HTML - FAQ

Q1 : Can you create a HTML form for both students and teachers where upon submission, the data will be displayed in a table?

```
<html>
 <head>
  <title>Student and Teacher Form</title>
 </head>
 <body>
  <form id="form">
   <label for="name">Name:</label>
   <input type="text" id="name" name="name"><br><br>
   <label for="age">Age:</label>
   <input type="number" id="age" name="age"><br><br>
   <label for="role">Role:</label>
   <select id="role" name="role">
    <option value="student">Student</option>
    <option value="teacher">Teacher</option>
   </select><br><br>
   <input type="submit" value="Submit">
  </form>
  <br><br>
  <table id="table">
   <tr>
    <th>Name</th>
    <th>Age</th>
    <th>Role</th>
   </tr>
  </table>
 </body>
 <script>
  const form = document.querySelector("#form");
  const table = document.querySelector("#table");
```

```
  form.addEventListener("submit", (event) => {
    event.preventDefault();

    const name = document.querySelector("#name").value;
    const age = document.querySelector("#age").value;
    const role = document.querySelector("#role").value;

    const newRow = table.insertRow();
    newRow.insertCell().innerHTML = name;
    newRow.insertCell().innerHTML = age;
    newRow.insertCell().innerHTML = role;
  });
 </script>
</html>
```

When the form is submitted, the JavaScript event listener will trigger and prevent the default form behavior. It will then retrieve the values from the form fields and add a new row to the table with the data.

Q2 : Difference between link and anchor tag?

The <a> (anchor) tag is used to create a hyperlink to another web page or to a specific location within the same page. The <a> tag has two main attributes: href and name. The href attribute specifies the URL of the page or resource to link to, while the name attribute is used to create a named anchor for linking to a specific location within the same page.

The <link> tag, on the other hand, is used to specify relationships between the current document and other resources, such as stylesheets, icons, and alternate versions of the document. The <link> tag is typically used within the <head> section of an HTML document and has attributes such as rel, href, and type.

In summary, the <a> tag is used for creating hyperlinks to other pages or within the same page, while the <link> tag is used for specifying relationships between the current document and other resources.

Q3 : When to add script tag in head and When to do it in body?

The placement of the <script> tag in an HTML document can affect how the JavaScript code within it is executed.

Generally, it is recommended to include the <script> tag in the <head> section of the document if the JavaScript code is needed for setting up the structure of the page or needs to be executed before the page has finished loading.

On the other hand, if the JavaScript code is related to a specific part of the page or needs to be executed after the page has finished loading, it is recommended to include the <script> tag just before the closing </body> tag. This ensures that the page has finished loading and the necessary elements are in place before the JavaScript code is executed.

In addition, you can also include the <script> tag with the async or defer attribute to specify how the script should be loaded and executed. These attributes can help optimize the loading of the page and avoid potential issues with executing the JavaScript code too early or too late.

Q4 : what are attributes in HTML ?

Attributes in HTML are additional values that provide more information or specific instructions to an HTML element. Attributes are specified within the opening tag of an HTML element, and they take the form of name-value pairs.

For example, the src attribute in an <img> element provides the source URL of the image to display, while the href attribute in an <a> element specifies the URL of the linked page.

Some common attributes include:

- ➔ class: used to apply CSS styles to an HTML element
- ➔ id: used to identify a specific element and apply styles or JavaScript to it
- ➔ src: used to specify the source URL for elements like <img> and <script>
- ➔ href: used to specify the URL for elements like <a> and <link>

➔ alt: used to provide alternative text for elements like <img>
➔ width and height: used to specify the size of elements like <img>
➔ style: used to apply inline styles to an HTML element

These are just a few examples of the many attributes available in HTML. Attributes can vary depending on the type of element and the purpose they serve.

Q5 : Explain about anchor tag and target attribute?

The <a> tag, also known as the anchor tag, is used to create hyperlinks in HTML. The <a> tag has a href attribute that specifies the URL of the page or resource to link to.

