browser

Assisted Practice

Styling Forms in HTML

Duration: 15 Min.

Problem statement:

You have been assigned a task to create an HTML form and apply styling using CSS to enhance its appearance and readability.

Outcome:

By the end of this demo, you will be able to create an HTML form and apply CSS styling to improve its layout, enhance readability, and optimize user experience by customizing form elements such as input fields, buttons, labels, and spacing.

Note: Refer to the demo document for detailed steps:
14_Styling_Forms_in_HTML

Assisted Practice: Guidelines

Steps to be followed:

1. Create a form and add styling
2. View the output in a browser

Quick Check

You are designing a web page for an online store, and you want to enhance its appearance by adding a background image. To ensure the background loads properly and is easy to update later, you decide to use CSS instead of the HTML <background> attribute. Which CSS property should you use to set the background image?

A. background-image

B. image-src

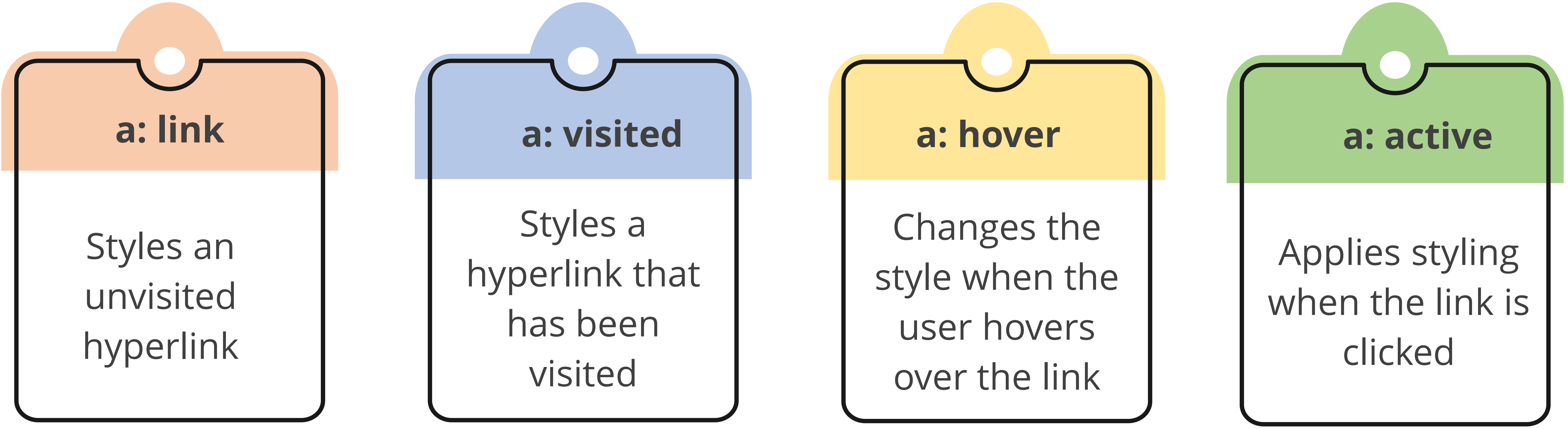
C. background-color

D. img-background

Customizing Link Styling and Element Positioning in CSS

Styling Links

The user can style links using any CSS attribute. There are four types of style links:



a: link

Styles an unvisited hyperlink

a: visited

Styles a hyperlink that has been visited

a: hover

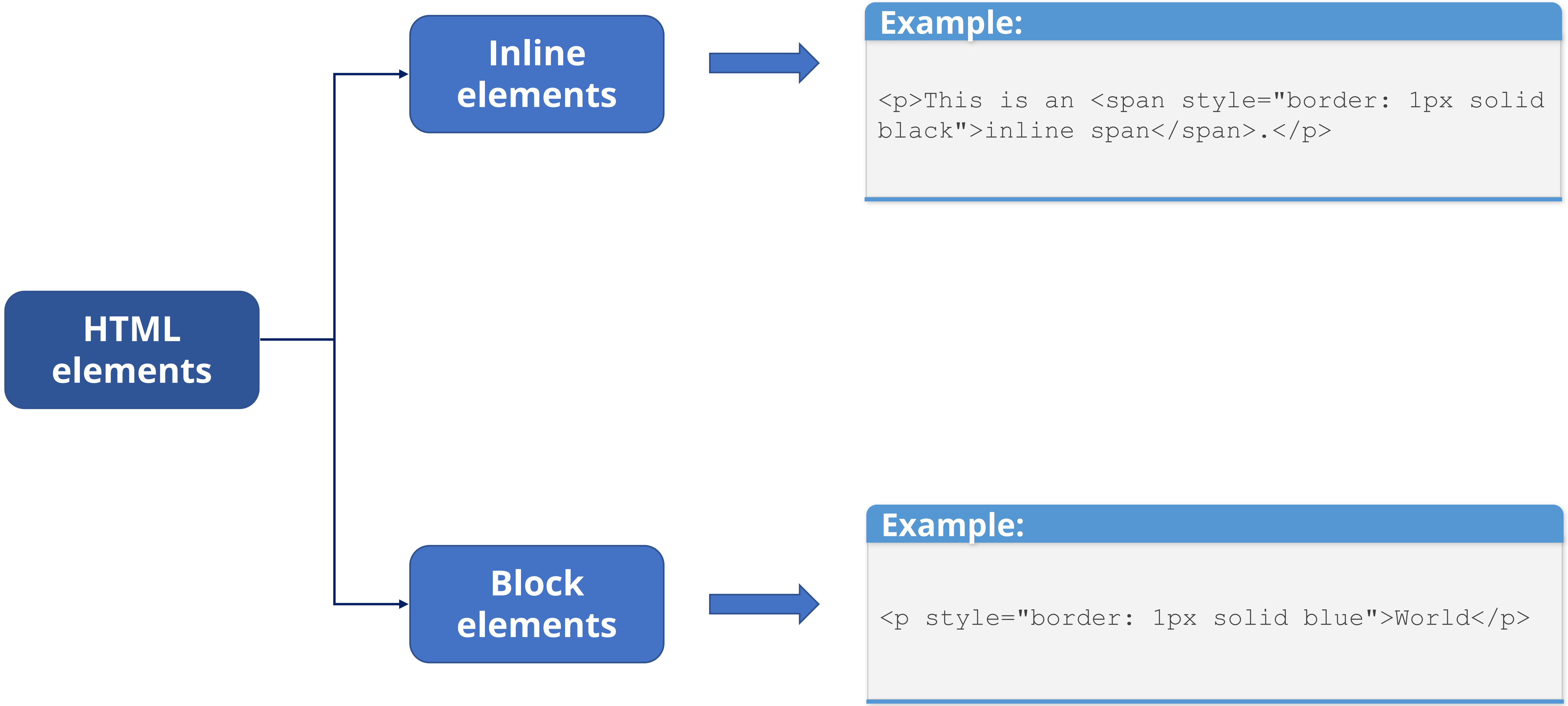
Changes the style when the user hovers over the link

a: active

Applies styling when the link is clicked

HTML Elements

They are the fundamental building blocks of a web page. They consist of a start tag, content, and an end tag, defining different types of content and structure on a web page.



HTML elements

Inline elements

Example:

```
<p>This is an <span style="border: 1px solid
black">inline span</span>.</p>
```

