



Web Technology – CACS 205

HTML- Hypertext Markup Language

What is HTML?

- HTML stands for **HyperText Markup Language**. It is a standard markup language for web page creation. It allows the creation and structure of sections, paragraphs, and links using HTML elements (the building blocks of a web page) such as tags and attributes.
- Tim Berners-Lee, a British computer scientist who was also the inventor of the World Wide Web, created HTML in the early 1990s.
- HTML has a lot of use cases, namely:
- **Web development**. Developers use HTML code to design how a browser displays web page elements, such as text, hyperlinks, and media files.
- **Internet navigation**. Users can easily navigate and insert links between related pages and websites as HTML is heavily used to embed hyperlinks.
- **Web documentation**. HTML makes it possible to organize and format documents, similarly to Microsoft Word.

HTML?

- **HTML** is an acronym which stands for Hyper Text Markup Language
- To understand "HTML" from front to back, let's look at each word that makes up the abbreviation:
- **Hypertext**: text (often with embeds such as images, too) that is organized in order to connect related items
- **Markup**: a style guide for typesetting anything to be printed in hardcopy or soft copy format
- **Language**: a language that a computer system understands and uses to interpret commands.

Overall, HTML determines the **structure** of web pages.

Important Terminology

HyperText

- The term HyperText in HTML simply means “Text within Text”. A text that contains links to other texts is called hypertext. It provides a way to connect two or more web pages with each other in a website.
- We can access it by clicking on the links. When you click on a link, you go to a new web page. Actually, you have clicked on a hypertext.
- **Markup language:**
- A markup language is a computer language that creates web pages (also called HTML document). It uses a set of tags to annotate the information in a document. In the HTML document, tags indicate how content should be displayed.
- For example, when a web browser reads the **** and **** HTML tags in a web page, the browser bolds the text between those tags because **** and **** tags are used to bold the text.

Important Terminology

Web Page:

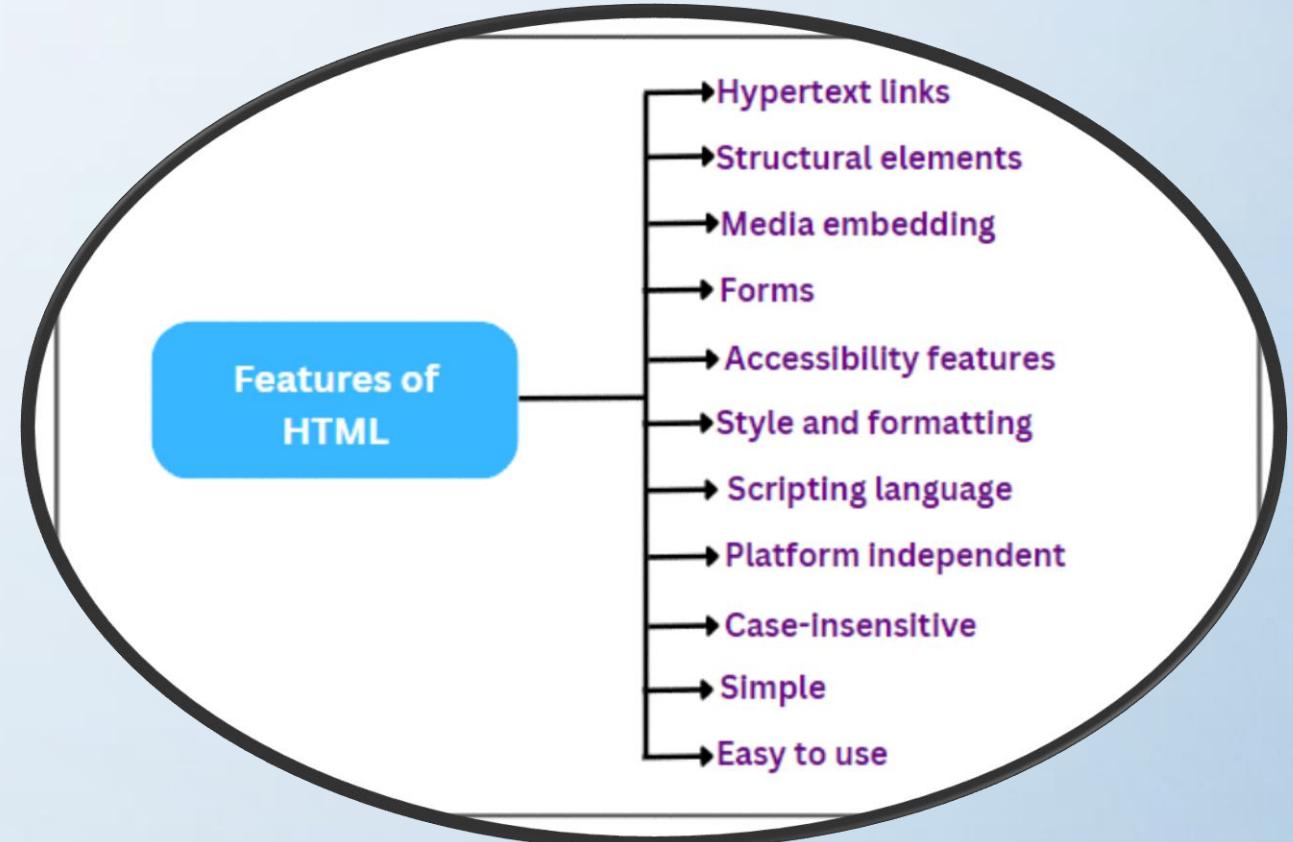
- A web page is a simply document commonly written in HTML and interpreted (i.e. read) by a web browser. It can be of static or dynamic type. We can identify by entering an URL. With HyperText Markup Language, we can only create a static web page.

Website:

- A website is a collection of related one or more web pages that are connected together and organized into a hierarchical structure. We host it on a web server and can access it through the internet using a web browser.
- In website, each web page contains text, images, videos, and other digital content presented on a specific format and style using HTML, CSS, and JavaScript. A website can be a static that contains a simple information or can be dynamic such as a blog or social media, etc.
- **Static Vs Dynamic Website(Self Study Task)**

Features of HyperText Markup Language

- Hypertext Markup Language is the standard markup language used to develop web pages. It has emerged over time and has various significant features that allow web developers to make web pages with rich content and functionality.



- **Hypertext links:** This feature allows us to create links between web pages so that users can navigate between different web pages with ease.
- **Structural elements:** This feature provides a wide range of elements that allow web developers to structure the content of a web page, such as headings, paragraphs, lists, links, tables, etc.
- **Media embedding:** This feature allows us to embed (i.e. insert) various types of media, such as images, audio, video, and graphics on a web page.
- **Forms:** HTML provides a set of form elements that allow us to create input fields, radio buttons, checkboxes, and other form elements that facilitate programmers to submit data.

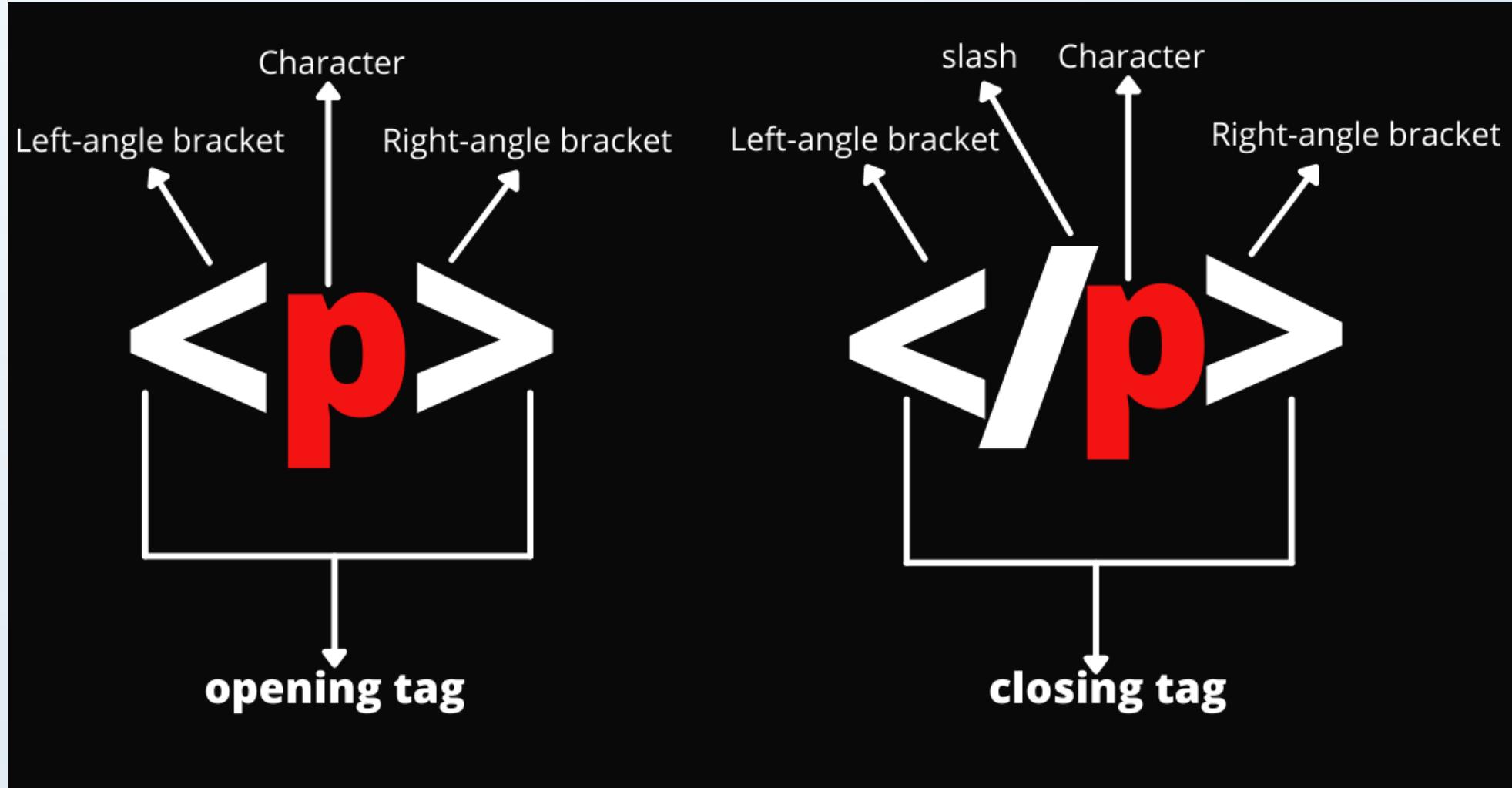
- **Accessibility features:** HTML offers such several features that make web pages accessible to people with disabilities, such as the facility to add alt text to images and to provide decent heading structure.
- **Style and formatting:** This feature provides a wide range of attributes and styles. It allows programmers to control the appearance of a web page, such as font size, color, symbols, emojis, layout, etc.
- **Scripting language:** It facilitates programmers to insert JavaScript code in a web page, which adds interactivity and functionality to a web page or website.
- **Platform:** Hypertext markup language is a platform-independent because we can display it on any platform like Windows, Linux, MacOS, etc.
- **Case-insensitive:** HTML is a case-insensitive language, meaning we can write tags either in lowercase or uppercase. We recommend you write all tags in lowercase for consistency, better readability, and understanding.

HTML Tags

HTML defines the markup for a particular web page, you'll want the text, images, or other embeds to appear in certain ways.

- For example, you might want some text to be big, other text to be small, and some to be bold, italic, or in bullet point form.
- HTML has "tags" that let you get this done. So, there are tags to create headings, paragraphs, bolded words, italicized words, and more.

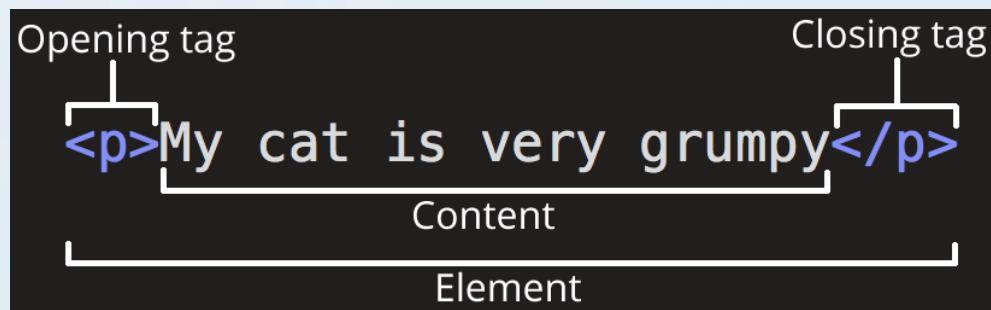
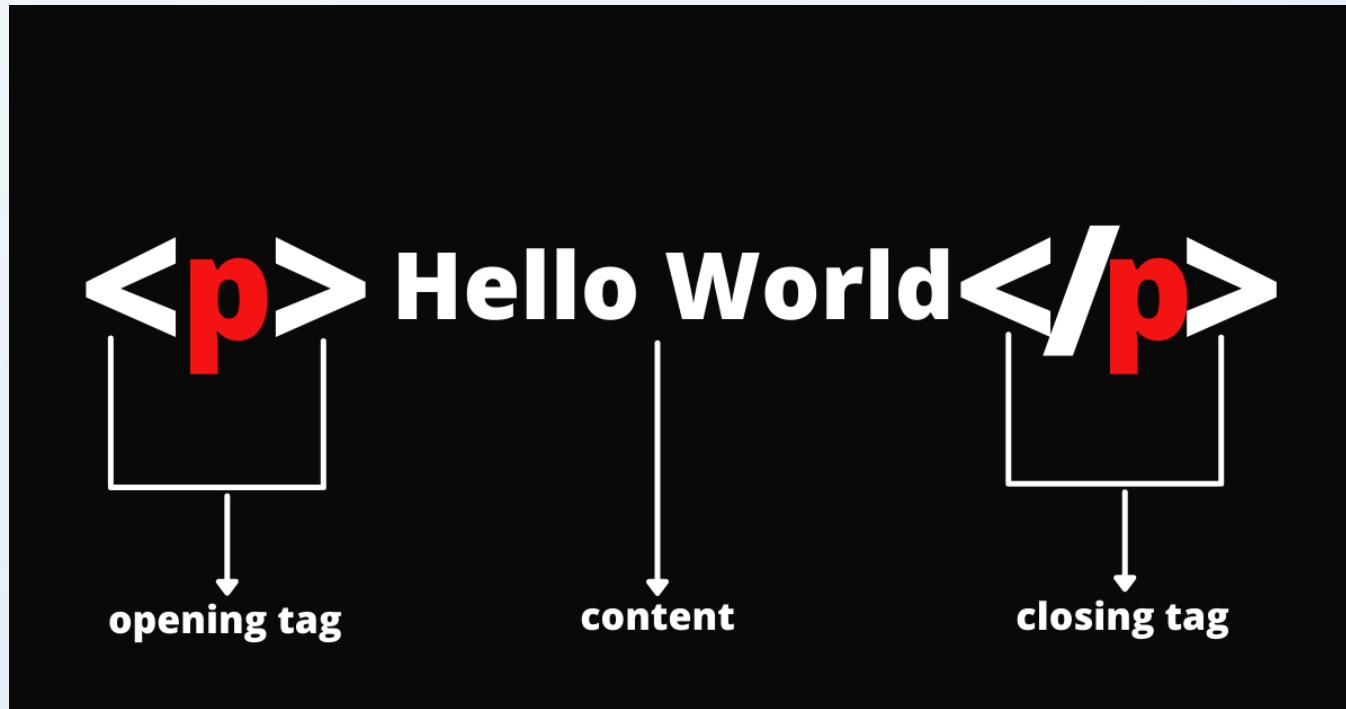
HTML Tag Structure



HTML Elements

- An element consists of the opening tag, a character, the content, and a closing tag.
- Some elements are **empty** – that is, they don't have a closing tag but instead have a source or link to content that you want to embed on the web page.
- An example of an empty element is , which you use to embed images on a web page.
- HTML elements are often used interchangeably with tags, but there's a small difference between the two.
- An element is a combination of the opening and closing tag, and then the content between them.