The target attribute is used in conjunction with the <a> tag and provides additional information about how the linked page should be displayed. The target attribute has several possible values, including:

- _self: The linked page will replace the current page and will be displayed in the same tab or window.
- _blank: The linked page will open in a new tab or window.
- _parent: The linked page will replace the parent frame of the current page.
- _top: The linked page will replace the entire page, breaking out of any frames.

For example:
Visit Example.com

In this example, clicking the link will open the https://www.example.com website in a new tab or window.

Q6 : Explain Doctype in HTML?

The DOCTYPE declaration in HTML is used to specify the type of document being used and the version of HTML that the document conforms to. The DOCTYPE declaration must be the first thing in an HTML document, before any HTML or head elements.

The DOCTYPE declaration is important because it helps the browser determine how to render the HTML document. The different types of DOCTYPEs can vary in syntax and content, and some older versions of HTML may not be supported by newer browsers.

Here are some common DOCTYPE declarations:

1. HTML5: The latest version of HTML, which is simple and easy to use, uses the following DOCTYPE declaration:

   <!DOCTYPE html>

2. HTML 4.01 Strict: A strict version of HTML 4.01 that only allows the use of elements and attributes that are considered to be standard and that do not include presentational or deprecated elements.

   <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">

3. HTML 4.01 Transitional: A version of HTML 4.01 that includes elements and attributes that have been deprecated in HTML 4.01 Strict, as well as some elements used for presentational purposes.
   <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

It's important to choose the right DOCTYPE for your HTML document to ensure that it is rendered correctly by different browsers.

Q7 : Difference between html and xhtml?

HTML (HyperText Markup Language) and XHTML (Extensible HyperText Markup Language) are both markup languages used for creating web pages. They share many similarities but there are some key differences between the two.

Syntax: XHTML has a stricter and more rigorous syntax compared to HTML. In XHTML, all elements must be properly nested and closed, and all attributes must have values.

Document Type Definition (DTD): XHTML uses a strict Document Type Definition (DTD) that requires the document to be well-formed and adhere to XML syntax rules. HTML, on the other hand, has looser syntax rules and uses a loose or transitional DTD.

Case sensitivity: XHTML is case sensitive, while HTML is not. In XHTML, all elements and attributes must be in lowercase.

Deprecated elements: HTML allows the use of deprecated elements, while XHTML does not. In XHTML, only those elements that are part of the current version of HTML are allowed.

Mobile devices compatibility: XHTML is more suitable for use on mobile devices than HTML, as it requires a well-formed document that can be processed by small devices with limited processing power.

Overall, XHTML is considered a more modern and forward-looking approach to web development compared to HTML, as it adheres to XML syntax and provides a more structured and consistent way of coding web pages. However, HTML remains the most widely used markup language on the web and has a large number of resources and tools available for its use.

Q8 : Explain about Responsive Web Design?

Responsive web design is a design approach that makes a website layout adaptable to different screen sizes and devices, such as desktop computers, laptops, tablets, and smartphones. The goal of responsive design is to create a seamless user experience across all devices by delivering a layout that is optimized for the user's device and screen size.

Here are some of the key elements of responsive web design:

1. Fluid grid: A responsive design uses a fluid grid system that scales the layout to fit the screen size. The grid system uses relative units like percentages to specify the width of elements, so the layout adjusts based on the screen size.

2. Media queries: Media queries are used to apply different CSS styles based on the screen size and device. With media queries, the design can change dynamically based on the screen size, orientation, and resolution, allowing the website to respond to different devices.

3. Flexible images and videos: Images and videos need to be flexible too, so they don't get cut off or become too large for smaller screens. To make images and videos responsive, you can use CSS to specify the maximum and minimum widths for elements and set their height to auto.

4. Text and font sizes: Text and font sizes also need to be adjustable for different devices. You can use relative units, such as ems or rems, to specify font sizes, so they scale along with the layout.

Responsive web design is important because it helps to provide a consistent and optimized user experience across all devices, improving the user's engagement and satisfaction with your website. Additionally, a responsive design helps to improve the website's search engine optimization, as search engines prefer websites that provide a good user experience on all devices.

Q9 : What are HTML semantic elements?

