**Q 1 ARE THE HTML TAGS AND ELEMENTS ARE SAME THINGS?**

ANS := HTML tags and elements are related but distinct concepts:

* **HTML Tags**: Tags are the building blocks of HTML. They are used to mark up the content of a webpage. Tags are written in angle brackets, such as <div> or <a>. Tags typically come in pairs (an opening tag and a closing tag) but some are self-closing (like <img>).
* **HTML Elements**: An element is a complete unit in HTML that includes the opening tag, any content, and the closing tag. For instance, <p>This is a paragraph.</p> is an HTML element. The element consists of the opening tag <p>, the content This is a paragraph., and the closing tag </p>.

In summary, tags are the individual components that define parts of an HTML document, while elements are the complete structures that result from using these tags.Top of FormBottom of Form

**Q 2 WHAT ARE TAGS AND ATTRIBUTES IN HTML?**

ANS := In HTML, tags and attributes are fundamental components used to structure and describe the content of a webpage. Here’s a detailed explanation of each:

HTML Tags

Definition: Tags are used to mark the beginning and end of an HTML element. They are written in angle brackets.

Types:-Opening Tag: Marks the start of an element. For example, `<p>` starts a paragraph element.

Closing Tag: Marks the end of an element and is similar to the opening tag but includes a slash. For example, `</p>` ends the paragraph element.

Self-Closing Tags: Some elements do not have closing tags and are self-contained. For example, `<img src="image.jpg" alt="Description">`.

HTML Attributes

Definition: Attributes provide additional information about an HTML element. They are always included in the opening tag of an element.

Syntax: Attributes are written as name-value pairs within the opening tag. The name and value are separated by an equals sign, and the value is enclosed in quotes. For example, in `<a href="https://example.com">`, `href` is an attribute name and `https://example.com` is its value.

Common Attributes:

- `id`: Provides a unique identifier for the element (e.g., `id="header"`).

- `class`: Specifies one or more class names for styling (e.g., `class="main-content"`).

- `src`: Specifies the source URL for an image (e.g., `src="image.jpg"`).

- `href`: Specifies the URL for a link (e.g., `href="https://example.com"`).

- `alt`: Provides alternative text for images (e.g., `alt="A description of the image"`).

Example: html

<a href="https://example.com" target="\_blank">Visit Example</a>

In this example:

- `<a>` is the tag for a hyperlink.

- `href` and `target` are attributes of the `<a>` tag.

`href="https://example.com"` specifies the URL the link points to.

- `target="\_blank"` specifies that the link should open in a new tab.

Understanding tags and attributes is crucial for creating and styling HTML documents effectively.

**Q 3 What are void elements in HTML?**

**ANS :-** Void elements, also known as self-closing elements, are HTML elements that do not have any content between an opening and closing tag, nor do they require a separate closing tag. These elements are self-contained and perform their function without needing to enclose any inner text or additional tags.

**Characteristics of Void Elements**

Self-contained : They do not have closing tags.

Purpose : They usually represent elements that don’t need to wrap around content or other HTML elements.

Examples of Void Elements

1. `<img>`: Defines an image.

<img src="image.jpg" alt="Description">

2. `<br>`: Inserts a line break.

<p>First line.<br>Second line.</p>

3. `<hr>`: Creates a horizontal rule or line.

4. `<input>`: Defines an input field in forms.

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

5. `<meta>`: Provides metadata about the HTML document.

<meta charset="UTF-8">

6. `<link>`: Links external resources, often used for stylesheets.

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

7. `<source>`: Specifies multiple media resources for elements like `<video>` and `<audio>`.

<video controls>

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

<source src="movie.ogg" type="video/ogg">

</video>

8. `<col>`: Specifies column properties for an HTML table.

<table>

<colgroup>

<col style="background-color:lightblue">

<col style="background-color:lightgreen">

</colgroup>

<tr>

<td>Cell 1</td>

<td>Cell 2</td>

</tr>

</table>

**Syntax in HTML5**

In HTML5, the self-closing syntax with a slash (e.g., `<img src="image.jpg" alt="Description" />`) is optional and not required. The simpler form without the trailing slash (e.g., `<img src="image.jpg" alt="Description">`) is perfectly valid.

Void elements are essential for creating and structuring web content efficiently, providing specific functions without the need for surrounding tags or content.

**Q 4 What are HTML Entities?**

**ANS :-** HTML entities are a way to represent characters in HTML that either cannot be easily typed on a keyboard or have special meanings in HTML. They are especially useful for displaying characters that might otherwise be interpreted as part of the HTML code.

1. Special Characters: Some characters have specific meanings in HTML (e.g., `<`, `>`, and `&`), and using entities allows you to display these characters as text without confusing the browser.

