Web Designing Assignment

Mohinkhan Malek

1. Are the HTML tags and elements the same thing?

HTML tags are like labels or markers we use to define different parts of a web page. Each tag tells the web browser what role that part of the page plays. For example, ** ** indicates a paragraph, **<h1>** signifies a top-level heading, and so on.

Now, an HTML element is a combination of tags and their content. It's like a container that holds information. An element consists of an opening tag, the actual content, and a closing tag. For instance, a paragraph element looks like this:

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

Here, marks the start of the paragraph, this is a paragraph. is the content, and marks the end. Together, they form the paragraph element.

In simpler terms, tags are like signposts, while elements are the signpost along with what's written on it. They work together to structure and present content on a webpage.

2. What are tags and attributes in HTML?

In HTML, tags are like labels that define different parts of a webpage, such as paragraphs, headings, images, and links. They are enclosed in angled brackets, like **<tagname>**. For example, ** ** creates a paragraph, **<h1>** creates a top-level heading, and **<img>** inserts an image.

Attributes, on the other hand, provide additional information about HTML elements. They are placed within the opening tag of an element and consist of a name and a value, separated by an equal's sign. Attributes modify the behavior or appearance of the element. For instance, the **href** attribute in an anchor tag (**<a>**) specifies the URL that the link should point to, and the **src** attribute in an image tag (**<img>**) specifies the image file's location. Here's an example:

```
<a href="https://www.example.com">Click here</a>
```

In this anchor tag, **href** is the attribute name, and **"https://www.example.com"** is the attribute value. Similarly, in an image tag:

```
<img src="image.jpg" alt="Description of the image">
```

Here, **`src`** is the attribute name, and **`"image.jpg"`** is the attribute value. The **`alt`** attribute provides alternative text for the image, which is displayed if the image cannot be loaded or for accessibility purposes.

3. What are void elements in HTML?

Void elements, also known as empty elements or self-closing elements, are HTML elements that do not have any content between an opening tag and a closing tag. Instead, they stand alone with just a single tag. Void elements are typically used to insert something into a document, like an image or a line break, rather than to define a section of content.

Common void elements include:

- **`<img>`**: Inserts an image.
- **`<br>`**: Creates a line break.
- **`<input>`**: Creates an input field.
- **`<meta>`**: Provides metadata about the HTML document.
- **`<link>`**: Links external resources such as stylesheets.

4. What are HTML Entities?

HTML entities are special codes used to represent reserved characters, characters that are not easily typed on a keyboard, or characters that have special meanings in HTML. These entities are written as text and are interpreted by web browsers to display the corresponding characters correctly.

For example, the **`<`** character is reserved in HTML for opening tags, so if you want to display it as regular text, you will use the HTML entity **`<`**. Similarly, if you want to display the copyright symbol (©), you will use the HTML entity **`&copy;`**.

Some common HTML entities include:

- **`<`** : Less than sign (<)
- **`>`** : Greater than sign (>)
- **`&`** : Ampersand (&)

- **`&quot;`** : Double quotation mark (")
- **`&copy;`** : Copyright symbol (©)
- **`&reg;`** : Registered trademark symbol (®)

HTML entities are particularly useful when you need to include special characters in your HTML document that might otherwise be misinterpreted by the browser. They ensure that these characters are displayed correctly regardless of the browser, or the character encoding used.

5. What are the different types of lists in HTML?

In HTML, there are three main types of lists:

Ordered Lists (`<ol>`): Ordered lists are used to present a list of items in a sequential order, typically with numbers or letters. Each item in an ordered list is automatically assigned a number or letter. You create an ordered list using the **`<ol>`** element, and each item in the list is defined using the **`<li>`** (list item) element. Example:

```
<ol>
  <li>First item</li>
  <li>Second item</li>
  <li>Third item</li>
</ol>
```

