M1

Markup Language (HTML): Introduction to HTML and HTML5

- Formatting and Fonts –Commenting Code – Anchors –

Backgrounds – Images – Hyperlinks – Lists – Tables – Frames -

HTML Forms.

<https://www.youtube.com/watch?v=G3e-cpL7ofc> CSS

What is HTML?

HTML is considered as official web standard. HTML is the main markup language of the web. It runs natively in every browser.

The HTML specifications are maintained and developed by the World Wide Web Consortium (W3C).

The main parts of our element are as follows:

1. **The opening tag:** This consists of the name of the element (in this case, p), wrapped in opening and closing **angle brackets**. This states where the element begins or starts to take effect — in this case where the paragraph begins.
2. **The closing tag:** This is the same as the opening tag, except that it includes a *forward slash* before the element name. This states where the element ends — in this case where the paragraph ends. Failing to add a closing tag is one of the standard beginner errors and can lead to strange results.
3. **The content:** This is the content of the element, which in this case, is just text.
4. **The element:** The opening tag, the closing tag, and the content together comprise the element.

Elements can also have attributes that look like the following:



Attributes contain extra information about the element that you don't want to appear in the actual content. Here, class is the attribute *name* and editor-note is the attribute *value*. The class attribute allows you to give the element a non-unique identifier that can be used to target it (and any other elements with the same class value) with style information and other things.

An attribute should always have the following:

1. A space between it and the element name (or the previous attribute, if the element already has one or more attributes).
2. The attribute name followed by an equal sign.
3. The attribute value wrapped by opening and closing quotation marks.

**Hypertext Markup Language**. It allows the user to create and structure sections, paragraphs, headings, links, and blockquotes for web pages and applications

HTML is not a programming language, meaning it doesn’t have the ability to create dynamic functionality. Instead, it makes it possible to organize and format documents, similarly to Microsoft Word.

Hypertext means a text that contains references (links) to other texts that viewers can access immediately.

Each HTML page consists of a set of **tags** (also called **elements**), referred as the building blocks of web pages. They create a hierarchy that structures the content into sections, paragraphs, headings, and other content blocks.

Most HTML elements have an opening and a closing that use the ***<tag></tag>*** syntax.

**Block-Level Tags**

The three block level tags every HTML document needs to contain are **<html>**, **<head>**, and **<body>**.

1. The **<html></html>** tag is the highest level element that encloses every HTML page.
2. The **<head></head>** tag holds meta information such as the page’s title and charset.
3. Finally, the **<body></body>** tag encloses all the content that appears on the page.

**Static websites** are ones that are fixed **and** display the same content for every user, usually written exclusively in HTML.

A **dynamic website**, on the other hand, is one that can display different content **and** provide user interaction, by making use of advanced programming **and** databases in addition to HTML.

Tags and attributes

* All HTML elements can have **attributes**
* The href attribute of <a> specifies the URL of the page the link goes to
* The src attribute of <img> specifies the path to the image to be displayed
* The width and height attributes of <img> provide size information for images
* The alt attribute of <img> provides an alternate text for an image
* The style attribute is used to add styles to an element, such as color, font, size, and more
* The lang attribute of the <html> tag declares the language of the Web page
* The title attribute defines some extra information about an element

The main difference between server-side scripting and client-side scripting is that the server side scripting involves server for its processing.

On the other hand, client-side scripting requires browsers to run the scripts on the client machine but does not interact with the server while processing the client-side scripts.

**Static Web pages:**  
Static Web pages are very simple. It is written in languages such as HTML, JavaScript, CSS, etc. For static web pages when a server receives a request for a web page, then the server sends the response to the client without doing any additional process. And these web pages are seen through a web browser. In static web pages, Pages will remain the same until someone changes it manually.

**Dynamic Web Pages:**  
Dynamic Web Pages are written in languages such as CGI (Common Gateway Interface- C, Perl, Python, or PHP) AJAX (Asynchronous JavaScript and XML- eXtensible Markup Language), ASP(Active Server Pages), ASP.NET, etc.

In dynamic web pages, the Content of pages is different for different visitors. It takes more time to load than the static web page. Dynamic web pages are used where the information is changed frequently, for example, stock prices, weather information, etc.

**Difference between <div> and <p> Tag**

| **Feature** | **<div>** | **<p>** |
| --- | --- | --- |
| **Meaning** | Division → a generic **container** for grouping elements. | Paragraph → represents a **block of text**. |
| **Semantic value** | Has **no semantic meaning** (just a block-level container). | Semantic → clearly indicates a paragraph of text. |
| **Default display** | Block-level element (creates a new block, but no extra spacing by default). | Block-level element with **default margin** above and below. |
| **Use case** | Used for layout, grouping content, applying CSS/JS. | Used for writing readable text content. |
| **Example** | html <div class="box"> <h2>Title</h2> <p>Some text...</p> </div> | html <p>This is a paragraph of text.</p> |
| **Accessibility / SEO** | Not meaningful by itself (needs roles, classes, or ARIA for meaning). | Recognized by browsers, screen readers, and search engines as actual text content. |