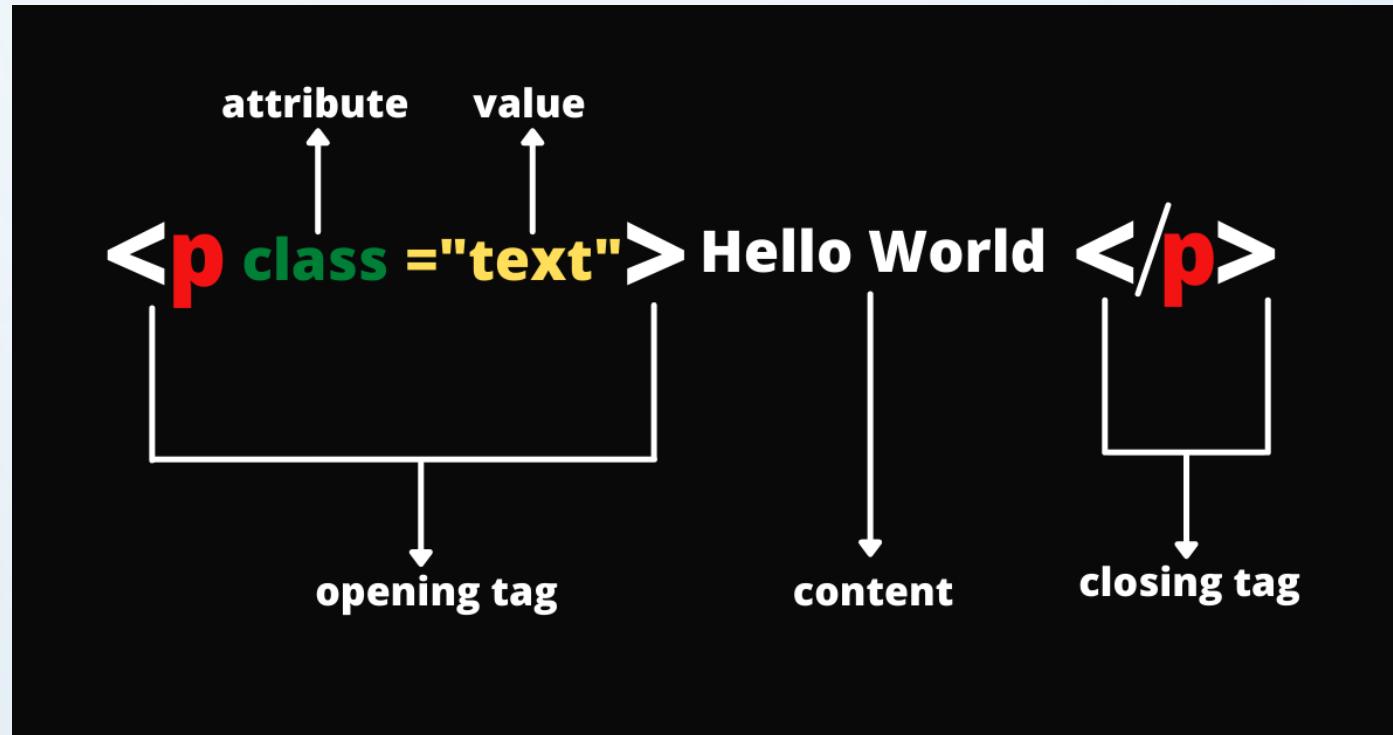
HTML Elements



HTML Attributes

- HTML attributes provide additional information about HTML elements.
- Attributes are placed in the opening tag and range from style and ids to classes. They take values, which convey more information about the element and help you do things such as styling and manipulation with JavaScript.
- Attributes usually come in name/value pairs like: **name="value"**
- This can be used to style the element or select it with JavaScript for interactivity.

HTML Attributes

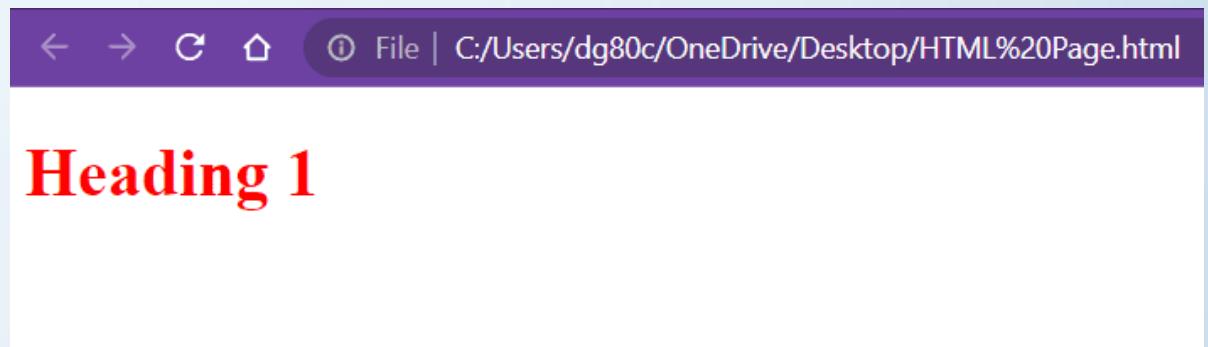


- Above example have **class** attribute with value **text**

HTML Attributes Example

```
<!DOCTYPE html>

<html>
  <head>
    <title>Attribute Example</title>
  </head>
  <body>
    <h1 style = "color: red">Heading 1</h1>
  </body>
</html>
```

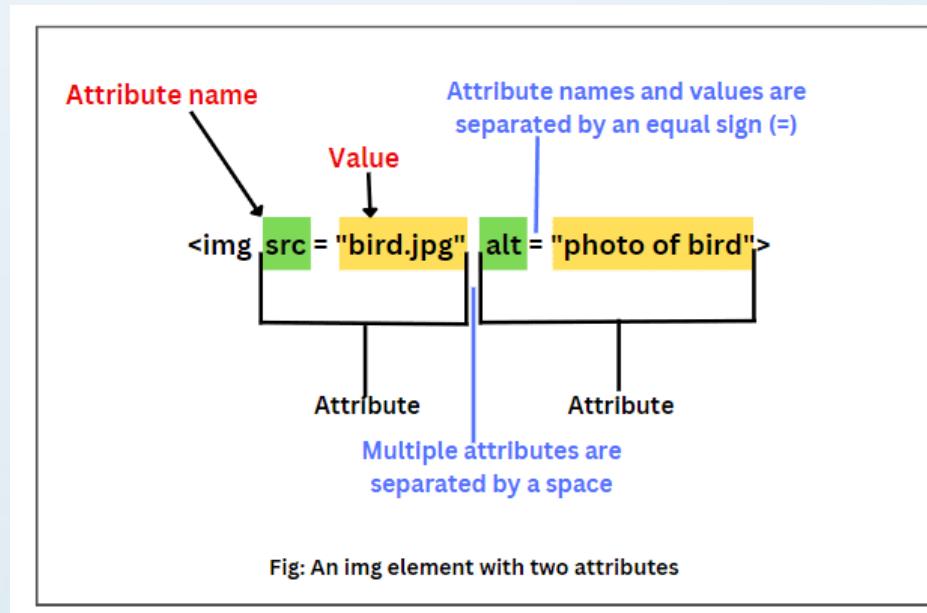


Using Multiple Attribute at Once

We can also put more than one attribute in an element in any order. Just keep them separated by spaces. The general syntax is as:

```
<element attribute1 = "value" attribute2 = "value">
```

Look at the below figure where an **img** element with two attributes.



Key points of HTML Attributes:

There are several important points about HTML attributes you should keep in mind while working with attributes.

- We can put attributes after the element name in the opening tag only, never in the closing tag.
- As per need, we can apply several attributes to an element, separated by spaces in the opening tag. Their order is not important.
- Most HTML attributes take values, which follow an equal sign (=). Some attribute values take a single descriptive word.
- An attribute's value can comprise a word, a string of text, a number, a URL, or a measurement, depending on the purpose of the attribute.

Key points of HTML Attributes:

- Enclosing attribute values in double quotes is a strong convention, but quotation marks are not requisite in HTML5. We may omit them.
- HTML allows us to enclose attribute values either in single or double quotation marks. Both are acceptable as long as the opening and closing marks match. Note that quotation marks in HTML files must be straight ("), not curly ("").

```
<a href="https://www.scientecheeasy.com">A link to HTML</a>
```

```
<a href='https://www.scientecheeasy.com'>A link to HTML</a>
```

In HTML5, we can also omit the use of quotation marks around attribute values. For example:

```
<a href=https://www.scientecheeasy.com>A link to HTML</a>
```

- We cannot make up an attribute for an element because HTML5 specification has already defined the attribute names and values for each element.

Types of HTML Attributes

- **Global attributes:** We can use these attributes with any HTML element. They provide a general-purpose way to add additional information to an element. Some commonly used global attributes are class, id, style, and title.
- **Event attributes:** We use these attributes to define how an element responds to user actions, such as clicks or mouse movements. Some commonly used event attributes are onclick, onmouseover, and onmouseout.
- **Element-specific attributes:** These attributes are unique for specific HTML elements that provide additional customization options for elements. For example, we use the src attribute to specify the URL of an image in the img element. Some commonly used element-specific attributes are src, href, and type.

Types of HTML Attributes



Global Attributes

- HTML Global Attributes are those attributes that are common to all HTML tags/elements.
- An attribute of any HTML tag is nothing but a part of it which can be used to add more information to any HTML tag, for example, in an anchor tag, href is an attribute which is used to provide the URL to which the text will be hyperlinked.
- Similarly, HTML tags can have global attributes to provide additional identification, information like CSS style classes, or some standard properties hidden(to hide HTML element), language, etc.
- Any HTML tag can have multiple attributes.
- Global attributes can be used with all tags; though on some HTML tags they do not have any effect.
- There are some exceptions too where these attributes are not relevant, some of the exceptional tags that do not support global attributes are <base> tag, <script> tag, <title> tag, etc.
- Global attributes can even be used with the tags that are not specified as HTML standard tags; i.e the non-standard tags also permit global attributes. Using these non-standard elements in a document indicates that a document is no longer HTML5 compliant.

Global Attributes

DESCRIPTION	
<u>id</u>	Sets a unique identifier for the element.
<u>class</u>	Sets one or more CSS classes to be applied to the element.
<u>style</u>	Sets the style for the element.
<u>data-*</u>	Defines additional data that can be used by JavaScript.
<u>hidden</u>	Specifies whether the element is hidden.
<u>title</u>	Sets a title that displays as a tooltip when element is hovered.
<u>tabindex</u>	Sets a tab sequence number relative to the other elements.
<u>lang</u>	Sets the language for the element.
<u>draggable</u>	Specifies whether the element can be dragged.
<u>accesskey</u>	Sets a shortcut key for the element.
<u>inputmode</u>	Sets keyboard configuration for when editing the element.
<u>spellcheck</u>	Specifies whether to spellcheck the element.
<u>autocapitalize</u>	Specifies to capitalize the data entered.
<u>contenteditable</u>	Specifies whether the element is editable.
<u>dir</u>	Sets the display direction: left-to-right or right-to-left.
<u>is</u>	Specifies that element behaves like a registered custom element.

Guidelines for using HTML Attributes

When dealing with attributes in HTML, it is essential to follow the guidelines that make your web pages both accessible and SEO-friendly. Some important guidelines that you should keep in mind:

- Always try to use descriptive attribute values that accurately describe the content of the element.
- You should use the correct type of attribute for the content being added. For example, use alt attribute for an image to provide alternative text for users who cannot see the image.
- Try to use CSS instead of inline styles whenever possible to separate presentation from content.
- Avoid using too many attributes on a single element because it can make your code difficult to read and maintain.
- Use event attributes sparingly, as they can make your code more complex and difficult to debug.

SEO Considerations for HTML Attributes

HTML attributes can affect your website's search engine optimization (SEO). Here are some important things you should keep in mind:

- Use descriptive alt attributes for images to help search engines so that they could understand the content of your web page.
- Add a title attribute to provide additional information about the link, which can improve the user experience and increase click-through rates.
- Avoid using keyword stuffing in your HTML attributes, as search engines can see as spammy this.

Common Mistakes to Avoid While Using Attributes

While using HTML attributes, there are some common mistakes you should avoid. They are a few as:

- Using inline styles instead of CSS.
- Usage too many attributes on a single element.
- Usage event attributes excessively.
- Using empty alt attributes on images.
- Using irrelevant or misleading attribute values.

SGML

The Standard Generalized Markup Language (SGML, defined in [\[ISO8879\]](#)), is a language for defining markup languages. HTML is one such "application" of SGML.

An SGML application consists of several parts:

- The SGML declaration. The SGML declaration specifies which characters and delimiters may appear in the application.
- The document type definition (DTD). The DTD defines the syntax of markup constructs. The DTD may include additional definitions such as numeric and named character entities.
- A specification that describes the semantics to be ascribed to the markup. This specification also imposes syntax restrictions that cannot be expressed within the DTD.
- Document instances containing data (contents) and markup. Each instance contains a reference to the DTD to be used to interpret it.

Versions of HTML

Different **Version of HTML** has different properties. But today we use HTML5, HTML5 is the latest version of HTML.

- HTML 1.0: released in 1991
- HTML 2.0: released in 1995
- HTML 3.2: released in 1997
- HTML 4.01: released in 1999
- XHTML: released in 2000
- HTML5: released in 2014

Versions of HTML

HTML 1.0

- **HTML 1.0** or first version of HTML was a version of SGML that had ability to link different document or pages using '**href**'.
- HTML 1.0 had 20 elements or tags, now latest version of HTML, ie HTML5 has a lot more.

Versions of HTML

HTML 2.0

- After HTML 1.0, the second version of HTML was released in 1994. **HTML 2.0** was an expansion of HTML 1.0.
- Internet Engineering Task Force (IETF) was behind it's creation.

Versions of HTML

HTML 3.2

- **HTML 3.2** was released In 1997. HTML 3.2 had many new features like tables, superscript, subscript etc.
- Two most important features introduced in HTML 3.2 were tables and text flow around images.
- Tables were widely used and programmers still use them but it is not recommended anymore.

Versions of HTML

HTML 4.01

- **HTML 4.01** was released In 1999. HTML 4.01 introduced features like scripting, style sheets, better tables, better forms frames and embedding objects.
- HTML 4.01 was a revised version of **HTML 4.0**, it also included features for the disabled people to enhance their interactivity with the Global world through Internet.

Versions of HTML

XHTML

- In 2000 **XHTML** was released. XHTML stands for Extensible Hyper Text Markup Language. XHTML has strict set of rules and it is basically an XML application of HTML.

Versions of HTML

HTML5

- So all of this added up and then after so many year **HTML5** was released in 2014. HTML5 is the best version of HTML up till now. HTML5 improved user interactivity so much and also lessened the burden of devices.
- HTML5 fully supports all kind of media application that are there. HTML5 supports both audio and video media content.
- HTML5 also provides full support for JavaScript to run in the background.

Versions of HTML

HTML 5.3

The latest version of HTML is **HTML 5.3**, released on 28 Jan 2021. HTML5.2 version is now the standard for web development and is widely supported by major modern web browsers, such as Chrome, Firefox, Opera, etc.