Unordered Lists (`<ul>`): Unordered lists are used to present a list of items without any particular order. Each item in an unordered list is typically preceded by a bullet point or some other marker. You create an unordered list using the **`<ul>`** element, and each item in the list is defined using the **`<li>`** (list item) element. Example:

```
<ul>
  <li>First item</li>
  <li>Second item</li>
  <li>Third item</li>
</ul>
```

Description Lists (`<dl>`): Description lists are used to present a list of items with associated descriptions or definitions. Each item in a description list consists of a term (defined using the **`<dt>`** element) followed by its description or definition (defined using the **`<dd>`** element). Example:

```
<dl>
```

```
  <dt>Term 1</dt>
  <dd>Description 1</dd>
  <dt>Term 2</dt>
  <dd>Description 2</dd>
  <dt>Term 3</dt>
  <dd>Description 3</dd>
</dl>
```

6. What is the ‘class’ attribute in HTML?

In HTML, the **`class`** attribute is used to assign one or more class names to an HTML element. Class names are essentially labels or identifiers that can be used to apply CSS styles or JavaScript behaviors to specific elements on a webpage.

Here's how the **`class`** attribute works:

1. **Defining a Class**: You define a class name by adding the **`class`** attribute to an HTML element and assigning it a value, which is the name of the class. Multiple classes can be assigned to an element by separating them with spaces. Example:

```
<div class="container">
        <! -- Content goes here -->
</div>
```

2. **Styling with CSS**: Once a class is defined in HTML, you can use CSS to style all elements with that class name. This allows you to apply consistent styles to multiple elements across your webpage. Example CSS:

```
.container {
        width: 80%;
        margin: 0 auto;
        padding: 20px;
        background-color: #f0f0f0;
}
```

3. **JavaScript Interaction**: Class names can also be used in JavaScript to target specific elements for interaction or manipulation. JavaScript can detect elements by their class names and perform actions on them. Example JavaScript:

```
var elements = document.getElementsByClassName("container");
// Do something with the elements...
```

Overall, the **`class`** attribute is a powerful tool in HTML that allows for better organization, styling, and interactivity on web pages.

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

The **`id`** and **`class`** attributes in HTML are both used to uniquely identify elements, but they serve different purposes and have distinct characteristics.

1. **ID Attribute**:

    - The **`id`** attribute is used to uniquely identify a single HTML element on a webpage. Each **`id`** attribute value must be unique within the HTML document, meaning no two elements can have the same **`id`**.
    - It is often used when there is only one instance of a particular element type on a page, or when an element needs to be specifically targeted for styling or scripting.
    - An element can have only one **`id`** attribute.
    - Example: **`<div id="header">...</div>`**

2. **Class Attribute**:

    - The **`class`** attribute is used to categorize multiple elements that share similar characteristics or styles. Multiple elements can have the same **`class`** attribute value, and elements can have multiple **`class`** attributes.
    - It is used when you want to apply the same style or behavior to multiple elements across a webpage.
    - Classes allow for more flexibility and reusability in styling and scripting, as they can be applied to various elements.

- Example: `<div class="container">...</div>`

In summary, the main differences between the **`id`** and **`class`** attributes are their uniqueness and usage scenarios. While **`id`** attributes uniquely identify individual elements, **`class`** attributes categorize and group multiple elements with similar characteristics.

8. What are the various formatting tags in HTML?

HTML provides a variety of formatting tags to style and structure content on a webpage. Some of the commonly used formatting tags include:

1. **Heading Tags (`<h1>` to `<h6>`)**: Used to define headings of different levels, ranging from the most important (**`<h1>`**) to the least important (**`<h6>`**).
2. **Paragraph Tag (`<p>`)**: Defines a paragraph of text.
3. **Bold Tag (`<b>`)**: Renders text in bold.
4. **Italic Tag (`<i>`)**: Renders text in italic.
5. **Underline Tag (`<u>`)**: Renders text with an underline.
6. **Strike Tag (`<strike>` or `<s>`)**: Renders text with a strikethrough effect.
7. **Strong Tag (`<strong>`)**: Indicates strongly emphasized text, typically rendered in bold.
8. **Emphasis Tag (`<em>`)**: Indicates emphasized text, typically rendered in italic.
9. **Superscript Tag (`<sup>`)**: Renders text in a superscript format, typically used for footnotes or mathematical expressions.
10. **Subscript Tag (`<sub>`)**: Renders text in a subscript format, often used for chemical formulas or mathematical expressions.
11. **Line Break Tag (`<br>`)**: Inserts a line break, forcing text to start on a new line.
12. **Horizontal Rule Tag (`<hr>`)**: Inserts a horizontal rule, typically used to separate content sections.