2. Characters Not Easily Typable: Certain characters may not be available on a standard keyboard, such as mathematical symbols or characters from other languages.

3. Consistency and Compatibility: Entities ensure that characters are displayed consistently across different browsers and systems.

**Q 5 What are different types of lists in HTML?**

**ANS** :- In HTML, lists are used to group and present related items in a structured way.

There are three main types of lists:

1. Ordered Lists (`<ol>`)

Purpose: Used for lists where the order of items is important.

Appearance: Items are usually numbered (or lettered, depending on the type attribute).

Structure:

- The `<ol>` tag creates the ordered list.

- The `<li>` tag represents each item in the list.

Example:

```html

<ol>

<li>First item</li>

<li>Second item</li>

<li>Third item</li>

</ol>

Attributes:

- `type`: Specifies the type of numbering (e.g., `1`, `A`, `a`, `I`, `i`).

- `start`: Specifies the start value of the first item.

Example with attributes:

```html

<ol type="A" start="3">

<li>Item A</li>

<li>Item B</li>

</ol>

2. Unordered Lists (`<ul>`)

Purpose: Used for lists where the order of items is not important. Appearance: Items are usually marked with bullets or other symbols.

Structure:

- The `<ul>` tag creates the unordered list.

- The `<li>` tag represents each item in the list.

Example:

```html

<ul>

<li>First item</li>

<li>Second item</li>

<li>Third item</li>

</ul>

Attributes:

- `type`: This attribute is not supported in HTML5. Use CSS for styling.

1. Description Lists (`<dl>`)

Purpose: Use for lists of terms and their descriptions.

Appearanc: Each term is paired with a description.

Structure:

- The `<dl>` tag creates the description list.

- The `<dt>` tag represents a term.

- The `<dd>` tag represents the description for the term.

Example:

Html

<dl>

<dt>Term 1</dt>

<dd>Description for Term 1</dd>

<dt>Term 2</dt>

<dd>Description for Term 2</dd>

</dl>

Example with All List Types

```html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>HTML Lists Example</title>

</head>

<body>

<h2>Ordered List</h2>

<ol>

<li>First item</li>

<li>Second item</li>

<li>Third item</li>

</ol>

<h2>Unordered List</h2>

<ul>

<li>First item</li>

<li>Second item</li>

<li>Third item</li>

</ul>

<h2>Description List</h2>

<dl>

<dt>HTML</dt>

<dd>A markup language used for creating web pages.</dd>

<dt>CSS</dt>

<dd>A stylesheet language used for describing the presentation of a document.</dd>

</dl>

</body>

</html>

Using these list elements helps organize content on a webpage in a clear and structured manner.

**Q 6 What is the ‘class’ attribute in HTML?**

**ANS** :- In HTML, the `class` attribute is used to assign one or more class names to an element. This allows you to apply CSS styles or JavaScript functionality to specific elements on a page. Here’s a basic overview:

Purpose

Styling: By assigning a class to an element, you can use CSS to target that class and apply specific styles. For example:

```html

<style>

.highlight {

background-color: yellow;

}

</style>

<p class="highlight">This text will have a yellow background.</p>

Syntax

```html

<element class="class-name">Content</element>

You can assign multiple classes to an element by separating them with spaces:

```html

<element class="class1 class2 class3">Content</element>

Example

Here’s a simple example combining HTML and CSS:

```html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<style>

.important {

color: red;

font-weight: bold;

}

.note {

color: blue;

font-style: italic;

}

</style>

<title>Class Attribute Example</title>

</head>

<body>

<p class="important">This is an important message.</p>

<p class="note">This is a note.</p>

</body>

</html>

In this example, the `.important` class makes text red and bold, while the `.note` class makes text blue and italic.

**Q 7 What is the difference between the ‘id’ attribute and the ‘class’ attribute of HTML elements?**

ANS :- The `id` and `class` attributes in HTML are both used for identifying and styling elements, but they serve different purposes and have different rules for usage.

`id` Attribute

1. Uniqueness:

The `id` attribute should be unique within a single HTML document. Each `id` value must be used only once per page. This uniqueness allows you to target specific elements precisely.

2. Usage:

CSS: You can use an `id` in CSS to apply styles to a single, unique element.

```html

<style>

header {

color: blue;

}

</style>

<h1 id="header">Welcome to my website</h1>

3. Specificity:

In CSS, `id` selectors have a higher specificity than class selectors. This means styles applied via `id` will override those applied via classes if there's a conflict.

`class` Attribute

1. Reusability:

The `class` attribute is used to apply styles or JavaScript functions to multiple elements. Unlike `id`, a `class` name does not have to be unique, and multiple elements can share the same class.

2. Usage:

CSS: Classes are used to apply styles to multiple elements at once.