HTML5

Difference between HTML and HTML5

**Difference between HTML and HTML5**

| **Aspect** | **HTML (Older versions)** | **HTML5 (Latest standard)** |
| --- | --- | --- |
| **Introduction** | Introduced in 1991, last major version was HTML4 (1999). | Introduced in 2014 by W3C/WHATWG as the modern standard. |
| **Doctype declaration** | Long and complicated: <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd"> | Simple: <!DOCTYPE html> |
| **Multimedia support** | Needed plugins like Flash, Silverlight for video/audio. | Has built-in <audio> and <video> tags. |
| **Graphics support** | No native support, needed plugins (e.g., Flash). | <canvas> and <svg> provide graphics & animations. |
| **New semantic tags** | No semantic tags (only <div> for layout). | New tags: <header>, <footer>, <article>, <section>, <nav>, <aside>. |
| **Form elements** | Limited input types (only text, password, checkbox, radio, etc.). | New input types: email, date, url, range, search, number, etc. |
| **APIs & features** | Very limited. | Rich APIs: Geolocation API, Web Storage (localStorage & sessionStorage), Web Workers, Drag & Drop, Offline support. |
| **Mobile friendly** | Not optimized for mobile devices. | Designed for **mobile-first web apps**, supports responsive design. |
| **Error handling** | Strict and inconsistent between browsers. | More forgiving, better error handling across modern browsers. |

**In short:**

* **HTML** was mainly for structuring text and links.
* **HTML5** is powerful → supports multimedia, graphics, mobile apps, and APIs **without extra plugins**.

<https://youtu.be/vHmUVQKXlVo>

Formatting

https://youtu.be/pEpqf\_OvIPM

Font

https://youtu.be/Qy\_VX9j4SV8

**comments<!--Write something which increase the readability and not displayed in the browser-- >**

**https://youtu.be/\_QPjxYJKeNc**

**Anchor tag**

**<a href=”give path”>display information</a>**

**Background**

[**https://youtu.be/QQj-ULba4\_0**](https://youtu.be/QQj-ULba4_0)

**List**

**https://youtu.be/lMcYfLAMGac**

**Table**

**https://youtu.be/Kzi6ZdUUtgY**

Frames

<https://youtu.be/LkquxqE1Lxg>

HTML Forms

<https://youtu.be/3VZiqUj_9z8>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **M24CA1B105 Web Development Lab** | | | | |
| **NO** | **Date** | **Day** | **Program** | **CO** |
| 1 | 10/9/2025 | Wednesday | Basic Html Tags <html> <head> <title> <body> <h1> to <h6> <p> <br> <hr> <b> / <strong> <i> / <em> <u> <small> <mark> <a href="URL">  <img src="image.jpg" alt="description"> | CO1 |
| 2 | 12/9/2025 | Friday | Prepare a Registration form to fill the following data for taking admission for any of the courses offered. [Courses - BA,B.Sc,BCA] Personal Information - Name ,Gender, DOB, Address, Phone , Guardian, Hobbies and extra curricular activities. [use  checkbox,Radio button,TextArea, Table...] Office Use .....Provide proper division to increase the aesthetic view | CO1 |
| 3 | 17/9/2025 | Wednesday | Create an HTML page to demonstrate the use of inline, internal, and external CSS. Use inline CSS to change the color of a heading. Use an internal style sheet to set the background color of the page and style the paragraphs. Use an external CSS file to apply a border and text alignment to a div element. | CO1 |
| 4 | 19/9/2025 | Friday | Write a JavaScript program for displaying a number is even or odd. | CO2 |
| 5 | 24/9/2025 | Wednesday | Perform validation on the Registration form using javascript Code. | CO2 |
| 6 | 8/9/2025 | Wednesday | Write a simple JavaScript program for  Cooking (takes 3 seconds to "finish")  kitchen Cleaning (without waiting for cooking to finish). | CO2 |
| 7 | 10/9/2025 | Friday | Repeat Lab | CO1 to CO2 |
| 8 | 24/10/2025 | Friday | Create a database named CollegeDB,Table Students with fields:id (INT, Primary Key), name (VARCHAR(50)),  age (INT), course (VARCHAR(30)) | CO3 |
| 9 | 29/10/2025 | Wednesday | Display all students enrolled in "Computer Science". Update the course of a student with id = 2 to "Data Science". Delete a student record with id = 3. | CO3 |
| 10 | 31/10/2025 | Friday | Create a list of students with IDs and names and render them. Use id as the key. | CO4 |
| 11 | 5/11/2025 | Wednesday | Create a React application to display a list of items using components. | CO4 |
| 12 | 7/11/2025 | Friday | Create a React class component called Counter that displays a number and has two buttons: Increment and Decrement. The component should manage its own state and update the displayed number when the buttons are clicked. | CO4 |
| 13 | 12/11/2025 | Wednesday | Create a reusable React component called Button that can be used in multiple places with different styles and behaviors. | CO4 |
| 14 | 14/11/2025 | Friday | Create a React application that demonstrates routing and navigation between multiple pages. | CO5 |
| 15 | 19/11/2025 | Wednesday | Build a React component called ProfileCard that displays a user’s profile using Bootstrap styling. The card should include a profile picture, name, short bio, and a button to “Follow”. | CO5 |
| 16 | 21/11/2025 | Friday | Repeat Lab | CO3 to CO5 |
| 17 | 3/12/2025 | Wednesday | Micro Project | CO1 to CO5 |
| 18 | 5/12/2025 | Friday | Micro Project | CO1 to CO5 |
| 19 | 10/12/2025 | Wednesday | Micro Project | CO1 to CO5 |
| 20 | 12/12/2025 | Friday | Micro Project | CO1 to CO5 |