These formatting tags allow developers to structure text and apply basic styling without the need for CSS. However, for more advanced styling and layout, CSS is generally preferred.

9. How is Cell Padding different from Cell Spacing?

Cell padding and cell spacing are both attributes used in HTML tables to control the spacing and padding around the content of table cells, but they serve different purposes:

1. **Cell Padding**

    - Cell padding controls the space between the content of a table cell and the cell's border. It specifies the amount of space to be added inside each cell.
    - Cell padding is defined using the **`cellpadding`** attribute on the **`<table>`** element or using CSS.
    - Example: **`<table cellpadding="5">`** or **`table {padding: 5px;}`**

2. **Cell Spacing**

    - Cell spacing controls the space between individual cells in a table. It specifies the distance between adjacent cells.
    - Cell spacing is defined using the **`cellspacing`** attribute on the **`<table>`** element or using CSS.
    - Example: **`<table cellspacing="5">`** or **`table {border-spacing: 5px;}`**

In summary, cell padding affects the space between the content of a cell and its border, while cell spacing affects the space between adjacent cells in the table. Both attributes can be adjusted to control the appearance and layout of tables in HTML.

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

In HTML tables, you can merge two or more rows or columns into a single row or column using the **`rowspan`** and **`colspan`** attributes, respectively. These attributes allow you to span a cell across multiple rows or columns.

Here's how you can use them:

1. **Rowspan (Merging Rows)**:
    - The **`rowspan`** attribute specifies the number of rows a cell should span vertically.
    - You add the **`rowspan`** attribute to the cell you want to merge and specify the number of rows it should span.
    - Example:

```
<table border="1">
  <tr>
```

```
    <td rowspan="2">Row 1, Cell 1</td>
    <td>Row 1, Cell 2</td>
  </tr>
  <tr>
    <td>Row 2, Cell 2</td>
  </tr>
</table>
```

- In this example, the cell containing "Row 1, Cell 1" spans two rows vertically.

2. **Colspan (Merging Columns)**

- The **`colspan`** attribute specifies the number of columns a cell should span horizontally.
- You add the **`colspan`** attribute to the cell you want to merge and specify the number of columns it should span.
- Example

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

- In this example, the cell containing "Row 1, Cell 1" spans two columns horizontally.

By using **`rowspan`** and **`colspan`** attributes appropriately, you can create complex table layouts with merged cells to achieve the desired visual presentation of your data.

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

Block-level Elements:

- ❖ Think of block-level elements like big building blocks. They start on a new line and take up the full width available to them.

- They stack on top of each other vertically, like a tower of blocks.
- Examples include paragraphs, headings, lists, and divisions (**<p>**, **<h1>** to **<h6>**, **<ul>**, **<ol>**, ** **).

Inline Elements:

- Inline elements are more like decorations you put on your text. They don't start a new line and only take up as much space as their content needs.
- They flow within the text, like adding a picture or making a word bold.
- Examples include links, bold or italic text, images, and input fields (**<a>**, ****, ****, **<img>**, **<input>**).

In simple terms, block-level elements are big and start new lines, while inline elements are smaller and don't start new lines. They're like different kinds of Lego blocks you use to build your webpage!