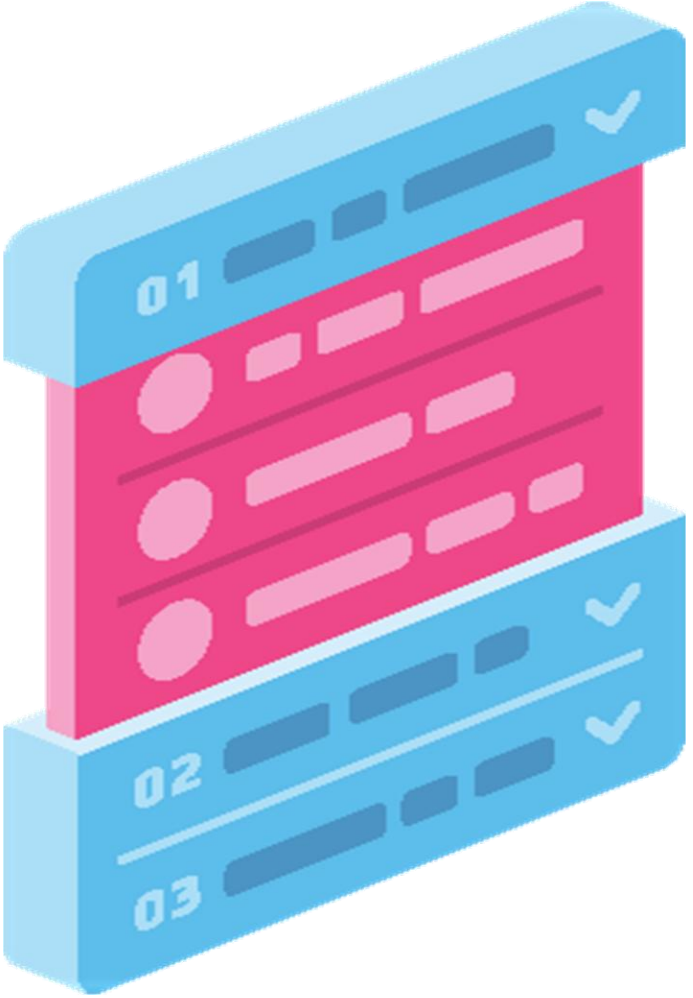
Block elements

Example:

```
<p style="border: 1px solid blue">World</p>
```

Block Element

A block-level element always begins on a new line, and browsers add a margin (a space) before and after the element by default.