```html

<style>

.button {

background-color: green;

color: white;

}

</style>

<button class="button">Click Me</button>

<button class="button">Another Button</button>

D

3. Specificity: In CSS, class selectors have lower specificity compared to `id` selectors. This means that if an `id` selector and a `class` selector apply conflicting styles to the same element, the `id` selector’s styles will take precedence.

**Q 8 What are the various formatting tags in HTML?**

**ANS:-** HTML provides several tags for formatting text to control its appearance and structure. These tags can be used to adjust things like text alignment, boldness, italics, underlining, and more. Here’s a rundown of the main formatting tags:

Text Formatting Tags

1. Bold:

- `<b>`: Represents text that is stylistically different from normal text, without conveying any special importance.

```html

<b>This is bold text.</b>

- `<strong>`: Represents important text. It is typically displayed in bold by default and conveys stronger emphasis.

```html

<strong>This is important text.</strong>

2. Italic:

- `<i>`: Represents text that is displayed in italics, often used for technical terms or foreign words.

```html

<i>This is italic text.</i>

`<em>`: Represents text that has stress emphasis, which is usually displayed in italics and conveys importance or emphasis.

```html

<em>This is emphasized text.</em>

3. Underline:

- `<u>`: Represents text that should be stylistically different from normal text by being underlined. Note that underlining is less common for emphasizing text due to its potential conflict with links, which are traditionally underlined.

```html

<u>This is underlined text.</u>

4. Strikethrough:

- `<s>`: Represents text that is no longer accurate or relevant (strikethrough).

```html

<s>This text is struck through.</s>

`<del>`: Represents text that has been deleted from a document. It also appears with a strikethrough by default.

html

<del>This text has been deleted.</del>

5. Superscript and Subscript:

- `<sup>`: Represents superscript text, which appears higher than the normal text line.

```html

E = mc<sup>2</sup>

- `<sub>`: Represents subscript text, which appears lower than the normal text line.

html

H<sub>2</sub>O

6. Code:

- `<code>`: Represents a fragment of computer code.

```html

<code>console.log('Hello, World!');</code>

- `<samp>`: Represents sample output from a computer program.

```html

<samp>Operation completed successfully.</samp>

7. Preformatted Text:

`<pre>`: Represents preformatted text, where whitespace and line breaks are preserved as they are in the HTML code.

```html

<pre>

This text

is preformatted

and preserves whitespace.

</pre>

8. Quotation:

- `<q>`: Represents a short inline quotation.

```html

He said, <q>This is a quote.</q>

`<blockquote>`: Represents a block-level quotation from another source, usually displayed with indentation.

```html

<blockquote>

This is a block-level quote.

</blockquote>

**Q 9 How is Cell Padding different from Cell Spacing?**

**ANS**:- In HTML tables, “cell padding” and “cell spacing” are two distinct properties that affect the layout and appearance of table cells. Here’s a breakdown of each:

Cell Padding

1. Definition:

Cell padding:-

refers to the space between the content of a table cell and its borders. It controls the internal spacing within each cell, making the content inside the cell more readable and visually appealing.

2. HTML Attribute (Deprecated):

- In older HTML, you could use the `cellpadding` attribute directly in the `<table>` tag. This attribute is now deprecated in favor of CSS.

html

<table cellpadding="10">

<tr>

<td>This cell has padding.</td>

</tr>

</table>

3. Effect:

- Increases the space around the content inside each cell, creating more space between the content and the cell border.

Cell Spacing

1. Definition:

-Cell spacing\*\* refers to the space between the borders of adjacent cells in a table. It controls the distance between separate cells, affecting the overall spacing between them.

2. HTML Attribute (Deprecated):

- In older HTML, you could use the `cellspacing` attribute directly in the `<table>` tag. This attribute is now deprecated in favor of CSS.

```html

<table cellspacing="10">

<tr>

<td>Cell 1</td>

<td>Cell 2</td>

</tr>

</table>

3. Effect:

- Increases the space between the borders of adjacent cells, affecting the overall layout and separation of cells within the table.

**Q 10 How can we club two or more rows or columns into a single row or column in an HTML table?**

**ANS**:- To combine or "club" two or more rows or columns into a single row or column in an HTML table, you use the `rowspan` and `colspan` attributes. These attributes allow you to merge cells across rows or columns, creating larger cells that span multiple areas of the table. Here’s how each attribute works:

Combining Columns with `colspan`

`colspan`: This attribute is used to make a cell span multiple columns. By specifying the number of columns a cell should span, you can merge several columns into a single cell.

Example:

```html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Colspan Example</title>

<style>

table {

border-collapse: collapse;

}

td, th {

border: 1px solid black;

padding: 8px;

}

</style>

</head>

<body>

<table>

<tr>

<th colspan="3">Header spanning 3 columns</th>

</tr>

<tr>

<td colspan="2">Cell spanning 2 columns</td>

<td>Cell 3</td>

</tr>

<tr>

<td>Cell 1</td>

<td>Cell 2</td>

<td>Cell 3</td>

</tr>

</table>

</body>

</html>

In this example:

- The header cell in the first row spans 3 columns.