12. How to create a Hyperlink in HTML?

Creating a hyperlink in HTML is straightforward. You use the **<a>** (anchor) element, along with the **href** attribute to specify the URL or destination where the link should take the user when clicked. Here's how you do it

```
<a href="https://www.example.com">Click here</a>
```

In this example:

- **<a>**: This is the anchor element, used to create the hyperlink.
- **href="https://www.example.com"**: This is the **href** attribute, which specifies the URL of the destination. Replace **"**https://www.example.com**"** with the actual URL you want to link to.
- **Click here**: This is the text that will be displayed to the user as the clickable link. You can replace it with any text you want.

When a user clicks on the link, it will take them to the specified URL in the same browser window/tab by default. If you want the link to open in a new window/tab, you can add the **target="_blank"** attribute

```
<a href="https://www.example.com" target="_blank">Click here</a>
```

This will open the linked page in a new browser window/tab when clicked.

13. What is the use of an iframe tag?

The **<iframe>** tag in HTML is used to embed another HTML document within the current document. It stands for "inline frame" and allows you to display content from another source, such as a webpage, a video, a map, or any other HTML document, directly within your webpage.

Here are some common uses of the **<iframe>** tag:

1. **Embedding External Content**: You can embed content from other websites or web services, such as embedding a YouTube video, a Google Map, or a social media widget.
2. **Creating Inline Frames**: You can use **<iframe>** to create inline frames within your webpage, allowing you to display multiple documents or content sections side by side.
3. **Displaying Advertisements**: Advertisements from ad networks or third-party providers can be displayed within an **<iframe>** to isolate their behavior from the main page.
4. **Security Isolation**: **<iframe>** elements provide a level of security by sandboxing the content within them, which helps prevent malicious code from affecting the main page.

Here's an example of how to use the **<iframe>** tag to embed another webpage

```
<iframe src="https://www.example.com" width="600" height="400" frameborder="0" scrolling="no"></iframe>
```

In this example:

- **src**: Specifies the URL of the webpage or content you want to embed.
- **width** and **height**: Specifies the dimensions of the iframe.
- **frameborder**: Specifies whether to display a border around the iframe (set to "0" to remove the border).
- **scrolling**: Specifies whether to display scrollbars if the content exceeds the dimensions of the iframe. Set to "no" to disable scrolling.

Keep in mind that while iframes offer flexibility in displaying external content, they should be used judiciously to maintain a good user experience and avoid security risks.

14. What is the use of a span tag? Explain with example?

The **** tag in HTML is used to apply styles or to group inline elements together without affecting the document's structure. It is a generic inline container that does not add any specific styling or formatting by itself. Instead, it allows you to target specific parts of text or content for styling or scripting purposes.

Here's an example of how the **** tag can be used:

```
<p>This is a <span style="color: red;">red</span> word. </p>
```

In this example:

- **<p>**: This is a paragraph element.
- **This is a**: This is the text content of the paragraph.
- ****: This is the **** element, used to wrap the word "red".
- **style="color: red;"**: This is an inline style applied to the **** element to change the color of the text to red.
- **red**: This is the word "red" that is wrapped inside the **** element.
- **word.**: This is the remaining text content of the paragraph.

In this example, the **** element is used to apply the color red to the word "red" within the paragraph. The **** element does not add any additional structure to the document but allows you to target specific parts of text for styling purposes.

You can also use the **** element with classes or IDs and apply CSS styles to those classes or IDs to achieve the desired appearance or functionality. For example:

```
<p>This is a <span class="highlight">highlighted</span> word. </p>

. highlight {
   background-color: yellow;
   font-weight: bold;
}
```

In this example, the **** element with the class "highlight" is used to apply a yellow background color and bold font weight to the word "highlighted" within the paragraph.

