HTML TUTORIAL

1. HTML introduction done
2. HTML DOCUMENT STRUCTURE done
3. HTML headings done
4. HTML paragraph done
5. HTML Hyperlinks done
6. HTML File Links done
7. HTML Styles done
8. HTML span done
9. HTML div done
10. HTML section done
11. HTML button done
12. HTML class done
13. HTML id done
14. HTML images done
15. HTML icons done
16. HTML tables done
17. HTML media

HTML INTRODUCTION

HTML, also known as Hypertext Markup Language. Used to create web pages by using some elements like hyperlinks, video, images, headings, and some other elements.

HTML DOCUMENT STRUCTURE

HTML stands for one of the most user-friendly and easy languages in web development. Important to have a basic knowledge of its structure.

Main elements of the HTML document Structure:

1. **Document Type Declaration (DOCTYPE):**<!DOCTYPE html> used to define the browser about which version of the HTML document is used.  
    **DOCTYPE IN HTML 5:** MINI COMPILER  
     
    **DOCTYPE IN OTHER PREVIOUS HTML VERSIONS:** MINI COMPILER
2. **HTML Element:**<html> is the base of every HTML document. which contains all the work of the page.
3. **Head Section:**<head> section contains <meta>, <title>, <links>, and other resource links too.
   1. **meta:**<meta> tag stands for Metadata which is used by browsers, search engines, and other web services so when users search on the browser related to your page they will reach your page.
   2. **title:**<title> used for adding or changing the title of the HTML document.
   3. **links:**<link> used to link CDN or a CSS file to the HTML file.
      1. **CDN**CDN stands for Content Delivery Network. it is a link to files from the server. CDN includes images, videos, text, and other resources.
      2. **CSS**CSS stands for Cascading Style Sheets. it is a style sheet language used for styling the page by pointing the tags like <body>, <h1>, and other tags.
4. **Body Section:**The main information like video, text, image, and audio are included in the body section this is the main tag that contains all the information

**Main compiler**

HTML Headings

HTML offer us six types of headings <h1>, <h2>, <h3>, <h4>, <h5>, and <h6>. Which represents different levels of headings

Here are the definitions of all six main headings:

<h1>: H1 represents the highest level of heading in HTML, Used to define the main heading or title of the entire page.

<h2>: H2 represents the second level of the heading in HTML, used to define subheadings that are less important than the main heading (H1).

<h3>: H3 represents the third-level of the heading in HTML,

<h4>: H4 represents the fourth level of the heading in HTML, mainly used to define topic with the H3 heading.

<h5> and <h6>: H5 and H6 represent the lowest level of the heading in HTML, mostly used to define bullen-points after the main headings or subheadings

**Main compiler**

HTML PARAGRAPH

The paragraph represents by the <p> element, used to add text to the page as a normal line we add in a normal document.

Here's how it works:

Mini compiler

Key point:

Paragraph Spacing: The browser adds some space before and after every <p> element.

Nested Tags: You can link pages, add video, images and to highlight word u can use <strong> element.

Mini compiler

HTML HYPERLINKS

HTML hyperlinks are like signposts that help you move between web pages or different parts of the same page. They are created using the <a> element (anchor element) and the href attribute (hypertext reference) to specify where they go.

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

MINI COMPILER

In this example, when you click "Click here," it takes you to "https://www.example.com."

Hyperlinks can also be used to link to sections within the same page by using the id attribute to identify the target element. Here's an example:

MINI COMPILER

Clicking "Go to Section 1" scrolls the page to the element with id="section1."

You can make a link open in a new tab or window by using the target attribute:

MINI COMPILER

When clicked, "Open in New Tab" opens the link in a new browser tab or window.

HTML FILE LINKS

To connect files like CSS and JavaScript to your HTML, you can use the <link> and <script> elements, respectively. Here's how you can do it:

Linking a CSS file:

Typically, put the CSS link in the <head> section of your HTML document.

Use this code, replacing 'your\_css\_file.css' with the actual path to your CSS file:

Linking a JavaScript file:

Usually, place the JavaScript link just before the closing </body> tag. This ensures that the JavaScript code is loaded after the HTML content is parsed.

Use this code, replacing 'your\_js\_file.js' with the actual path to your JavaScript file:

Main compiler

HTML STYLES

In HTML, you can use CSS (Cascading Style Sheets) to style your webpage elements. CSS helps you manage how elements look, including their color, size, font, layout, and more. There are several methods to apply styles in HTML:

Inline styles: You can set styles directly in HTML elements by using the 'style' attribute.

Main compiler with 3 slides

Internal styles: You can create styles inside the <style> tag located in the <head> section of your HTML document.

External styles: You can make a separate CSS file with the .css extension and connect it to your HTML document using the <link> tag.

HTML SPAN

In HTML, the <span> element is an inline-level container used to group and apply styles or manipulate specific parts of text within a larger content block. It doesn't carry any specific meaning by itself but enables targeting and styling of text for design or scripting purposes.

Here's an example of how you can use the <span> element:

Mini compiler

In the previous example, the word 'blue' is enclosed within a <span> element with an inline style applied, changing the text color to blue. This enables you to style only that particular portion of text differently from the rest of the paragraph.

You can also assign an id or class attribute to a <span> element, allowing you to style it with CSS or manipulate it with JavaScript.