- The second row contains a cell that spans 2 columns, with another cell next to it.

Combining Rows with `rowspan`

`rowspan`: This attribute is used to make a cell span multiple rows. By specifying the number of rows a cell should span, you can merge several rows into a single cell.

Example:

```html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Rowspan Example</title>

<style>

table {

border-collapse: collapse;

}

td, th {

border: 1px solid black;

padding: 8px;

}

</style>

</head>

<body>

<table>

<tr>

<th rowspan="2">Header spanning 2 rows</th>

<th>Header 1</th>

<th>Header 2</th>

</tr>

<tr>

<td>Cell 1</td>

<td>Cell 2</td>

</tr>

<tr>

<td>Cell 3</td>

<td>Cell 4</td>

<td>Cell 5</td>

</tr>

</table>

</body>

</html>

In this example:

- The header cell in the first column spans 2 rows.

- The remaining cells adjust to fit around the merged cell.

Combining `rowspan` and `colspan`

You can also use both `rowspan` and `colspan` together in a single cell to create more complex table layouts.

`colspan`: Use this attribute to make a cell span multiple columns in the same row.

`rowspan`: Use this attribute to make a cell span multiple rows in the same column.

**Q 11 What is the difference between a block-level element and an inline element?**

**ANS:-** In HTML , block-level elements and inline elements are fundamental to understanding how content is structured and displayed on a webpage. Here’s a detailed comparison:

Block-Level Elements

Definition:

- Block-level elements are elements that occupy the full width of their parent container and start on a new line. They are used to structure the layout and create larger sections of content.

Characteristics:

Width and Height: Block-level elements can have their width and height set explicitly. By default, they stretch to fill the width of their container.

Line Break: Automatically start on a new line, causing a line break before and after the element.

Default Behavior: Stack vertically, one below the other.

Examples: `<div>`, `<p>`, `<h1>`, `<ul>`, `<li>`, `<header>`, `<footer>`, `<section>`, `<article>`

Usage Example:

```html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Block-Level Example</title>

<style>

.block {

background-color: lightblue;

padding: 10px;

margin-bottom: 10px;

}

</style>

</head>

<body>

<div class="block">This is a block-level element.</div>

<div class="block">This is another block-level element.</div>

</body>

</html>

In this example, each `<div>` starts on a new line and fills the width of its container.

**Inline Elements**

Definition:

- Inline elements do not start on a new line and only take up as much width as necessary for their content. They flow within the text and do not disrupt the layout of surrounding content.

Characteristics:

Width and Height: Cannot be directly set; their width is determined by the content they enclose.

Line Break: Do not create line breaks before or after the element; they flow along with the text.

Default Behavior: Stack horizontally within their container.

Examples: `<span>`, `<a>`, `<strong>`, `<em>`, `<img>`, `<br>`, `<code>`

Usage Example:

```html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Inline Example</title>

<style>

.inline {

background-color: lightgreen;

padding: 5px;

}

</style>

</head>

<body>

<p>This is <span class="inline">an inline element</span> within a paragraph.</p>

<p>Another <a href="#">inline link</a> in the text.</p>

</body>

</html>

In this example, the `<span>` and `<a>` elements do not disrupt the flow of the text and only occupy the space needed for their content.

**Q 12 How to create a Hyperlink in HTML?**

**ANS**:- Creating hyperlinks in HTML is done using the `<a>` (anchor) element. This element allows you to link to another webpage, a specific section within a page, a file, an email address, or other resources.

**Q 13 WHAT ARE THE USE OF AN IFRAM TAG?**

**ANS :-** The `<iframe>` tag in HTML is used to embed another HTML document within the current page. This creates a nested browsing context that allows you to display content from another source or website directly within your webpage. Here are some key uses and features of the `<iframe>` tag:

Uses of `<iframe>`

1. Embedding External Content: You can use an `<iframe>` to display content from external sources, such as other web pages, videos, or interactive maps.

- Example: Embedding a YouTube video:

```html

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

2. Including Interactive Content: `<iframe>` can be used to embed interactive elements like forms or applications from other websites.