15. How to insert a picture into a background image of a web page?

To insert a picture into the background of a webpage, you can use CSS background properties. Here's how you can do it:

1. **Using CSS `background-image` Property**: You can use the CSS **`background-image`** property to specify an image to be used as the background of an element. This can be applied to the **<body>** element to set the background image for the entire webpage. Example CSS

```
body {
  background-image: url('background-image.jpg');
  background-size: cover;
  background-position: center;
}
```

2. **Using Inline CSS**: You can also apply the **`background-image`** property directly within the HTML using the **`style`** attribute. Example HTML

```
<body style="background-image: url('background-image.jpg');">
  <! -- Content of the webpage -->
</body>
```

3. **Using External Stylesheet**: Alternatively, you can define the background image in an external CSS file and link it to your HTML document using the **<link>** element in the **<head>** section. Example HTML

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

```
<body>
  <! -- Content of the webpage -->
</body>
```

Example CSS (styles.css):

```
body {
  background-image: url('background-image.jpg');
  }
```

By using one of these methods, you can insert a picture into the background of a webpage and style it according to your design preferences.

16. How are active links different from normal links?

Active links and normal links are similar in appearance and functionality, but they have different states based on user interaction:

1. **Normal Links**

   - Normal links are the default state of links on a webpage.
   - They are displayed as clickable text or elements that users can interact with to navigate to another page or resource.
   - Normal links typically appear with their default styling, such as underlined text or a different color to indicate that they are clickable.

2. **Active Links**

   - Active links are links that are currently being clicked or interacted with by the user.
   - They represent the moment when the user has clicked or activated the link but hasn't released the mouse button yet.
   - Active links may have a different appearance compared to normal links to provide visual feedback to the user that they are being interacted with. For

example, they may change color, have a different background, or have a different border style to indicate the active state.

In summary, while normal links are the default clickable elements on a webpage, active links represent the state when the user is actively interacting with them, such as when they are being clicked or tapped. Active links help provide visual feedback to users about their interactions with the links on the webpage.

17. What are the different tags to separate sections of text?

In HTML, there are several tags that can be used to separate sections of text or content within a webpage. Some common tags for this purpose include:

1. **Paragraph Tag (`<p>`)**

    - The <p> tag is used to define paragraphs of text. It separates blocks of text into distinct paragraphs.

2. **Heading Tags (`<h1>` to `<h6>`)**

    - Heading tags are used to define headings of different levels, ranging from the most important (<h1>) to the least important (<h6>).
    - They are typically used to introduce sections or subsections of content.

3. **Div Tag (`<div>`)**

    - The tag is a generic container that can be used to divide content into logical sections.
    - It is often used to group and separate sections of content for styling or scripting purposes.

4. **Section Tag (`<section>`)**

    - The <section> tag is used to define sections of content within a document.
    - It is commonly used to group related content together, such as articles, blog posts, or chapters.

5. **Article Tag (`<article>`)**

- The **`<article>`** tag is used to define self-contained content that can be independently distributed or reused.
- It is typically used to markup blog posts, news articles, forum posts, or similar content.

6. **Header Tag (`<header>`)** and **Footer Tag (`<footer>`)**

    - The **`<header>`** and **`<footer>`** tags are used to define header and footer sections of a webpage, respectively.
    - They are commonly used to contain introductory or concluding content, such as site navigation links, copyright information, or author information.

These tags provide semantic meaning to the content within a webpage and help organize and structure the information for better readability and accessibility.

18. What is SVG?

SVG stands for Scalable Vector Graphics. It is a markup language used to create vector-based graphics for the web. Unlike raster images (such as JPEG or PNG), which are made up of pixels and can lose quality when resized, SVG images are based on mathematical calculations and can be scaled to any size without losing quality.

Here are some key features of SVG:

1. **Scalability**: SVG images can be scaled up or down without losing quality, making them ideal for use in responsive web design.
2. **Vector-based**: SVG graphics are made up of geometric shapes (such as lines, curves, and polygons) defined by mathematical equations. This allows for precise rendering and smooth scaling.
3. **Text support**: SVG supports text elements, allowing for the inclusion of text within graphics. Text in SVG can be styled using CSS, making it easy to customize.
4. **Interactivity**: SVG supports interactivity and animation through JavaScript. Elements within an SVG graphic can be manipulated dynamically, allowing for interactive user experiences.
5. **Accessibility**: SVG graphics are accessible to screen readers and other assistive technologies, making them suitable for creating accessible web content.
6. **Compact file size**: SVG files are typically smaller in size compared to raster images, especially for complex graphics. This can contribute to faster page loading times and reduced bandwidth usage.