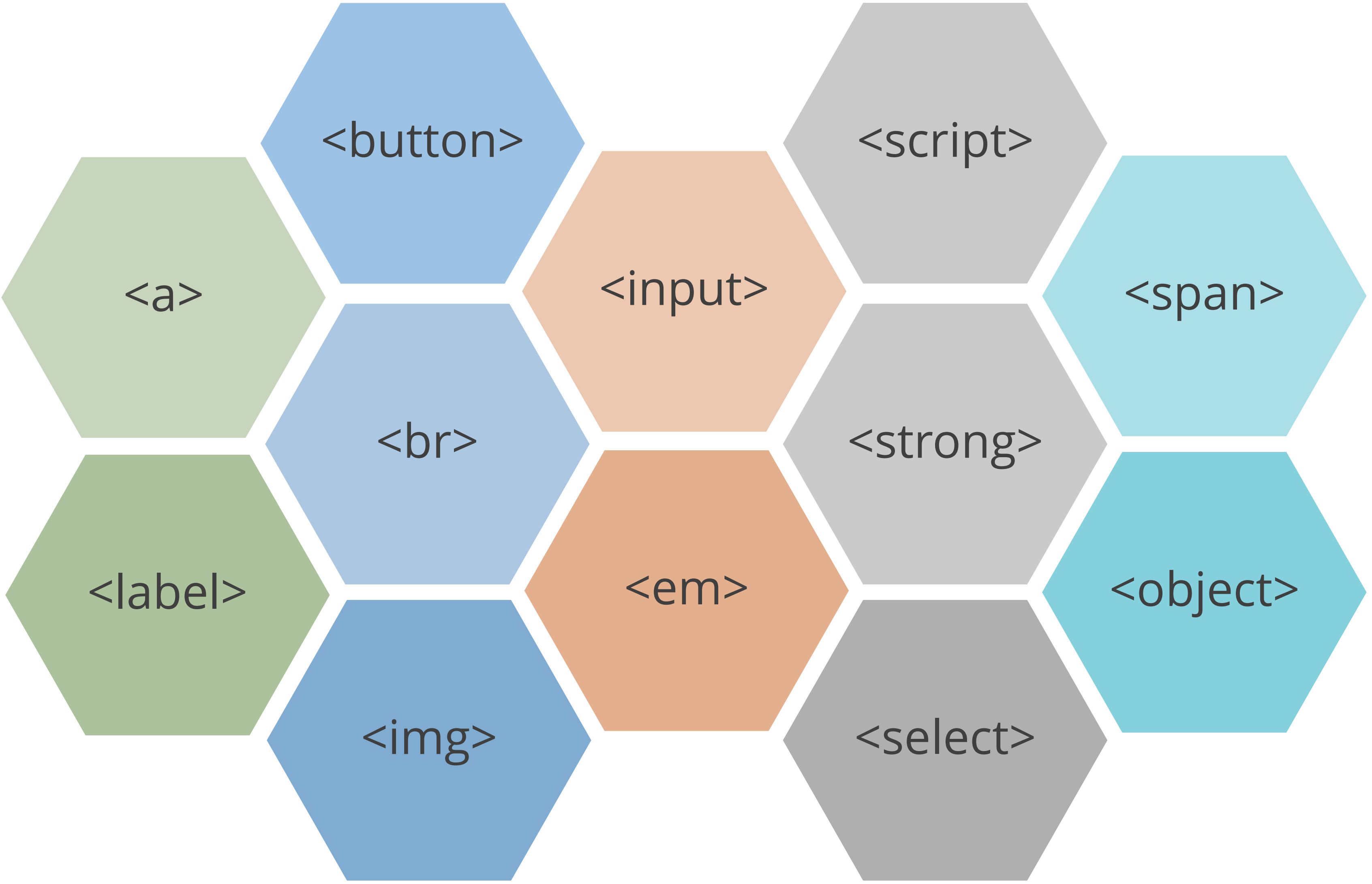
Block Element

The features of block elements are as follows:

New line behavior	Every element at the block level will begin a new line and stack down the page.
Div element usage	The best example is the <div> tag, which groups the components and adds CSS style to the group of elements.
Paragraph tag behavior	Every new paragraph tag will have its own line and display vertically.
Example	An example of a block-level element is the p element.

Inline Element

Inline elements appear in a single line. A few examples of inline elements are as follows:



<a>

<label>

<button>

<br>

<img>

<input>

<em>

<script>

<strong>

<select>

<span>

<object>

CSS Positioning

The position attribute describes the kind of positioning technique that is applied to an element (static, relative, absolute, fixed, or sticky).

Assisted Practice

Implementing Links Blocks and Float in HTML

Duration: 15 Min.

Problem statement:

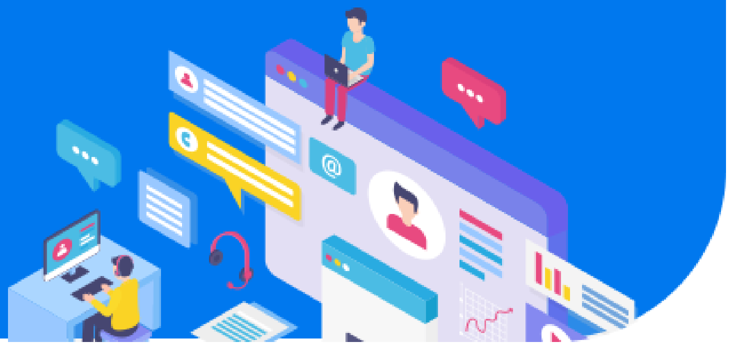
You have been assigned a task to create an HTML file that incorporates links, block elements, and float properties to enhance web page layout and styling.

Outcome:

By the end of this demo, you will be able to create an HTML file that incorporates links for navigation, use block elements to structure content, and apply the float property to control element positioning and improve web page layout and styling.

Note: Refer to the demo document for detailed steps:
15_Implementing_Links_Blocks_and_Float_in_HTML

Assisted Practice: Guidelines



Steps to be followed:

1. Create a form and add styling
2. View the output in a browser

Quick Check

You are designing a web page and want to structure the content so that each section starts on a new line and extends across the full width of the container. You also need to group multiple elements together to apply CSS styles consistently. Which type of HTML element should you use?

A. Inline element

B. Block-level element

C. Self-closing element

D. Inline-block element

Key Takeaways

- HTML is the standard markup language used for creating and structuring content on the World Wide Web (WWW).
- HTML tags are similar to keywords, which help differentiate HTML content from other content.
- Plug-ins (Helper applications) extend the functionalities of a web browser. They can be added to web pages with the <object> or <embed> tags.
- CSS is used to set the style of web pages that contain HTML elements. It sets the background color, font size, font family, and color of a web page.
- Block-level element always begins on a new line, and browsers add a margin (a space) before and after the element by default.

Developing a Web-Based Client Meeting Scheduler

Project Agenda: To design and implement a fully functional web-based client meeting scheduler that enables users to create client profiles and schedule meetings efficiently

Description: You are responsible for designing and implementing a web application that allows users to create client profiles, input relevant details, and schedule meetings seamlessly using a structured form and an interactive calendar-based interface.

LESSON-END PROJECT

Developing a Web-Based Client Meeting Scheduler

Steps to be performed:

1. Create a development folder
2. Develop and implement the web application

Expected deliverables: A fully functional and comprehensive web-based client meeting scheduler with profile creation and booking features. The project will include well-structured HTML and CSS files for maintainability, along with a responsive, well-designed interface featuring a date and time picker. The application will be validated, tested, and built with scalability in mind, allowing for future enhancements such as email notifications and database integration.

Thank You