- Example: Embedding a Google Maps location:

```html

<iframe src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d3153.6795805691794!2d-122.40182348468141!3d37.79353897975542!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x8085809bde0bd99d%3A0x98b3a7f4b758b7a!2s123%20Main%20St%2C%20San%20Francisco%2C%20CA%2094111!5e0!3m2!1sen!2sus!4v1631796640500!5m2!1sen!2sus" width="600" height="450" style="border:0;" allowfullscreen="" loading="lazy"></iframe>

3. Displaying Documents: You can use an `<iframe>` to embed documents such as PDFs or other web-based resources.

- Example: Embedding a PDF document:

```html

<iframe src="document.pdf" width="600" height="400"></iframe>

4. Creating Widgets: Many websites offer widgets that can be embedded via `<iframe>`, such as weather widgets or social media feeds.

Key Attributes of `<iframe>`

- `src`: Specifies the URL of the document to be embedded.

- `width` and `height`: Define the size of the iframe.

- `frameborder`: Specifies whether to display a border around the iframe (deprecated in HTML5; use CSS instead).

- `allowfullscreen`: Allows the iframe content to be displayed in fullscreen mode.

- `loading`: Controls the lazy-loading of the iframe content (e.g., `loading="lazy"`).

`name`: Provides a name for the iframe that can be targeted by links or forms.

- `sandbox`: Applies extra restrictions on the content in the iframe for security purposes.

Example with attributes:

```html

<iframe src="https://example.com" width="600" height="400" frameborder="0" allowfullscreen loading="lazy"></iframe>

Considerations

Security: Be cautious when embedding content from unknown or untrusted sources, as it can pose security risks. Use the `sandbox` attribute to apply security restrictions.

Performance: Overusing `<iframe>`s or embedding heavy content can impact page performance.

Cross-Origin Restrictions: Some content may have restrictions or may not function properly due to cross-origin policies.

The `<iframe>` tag is a versatile tool for embedding external content and interactive elements into a webpage.

**Q 14 What is the use of an iframe tag?**

**ANS:-** The `<iframe>` tag in HTML is used to embed another HTML document within the current page. It creates a nested browsing context that allows you to display content from a different source or location directly within your webpage. Here are the primary uses and benefits of the `<iframe>` tag:

Primary Uses of `<iframe>`

1. Embedding External Content:

- You can use `<iframe>` to include content from another website or service without navigating away from your page. For example, you might embed a YouTube video, a Google Map, or an external web application.

```html

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

2. Displaying Interactive Applications:

- If you have an external application or interactive tool (such as a calculator or a survey form) that you want to incorporate into your site, `<iframe>` allows you to embed it.

```html

<iframe src="https://example.com/interactive-tool" width="600" height="400"></iframe>

```

3. Embedding Documents:

You can embed other types of documents, such as PDFs or spreadsheets, using `<iframe>`, allowing users to view these documents directly within the page.

```html

<iframe src="path/to/document.pdf" width="600" height="500"></iframe>

4. Integrating Widgets:

- Widgets like social media feeds, chat widgets, or ad banners can be embedded in your page using `<iframe>`. These are often provided by third-party services.

```html

<iframe src="https://widgetprovider.com/widget" width="300" height="250"></iframe>

Attributes of `<iframe>`

`src`: Specifies the URL of the document to embed. This can be an external webpage, a local document, or any HTML file.

```html

<iframe src="https://www.example.com"></iframe>

`width` and `height`-: Define the size of the iframe. Values can be in pixels or percentages.

```html

<iframe src="https://www.example.com" width="800" height="600"></iframe>

```frameborder`: (Deprecated) Controls the border around the iframe. Use CSS to style borders instead.

`html

<iframe src="https://www.example.com" frameborder="0"></iframe>

`allowfullscreen`: Allows the iframe content to be viewed in full-screen mode. Useful for embedded videos.

```html

<iframe src="https://www.youtube.com/embed/example" allowfullscreen></iframe>

`loading`: (HTML5) Specifies how the browser should load the iframe. Values are `eager` (default) or `lazy`.

```html

<iframe src="https://www.example.com" loading="lazy"></iframe>

`name`: Assigns a name to the iframe, which can be used for targeting it with links or scripts.

```html

<iframe src="https://www.example.com" name="myiframe"></iframe>

`sandbox`: Provides security restrictions for the iframe content, such as disallowing scripts or forms.

```html

<iframe src="https://www.example.com" sandbox="allow-scripts"></iframe>

```

Security Considerations

Using `<iframe>` comes with some security implications:

Clickjacking : Protect against clickjacking attacks by using the `X-Frame-Options` header on the embedded content or applying the `sandbox` attribute.

Cross-Site Scripting (XSS): Ensure that the content embedded in the iframe is from a trusted source to prevent XSS attacks.