SVG is widely supported by modern web browsers and can be used inline within HTML documents or included as external files. It is commonly used for icons, logos, illustrations, data visualizations, and other types of graphical content on the web.

19. What is the difference between HTML and XHTML?

HTML (Hypertext Markup Language) and XHTML (Extensible Hypertext Markup Language) are both markup languages used to create web pages, but they have some key differences in syntax and rules:

1. **Syntax**:
   - HTML syntax is more forgiving and allows for certain mistakes or omissions, such as unclosed tags or lowercase tag names.
   - XHTML syntax is stricter and requires well-formed XML syntax. All elements must be properly nested and closed, and attribute values must be enclosed in quotes. For example, **`<img>`** must be written as **`<img />`** in XHTML.
2. **Document Structure**:
   - In HTML, the document structure is less rigid, and elements like <html>, <head>, and <body> are optional.
   - XHTML requires a well-defined document structure with a root <html> element, a <head> element containing metadata, and a <body> element containing the document's content.
3. **Case Sensitivity**:
   - HTML is case-insensitive for element and attribute names. That means <p>, <P>, and <P> are treated as the same.
   - XHTML requires element and attribute names to be lowercase. Using uppercase letters in XHTML may result in parsing errors.
4. **Quoting Attributes**:
   - In HTML, attribute values can be enclosed in single quotes (') or double quotes (") or left unquoted if they don't contain spaces or special characters.
   - In XHTML, attribute values must always be enclosed in double quotes ("), and single quotes (') are not allowed.
5. **Namespace**:
   - XHTML is based on XML, which allows for the use of namespaces. This enables XHTML documents to incorporate elements and attributes from other XML vocabularies.
6. **Error Handling**:
   - HTML browsers tend to be more forgiving of syntax errors and may still render pages even if they contain errors.

- XHTML browsers are stricter and may refuse to render pages with syntax errors.

In summary, XHTML is an XML-based reformulation of HTML with stricter syntax rules and a well-defined document structure. While XHTML offers benefits such as improved interoperability and compatibility with other XML-based technologies, it requires developers to adhere to stricter syntax rules compared to traditional HTML.

20. What are logical and physical tags in HTML?

In HTML, the terms "logical tags" and "physical tags" refer to different ways of structuring and styling content on a webpage:

1. **Logical Tags**

    - Logical tags describe the purpose or meaning of the content, rather than its presentation or appearance.
    - They focus on the semantic meaning of the content and are independent of any specific styling or formatting.
    - Examples of logical tags include **`<header>`**, **`<nav>`**, **`<main>`**, **`<article>`**, **`<section>`**, **`<aside>`**, **`<footer>`**, **`<h1>`** to **`<h6>`**, **`<p>`**, **`<ul>`**, **`<ol>`**, **`<li>`**, **`<table>`**, **`<form>`**, **`<input>`**, **`<button>`**, etc.
    - Logical tags help improve accessibility, search engine optimization (SEO), and maintainability of the code by providing clear and meaningful structure to the content.

2. **Physical Tags**

    - Physical tags describe the appearance or presentation of the content on the webpage.
    - They focus on how the content is displayed, styled, or formatted, rather than its semantic meaning.
    - Examples of physical tags include <b> (bold), <i> (italic), <u> (underline), <font>, , <strike>, <big>, <small>, etc.
    - Physical tags are generally considered outdated and less recommended for use in modern HTML documents. Instead, CSS (Cascading Style Sheets) is preferred for styling and formatting content, allowing for separation of content and presentation.

In summary, logical tags provide semantic meaning and structure to the content, while physical tags describe its appearance or presentation. Using logical tags is generally preferred in modern HTML documents as it helps improve accessibility, SEO, and maintainability, while also allowing for better separation of concerns between content and presentation.