HTML DIV

In HTML, the <div> element, short for 'division,' serves as a block-level container for organizing and grouping content on a web page. Unlike the <span> element, which is an inline-level container, the <div> element is commonly used to create larger content sections or divisions.

Here's how you can utilize the <div> element:

Mini compiler

In the previous example, the <div> element is employed to group the heading <h1> and paragraph <p> elements. It forms a block-level container that can be styled or accessed with CSS and JavaScript.

The <div> element can also be assigned an id or class attribute, just like other HTML elements, enabling you to apply particular styles or select them with CSS and JavaScript.

HTML SECTION

In HTML, the <section> element is a semantic tag used to define a distinct section of content within a web page. It's commonly utilized to group related content together, providing it with a specific meaning or purpose. The <section> element aids in enhancing the organization and structure of the HTML document.

Here's an example of how you can employ the <section> element:

Main compiler

In the example on the right side, the <section> element is utilized to enclose content that pertains to providing information about a company. Inside it, you'll find a heading <h2> element and a paragraph <p> element.

Additionally, you can nest <section> elements within other <section> elements or other semantic elements to create a hierarchical structure.

HTML BUTTON

In HTML, the <button> element is used to create interactive buttons on a web page. These buttons can trigger actions or perform specific functions when clicked. The <button> element is versatile and can be customized with text, icons, or other HTML elements.

Here's an example of how to use the <button> element:

Mini compiler

In the example on the right, a simple <button> element is created with the text "Click Me" displayed on it. The type="button" attribute is added to explicitly specify that it's a button element.

You can also use the <button> element to wrap other HTML content, such as text, images, or icons:

In this example, an <img> element is placed inside the <button> element along with the text "Submit." This allows you to show an icon or an image on the button.

Mini compiler 2

The <button> element can also be used to execute JavaScript functions or navigate to a different page when clicked. This can be accomplished using JavaScript event handlers, like onclick:

HTML CLASS

In HTML, the class attribute helps you add one or more class names to an element. This makes it easy to style it with CSS or find it using JavaScript or CSS selectors. It's like giving elements labels to group them based on their similarities.

Here's an example:

Mini compiler

In the example above, we have a <p> element with a class attribute set to "highlight." This lets you style or add functionality to this specific paragraph using CSS or JavaScript.

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

For instance, the <div> element has two classes: "box" and "large." This way, you can style elements using multiple class names, which makes your styling more flexible and reusable.

In CSS, you target elements with a specific class using the dot notation:

In this CSS example, the .highlight selector is used to style any element with the class "highlight" by making the text yellow.

HTML ID

In HTML, the id attribute gives a special name to an element, making it unique. This unique name helps you identify and control the element with JavaScript or CSS.

Here's an example:

Mini compiler

In the example above, the <h1> element has an id attribute set to "main-heading." This unique name lets you target this heading specifically for styling or other actions.

To target an element with a particular id in CSS, use the hash (#) notation:

Mini compiler

In this CSS example, the #main-heading selector is used to style the element with the id "main-heading" by making its text blue.

HTML image  
  
In HTML, you use the <img> element to put pictures on your webpage. It's a simple tag that doesn't need a closing tag.

Here's how to use it:

Mini compiler

In this example, the src attribute tells the browser where to find the image. The alt attribute provides text to show if the image can't load or for accessibility.

You can also set the image's width and height with the width and height attributes:

Mini compiler

For instance, width="300" and height="200" make it 300 pixels wide and 200 pixels tall. Be careful with this, as it can distort the image.

You can style the <img> element with CSS classes or inline styles:

Mini compiler

In this example, the <img> has a class attribute set to "rounded" for styling.

Depending on the image's location, you can use relative or absolute paths for the src attribute:

Mini compiler

In this example, src uses an absolute path from the website's root.

The <img> element is crucial for adding images to your webpage. It offers options to control the image source, size, alternative text, and appearance.

HTML icons

In HTML, you can make your web page visually appealing by adding icons. Here are two common ways to do it:

1. Using Icon Libraries: Icon libraries like Font Awesome offer a variety of icons to use in your HTML. To include Font Awesome icons, follow these steps:
   1. Add the Font Awesome stylesheet to your HTML's <head> section with this line:

Mini compiler

* 1. Once the Font Awesome stylesheet is added, use icons in your HTML with the <i> element. For example:  
     mini compiler

1. Using Custom Icon Images: If you have your own icon images, you can include them with the <img> element. Make sure you have the image file (e.g., PNG, SVG) ready and use this code:

Mini compiler

Replace 'path/to/icon.png' with your actual image file location. The 'alt' attribute is for accessibility.

You can also apply CSS styles to customize the <img> element's appearance.

Remember to respect licensing when using icons or images from external sources.

By using icon libraries or custom images, you can enhance your web page's design with attractive icons.

HTML tables

In HTML, tables help you organize information into rows and columns. The <table> element is the main container for the table, and it's often followed by <thead>, <tbody>, and <tfoot> elements to structure the table's content. Here's a basic table structure example:

Main compiler

* <thead> defines the table header with column headings in <th> elements.
* <tbody> contains the main data rows within <tr> (table row) and <td> (table cell) elements.
* You can use <tfoot> for a table footer with summary information.