Example

Here’s a simple example of embedding a Google Map into a webpage:

```html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Embedding Google Map</title>

</head>

<body>

<h1>Our Location</h1>

<iframe

src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d3153.900201833436!2d-122.4194156846813!3d37.77492927975981!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x80858177d67b7e1%3A0x5b0f622fe2b515e5!2sSan+Francisco%2C+CA!5e0!3m2!1sen!2sus!4v1634727456760!5m2!1sen!2sus"

width="600"

height="450"

style="border:0;"

allowfullscreen=""

loading="lazy">

</iframe>

</body>

</html>

In this example, the `<iframe>` is used to embed a Google Map that displays a specific location.

Using `<iframe>` effectively can enrich your web pages by incorporating diverse content while maintaining user engagement.

**Q 15 WHAT IS THE USE OF SPAN TAG? EXPLAIN WITH EXAMPLES?**

ANS :- The `<span>` tag in HTML is an inline container used to apply styles or scripts to a portion of text or other inline elements within a block-level element, such as a paragraph or a div. It does not inherently alter the layout or presentation of the content but provides a way to target and style specific sections of content.

Uses of the `<span>` Tag

1. Applying Styles: The `<span>` tag is often used to apply CSS styles to a specific part of a text. This is useful when you want to change the appearance of a small section of text without affecting the rest of the content.

Examples of Using `<span>`

1. Applying CSS Styles

You can use the `<span>` tag to apply different styles to specific parts of your text. This is useful for highlighting, changing colors, or applying other CSS properties to only a portion of the text.

Example: Highlighting Text

```html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Span Example</title>

<style>

.highlight {

background-color: yellow;

font-weight: bold;

}

</style>

</head>

<body>

<p>Here is a sentence with a <span class="highlight">highlighted section</span> to show the use of the span tag.</p>

</body>

</html>

In this example, the text "highlighted section" is highlighted with a yellow background and bold font.

2. Adding IDs and Classes for Targeting

The `<span>` tag can be used to assign specific IDs or classes to parts of the text for styling or scripting.

Example: Styling with Classes

```html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Span Example</title>

<style>

.important-text {

color: red;

font-size: 1.2em;

}

</style>

</head>

<body>

<p>Make sure to read the <span class="important-text">important information</span> before proceeding.</p>

</body>

</html>

In this example, the text "important information" is styled to be red and slightly larger.

**Q 16 HOW TO INSERT A PICTURE INTO A BACKGROUND IMAGE OF A WEB PAGE?**

ANS :- To insert a picture as a background image for a web page, you typically use CSS. The CSS `background-image` property is used to set an image as the background of an HTML element. Here’s a step-by-step guide on how to do this:

1. Using CSS to Set a Background Image

You can set a background image for the entire web page or for specific elements like a `div`, `header`, or `section`. Here’s how to do it:

For the Entire Web Page

To set a background image for the entire page, you would typically apply the background image to the `body` element. You can do this either inline within a `<style>` tag or in an external stylesheet.

Example using inline CSS:

`html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Background Image Example</title>

<style>

body {

background-image: url('background.jpg');

background-size: cover; / Adjusts the size of the background image /

background-position: center; /Centers the background image /

background-repeat: no-repeat; / Prevents the image from repeating /

}

</style>

</head>

<body>

<h1>Welcome to My Web Page</h1>

<p>This is a sample paragraph with a background image.</p>

</body>

</html>

Example using an external stylesheet (styles.css):

```css

/ styles.css /

body {

background-image: url('background.jpg');

background-size: cover;

background-position: center;

background-repeat: no-repeat;

}

````html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Background Image Example</title>

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

</head>

<body>

<h1>Welcome to My Web Page</h1>

<p>This is a sample paragraph with a background image.</p>

</body>

</html>

For Specific Elements

If you want to set a background image for a specific element, such as a `div`, you can target that element with a CSS class or ID.

Example:

```html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Background Image Example</title>

<style>

.background-container {

width: 100%;

height: 400px;

background-image: url('background.jpg');

background-size: cover;

background-position: center;

background-repeat: no-repeat;

}

</style>

</head>

<body>

<div class="background-container">

<h1>Content Over Background Image</h1>

<p>This content is overlaid on a background image.</p>

</div>

</body>

</html>

**Q 17 HOW ARE ACTIVE LINKES DIFFERENT FROM NORMAL LINKS?**

ANS:- Active links differ from normal links primarily in terms of their state and how they are visually presented to users. In HTML and CSS, the presentation and behavior of links can be controlled through various pseudo-classes that represent different states of the link.

**Q 18 WHAT ARE DIFFERENT TAGS TO SEPARATE SECTIONS OF TEXT?**

ANS :- In HTML, there are several tags that you can use to separate and organize sections of text. Each tag serves a specific purpose and helps structure the content in a meaningful way. Here’s a list of commonly used tags for separating and organizing text:

1. `<h1>` to `<h6>` Tags

Purpose: Define headings of various levels. `<h1>` is the highest level, and `<h6>` is the lowest.