Here's the anatomy/structure of a basic HTML 4 page:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"  
      "http://www.w3.org/TR/html4/strict.dtd">  
  
<HTML>  
  
<HEAD>  
  
  <TITLE>The document title</TITLE>  
  
</HEAD>  
  
<BODY>  
  
  <h1>This is a heading</h1>  
  
  <p>Document content goes here.....</p>  
  
</BODY>  
  
</HTML>
```

Here's the anatomy/structure of a basic HTML 5 page:

```
<!DOCTYPE html>

<html>

<head>

<title>This is document title</title>

</head>

<body>

<h1>This is a heading</h1>

<p>Document content goes here.....</p>

</body>

</html>
```

Standard format of HTML Document

```
<!DOCTYPE html>

<html>

    <head>

        <meta charset="UTF-8" />

        <meta http-equiv="X-UA-Compatible" content="IE=edge" />

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

        <title>Definition of HTML</title>

    </head>

    <body>

        <!--Page content such as text and images goes in here-->

    </body>

</html>
```

Let's learn individual element of HTML document

<!Doctype html>:

- Specifies that we're using HTML5 in this code. Before the introduction of HTML5, you had to explicitly state which version of HTML you were coding in with the <!Doctype> tag.
- For example, HTML4.0, 3.2, and so on. But now we no longer need it. When “html” is written in the code, the browser automatically assumes that you are coding in HTML5.

<html></html>:

- the root, or top-level element of every HTML document. Every other element must be wrapped in it.

Let's learn individual element of HTML document

<head></head>:

- one of the most crucial parts of the HTML document.
- Web crawlers look inside the head tags to get important information about the page.
- It contains info such as the page title, stylesheets, meta information for SEO, and lots more.

<title></title>:

- This defines the title of the web page. It is always shown in the browser tab.

Let's learn individual element of HTML document

<meta />:

- This is an empty element that conveys meta-information about the page. Such information may include the author, what type of encoding it's using (almost always UTF-8), responsiveness, compatibility, and a lot more.
- Web crawlers always look at the meta tag to get information about the web page, which will play a crucial role in SEO.

<body></body>:

- All the content of the HTML document is located inside the body tag. There can only be one <body> tag on the whole page.

Understanding of HTML Structure Summary

Let us understand the brief description of the above HTML structure example.

- The declaration of <!DOCTYPE html> defines the document type that instructs the browser about the latest version of HTML5.
- The <html> element is used to enclose the entire HTML document. It is the root element of the HTML web page and informs the web browser about the HTML document.
- The <head> element contains metadata that gives the information about the document, such as the title, which is displayed in the browser's title bar. You must close it before the body tag opens.

Understanding of HTML Structure Summary

- The `<title>` element is used to specify the title of the HTML document at the top of the browser window. You must place this tag within the head tag and should close immediately.
(Optional)
- The `<body>` element contains the main content of the HTML document, such as text, images, links, and other media. The content between the body tag is visible to the user on the web browser.
- The `<h1>` element is used to define a top-level heading of the web page. We mainly use it for the main heading of the page.
- The `<p>` element is used to define a text paragraph of the webpage.

HTML Text Editor | HTML In Notepad

- An HTML document is nothing but a text file, so we can use any text editor to create an HTML page.
- An HTML text editor is a software application that allows us to create, edit, and save the HTML code. However, it is also possible to write HTML code using a simple text editor like Notepad or Notepad++.
- Types of HTML Text Editors
 - Integrated Development Environments (IDEs) like Visual Studio Code, Sublime Text, and Atom.
 - Online HTML text editors like CodePen and JSFiddle.
 - WYSIWYG editors like Dreamweaver.

Create a webpage with HTML using Notepad.

Step 1: On your computer system, just click on the Start button to open Notepad++ and then click on Notepad if the shortcut icon for Notepad++ is not on the desktop. You can do this by typing “Notepad” in the Windows search bar.

Create a webpage with HTML using Notepad.

Step2 : Now write the following code snippet in your document.

```
<DOCTYPE html>

<html>

    <head>
        <title>My first HTML webpage</title>
    </head>

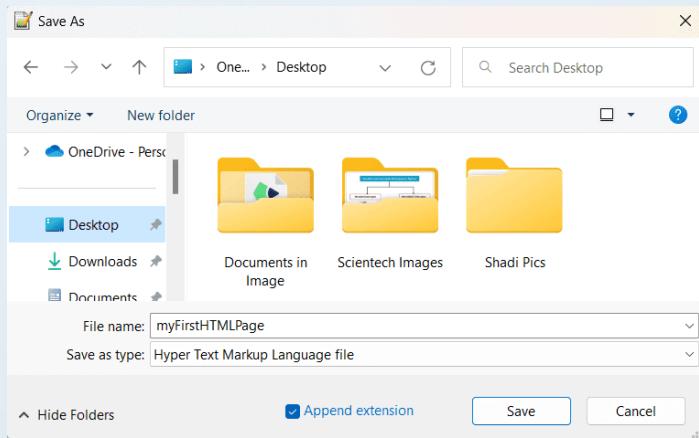
    <body>
        <h1>Welcome to my website!</h1>
        <p>This is my first HTML web page.</p>
    </body>

</html>
```

Create a webpage with HTML using Notepad.

Step 3: After writing the above HTML code, you need to save it with a .html extension. To save file, navigate File menu -> Save. A Save As dialog box will open on your computer system.

Step 4: Enter a name for your file name “myFirstHTMLPage” and Save as type: select extension “*.html; *.htm;” option and then click on the save button as shown in the below screenshot. Save your HTML file to a location on your computer where you can easily find it.



Create a webpage with HTML using Notepad.

Note that when you will save the file for the first time, by default, Notepad adds a .txt extension to the document name.

Step 5: To view the output of the above code snippet, go to the location where you saved the HTML file. I have saved “myFirstHTMLPage” html file on the desktop.

Writing HTML code with Sublime Text Editor

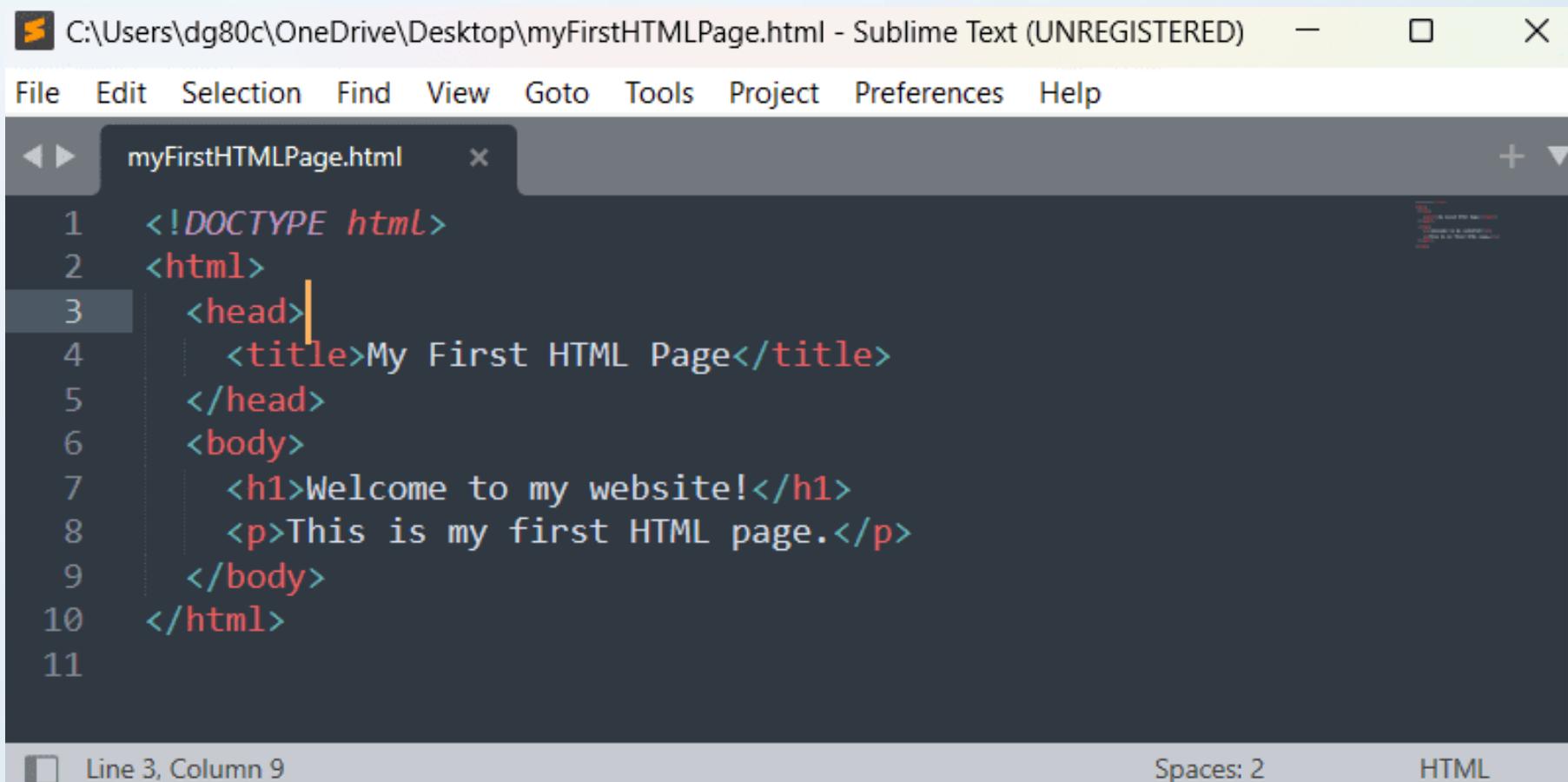
- To work with Sublime Text editors, first you need to download and install from the internet on your computer. To download Sublime Text editor for Windows 10, go to the link: [Download Sublime Text Editor](#)
- Once your download is finished, you need to install it on your computer system. After the complete installation of Sublime text editor, you need to follow the simple steps to use it. The steps are as:

Step 1: To open Sublime Text editor, navigate to Start screen → type Sublime Text → Open it. To open a new page, press CTRL+N.

Step 2: To save your page in Sublime Text editor, press Ctrl+S or go to File option → save. Use .html or .htm extension to save your file. We recommend saving the file first, then write the code because sublime editor will give you suggestions to write code after saving the page.

Writing HTML code with Sublime Text Editor

- **Step 3:** Write the following code in Sublime Text editor.



The screenshot shows the Sublime Text editor interface with the following details:

- Title Bar:** C:\Users\dg80c\OneDrive\Desktop\myFirstHTMLPage.html - Sublime Text (UNREGISTERED)
- Menu Bar:** File Edit Selection Find View Goto Tools Project Preferences Help
- File List:** myFirstHTMLPage.html
- Code Area:** Displays the following HTML code:

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>My First HTML Page</title>
5   </head>
6   <body>
7     <h1>Welcome to my website!</h1>
8     <p>This is my first HTML page.</p>
9   </body>
10 </html>
11
```
- Status Bar:** Line 3, Column 9, Spaces: 2, HTML

Writing HTML code with Sublime Text Editor

- **Step 4:** To execute written HTML code or see the output this code, just right click by mouse on sublime text page and click on “Open in Browser” open. The output of the HTML code will appear in the default web browser.



Types of HTML Tags

There are mainly two types of tags available in the HTML. They are as:

- **Paired tag:** This tag has both start and end tags. For example, `` and `` tags are the paired tags.
- **Empty tag:** This tag does not require to be closed. It has only start tag. For example, `
` tag is an empty tag.

Types Of Elements:

Block Level: Block elements are meant to structure the main parts of your page, by dividing your content in coherent blocks with full width.

- paragraphs <p>
- lists: UL and LI
- headings <h1> to <h6>, articles <article>
- sections <section>
- long quotes <blockquote>
- Division <div>
- Form <form>
- Table <table>

Types Of Elements:

Inline Level: Inline elements are meant to differentiate part of a text, to give it a particular function or meaning. Inline elements usually comprise a single or few words.

- links <a>
- image
- span
- button <button>
- input <input>
- label <label>
- textarea <textarea>
- emphasised words
- important words
- short quotes <q>
- Strong

Example

Block-level

This is a paragraph.

This is another paragraph.

Paragraphs are block-level elements, so they stack vertically.

Inline

Links are

inline elements,

so they fit side-by-side.

Example Code: Block Element

```
<!DOCTYPE html>

<html>
  <body>
    <head>
      <title>Block Element</title>
    </head>
    <h1>Hello</h1>
    <p>This para will be displayed on new line because it is a block level element</p>
  </body>
</html>
```

Hello

This para will be displayed on new line because it is a block level element

Example Code: Inline Element

```
<!DOCTYPE html>
<html>
<body>
<head>
<title>Inline Element</title>
</head>
<span>Hello !!!</span>
<span>This text be displayed on same line because it is a inline element</span>
<a href="https://www.google.com">Hello</a>
<a href="https://www.google.com">Hello</a>
</body>
</html>
```

Hello !!! This text be displayed on same line because it is a inline element [Hello](#) [Hello](#)

BLOCK-LEVEL ELEMENTS

<address>	<article>	<aside>	<blockquote>	<canvas>	<dd>	<div>
<dl>	<dt>	<fieldset>	<figcaption>	<figure>	<footer>	<form>
<h1>-<h6>	<header>	<hr>		<main>	<nav>	<noscript>
	<p>	<pre>	<section>	<table>	<tfoot>	
<video>						

INLINE ELEMENTS

<a>	<abbr>	<acronym>		<bdo>	<big>	
<button>	<cite>	<code>	<dfn>		<i>	
<input>	<kbd>	<label>	<map>	<object>	<output>	<q>
<samp>	<script>	<select>	<small>			<sub>
<sup>	<textarea>	<time>	<tt>	<var>		

Commonly used Basic Tags

Here are some of the commonly used basic tags in HTML that you should keep it mind.

- <html>
- <head>
- <title>
- <body>
- <h1> to <h6>
- <p>
-

- <hr>
- <!--...-->

Commonly used Char Tags

Here are some of the commonly used char tags in HTML that you should keep it mind.

-
- <i>
-
- <big>
-
- <small>
- <sup>
- <sub>
- <bdo>
- <u>

Commonly used Output Tags

The most commonly used output tags are as:

- <pre>
- <code>
- <kbd>
- <var>
- <dfn>
- <samp>

Commonly used Block Tags

The most commonly used block tags are as:

- <acronym>
- <abbr>
- <address>
- <blockquote>
- <q>
- <cite>
- <ins>
-
- <s>
- <a>

Commonly used Input, List, Image, and Table Tags

The most commonly used input tags are as:

- <form>
- <input>
- <button>
- <select>
- <optgroup>
- <option>
- <label>

List tags:

-
-
-
- <dir>
- <dl>
- <dt>
- <dd>
- <menu>

Image tags:

-
- <map>
- <area>

Table tags:

- <table>
- <caption>
- <th>
- <tr>
- <td>
- <thead>
- <tbody>

HTML Comments

- HTML has a mechanism to write comments in the code. Browsers ignore comments, effectively making comments invisible to the user.
- The purpose of comments is to allow you to include notes in the code to explain your logic or coding.
- This is very useful if you return to a code base after being away for long enough that you don't completely remember it.
- To write an HTML comment, wrap it in the special markers <!-- and -->. For example:

```
<p>I'm not inside a comment</p>  
<!-- <p>I am!</p> -->
```

Entity references: Including special characters in HTML

- In HTML, the characters <, >, ',' and & are special characters. They are parts of the HTML syntax itself. So how do you include one of these special characters in your text? For example, if you want to use an ampersand(&) or less-than sign(<), and not have it interpreted as code.
- You do this with character references. These are special codes that represent characters, to be used in these exact circumstances. Each character reference starts with an ampersand (&), and ends with a semicolon (;).