HTML semantic elements are HTML elements that have a specific meaning to both the browser and the developer. These elements help to improve the accessibility and structure of a web page.

Some of the most commonly used HTML semantic elements are:

<header>: Defines a container for the header content, such as the logo, navigation menu, and other site-wide elements.

<nav>: Defines a container for navigation links.

<main>: Defines the main content of a document, excluding header, footer, and navigation.

<article>: Defines a self-contained content piece, such as a blog post, a news article, or a forum post.

<section>: Defines a content section that has a specific theme and is often used to group related content together.

<aside>: Defines content that is tangentially related to the main content and is typically displayed as a sidebar.

<footer>: Defines the footer content, such as the copyright information, links to legal documents, and site-wide elements.

<figure>: Defines a content piece that is self-contained and is typically accompanied by a caption.

<figcaption>: Defines a caption for a <figure> element.

Using semantic elements helps to improve the accessibility and search engine optimization of a web page, making it easier for users and search engines to understand the structure and content of the page.

Q10 : In how many ways you can style a HTML page?

There are several ways to style an HTML page:

Inline styling: Inline styles can be added to individual HTML elements using the style attribute. This method is useful for styling specific elements on a page, but can quickly become difficult to manage as the number of styles grows.

Internal stylesheet: An internal stylesheet is added to the <head> of an HTML document using the <style> tag. Styles defined in an internal stylesheet apply to all elements on the page.

External stylesheet: An external stylesheet is a separate file that contains all of the styles for a website, linked to the HTML document using the <link> tag in the <head> of the document. This method is preferred for large websites, as it allows for easier maintenance and management of styles.

CSS Frameworks: CSS frameworks are pre-prepared libraries that are meant to allow for easier, more standards-compliant styling of web pages using CSS. Some popular CSS frameworks include Bootstrap, Foundation, and Materialize.

CSS Preprocessors: CSS preprocessors, such as Sass and Less, allow you to write styles using a more advanced syntax, and then compile them into standard CSS. This method provides additional functionality and makes it easier to manage and maintain large stylesheets.

JavaScript-based styling: JavaScript can also be used to dynamically style elements on a page, either by manipulating styles directly or by adding and removing classes with JavaScript. This method is typically used in combination with other styling methods, such as CSS.

Q11 : How can we access DOM?

The Document Object Model (DOM) is a programming interface for HTML and XML documents. You can access the DOM to manipulate and modify the content and structure of a web page. There are several ways to access the DOM in JavaScript:

Using the document object: The document object is the top-level object in the DOM and represents the entire document. You can use it to access the elements, attributes, and content of a web page. For example, you can use the document.getElementById() method to access an element by its ID, or the document.querySelector() method to access an element based on a CSS selector.

Using the window object: The window object is the top-level object in the browser and represents the entire window or tab. You can use it to access the DOM and the properties of the window. For example, you can use the window.innerHeight property to access the height of the window or the window.document property to access the document object.

Using the element object: Each element in the DOM has its own element object, which you can use to access its properties, attributes, and content. For example, you can use the element.style property to access the inline style of an element, or the element.classList property to access the classes of an element.

Using the node object: The node object is a general-purpose object that represents an element, attribute, text node, or comment in the DOM. You can use it to access the properties and methods common to all nodes in the DOM. For example, you can use the node.nodeName property to access the name of a node, or the node.appendChild() method to add a new node as a child of an existing node.

These are the main ways to access the DOM in JavaScript. By using these methods, you can interact with the content and structure of a web page and create dynamic and interactive experiences for users.

Q12 : What are features of HTML5?

HTML5 is the latest version of the HTML (Hypertext Markup Language) standard, and it introduces a number of new features and capabilities to the web development world. Some of the key features of HTML5 include:

Semantic Elements: HTML5 introduces a number of new semantic elements, such as header, footer, nav, section, article, aside, and figure, which make it easier to structure the content of a web page. These elements provide additional meaning to the content and make it easier for search engines and other technologies to understand the content of a web page.

Video and Audio: HTML5 includes new elements for embedding video and audio content directly in a web page, without the need for plugins like Flash. This makes it easier to include multimedia content in a web page and provides a more consistent experience across different devices and platforms.