Usage: Organize content into sections with different levels of importance.

```html

<h1>Main Heading</h1>

<h2>Subheading</h2>

<h3>Sub-subheading</h3>

2. `<p>` Tag

Purpose: Defines a paragraph. Used to separate blocks of text.

Usage: Groups related sentences or thoughts into distinct paragraphs.

`html

<p>This is a paragraph of text. It contains a few sentences that are grouped together.</p>

<p>This is another paragraph, separated from the previous one.</p>

3. `<div>` Tag

Purpose: Defines a division or section. It’s a block-level container that can be used to group content and apply styles or scripts.

Usage: Often used with CSS to style sections of a page or with JavaScript for manipulation.

```html

<div class="section">

<h2>Section Title</h2>

<p>This is a section of content within a div element.</p>

</div>

4. `<section>` Tag

Purpose: Defines a section of a document. Represents a standalone section that could be independently distributed or reused.

Usage: Used to group related content, often with its own heading.

```html

<section>

<h2>Section Heading</h2>

<p>Content of the section.</p>

</section>

5. `<article>` Tag

Purpose: Represents a self-contained piece of content that could be distributed or syndicated independently.

Usage: Often used for blog posts, news articles, or user comments.

```html

<article>

<h2>Article Title</h2>

<p>This is the content of the article.</p>

</article>

6. `<aside>` Tag

Purpose: Represents content that is tangentially related to the content around it. Typically used for sidebars or additional information.

Usage: Includes additional information, such as a sidebar or related links.

```html

<aside>

<h3>Related Information</h3>

<p>This content is related but not central to the main content.</p>

</aside>

7. `<nav>` Tag-

Purpose: Defines a navigation section containing links.

Usage: Used to group navigation links or menus.

```html

<nav>

<ul>

<li><a href="#home">Home</a></li>

<li><a href="#about">About</a></li>

<li><a href="#contact">Contact</a></li>

</ul>

</nav>

8. `<footer>` Tag

Purpose: Represents the footer of a section or the entire document. Typically contains metadata, contact information, or links.

Usage: Used to group footer content, such as copyright information.

```html

<footer>

<p>&copy; 2024 My Website</p>

</footer>

9. `<header>` Tag

Purpose: Represents the header of a section or the entire document. Typically contains introductory content, logos, or navigation.

Usage: Used to group header content, such as site titles or navigation menus.

```html

<header>

<h1>My Website</h1>

<nav>

<!-- Navigation links -->

</nav>

</header>

10. `<blockquote>` Tag

Purpose: Represents a block of text that is a quotation from another source.

Usage: Used to quote text from other sources, often with a citation.

```html

<blockquote>

<p>"This is a quoted text from another source."</p>

<footer>- Author Name</footer>

</blockquote>

11. `<hr>` Tag

Purpose: Represents a thematic break or horizontal rule. Used to separate content or sections visually.

Usage: Adds a horizontal line to indicate a thematic shift.

```html

<p>Some text above the line.</p>

<hr>

<p>Some text below the line.</p>

**Q 19 WHAT IS SVG?**

ANS :- SVG, which stands for Scalable Vector Graphics, is a file format and technology used to create vector-based graphics for the web. Unlike raster images (such as JPEGs or PNGs), which are composed of pixels, SVG images are made up of vectors, which are mathematical descriptions of shapes like lines, circles, and polygons. This allows SVG images to be scaled to any size without losing quality, making them ideal for responsive designs and high-resolution displays.

**Q 20 WHAT IS DIFFERENT BETWEEN HTML AND XHTML?**

ANS := HTML (HyperText Markup Language) and XHTML (eXtensible HyperText Markup Language) are both markup languages used to structure and present content on the web. While they share many similarities, there are some important differences between them:

1. Syntax Rules

HTML:

- HTML is more lenient with syntax rules. For example, it allows for optional closing tags and does not enforce case sensitivity.

- Tags can be written in uppercase or lowercase, and some tags can be left unclosed (e.g., `<li>`, `<p>`).

```html

<html>

<head>

<title>Title</title>

</head>

<body>

<p>Paragraph

</body>

</html>

XHTML:

- XHTML is a stricter and more rigorous version of HTML that adheres to XML (eXtensible Markup Language) rules.

- All tags must be properly nested, and all elements must be closed. Tags are case-sensitive and must be written in lowercase.

- XHTML requires that all tags be properly closed (e.g., `<br />`, `<img src="image.jpg" />`).

```xml

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<title>Title</title>

</head>

<body>

<p>Paragraph</p>

</body>

</html>

2. Document Structure

HTML:

- HTML documents do not require a strict XML declaration or namespace. The `<html>` element does not need to specify a namespace.

XHTML:

- XHTML documents must include an XML declaration (`<?xml version="1.0" encoding="UTF-8"?>`) at the beginning.

- The `<html>` element must include the XML namespace declaration: `xmlns="http://www.w3.org/1999/xhtml"`.