Entity references: Including special characters in HTML

- The character reference equivalent could be easily remembered because the text it uses can be seen as less than for '<', quotation for ' "' and similarly for others.
- <https://www.htmlquick.com/reference/character-entity-reference.html>

Literal character

<

>

"

'

&

Character reference equivalent

<

>

"

'

&

Entity references: Including special characters in HTML:Example

<p>In HTML, you define a paragraph using the <p> element.</p>

<p>In HTML, you define a paragraph using the <p> element.</p>

- In the above code, you can see that the first paragraph has gone wrong. The browser interprets the second instance of <p> as starting a new paragraph. The second paragraph looks fine because it has angle brackets with character references.

In HTML, you define a paragraph using the
element.

In HTML, you define a paragraph using the <p> element.

Paragraph Element

- The `<p>` HTML element represents a paragraph.
- Paragraphs are usually represented in visual media as blocks of text separated from adjacent blocks by blank lines and/or first-line indentation, but HTML paragraphs can be any structural grouping of related content, such as images or form fields.

`<p>`Geckos are a group of usually small, usually nocturnal lizards. They are found on every continent except Antarctica.`</p>`

`<p>`Some species live in houses where they hunt insects attracted by artificial light.`</p>`

Geckos are a group of usually small, usually nocturnal lizards. They are found on every continent except Antarctica.

Some species live in houses where they hunt insects attracted by artificial light.

Paragraph Element

Attribute : This element only includes the global attributes.

<p>

This is the first paragraph of text. This is the first paragraph of text. This is the first paragraph of text. This is the first paragraph of text.

</p>

<p>

This is the second paragraph. This is the second paragraph. This is the second paragraph. This is the second paragraph.

</p>

This is the first paragraph of text. This is the first paragraph of text. This is the first paragraph of text. This is the first paragraph of text.

This is the second paragraph. This is the second paragraph. This is the second paragraph. This is the second paragraph.

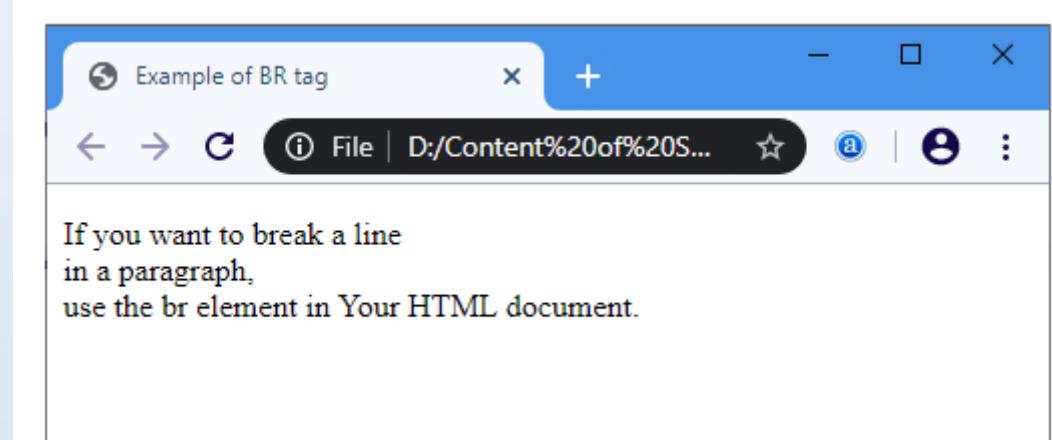
HTML
 tag

- The **
 tag** in **HTML** document is used to create a line break in a text.
- It is generally used in poem or address where the division of line is necessary.
- It is an **empty tag**, which means it does not need a company of end tag. If you place the
 tag in the HTML code, then it works the same as pressing the enter key in a word processor.

Difference between HTML
 and

- You can use HTML br tag two ways:
 or
. It is recommended to use closed br tag
 because it is supported in HTML and XHTML both.

```
<!DOCTYPE HTML>
<html>
<head>
<title>
Example of BR tag
</title>
</head>
<body>
<p>If you want to break a line <br> in a paragraph, <br> use the BR element in <br> your HTML do
</body>
</html>
```



HTML tag (Not Supported in HTML5)

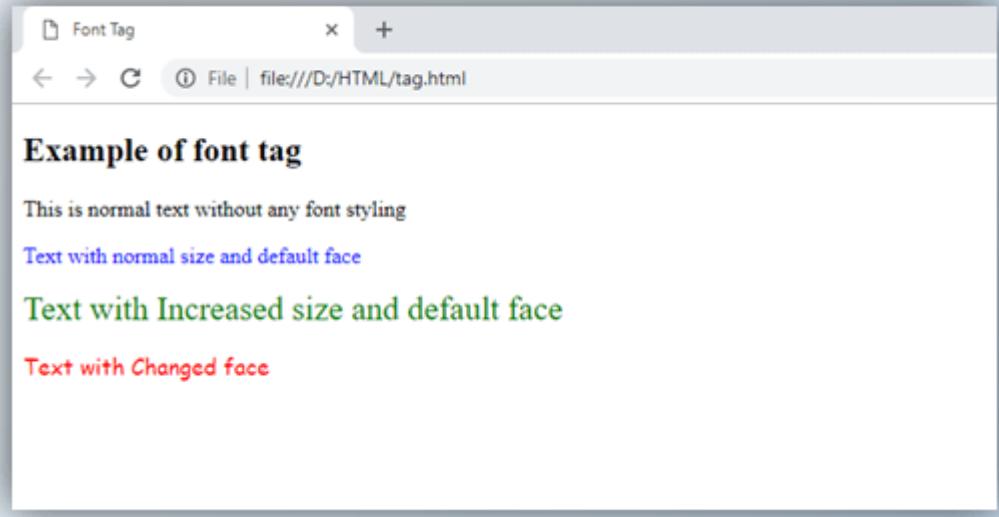
- HTML tag is used to define the font style for the text contained within it. It defines the font size, color, and face of the text in an HTML document.
- Syntax: Content....
- Tag-specific attribute

Attribute	Value	Description
color	rgb(X,X,X) #xxxxxx color_name	It specifies the color of the content. (Not Supported in HTML5)
face	font_family	It specifies the typeface of the content. (Not Supported in HTML5)
size	number	It specifies the size of the content. (Not Supported in HTML5)

HTML tag (Not Supported in HTML5)

```
<!DOCTYPE html>

<html>
<head>
<title>Font Tag </title>
</head>
<body>
<h2>Example of font tag</h2>
<p>This is normal text without any font styling</p>
<p>
<font color="blue">Text with normal size and default face</font>
</p>
<p>
<font size="5" color="green">Text with Increased size and default face</font>
</p>
<p>
<font color="red" face="cursive">Text with Changed face</font>
</p>
</body>
</html>
```



Structure Tags

- There are a group of HTML elements that are created to broadly give our documents more structure. You've already used the first section element:

Body

- The body element contains all of the content of our document.
- This typically encompasses everything except the meta element and its sub elements.

Structure Tags

Div Tag

The HTML division tag, called "div" for short, is a special element that lets you group similar sets of content together on a web page. You can use it as a generic container for associating similar content.

The div tag is one of the most used tags and doesn't seem to be going anywhere despite the introduction of semantic elements (these elements let you use several tags as a container).

When to Use the div Tag

The div tag is multi-purpose – you can use it to do several things on a web page. You'll mostly use it in web layouts and CSS art, but it's super flexible.

Example

```
<div class="square"></div>
```

Structure Tags

Div Tag

The HTML division tag, called "div" for short, is a special element that lets you group similar sets of content together on a web page. You can use it as a generic container for associating similar content.

The div tag is one of the most used tags and doesn't seem to be going anywhere despite the introduction of semantic elements (these elements let you use several tags as a container).

When to Use the div Tag

The div tag is multi-purpose – you can use it to do several things on a web page. You'll mostly use it in web layouts and CSS art, but it's super flexible.

Example

```
<div class="square"></div>
```

Structure Tags-Semantics Tag

The `<header>` tag

The `<header>` element is used as a container for page heading content. The tag typically contains heading tags (`<h1>` - `<h6>`), a logo image, or other content that introduces the content. It is possible to have more than one `<header>` tag per page; however, the `<header>` cannot be coded within a `<footer>` or another `<header>` tag.

The `<nav>` tag

The `<nav>` tag is a block level element used to define major blocks of navigation links or navigation menus.

The `<main>` tag

The `<main>` tag is used to serve a container for major page content that is not repeated in other parts of the page. A web page can include only one `<main>` element. Additionally, it cannot be included as a child element within a `<header>`, `<nav>`, `<footer>`, `<article>`, or `<aside>` tag.

Structure Tags-Semantics Tag

The **<section>** tag

The **<section>** tag defines generic sections in a web page such as headers, footers, or any other sections of the document. In this context, a section is a thematic grouping of content.

The **<article>** tag

The **<article>** element represents a complete, or self-contained, composition in a document, page, application, or site that is, in principle, independently distributable or reusable.

The **<aside>** tag

The **<aside>** tag is a block level element that defines content aside from the content it is placed in. The aside content should be related to the surrounding content.

Structure Tags-Semantics Tag

The **<footer>** tag

The **<footer>** tag is a block level element that defines footer information for an entire web page or section of the document. The content of the **<footer>** typically consists of contact information, copyright, links, or logos. In some cases, a web page may contain multiple **<footer>** tags.

The **<figure>** tag

The **<figure>** tag specifies self-contained content, like illustrations, diagrams, photos, or code sections. By default, this is a block level element with 40px right and left margin settings.

The **<figcaption>** tag

The **<figcaption>** is a block level tag that defines a caption for a **<figure>** element discussed in the previous section. The **<figcaption>** tag is normally coded as the first or last child element of the **<figure>** tag.

Formatting Tags

HTML provides us with several tags to help format text, **** for bold, **<i>** for italic, **<u>** for underlining, etc. These tags are divided into two categories:

HTML Formatting tags

- ** tag - Bold Text
- <i>** tag - Italic Text
- <u>** tag - Underlined Text
- ** tag - Strong Text
- ** tag - Emphasized Text
- <mark>** tag - Highlighted Text
- <sup>** tag - Superscript Text
- <sub>** tag - Subscript Text
- ** tag - Deleted Text
- <ins>** tag - Inserted Text
- <big>** tag - Big Text
- <small>** tag - Small Text

Formatting Tags

Element name	Description
	This is a physical tag, which is used to bold the text written between it.
	This is a logical tag, which tells the browser that the text is important.
<i>	This is a physical tag which is used to make text italic.
	This is a logical tag which is used to display content in italic.
<mark>	This tag is used to highlight text.
<u>	This tag is used to underline text written between it.
<tt>	This tag is used to appear a text in teletype. (not supported in HTML5)
<strike>	This tag is used to draw a strikethrough on a section of text. (Not supported in HTML5)
<sup>	It displays the content slightly above the normal line.
<sub>	It displays the content slightly below the normal line.
	This tag is used to display the deleted content.
<ins>	This tag displays the content which is added
<big>	This tag is used to increase the font size by one conventional unit.
<small>	This tag is used to decrease the font size by one unit from base font size.

Formatting Tags

Bold Text

- HTML and formatting elements
- The HTML element is a physical tag which displays text in bold font, without any logical importance. If you write anything within element, it is shown in bold letters.

<p> Write Your First Paragraph in bold text.</p>

Output:

This is an important content, and this is normal content

Formatting Tags

Italic Text

- **HTML <i> and formatting elements**
- The HTML <i> element is physical element, which display the enclosed content in italic font, without any added importance. If you write anything within <i>.....</i> element, is shown in italic letters.

<p> <i>Write Your First Paragraph in italic text.</i></p>

Output:

Write Your First Paragraph in italic text.

Formatting Tags

Italic Text

- **HTML <i> and formatting elements**
- The HTML <i> element is physical element, which display the enclosed content in italic font, without any added importance. If you write anything within <i>.....</i> element, is shown in italic letters.

<p> <i>Write Your First Paragraph in italic text.</i></p>

<p>This is an important content, which displayed in italic font.</p>

Output:

Write Your First Paragraph in italic text.

This is an important content, which displayed in italic font.

Formatting Tags

HTML Marked formatting

- If you want to mark or highlight a text, you should write the content within <mark>.....</mark>.

<h2> I want to put a <mark> Mark</mark> on your face</h2>

I want to put a **Mark** on your face

Formatting Tags

Underlined Text

- If you write anything within `<u>.....</u>` element, is shown in underlined text.

`<p> <u>Write Your First Paragraph in underlined text.</u></p>`

Write Your First Paragraph in underlined text.

Strike Text

- Anything written within `<strike>.....</strike>` element is displayed with strikethrough. It is a thin line which cross the statement.

`<p> <strike>Write Your First Paragraph with strikethrough</strike>.</p>`

Write Your First Paragraph with strikethrough.

Formatting Tags

Monospaced Font

- If you want that each letter has the same width then you should write the content within <tt>.....</tt> element.

<p>Hello <tt>Write Your First Paragraph in monospaced font.</tt></p>

Hello Write Your First Paragraph in monospaced font.

Superscript Text

- If you put the content within ^{^{.....}} element, is shown in superscript; means it is displayed half a character's height above the other characters.

<p>Hello ^{Write Your First Paragraph in superscript.}</p>

Hello Write Your First Paragraph in superscript.

Formatting Tags

Subscript Text

- If you put the content within `_{.....}` element, is shown in subscript ; means it is displayed half a character's height below the other characters.

`<p>Hello _{Write Your First Paragraph in subscript.}</p>`

Hello Write Your First Paragraph in subscript.

Deleted Text

- Anything that puts within `.....` is displayed as deleted text.

`<p>Hello Delete your first paragraph.</p>`

Hello Delete your first paragraph.

Formatting Tags

Inserted Text

- Anything that puts within <ins>.....</ins> is displayed as inserted text.

Delete your first paragraph.<ins>Write another paragraph.</ins></p>

Delete your first paragraph. Write another paragraph.

Larger Text

- If you want to put your font size larger than the rest of the text then put the content within <big>.....</big>. It increase one font size larger than the previous one.

Hello Write the paragraph in larger font.

<p>Hello <big>Write the paragraph in larger font.</big></p>

Formatting Tags

Smaller Text

- If you want to put your font size smaller than the rest of the text then put the content within `<small>.....</small>`tag. It reduces one font size than the previous one.

```
<p>Hello <small>Write the paragraph in smaller font.</small></p>
```

Hello Write the paragraph in smaller font.

HTML Heading

- A HTML heading or HTML **hn** tag can be defined as a title or a subtitle which you want to display on the webpage. When you place the text within the heading tags `<h1>.....</h1>`, it is displayed on the browser in the bold format and size of the text depends on the number of heading.
- There are six different HTML headings which are defined with the `<h1>` to `<h6>` tags, from highest level h1 (main heading) to the least level h6 (least important heading).
- h1 is the largest heading tag and h6 is the smallest one. So h1 is used for most important heading and h6 is used for least important.

HTML Heading

<h1>Heading no. 1</h1>

<h2>Heading no. 2</h2>

<h3>Heading no. 3</h3>

<h4>Heading no. 4</h4>

<h5>Heading no. 5</h5>

<h6>Heading no. 6</h6>

Heading no. 1

Heading no. 2

Heading no. 3

Heading no. 4

Heading no. 5

Heading no. 6

HTML Heading

<h1>Heading no. 1</h1>

<h2>Heading no. 2</h2>

<h3>Heading no. 3</h3>

<h4>Heading no. 4</h4>

<h5>Heading no. 5</h5>

<h6>Heading no. 6</h6>

Heading no. 1

Heading no. 2

Heading no. 3

Heading no. 4

Heading no. 5

Heading no. 6

HTML Phrase tag

The HTML phrase tags are special purpose tags, which defines the structural meaning of a block of text or semantics of text.