Canvas: HTML5 introduces a new canvas element, which allows for dynamic and interactive graphics on the web. The canvas element provides a way for developers to programmatically draw shapes, images, and animations in a web page.

Local Storage: HTML5 introduces a new way to store data on the client-side, called local storage. This allows web applications to store data on the user's device and access it later, even when the user is offline.

Geolocation: HTML5 introduces new APIs for accessing the user's location information, making it possible to create location-aware web applications.

Web Workers: HTML5 introduces a new way to run background scripts in a web page, called web workers. This allows for heavy computation to be performed in the background, without affecting the performance of the main page.

Drag and Drop: HTML5 introduces new APIs for implementing drag-and-drop functionality in a web page, making it easier to create interactive and user-friendly interfaces.

These are just a few of the many features of HTML5. Overall, HTML5 provides a more rich and powerful platform for web development, making it easier to create dynamic and interactive experiences for users.

Q13 : How to handle events in HTML?

Events in HTML can be handled using JavaScript. To handle an event, you need to:

1. Identify the element that you want to bind the event to. You can do this using its id or class attribute.
2. Specify the event you want to bind to. For example, a button click, a form submit, or an element hover.
3. Write a JavaScript function to handle the event. This function will be executed when the event occurs.

Here's an example of how to handle a button click event:

HTML:

```
<button id="myButton">Click Me</button>
```

JavaScript:

```
document.getElementById("myButton").addEventListener("click", function() {
  alert("Button was clicked!");
});
```

In this example, we first use document.getElementById to select the button element with id="myButton". Then, we use addEventListener to bind a click event to the button, and specify the function that should be executed when the event occurs.

Q14 : What are formatting tags?

Formatting tags in HTML are elements that are used to define the appearance and layout of text and other content on a web page. These tags allow you to specify how text should be displayed, including font size, font style, text color, and more. Some common formatting tags in HTML include:

- <p>: The paragraph tag, used to define a paragraph of text.
- : The strong tag, used to make text bold.
- : The emphasis tag, used to italicize text.
- <u>: The underline tag, used to underline text.
- <sup>: The superscript tag, used to create superscript text.
- <sub>: The subscript tag, used to create subscript text.
-
: The line break tag, used to insert a line break in the text.
- <hr>: The horizontal rule tag, used to insert a horizontal rule on the page.
- <font>: The font tag, used to specify font size, color, and face.

In addition to these tags, there are many other formatting tags available in HTML, and new tags are constantly being added with each new version of HTML. However, it's important to note that these formatting tags are being phased out in favor of CSS, which provides more control and versatility in styling text and other content on a web page.

Q15 : How can I change the color of the last anchor element only, where the element may be dynamically added?

You can change the color of the last anchor element by using JavaScript and accessing it through the DOM (Document Object Model). Here's an example of how you could do this:

```
<script>
  document.addEventListener("DOMContentLoaded", function() {
    let anchorElements = document.getElementsByTagName("a");
    let lastAnchor = anchorElements[anchorElements.length - 1];
    lastAnchor.style.color = "red";
  });
</script>
```

In this example, the DOMContentLoaded event is used to wait for the DOM to be fully loaded before executing the script. Then, getElementsByTagName is used to get an array of all the a elements on the page. Finally, lastAnchor is assigned the value of the last element in the anchorElements array, and its color property is set to red.

This way, regardless of how many anchor elements are on the page or when they are added, the script will always find the last one and change its color to red.

Q16 : Difference between inline and block level components?

In HTML, there are two main types of elements: inline elements and block-level elements.

Inline elements: Inline elements are elements that are placed inline with the text, and only take up as much width as necessary to display their content. Examples of inline elements include the a and span tags.

Block-level elements: Block-level elements create a rectangular block that takes up the full width of their parent container, and create a new line before and after the element. Examples of block-level elements include the div, p, and h1 tags.