3. Error Handling

HTML:

- Browsers are quite forgiving of errors in HTML. They will attempt to render the content even if there are syntax issues or unclosed tags.

XHTML:

- XHTML is stricter about errors. If an XHTML document contains errors, it may not render properly in browsers. XHTML must be well-formed to be processed correctly.

4. Compatibility

HTML:

- HTML is widely supported by all browsers and is more forgiving of errors. It is the standard for most web content.

XHTML:

- XHTML can be used in both standards-compliant and legacy browsers, but browsers may handle XHTML as HTML if the content is served with an incorrect MIME type (`text/html` instead of `application/xhtml+xml`).

5. Use of Attributes

HTML:

- HTML attributes do not require quotes and can be written in any order.

```html

<input type=text value=Submit>

XHTML:

- XHTML attributes must be enclosed in quotes and written in a consistent order.

```xml

<input type="text" value="Submit" />

6. Self-Closing Tags

HTML:

- HTML allows self-closing tags without a trailing slash.

```html

<img src="image.jpg">

XHTML:

- XHTML requires self-closing tags to have a trailing slash.

```xml

<img src="image.jpg" />

7. Case Sensitivity

HTML:

- HTML is not case-sensitive. You can use uppercase or lowercase tags and attributes interchangeably.

```html

<TITLE>Title</TITLE>

XHTML:

- XHTML is case-sensitive. Tags and attributes must be written in lowercase.

```xml

<title>Title</title>

**Q 21 WHAT ARE LOGICAL AND PHYSICAL TAGS IN HTML?**

ANS :- In HTML, tags can be categorized into different types based on their purpose and behavior. Two such categories are logical tags and physical tags. Understanding these categories helps in structuring content effectively for both styling and semantic purposes.

Logical Tags

Logical tags are used to define the meaning or role of content rather than its appearance. They are often used to convey the structure or semantics of the document, which is important for accessibility, search engine optimization (SEO), and maintaining a well-organized content structure. Logical tags describe the purpose or context of the content they enclose.

Examples of Logical Tags:

1. `<strong>` and `<em>`

`<strong>`: Indicates that the text has strong importance, usually rendered in bold.

`<em>`: Indicates emphasized text, usually rendered in italics.

```html

<p>This is a <strong>very important</strong> message.</p>

<p>Please <em>read carefully</em>.</p>

2. `<blockquote>`

- Represents a section of text that is a quotation from another source.

```html

<blockquote>

<p>"To be yourself in a world that is constantly trying to make you something else is the greatest accomplishment."</p>

<footer>- Ralph Waldo Emerson</footer>

</blockquote>

3. `<cite>`

- Represents the title of a creative work or the source of a quotation.

```html

<p>The novel <cite>To Kill a Mockingbird</cite> is a classic.</p>

4. `<code>`

- Represents a fragment of computer code.

```html

<p>Use the <code>printf()</code> function to print output in C.</p>

5. `<dfn>`

- Represents the defining instance of a term.

```html

<p><dfn>HTML</dfn> stands for HyperText Markup Language.</p>

6. `<header>`, `<footer>`, `<article>`, `<section>`, `<nav>`

- These tags provide semantic structure to a webpage, helping to define the layout and meaning of different parts of the content.

```html

<header>

<h1>My Website</h1>

<nav>

<ul>

<li><a href="#home">Home</a></li>

<li><a href="#about">About</a></li>

</ul>

</nav>

</header>

<section>

<h2>About Us</h2>

<p>Content about the company.</p>

</section>

<footer>

<p>&copy; 2024 My Website</p>

</footer>

Physical Tags

Physical tags, also known as presentational tags, are used to define the appearance or style of content directly. These tags are less about the meaning of the content and more about how it should be displayed. In modern web development, CSS (Cascading Style Sheets) is preferred for styling, but physical tags are still used in HTML for controlling the visual presentation.

Examples of Physical Tags:

1. `<b>` and `<i>`

`<b>`: Renders text in bold, but without implying any added importance.

`<i>`: Renders text in italics, but without implying any added emphasis.

```html

<p>This is <b>bold</b> text and this is <i>italic</i> text.</p>

2. `<u>`

- Renders text with an underline.

```html

<p>This text is <u>underlined</u>.</p>

3. `<font>`

- Defines the font, size, and color of text (deprecated in HTML5).

```html

<font color="red" size="4">This text is red and size 4.</font>

4. `<big>` and `<small>`

`<big>`: Renders text larger than the surrounding text.

`<small>`: Renders text smaller than the surrounding text.

```html

<p>This is <big>big</big> text and this is <small>small</small> text.</p>

5. `<center>`

- Centers text horizontally (deprecated in HTML5).

```html

<center>This text is centered.</center>