- Abbreviation tag : <abbr>
- Acronym tag: <acronym> (not supported in HTML5)
- Marked tag: <mark>
- Strong tag:
- Emphasized tag :
- Definition tag: <dfn>
- Quoting tag: <blockquote>
- Short quote tag : <q>
- Code tag: <code>
- Keyboard tag: <kbd>
- Address tag: <address>

HTML Phrase tag

Text Abbreviation tag

- This tag is used to abbreviate a text. To abbreviate a text, write text between `<abbr>` and `</abbr>` tag.

```
<p>An <abbr title = "Hypertext Markup language">HTML </abbr>language is used to create web pages. </p>
```

An HTML language is used to create web pages.

Code tags:

- An HTML `<code> </code>` element is used to display the part of computer code.

```
<p><code>
class Simple{
    public static void main(String args[]){
        System.out.println("Hello Java");
    }
}</code> </p>
```

Code tags

```
class Simple{ public static void main(String
args[]){ System.out.println("Hello Java"); }}
```

HTML Phrase tag

Keyboard Tag:

- The keyboard tag is, <kbd>, and it indicates that a section of content is a user input from the keyboard.

<p>Please press <kbd>Ctrl</kbd> + <kbd>Shift</kbd> + t<kbd></kbd> to restore page on chrome.</p>

Keyboard input.

Please press Ctrl + Shift + t to restore page on chrome.

Address tag:

The <address> tag provides the contact information about the author of the content.

<address> You can ask your queries by contact us on test@gmail.com

 visit at:
ktm - 44066.

</address>

Address Tag

*You can ask your queries by contact us on
i2tutorials@gmail.com
visit at:
hyderabad - 300038.*

CSS(Cascading Stylesheet)

- CSS (Cascading Stylesheet) is used to apply the style on the web page which is made up of HTML elements. It creates the look of the webpage.
- CSS provides different style properties such as background color, padding, margin, border-color, etc, to style a webpage.
- **Three different ways to apply CSS (Types)**
 - Inline CSS
 - Internal or Embedded CSS
 - External CSS

CSS- Inline CSS

- Inline-styles are used to apply CSS in a single element. It can apply style uniquely in each element.
- To apply inline CSS, you need to use **style** attributes within the HTML element. We can use as many properties as we want, but each property should be separated by a semicolon (;).

```
<h3 style="color: red;  
          font-style: italic;  
          text-align: center;  
          font-size: 50px;  
          padding-top: 25px;">Learning HTML using Inline Styles</h3>
```

Learning HTML using Inline Styles

CSS- Internal CSS

- An Internal or Embedded stylesheets contains the CSS properties for a webpage in <head> section of HTML document. To use Internal styles (CSS), we can use class and id attributes(use different selector in CSS).

CSS- Internal CSS

```
<head>
    <style>
        /*Internal CSS using element name*/
        body{background-color:lavender;
        text-align: center;}
        h2{font-style: italic; font-size: 30px;
        color: #f08080;}
        p{font-size: 20px;}
```

```
/*Internal CSS using class
name*/
.blue{color: blue;}
.red{color: red;}
.green{color: green;}
</style>
</head>
```

CSS- Internal CSS

```
<body>  
<h2>Learning HTML with internal CSS</h2>  
<p class="blue">This is a blue color paragraph</p>  
<p class="red">This is a red color paragraph</p>  
<p class="green">This is a green color paragraph</p>  
</body>
```



CSS- External CSS

External CSS:

An external style sheet holds all the style rules in a separate document (file) that you can link from any HTML document on your site.

There are two files that need to create to apply external CSS

- Create a CSS document (file) and save it using the .css extension.
- Link the CSS file in your HTML file using the tag in the header section of the HTML document.

CSS- External CSS

```
<!-- Inside head-->

<head>

    <link rel="stylesheet" type="text/css" href="css/style.css">

</head>

<!-- Inside body-->

<body>

    <h2>Learning HTML with External CSS</h2>

    <p class="blue">This is a blue color paragraph</p>

    <p class="red">This is a red color paragraph</p>

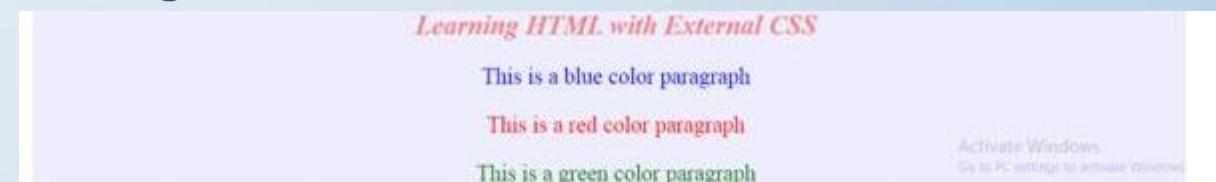
    <p class="green">This is a green color paragraph</p>

</body>
```

CSS- External CSS

```
body{  
background-color:lavender;  
text-align: center;  
}  
  
h2{  
font-style: italic;  
size: 30px;  
color: #f08080;  
}  
2/22/2024
```

```
p{  
font-size: 20px;  
}  
.blue{  
color: blue;  
}  
.red{  
color: red;  
}  
.green{  
color: green;  
}
```



CSS Selector

- To apply CSS to an element you need to select it. CSS provides you with a number of different ways to do this
- **CSS selectors** are used to select the content you want to style. Selectors are the part of CSS rule set. CSS selectors select HTML elements according to its id, class, type, attribute etc.
 - Simple CSS selectors
 - Attribute selectors
 - Grouping CSS selector
 - CSS Combinators
 - Pseudo-class selectors
 - Pseudo-element selectors

Simple CSS Selectors

- Selectors allow you to target and select specific parts of your document for styling purposes.
- Simple selectors directly select one or more elements:
 - By using the universal selector, *.
 - Based on the name/type of the element.
 - Based on the class value of the element.
 - Based on the ID value of the element.

CSS Universal Selector

- The universal selector, also known as a wildcard, selects everything - every single element in the document.

To use the universal selector, use the asterisk character, *.

```
* {  
    property: value;  
}
```

You can use the universal selector to reset the browser's default padding and margin to zero at the top of the file before you add any other styles:

```
* {  
    padding: 0;  
    margin: 0;  
}
```

CSS Type Selector

- The CSS type selector selects all HTML elements of the specified type.
- To use it, mention the name of the HTML element.
- For example, if you wanted to apply a style to every single paragraph in the HTML document, you would specify the p element:

```
p {  
    property: value;  
}
```

- The code above matches and selects all p elements within the document and styles them.

CSS Class Selector

- The class selector matches and selects HTML elements based on the value of their given class. Specifically, it selects every single element in the document with that specific class name.
- With the class selector, you can select multiple elements at once and style them the same way without copying and pasting the same styles for each one separately.
- Classes are reusable, making them a good option for practicing DRY development. DRY is a programming principle and is short for '**'Don't Repeat Yourself'**'. As the name suggests, the aim is to avoid writing repetitive code whenever possible.

CSS Class Selector

- To select elements with the class selector, use the dot character, ., followed by the name of the class.

```
.my_class {  
    property: value;  
}
```

- In the code above, elements with a class of my_class are selected and styled accordingly.

```
.pastoral { color: green } /* all elements with class~=pastoral */
```

CSS ID Selector

- The ID selector selects an HTML element based on the value of its ID attribute.
- Keep in mind that the ID of an element should be unique in a document, meaning there should only be one HTML element with that given ID value. You cannot use the same ID value on a different element besides that one.
- To select an element with a specific ID, use the hash character, #, followed by the name of the ID value:

```
#my_id {  
    property: value;  
}
```

The code above will match only the unique element with the ID value of my_id.

CSS ID Selector

It's worth mentioning that it is best to try and limit the use of this selector and opt for using the class selector instead. Applying styles using the ID selector is not ideal because the styles are not reusable.

CSS

```
#rad {  
border: 1px solid blue;  
}
```

HTML

```
<div id="rad"></div>
```

CSS Attribute Selectors

The [attribute] Selector

- To use the attribute selector, use a pair of square brackets, [], to select the attribute you want.
- The general syntax for attribute selectors is the following:

element[attribute]

- This selector selects an element if the given attribute exists.

In the following example, elements that have the attribute attr present are selected, regardless of the specific value of attr:

```
a[attr] {  
    property: value;
```

```
}
```

CSS Attribute Selectors

```
a[attr] {  
    property: value;  
}
```

- In the example above, a elements with the attr attribute name are selected, regardless of the value of attr.
- You can be more specific with your styling.
- The [attribute="value"] Selector
- You can specify the value of the attribute using the following syntax:

element[attribute="value"]

CSS Attribute Selectors

- So, if you want to style a elements with an attr attribute that has an exact value of 1, you would do the following:

```
a[attr="1"] {  
    property: value;  
}
```

- This code above matches a elements where the attr attribute name has an exact value of 1.

CSS Attribute Selectors

The [attribute^="value"] Selector

You could also specify that the value of the attribute starts with a specific character using the following syntax:

```
element[attribute^="value"]
```

For example, if you wanted to select and style any a elements that have an attr attribute with a value that starts with www, you would do the following:

```
a[attr^="www"] {  
    property: value;  
}
```

The code above selects any a elements where the attr attribute name has a value that starts with www.

CSS Attribute Selectors

The [attribute\$="value"] Selector

You could also specify that the value of the attribute ends with a specific character using the following syntax:

```
element[attribute$="value"]
```

For example, if you wanted to select a elements that have an attr attribute name with a value that ends with .com, you would do the following:

```
a[attr$=".com"] {  
    property: value;  
}
```

CSS Attribute Selectors

The [attribute*="value"] Selector

You can also specify that the attribute value contains a specific substring - this selector is known as the Attribute Contains **Substring Selector** and has the following syntax:

element[attribute*="value"]

In this case, the string value needs to be present in the attribute's value followed by any number of other characters - value doesn't need to be a whole word.

For example, if you wanted to select elements that have an attr attribute with a value that contains the string free, you would do the following:

```
a[attr*="free"] {  
    property:value;
```

CSS Attribute Selectors

The code above selects a elements with an attr attribute name when the string free is present in the attribute's value - even as a substring (a substring is a word inside another word).

As long as the attribute's value contains free, then the HTML element is selected - this could match an attr attribute with a value of free, freeTest, or freediving, for example.

CSS Attribute Selectors

The [attribute~="value"] Selector

You can also specify that the selector matches an attribute value that contains a **whole word** using the following syntax:

```
element[attribute~="value"]
```

In this case, the string value needs to be a whole word.

For example, if you wanted to select a elements that have an attr attribute name with a value that contains the word free, you would do the following:

```
a[attr~="free"] {  
    property: value;  
}
```

The code above would match an attr attribute with a value of free that contains different kinds of whitespace

CSS Attribute Selectors

The code above would match an attr attribute with a value of free that contains different kinds of whitespace.

The code wouldn't select elements with an attr value of freeTestCode or freediving like you saw in an earlier example because free needs to be a whole word on its own - not a substring.

CSS Group Selectors

- With the grouping selector, you can target and style more than one element at once.
- To use the grouping selector, use a comma, , to group and separate the different elements you want to select.
- For example, here is how you would target multiple elements such as divs, ps, and spans all at once and apply the same styles to each of them:

```
div, p, span {  
    property: value;  
}
```

- The code above matches all div, p, and span elements on the page, and those three elements will share the same styling.

CSS Combinator Selectors

- Combinators allow you to combine two elements based on the relationship between the elements and their location in the document.
- Essentially, you can combine two simple selectors in a way that explains the relationship between those CSS selectors.
Combinators are a type of selector that specifies and describes the relationship between the two selectors.
- There are four types of combinators:
 - The descendant combinator.
 - The direct child combinator.
 - The general sibling combinator.
 - The adjacent sibling combinator

Descendant Selector/Combinator

- As the name implies, the descendant combinator selects only the descendants of the specified element.
- Essentially, you first mention the parent element, leave a space, and then mention the descendant of the first element, which is the child element of the parent. The child element is an element inside the parent element.

```
<body>
  <div>
    <h2>I am level 2 heading</h2>
    <p>I am a paragraph inside a div</p>
    <span>I am a span</span>
    <p>I am a paragraph inside a div</p>
  </div>
  <p>I am a paragraph outside a div</p>
</body>
```

```
div p {
  property: style;
}
```

Direct Child Combinator

- The direct child combinator, also known as the direct descendant, selects only the direct children of the parent.
- To use the direct child combinator, specify the parent element, then add the > character followed by the direct children of the parent element you want to select.

```
<body>
  <div>
    <a href="#">I am a link</a>
    <a href="#">I am a link</a>
    <p><a href="#">I am a link inside a paragraph</a></p>
  </div>
</body>
```

```
div > a {
  property: value;
}
```

Direct Child Combinator

- There is a div which is the parent element.
- Inside the parent element, there are two a elements which are the direct children of the div.
- There is also another a element inside a p element. The p element is a child of the div, but the a element inside the paragraph is not a direct child of the div.
- So, to access only the a elements that are direct children of div, you would do the following:
- The code above matches the a elements directly nested inside the div element and are immediate children.

General Sibling Combinator

- The general sibling combinator selects siblings.
- You can specify the first element and a second one that comes after the first one. The second element doesn't need to come right after the first one.
- To use the general sibling combinator, specify the first element, then use the ~ character followed by the second element that needs to follow the first one.

```
<body>
  <div>
    <p>I am paragraph inside a div</p>
  </div>
  <p>I am a paragraph outside a div</p>
  <h3>I am a level three heading</h3>
  <p>I am a paragraph outside a div</p>
</body>
...>
```

General Sibling Combinator

The div has a p element nested inside it. That specific p element is a child of div.

- There are also two paragraphs with the text I am a paragraph outside a div and an h3 element. All those three elements are *siblings* of the div element.
- So, to select the p elements that are *siblings* of the div element, you would do the following:

```
div ~ p {  
    property: value;  
}
```

- The code above styles both p elements that come after the div.
- It styles even the p element that does not come directly after the div element, such as the p that follows the h3 element. It does so because it still comes after div.

Adjacent Sibling Combinator

- The adjacent sibling combinator is more specific than the general sibling combinator.
- This selector matches only the immediate siblings. Immediate siblings are the siblings that come right after the first element.
- To use the adjacent sibling combinator, specify the first element, then add the + character followed by the element you want to select that immediately follows the first element.

```
<body>
  <div>
    <p>I am paragraph inside a div</p>
  </div>
  <p>I am a paragraph outside a div</p>
  <h3>I am a level three heading</h3>
  <p>I am a paragraph outside a div</p>
</body>
```

```
div + p {
  property: value;
}
```

Adjacent Sibling Combinator

- Although the p element that follows the h3 element is a sibling of the div element, it is not a direct sibling like the p element that comes before the h3.
- So, to target only the p element that comes directly after the div, you would do the following:

```
<body>
  <div>
    <p>I am paragraph inside a div</p>
  </div>
  <p>I am a paragraph outside a div</p>
  <h3>I am a level three heading</h3>
  <p>I am a paragraph outside a div</p>
</body>
```

```
div + p {
  property: value;
}
```

Pseudo-Class Selectors

- Pseudo-class selectors select elements that are in a specific state.
- Some examples of the state the element can be in are:
 - The element is being hovered over by the mouse pointer.
 - The element is the first of its type.
 - The link has been visited before from that specific browser.
 - The link has not been visited before from that specific browser.
 - The checkbox/radio button has been checked.
- Pseudo-class selectors start with a colon, :, followed by a keyword that reflects the state of the specified element.

Pseudo-Class Selectors

- The general syntax looks something like the following:

```
element:pseudo-class-name {
```

```
    property: value;
```

```
}
```

- Pseudo-Class Selectors for Links**

- There are various selectors based on link state information.