Here's an example to illustrate the difference between inline and block-level elements:

HTML:

```
<p>This is a <span>span</span> element inside a
<strong>block-level</strong> element.</p>
```

CSS:

```
p {
  background-color: lightgray;
  padding: 10px;
}

span {
  background-color: yellow;
  display: inline;
}

strong {
  background-color: lightblue;
  display: block;
}
```

In this example, the p tag is a block-level element, so it creates a rectangular block with a gray background and padding. The span tag is an inline element, so it only takes up as much width as necessary to display its content, and has a yellow background. The strong tag is also a block-level element, so it creates a new line before and after itself and takes up the full width of its parent container, with a light blue background.

Q17 : What are tags and elements in HTML?

In HTML, a tag is a keyword surrounded by angle brackets < and > that specifies a specific type of element. An element is the combination of a start tag, content, and an end tag that defines a specific type of content on a web page.

For example, the following code creates a paragraph element:

This is a paragraph of text.

Here, is the start tag, This is a paragraph of text. is the content, and is the end tag. The combination of the start tag, content, and end tag define a specific type of content on the web page.

HTML includes many different tags, each with its own specific meaning and function, such as headings (<h1>, <h2>, etc.), paragraphs ( ), lists (<ul>, <ol>, etc.), images (<img>), links (<a>), and more.

It's important to understand the difference between tags and elements in HTML in order to effectively create and structure content on a web page.

Q18 : How can the text "Mobile no. - 1234567890" be displayed on a website using HTML and provide the ability to make a call to the number when clicked without using JavaScript?

To display the text "Mobile no. - 1234567890" and provide the ability to make a call to the number when clicked without using JavaScript, you can use an HTML a tag with the tel: protocol. The code would look like this:

Mobile no. - 1234567890

When the link is clicked, it will initiate a call to the specified number. The tel: protocol is supported by most modern browsers and mobile devices, so this should work on both desktop and mobile devices.

Q19 : What are private and global attributes in HTML?

In HTML, there are no specific terms like "private" and "global" attributes. However, you can understand the concept of global and private attributes in terms of the scope of an attribute.

A global attribute is one that can be used on any HTML element, regardless of the type of element it is. For example, the class attribute is a global attribute that can be used on any HTML element to specify a class name.

A private attribute, on the other hand, is specific to a particular element type and cannot be used on other element types. For example, the cols attribute is a private attribute that can only be used on the textarea element to specify the number of columns in the text area.

In general, it's best to use global attributes whenever possible, as they provide more flexibility and can be used on a wider range of elements. However, private attributes may be necessary in some cases to provide more specific control over an element.

Q20 : How to create a nested web-page in HTML?

To create a nested web-page in HTML, you'll need to use an iframe. An iframe (inline frame) is an HTML element that allows you to embed one HTML document inside another.

Here's an example of how to create an iframe in HTML:

```
<iframe src="nested-page.html" width="600" height="400"></iframe>
```

In this example, the src attribute specifies the URL of the nested web-page, and the width and height attributes set the dimensions of the iframe.

You can also style the iframe using CSS, for example to add borders, background color, or to set its position on the page. Here's an example of how to style an iframe:

```
iframe {
  border: 1px solid black;
  background-color: white;
```

```
  position: absolute;
  top: 50px;
  left: 50px;
}
```

In this example, the iframe has a black border, a white background color, and is positioned 50 pixels from the top and left of the containing element.

Q21 : How to create a hyperlink in HTML?

A hyperlink, also known as a link, is used to connect one web page to another. To create a hyperlink in HTML, you can use the <a> (anchor) element.

Here's an example of how to create a hyperlink in HTML:

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

In this example, the <a> element has a href attribute, which specifies the URL of the page that the link should navigate to when clicked. The text between the opening and closing <a> tags is the text that will be displayed as the link.

You can also create links to other types of resources, such as email addresses, phone numbers, and files, by using different URL schemes. For example:

```
<a href="mailto:info@example.com">Email Us</a>
<a href="tel:+1-555-555-5555">Call Us</a>
<a href="file.pdf">Download PDF</a>
```

In these examples, the first link opens the user's email client and creates a new email to info@example.com, the second link opens the user's phone application and dials the specified number, and the third link downloads the file "file.pdf"