- The :link selector applies styling when the element has not been visited before:

```
a:link {
```

```
    property: value;
```

```
}
```

Pseudo-Class Selectors

- The :visited selector applies when the element has been visited before in the current browser:

```
a:visited {  
    property: value;  
}
```

- The :hover selector applies when the mouse pointer hovers over an element:

```
a:hover {  
    property: value;  
}
```

Pseudo-Class Selectors

- The :focus selector applies when a user has tabbed onto an element:

```
a:focus {  
    property:value;  
}
```

- The :active selector applies when the element is selected after being clicked on and after holding down a mouse button:

```
a:active {  
    property: value;  
}
```

Pseudo-Class Selectors

For input

```
input:focus {  
    property: value;  
}  
  
input:required {  
    property: value;  
}  
  
input:checked {  
    property: value;  
}  
  
input:disabled {  
    property: value;  
}
```

Pseudo-Class Selectors for Position

```
a:first-child {  
    property: value;  
}  
a:last-child {  
    property: value;  
}  
a:nth-child(n) {  
    property: value;  
}  
a:nth-child(even) {  
    property: value;  
}  
p:first-of-type {  
    property: value;  
}  
p:last-of-type {  
    property: value;  
}
```

Pseudo-Element Selectors

- Pseudo-element selectors are used for styling a specific part of an element - you can use them to insert new content or change the look of a specific section of the content.
- For example, you can use a pseudo-element to style the first letter or the first line of an element differently. You can also use pseudo-elements to add new content before or after the selected element.
- In contrast to pseudo-classes that are preceded by a : character, pseudo-elements are preceded by a :: character.
- The :: character is followed by a keyword that allows you to style a specific part of the selected element.

Pseudo-Element Selectors

- The general syntax looks something like the following:

```
element::pseudo-element-selector {  
    property:value;  
}
```

- Make sure to use the :: character instead of the : one when using pseudo-element selectors - this will help distinguish pseudo-classes from pseudo-elements.

Pseudo-Element Selectors

The ::before Pseudo-Element

- You can use the ::before pseudo-element to insert content before an element:

```
p::before {  
    property: value;  
}
```

The ::after Pseudo-Element

- And you can use the ::after pseudo-element to insert content at the end of an element:

```
p::after {  
    property: value;  
}
```

Pseudo-Element Selectors

The ::first-letter Pseudo-Element

- You can also use the ::first-letter pseudo-element to select the first letter of a paragraph, which is helpful when you want to style the first letter in a certain way:

```
p::first-letter {  
    property: value;  
}
```

The ::first-line Pseudo-Element

- And you can use the ::first-line pseudo-element to select the first line of a paragraph:

```
p::first-line {  
    property: value;  
}
```

Example: <https://www.educba.com/css-pseudo-elements/>

Commonly used CSS properties:

- background-color – Background color of that element
- color – Color of text of an element
- padding – Space between content and the border
- margin – Space around an element
- font-family – Font for a particular element
- Font-size – Font size for a particular element
- text-align – Align the text in a selected position

How CSS works?

- It is often a curious question, how does CSS interact and alters the HTML. There is a complete mechanism that is followed. The HTML is converted in DOM (Document Object Model) in the web browser.
- DOM is basically a layout of HTML tags to make accessing them sequentially.
- DOM tree enables you to trigger the specific selectors with child components.
- The CSS is loaded and parsed to the DOM tree. This is where the CSS interacts not with the HTML but with the DOM.

It is finally rendered and appears as a complete webpage.

History of CSS

- CSS came first into existence in 1994.
- CSS evolved all these years and W3C maintained the standards with **CSS1, CSS2 and CSS3**.
- They were built on top of each other with better adaptability and more features.
- Each one had issues that were solved by the next standard.
- CSS got major adaptability issues with multiple web browsers coming up all these years starting from Internet Explorer, Opera, Firefox, Chrome and Edge.

History of CSS

CSS 1

- CSS 1 was adopted in 1996. It was difficult and less adapted by then recent browsers such as Internet Explorer 3, Internet Explorer 4 and Netscape 4.x.
- CSS 1 had font properties and specification for typeface and emphasis. The text attributes were supported such as spacing of letters and line of text.
- Alignment of text, positioning and tables were also added. Margin, padding, border and positioning for elements was also implemented. This recommendation was not maintained by W3C.

History of CSS

CSS2

- CSS2 is the improvement of CSS1. It removed the not fully interoperable features. It also included the browser extensions.
- It had many new features such as absolute, relative and fixed positioning of elements. It supported different media types. It also included new font properties such as shadow.

CSS3

- CSS3 is the most recent and currently used. It has XHTML specification. CSS3 has its major focus on modularization and separation of concerns.
- Different modules now go through different stages of the recommendation process. CSS3 has support for almost all recent web browsers.

History of CSS

- It has even included new selectors along with new combinator and new pseudo-elements.
- CSS3 has several new CSS properties. It supports animation which is not a part of earlier recommendations.
- There were various properties added such as transforms, gradients, animation and transition for animation effect in the website.
- Recent add-ons are like border-radius, box-shadow, flex-box and CSS grid.

Major Difference Between CSS1,CSS2,CSS3

- CSS1 had a major difficulty with adaptation and consistency across different web browsers. The number of web browsers has also increased largely. CSS2 still has browser extension issues. CSS3 has complete support for almost all recent web browsers.
- CSS1 had limited styling options and old design influences. The properties and add-ons have increased with CSS2 and further expanded with CSS3. CSS3 has support to add animations to your modern websites.
- CSS3 has compatibility with external font styles through google fonts and typecast. It was not possible with earlier CSS1 and CSS2.
- The selectors in CSS3 has increased while CSS1 and CSS2 only had simple selectors.
- CSS1 AND CSS2 didn't have provision to specifically design the web layout. It is possible with the CSS3 grid system and template layout module. It helped in creating layouts according to user components.
- CSS3 has now been split into various documents known as modules. Previously it was all a single document and limited features. The modularisation helped in working on a particular specification and faster evolution.
- CSS3 is compatible with Internet Explorer 9 whereas CSS1 used to be compatible with Internet Explorer 3.
- CSS3 supports responsive design. Responsive design is referred to as designing a website in such a manner that it looks good on all screen sizes. It should not break or misalign components on changing the screen size.

Advantages of CSS

Better Website Speed

- For a website to function efficiently, it should have a faster load time. In modern times, people usually wait for just a couple of seconds for a website to load. So, it's important to ensure faster speed. For companies wanting to ensure a faster and smooth website experience, CSS becomes paramount to their success.

Easier to Maintain

- CSS is easy to maintain due to less maintenance time. This is because a single line code change affects the entire web page. Also, if improvements are required, then less effort is required to affect changes in the webpage code.

Advantages of CSS

Consistent Design

- You would have seen many websites that are elegant and user-friendly. One thing common to all these websites is consistency in design. CSS enables developers to ensure the style elements are applied consistently across several web pages.

Time-Saving

- Due to faster speed and easier maintenance, CSS saves a lot of time and effort in the web development process due to faster loading time. Here, lesser time ensures designer efficiency.

Advantages of CSS

Better Device Compatibility

- People use different smart devices to view a particular website. It can be a smartphone, PC or laptop. For this purpose, websites are required to be device compatible. CSS ensures the task is done smoothly by providing better compatibility.

Positioning of Design Elements

- You can change the position of an HTML tag with the help of CSS. You can place the elements like an image on any part of the webpage as and when required.

Disadvantages of CSS

Confusion due to many CSS levels

- Beginners are more vulnerable to this issue. They might get confused while opting to learn CSS as there are many levels of CSS such as CSS2, CSS3, etc.

Cross-Browser Issues

- Different browsers work differently. So, you have to check that changes implemented in the website via CSS codes are reflected properly among all browsers.

Security Issues

- Security is important in today's world driven by technology and data. One of the major disadvantages of CSS is that it has limited security.

Disadvantages of CSS

Extra Work for Developers

- Design services are required to consider and test all CSS codes across different browsers for compatibility. Due to developers testing compatibility for different browsers, their workload increases.

Defining Colors in CSS

- **Color** Keywords :Red, green, black, lightgreen, white, yellow
- **RGB** : RGB colors have three values: **red**, **green**, and **blue**
- **RGBA** :it's just like RGB, except with the addition of a fourth value: the **alpha channel**.
- **HSL**: HSL stands for: hue, saturation, and lightness
- **HSLA** : HSLA is simply the HSL color model with the addition of an alpha channel.
- **Hexadecimal** : Hex numbers can be 0-9 and A-F.

Color Keywords

- The first and easiest way to specify a color is using one of the 140 predefined color **keywords** specified in CSS.

Color	Keyword	Hex Value
black	black	#000000
gray	gray	#808080
silver	silver	#c0c0c0
white	white	#ffffff
maroon	maroon	#800000
red	red	#ff0000
purple	purple	#800080
fuchsia	fuchsia	#ff00ff
green	green	#008000
lime	lime	#00ffff
olive	olive	#808000
yellow	yellow	#ffff00
navy	navy	#000080
blue	blue	#0000ff
teal	teal	#008080
aqua	aqua	#0000ff
orange	orange	#ffa500

RGB

- RGB, which stands for red, green, and blue is the color model that monitors use. Since in web design we're primarily concerned with what web pages look like on screens, RGB is the color model we use.
- RGB colors have three values that represent: red, green, and blue
- Each value can be a number between 0 and 255 or a percentage from 0 to 100%
- A value of 0 means none of that color is being used
- A value of 255 or 100% means all of that color is being used
- A 0 for all three color values will be black
- A 255 or 100% for all three color values will be white

RGB Example

- A black paragraph

```
p { color: rgb(0, 0, 0); }      /* black */
```

- A white h1

```
h1 { color: rgb(255, 255, 255); }      /* white */
```

- /* Percentages work too */

```
h1 { color: rgb(100%, 100%, 100%); }      /* white */
```

- A purple unordered list

```
ul { color: rgb(128, 80, 200); }      /* purple */
```

RGBA Example

- The alpha value represents the level of transparency that the rgb color should have. It can be a value from 0 to 1 or a percentage from 0 to 100%. Note that you must specify RGBA instead of RGB.

```
<ul>
  <li style="background-color: rgba(255, 0, 0, 1)"></li>
  <li style="background-color: rgba(255, 0, 0, .9)"></li>
  <li style="background-color: rgba(255, 0, 0, .8)"></li>
  <li style="background-color: rgba(255, 0, 0, .7)"></li>
  <li style="background-color: rgba(255, 0, 0, .6)"></li>
  <li style="background-color: rgba(255, 0, 0, .5)"></li>
  <li style="background-color: rgba(255, 0, 0, .4)"></li>
  <li style="background-color: rgba(255, 0, 0, .3)"></li>
  <li style="background-color: rgba(255, 0, 0, .2)"></li>
  <li style="background-color: rgba(255, 0, 0, .1)"></li>
</ul>
```



RGB Example

#rrggb

- Where rr (red), gg (green) and bb (blue) are hexadecimal integers between 00 and ff, specifying the intensity of the color.
- For example, #ff0000 is displayed as red, because red is set to its highest value (ff), and the other two (green and blue) are set to 00.
- Another example, #00ff00 is displayed as green, because green is set to its highest value (ff), and the other two (red and blue) are set to 00.
- To display black, set all color parameters to 00, like this: #000000.
- To display white, set all color parameters to ff, like this: #ffffff.

Types of Units in CSS

Self Study task: Type of units in CSS

Relative vs Absolute Path

- Absolute paths specify the complete location of a file or directory within a computer's file system, starting from the root directory.
- Relative paths indicate the location of a file or directory about the current working directory without specifying the entire path from the root directory.
- Both absolute and relative paths are used to locate files and directories within a computer's file system, but they differ in their reference points and level of detail provided.

Relative vs Absolute Path

Linux absolute path

```
/home/users/c/computerhope/public_html/cgi-bin
```

Linux relative path

```
./public_html/cgi-bin
```

Internet URL absolute path

```
https://www.computerhope.com/oh.htm
```

Internet URL relative path

```
oh.htm
```

Windows absolute path

```
C:\Windows\calc.exe
```

Windows non absolute path (relative path)

```
calc.exe
```

<a> tag

- An anchor tag is a HTML element that creates a link to a target URL.
- When correctly implemented, the link can wrap around text, images, or as buttons, so that users can interact with it and visit the link's destination.
- It is a best practice to provide context about the link's destination, either in the form of clear anchor text or with a descriptive image.
- This way, website visitors know what to expect, and Search Engine crawlers can understand the connection across both URLs.

<a> tag

Example HTML

```
<a href="https://www.example.com">My sample page </a>
```

- An anchor tag requires the **href** attribute which specifies the URL to be linked to. Other attributes can help improve the link's appearance and behavior, but are not required.

By default, links will appear as follows in all browsers:

- An unvisited link is underlined and blue
- A visited link is underlined and purple
- An active link is underlined and red

Attribute	Value	Description	rel	alternate author bookmark external help license next nofollow noreferrer noopener prev search tag	Specifies the relationship between the current document and the linked document
download	<i>filename</i>	Specifies that the target will be downloaded when a user clicks on the hyperlink			
href	<i>URL</i>	Specifies the URL of the page the link goes to			
hreflang	<i>language_code</i>	Specifies the language of the linked document			
media	<i>media_query</i>	Specifies what media/device the linked document is optimized for			
ping	<i>list_of_URLs</i>	Specifies a space-separated list of URLs to which, when the link is followed, post requests with the body ping will be sent by the browser (in the background). Typically used for tracking.			
referrerpolicy	no-referrer no-referrer-when-downgrade origin origin-when-cross-origin same-origin strict-origin-when-cross-origin unsafe-url	Specifies which referrer information to send with the link	target	_blank _parent _self _top	Specifies where to open the linked document
			type	<i>media_type</i>	Specifies the media type of the linked document

Attribute	Value	Description	rel	alternate author bookmark external help license next nofollow noreferrer noopener prev search tag	Specifies the relationship between the current document and the linked document
download	<i>filename</i>	Specifies that the target will be downloaded when a user clicks on the hyperlink			
href	<i>URL</i>	Specifies the URL of the page the link goes to			
hreflang	<i>language_code</i>	Specifies the language of the linked document			
media	<i>media_query</i>	Specifies what media/device the linked document is optimized for			
ping	<i>list_of_URLs</i>	Specifies a space-separated list of URLs to which, when the link is followed, post requests with the body ping will be sent by the browser (in the background). Typically used for tracking.			
referrerpolicy	no-referrer no-referrer-when-downgrade origin origin-when-cross-origin same-origin strict-origin-when-cross-origin unsafe-url	Specifies which referrer information to send with the link	target	_blank _parent _self _top	Specifies where to open the linked document
			type	<i>media_type</i>	Specifies the media type of the linked document

<a> tag attribute : name

The name attribute of the anchor tag can be used to enable users to “jump” to a specific point on a page (jump marker, anchor). This is especially useful with large pages or subdivisions.

The HTML code looks like this:

```
<a name="to top"></a> or
```

```
<a name="Content"></a>Content
```

In the first code sample, you link from the bottom of a page back to the beginning, so users can quickly get to the top of the page without having to scroll for a long time. In the second example, users can directly access a part of the page, for example a subdivision point. By clicking, users are guided via name attribute directly to the subject.

```
<a href="#Content">Content</a>
```

By simply setting a hash tag (#) at the anchor name, the browser can identify a jump within the page.

You can also link to a specific location on another page internally. This would be defined as follows:

```
<a href="anotherpage.html#name">Linktext</a>
```

<a> tag attribute : Target

The target attribute specifies how the destination page or the target document should be opened. “target=“ _ blank “ is used for the opening of the target page in a new tab. This is the usual option when using target attributes for linking to other pages.

```
<a href="http://www.mypage.com" target="_blank">Linktext</a>
```

Value	Description
_blank	Opens the linked document in a new window or tab
_self	Opens the linked document in the same frame as it was clicked (this is default)
_parent	Opens the linked document in the parent frame
_top	Opens the linked document in the full body of the window
<i>framename</i>	Opens the linked document in the named iframe

<a> tag attribute : Title

- The title attribute gives users an important indication as to where they will be directed to when they click on a link.
- If the user hovers with the mouse over a link reference, the title text, which was defined in the attribute, will be displayed. This can be a **tool tip**, or description of the resource to which the link directs. The data from the title attribute makes the site more user-friendly.
- The title attribute can also be used for images and graphic files to give a brief description of what is shown in the picture.

```

```

The title attribute for Anchor Tags generally has no direct effect on SEO. However, the use of title attributes and elements on links or images can increase the click-through rate and thus indirectly affect the user signals that Google receives from the website.

<a> tag Style

When referencing the special attributes of the anchor tag, such as **link**, **visited**, **hover**, and **active**.

These 4 special attributes are explained below:

- **link**- This references the text of the link itself. Note that there are 2 ways in CSS in which we can modify the text of a link. We can either just directly reference the class name of the link. Referencing the attribute link is the same as referencing the class name directly. `.classname { }` and `.classname:link{ }` both reference the same text of the link.
- **visited**- This refers to a link on a page after the user has clicked on it. After the user clicks on a link, the link is said to be visited. After this link is visited, you can change its appearance to anything you want it to be. Most of the times, simply the color is changed, so that the user visiting a site can know that s/he has already clicked on that link.

<a> tag Style

- **hover**- This refers to a link on a page when a user hovers the mouse over the link. When a user has the mouse pointer on the link, the webmaster can change the link's appearance to anything. Usually the color of the link is programmed so that it changes to a different color and the underline of the link may disappear. Again, as a programmer, you can change this to anything with this attribute.
- **active**- This refers to a link the split second after it is clicked, to show that the user clicked it and that the click has gone through. Usually, again, this is simply a color change of the text of the link to some other color.

<a> tag Style

```
a:link {color:orange;}  
a :visited {color:green;}  
a:hover {color:yellow;}  
a:active {color:red;}
```

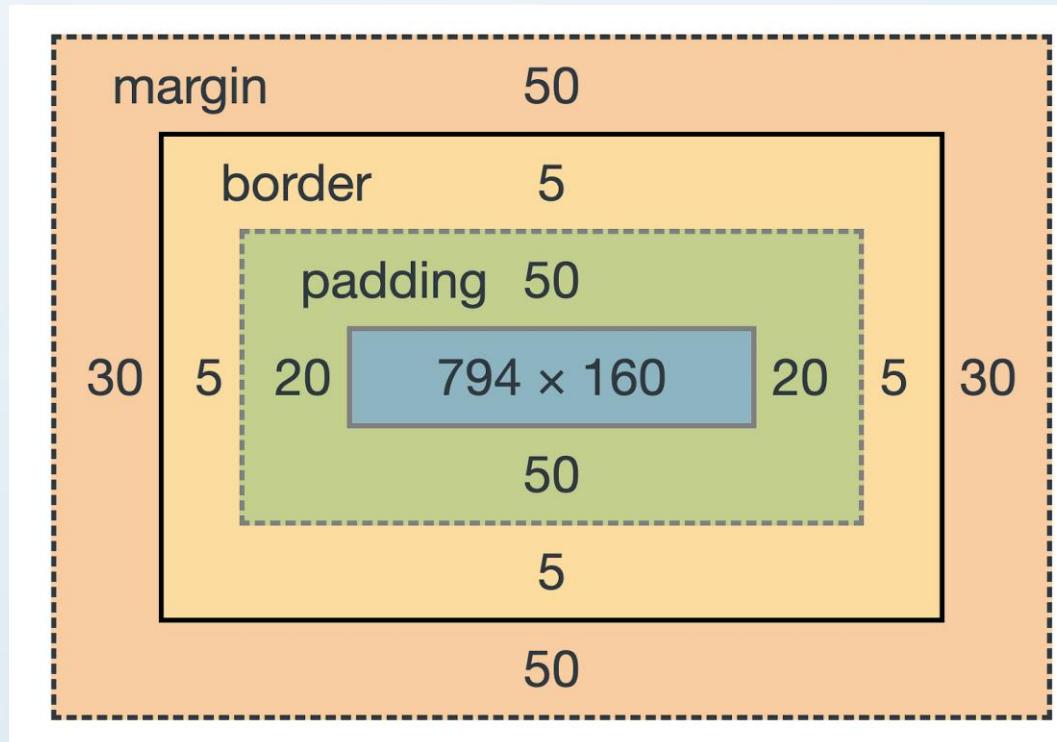
```
a[href^="http"] {  
background: url("external-link-52.png") no-repeat 100% 0;  
background-size: 16px 16px;  
padding-right: 19px;  
}
```

Links as Buttons

```
a {  
background-color: red;  
color: white;  
padding: 1em 1.5em;  
text-decoration: none;  
text-transform: uppercase;  
}  
  
<a href="#">Top It Center in Nepal</a>
```

CSS Box Model and its Properties

- The CSS box model is a container that contains multiple properties including borders, margin, padding, and the content itself.
- It is used to create the design and layout of web pages.



Properties of the Box Model

Content

- The content area consists of content like image, text, or other forms of media content. The height and width properties help to modify the box dimensions.

Padding

- The padding area is the space around the content area and within the border-box. It can be applied to all sides of the box or to the specific, selected side(s) - top, right, bottom, and/or left.

Border

- The border area surrounds the padding and the content, and can be applied to all the sides of the box or to selected side(s) - top, right, bottom, and/or left.

Properties of the Box Model

Margin

- The margin area consists of space between the border and the margin. The margin does not possess its own background color and is completely transparent. It shows the background color of the element, like the body element.

Code: <p>I am a paragraph of text that has a few words in it.</p>

```
p {  
width: 100px;  
height: 50px;  
padding: 20px;  
border: 1px solid;  
}
```

Margin

CSS **margin attributes** are used to generate space around items that are not bound by any defined borders.

Property of CSS Margin	Function
margin	This property is used to configure all of the properties in single declaration.
margin-left	It is used to specify the left margin of an element.
margin-right	It is used to specify the right margin of an element.
margin-top	It is used to specify the top margin of an element.
margin-bottom	It is used to specify the bottom margin of an element.

Syntax for Margin Property in CSS

One Value: margin: all;

Two Values: margin: top_bottom
left_right;

Three Values: margin: top right_left
bottom;

Four Values: margin: top right bottom
left;

Padding

CSS **padding values** are used to provide space around an element's content while staying within any established bounds.

- **The following values can be assigned to all padding properties:**
- **Length:** defines padding in px, pt, cm, and so on.
- **Percent:** defines padding as a percentage of the contained elements.
- **inherit:** defines that the padding must be inherited from the parent element.

Border

- CSS border property is used to set the border of an HTML element.
- The border property is shorthand for three sub-properties that define the style, color, and width of a border.

Example:

```
border: 1px solid red;
```

Syntax:

```
border: border-width border-style color;
```

Border-Width

Border-width:

Sets the thickness of the border. Defaults to medium if absent.

`border-width: thin | medium | thick;`

OR

`border-width: border-top-width border-right-width border-bottom-width border-right-width;`

Example : `border-width: 0 4px 8px 12px;`

border-style

Sets the style of the border. Defaults to none if absent.

`border-style: none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset`

Border-Width

color : Sets the color of the border. Defaults to current color if absent.

border-color: red;

border-color: border-top-color border-right-color border-bottom-color border-right-color

border-color: red yellow green pink;

CSS Individual Borders

The border sub-properties and property can also be applied to an individual side of a web element.

border-left: green;

border-top: pink;

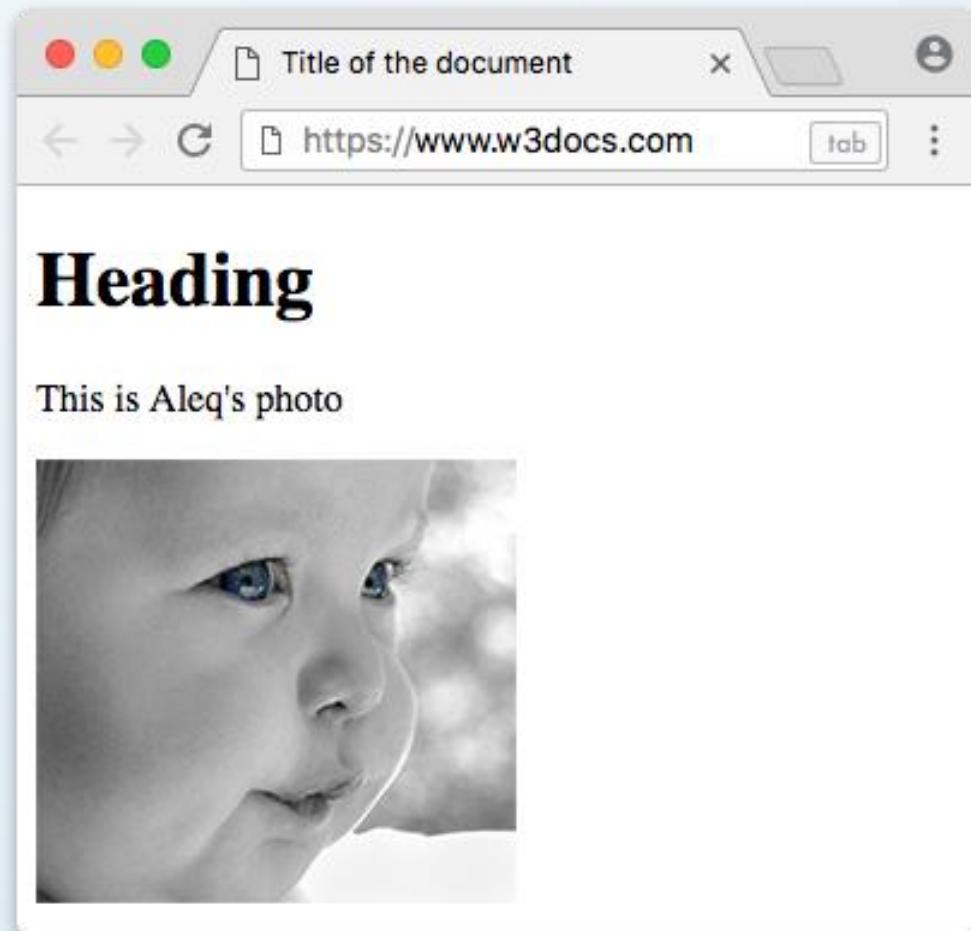
border-right: blue;

border-bottom: skyblue;

HTML Tag

- The tag is used to insert an image into an HTML document.
- The image itself isn't inserted directly into the document, the browser inserts an HTML image from the source specified in the tag.
- There are two required **attributes** for an element: **src** which is used to show the image source, and **alt** which defines an alternate text for the image.
- To make HTML images clickable, you should place the tag inside the <a> tag, which is used for inserting an HTML image link.

HTML Tag



Src and Alt Attributes : Img tag

- The src (source) attribute shows the image source. It is required, as it defines the path to the image. The value of the href attribute can be either the file name or its URL.
- The alt attribute defines an alternate name for the image. It is required for the tag too. Its value is a descriptive text displayed in the browser before the image is loaded. The browser also shows this text when you hover over the image.

Src and Alt Attributes : Img tag

Using "data:image/[type];base64,[base64-string]" for src attribute

- The data:image/[type];base64,[base64-string] format can be used as the value of the src attribute of an img tag to display an image directly from the HTML code, without having to load it from an external file.
-

Vspace and Hspace attribute

- The Hspace and Vspace attribute can be used to introduce "rounaround" or buffer space around an inland image.
- The Hspace attribute is used to insert a buffer of horizontal space on the left and right of an image.
- The Vspace attribute is sued to insert a buffer of vertical space in between the top and bottom of the image and other objects

Vspace and Hspace attribute

- The Hspace and Vspace attribute can be used to introduce "rounaround" or buffer space around an inland image.
- The Hspace attribute is used to insert a buffer of horizontal space on the left and right of an image.
- The Vspace attribute is sued to insert a buffer of vertical space in between the top and bottom of the image and other objects

Other attributes of img tag

Attribute	Value	Description
align	left right top bottom middle	Defines the alignment of the image in reference to surrounding elements. Not supported in HTML5.
alt	text	Defines the alternate text for the image.
border	pixels	Defines the width of the border around the image. Not supported in HTML5.

Other attributes of img tag

crossorigin		Defines, whether the CORS (a technology, that allows a web page to access resources from another domain) is used when loading the image. Images, uploaded via CORS, can be used in the <code><canvas></code> element without limiting the functionality of the latter.
	anonymous	CORS requests from this element will not pass credentials.
	use-credentials	CORS requests from this element will pass credentials. New attribute in HTML5.
height	pixels	Defines the height of the image.
hspace	pixels	Defines spaces at the left and right sides of the image. Not supported in HTML5.
ismap	ismap	Specifies that the contents of the tag is a server-side image map.

Other attributes of img tag

longdesc	URL	Specifies the URL address with a detailed description of the image (For a short description of the image, use the alt attribute. Not supported in HTML5.
src	URL	Defines the source of the image.
usemap	#mapname	Specifies a link to the <map> element, which contains the coordinates for the client map image.
vspace	pixels	Defines spaces at the top and bottom of the image. Not supported in HTML5.
width	pixels	Defines the width of the image.

CSS object-fit property

- This CSS property specifies how a video or an image is resized to fit its content box. It defines how an element fits into the container with an established width and height.
- It is generally applied to images or videos. This property specifies how an element reacts to the width and height of its container.

object-fit: fill | contain | cover | none | scale-down | initial | inherit;

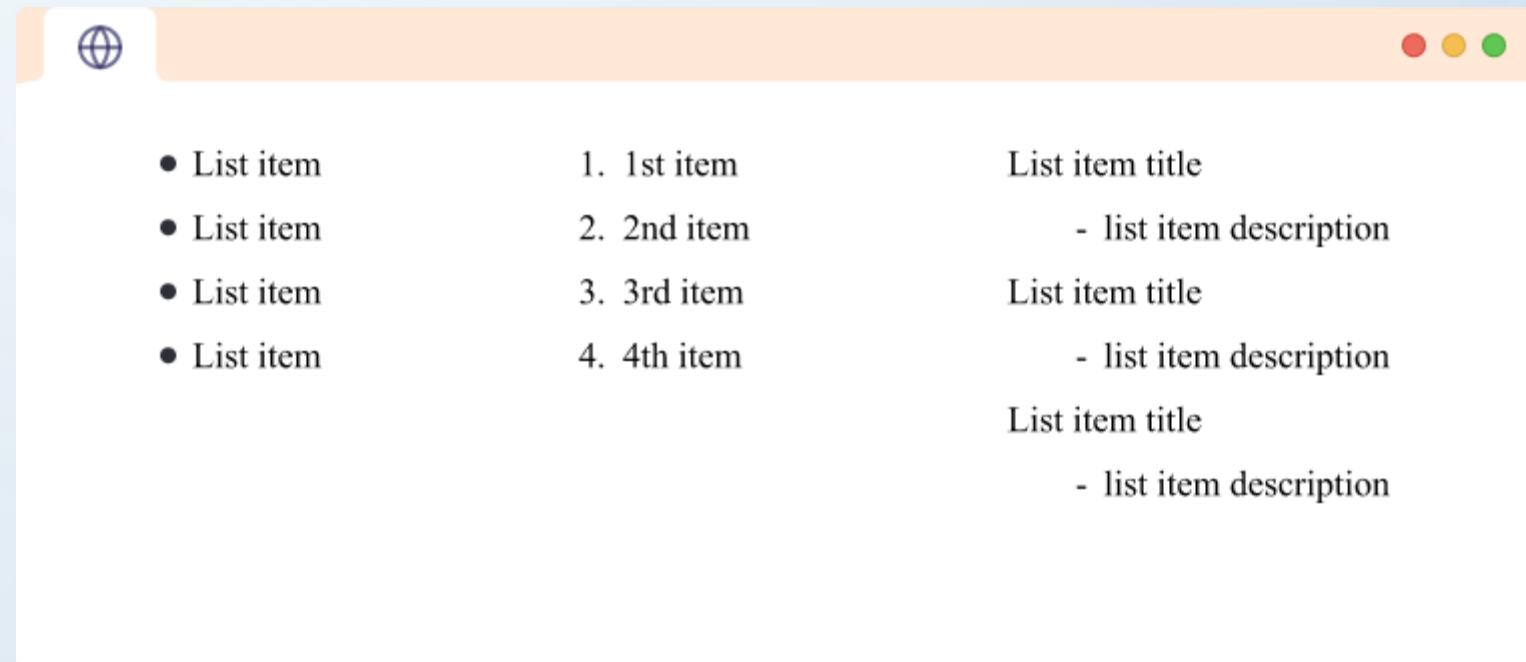
```
img {  
  object-fit: cover;  
}
```

HTML Lists

- In HTML, there are three types of lists: unordered, ordered and description lists. Each of them is defined using different tags.

List of tags for html list

- UL
- OL
- LI
- DL
- DD
- DT



HTML Unordered Lists

- We use unordered lists to group items having no numerical order. When changing the order of list items, the meaning will not change.
- To create an unordered list, we use the `` tag. This tag comes in pairs, the content is written between opening `` and closing `` tags.
- Each element of an unordered list is declared inside the `` tag.

```
<ul>
  <li>This is a list item</li>
  <li>This is another list item</li>
  <li>This is one more list item</li>
</ul>
```

HTML Unordered Lists

- The items in unordered lists are marked with bullets (small black circles) by default. However, the default bullet style for the list items can be changed using a type attribute.
- The type attribute is used to change the default bullet style for the list items.

Unordered Lists Marker

- We use the CSS list-style-type property to change the marker that marks the list item.

Dot	Circle	Square	None
● Apple	○ Apple	■ Apple	Apple
● Mango	○ Mango	■ Mango	Mango
● Orange	○ Orange	■ Orange	Orange
● Banana	○ Banana	■ Banana	Banana

The valid options for markers are

Icon	Marker	Description
• (default)	disc	sets the marker to a dot
○	circle	sets the marker to a hollow circle
■	square	sets the marker to a filled black square
	none	removes the marker altogether

```
<ul style="list-style-type: square;">  
  <li>Cold Drinks</li>  
  <li>Hot Drinks</li>  
  <li>Ice-Creams</li>  
</ul>
```

```
<ul style="list-style-type: disc;">  
  <li>Coca-Cola</li>  
  <li>Fanta</li>  
  <li>Ice Tea</li>  
</ul>
```

```
<ul style="list-style-type: circle;">  
  <li>Coca-Cola</li>  
  <li>Fanta</li>  
  <li>Ice Tea</li>  
</ul>
```

Nesting Lists

```
<ul>
  <li>
    Coffee
    <ul>
      <li>Cappuccino</li>
      <li>Americano</li>
      <li>Espresso</li>
    </ul>
  </li>
  <li>
    Tea
    <ul>
      <li>Milk Tea</li>
      <li>Black Tea</li>
    </ul>
  </li>
  <li>Milk</li>
</ul>
```

- Coffee
 - Cappuccino
 - Americano
 - Espresso
- Tea
 - Milk Tea
 - Black Tea
- Milk

HTML Ordered Lists

- HTML ordered list is used for listing items that are marked with numbers. It starts with the `` tag. This tag comes in pairs, the content is written between opening `` and closing `` tags.
- Each item in the ordered list starts with opening `` tag and ends with `` closing tag.

```
<ol>
  <li>This is List item number 1</li>
  <li>This is List item number 2</li>
  <li>This is List item number 3</li>
</ol>
```

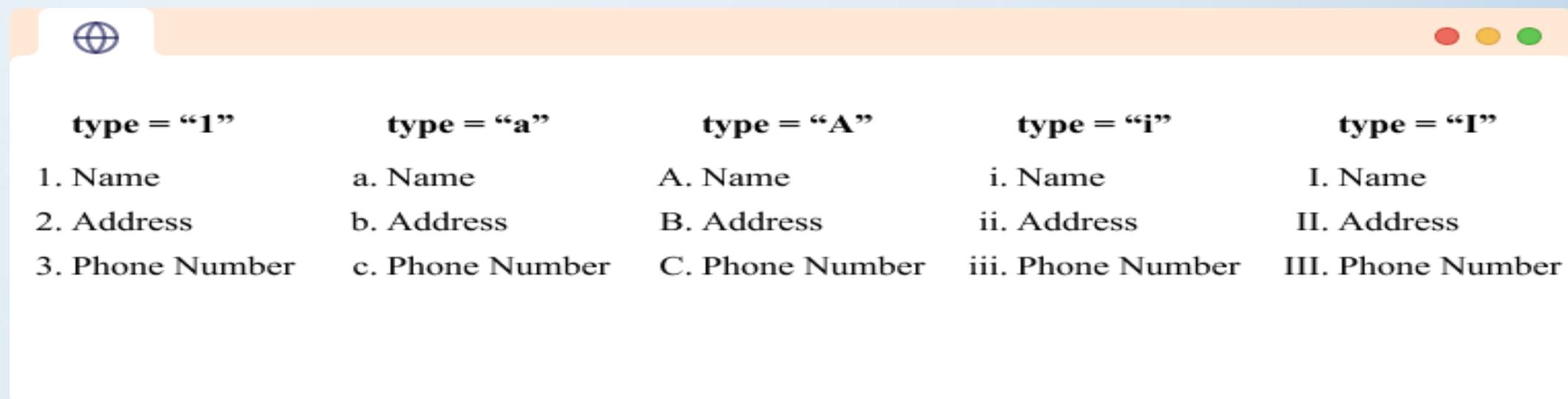
1. This is List item number 1
2. This is List item number 2
3. This is List item number 3

HTML Ordered Lists

- The items in the ordered list are marked with numbers by default. If you want to create ordered list with alphabet or Roman numbers, you just need to add type="a" or type="I" to the tag.
- **Ordered Lists Type**
- We use the **type** attribute to change the marker for the list. There are five types of numbering in the ordered list. They are

HTML Ordered Lists

Type	Description
"1"(Default)	The list is numbered with numbers.
"a"	The list is numbered with lower-case alphabets.
"A"	The list is numbered with upper-case alphabets.
"i"	The list is numbered with lower-case roman numerals.
"I"	The list is numbered with upper-case roman numerals.



HTML Ordered Lists:Attribute

start Attribute

- We use the start attribute to change the starting point for the numbering of the list. For example

```
<ol start='5'>
  <li>Harry</li>
  <li>Ron</li>
  <li>Sam</li>
</ol>
```

5. Harry
6. Ron
7. Sam

```
<ol type="a" start='5'>
  <li>Harry</li>
  <li>Ron</li>
  <li>Sam</li>
</ol>
```

e. Harry
f. Ron
g. Sam

HTML Ordered Lists:Attribute

reversed Attribute

- We can use the reversed attribute on the ordered list to reverse the numbering on the list. For example,

```
<ol reversed>
  <li>Cat</li>
  <li>Dog</li>
  <li>Elephant</li>
  <li>Fish</li>
</ol>
```

4. Cat
3. Dog
2. Elephant
1. Fish

```
<ol reversed type="I" start="10">
  <li>Cat</li>
  <li>Dog</li>
  <li>Elephant</li>
  <li>Fish</li>
</ol>
```

X. Cat
IX. Dog
VIII. Elephant
VII. Fish

Nesting Lists

```
<ol type="I">
  <li>
    Chapter 1
    <ol type="a">
      <li>Lesson 1</li>
      <li>Lesson 2</li>
    </ol>
  </li>
  <li>
    Chapter 2
    <ol type="a">
      <li>Lesson 1</li>
      <li>Lesson 2</li>
      <li>Lesson 3</li>
    </ol>
  </li>
  <li>
    Chapter 3
    <ol type="a">
      <li>Lesson 1</li>
    </ol>
  </li>
</ol>
```

I. Chapter 1

a. Lesson 1

b. Lesson 2

II. Chapter 1

a. Lesson 1

b. Lesson 2

c. Lesson 3

III. Chapter 1

a. Lesson 1

Ordered List inside Unordered List

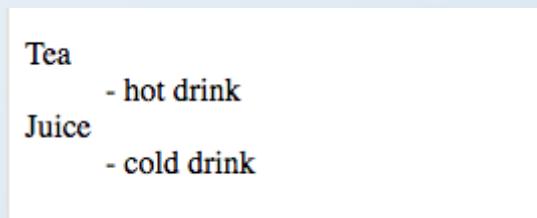
```
<ul>
  <li>
    Coffee
    <ol>
      <li>Cappuccino</li>
      <li>Americano</li>
      <li>Espresso</li>
    </ol>
  </li>
  <li>
    Tea
    <ol>
      <li>Milk Tea</li>
      <li>Black Tea</li>
    </ol>
  </li>
  <li>Milk</li>
</ul>
```

- Coffee
 1. Cappuccino
 2. Americano
 3. Espresso
- Tea
 1. Milk Tea
 2. Black Tea
- Milk

HTML Description Lists

- HTML description list is used to arrange terms or names with a description the same way as they are arranged in a dictionary.
- To create a description list, we use the <dl> tag. This tag comes in pairs.
- In <dl>, we use <dt> tags for a term/name in a description list and <dd> for a description of a term/name in a description list.

```
<dl>
  <dt>Tea</dt>
  <dd>- hot drink</dd>
  <dt>Juice</dt>
  <dd>- cold drink</dd>
</dl>
```



Horizontal List with CSS

```
<style>
  ul {
    list-style-type: none;
    margin: 0;
    padding: 0;
    overflow: hidden;
    background-color: #F44336;
  }
  li {
    float: left;
  }
  li a {
    display: block;
    color: white;
    text-align: center;
    padding: 16px;
    text-decoration: none;
  }
  li a:hover {
    background-color: #981816;
  }
</style>
```

```
<ul>
  <li>
    <a href="#home">Home</a>
  </li>
  <li>
    <a href="https://www.w3docs.com/tool/">Tools</a>
  </li>
  <li>
    <a href="https://www.w3docs.com/snippets">Snippets</a>
  </li>
  <li>
    <a href="https://www.w3docs.com/quiz/">Quizzes</a>
  </li>
  <li>
    <a href="https://www.w3docs.com/string-functions/">String Functions</a>
  </li>
</ul>
```



More Code Example for List

- <https://sharkcoder.com/blocks/list>
- <https://learn.shayhowe.com/html-css/creating-lists/>
- <https://developer.mozilla.org/en-US/docs/Web/CSS/list-style>

Create following list using HTML/CSS

Learning Web Development

- I. Background Skills
 - A. Unix Commands
 - B. Vim Text Editor
- II. HTML
 - A. Minimal Page
 - B. Headings
 - C. Tags
 - D. Lists
 - i. Unordered
 - ii. Ordered
 - iii. Definition
 - iv. Nested
 - E. Links
 - i. Absolute
 - ii. Relative
 - F. Images
- III. CSS
 - A. Anatomy
 - B. Basic Selectors
 - i. Element
 - ii. Class
 - iii. ID
 - iv. Group
 - C. The DOM
 - D. Advanced Selectors
 - E. Box Model
- IV. Programming
 - A. Python
 - B. JavaScript
- V. Database
 - A. Flat File
 - B. Relational

Manage Categories

- Basketball Sport
 - Basketball Shoes
 - Jordon
 - Lebron
 - Basketball Shorts
 - Long Basketball Shorts
 - Short Basketball Shorts
- Golf Sport
 - Bags
 - Nike Bags
 - Clubs
 - Nike Clubs
- Tennis Sport
 - Racquets
 - Head
 - Prince
 - Wilson
 - Tennis Shoes
 - Mens Tennis Shoes
 - Cheap Mens Tennis Shoes
 - Expensive Mens Tennis Shoes
 - Womens Tennis Shoes
 - Women Practical Tennis Shoes
 - Women Stylish Tennis Shoes

CSS Property for List

The CSS properties to style the lists are given as follows:

list-style: Shorthand property for List.

list-style-type: This property is responsible for controlling the appearance and shape of the marker.

list-style-image: It sets an image for the marker instead of the number or a bullet point.

list-style-position: It specifies the position of the marker.

list-style: It is the shorthand property of the above properties.

marker-offset: It is used to specify the distance between the text and the marker. It is unsupported in IE6 or Netscape 7.

CSS Property for List

Shorthand Property

```
ul { list-style: <list-style-type> || <list-style-position> || <list-style-image>; }
```

Example:

```
ul { list-style: square outside none; }
```

Values for list related property

- **List-style-type:** disc,circle,square,decimal,decimal-leading-zero,lower-roman,upper-roman,lower-greek,lower-latin,upper-latin,armenian,Georgian,lower-alpha,upper-alpha,none
- **List-style-position:** inside,outside
- **List-style-image:** list-style-image: url(images/bullet.png);
- **Marker-offset:2px/cm/inch;**

CSS Background Property

- The CSS background property is a shorthand for specifying the background of an element.
- `background-color`, `background-image`, `background-repeat`, `background-position`, `background-clip`, `background-size`, `background-origin` and `background-attachment` together comprise the CSS background properties.
- The syntax of CSS background property is as follows–

```
Selector {  
  background: /*value*/  
}
```

Background Property CSS

All Properties

#1 background-image

#2 background-size

#3 background-position

#4 background-repeat

#5 background-color

#6 background-origin

#7 background-clip

#8 background-attachment

#9 **background**

Background Property CSS

- **background-image**: Specifies the background image to be displayed. It can be a URL pointing to an image file or a gradient created using the **linear-gradient()** or **radial-gradient()** functions.
- **background-color**: Sets the background color for an element. It can be specified using color names, hexadecimal values, RGB values, or HSL values.
- **background-repeat**: Determines how the background image is repeated if it is smaller than the element's size. Values include **repeat (default)**, **repeat-x**, **repeat-y**, and **no-repeat**.
- **background-position**: Specifies the starting position of the background image. It can be set using keywords like **left**, **center**, **right**, **top**, **bottom**, or using **percentage** or **length** values.
- **background-size**: Sets the size of the background image. It can be specified using keywords like **cover**, **contain**, or using length values, such as **px**, **em**, or **percentage** values.

Background Property CSS

- **background-attachment:** Determines whether the background image scrolls with the content or remains fixed. Values include **scroll** (default) and **fixed**.
- **background-origin:** Defines the positioning area of the background image relative to the element's **padding box, border box, or content box**.
- **background-clip:** Specifies the area where the background image or color is visible. It can be set to **border-box, padding-box, or content-box**.
- **background-blend-mode:** Allows blending of the background image or color with the content behind it. It provides effects like transparency, color mixing, and more.
- **background-gradient:** Represents a CSS gradient that can be used as the background. Gradients can be linear or radial and can include multiple color stops.

Background-color Property CSS

- In CSS (Cascading Style Sheets), the background color property is used to define the background color of an element on a web page. It allows you to specify a color value or keyword that determines the background color of the content area of an element.
- The background color property can be applied to various HTML elements, such as `<body>`, `<div>`, `<p>`, ``, etc. Here's the basic syntax:

```
selector {  
  background-color: value;  
}
```

- The selector is the HTML element or class/id selector to which you want to apply the background color. The value represents the desired color for the background.

Example

```
p {  
  background-color: yellow;  
}
```

Background-image Property CSS

To set a background image using CSS, you can use the background-image property.

Here's an example of how you can add a background image to an element:

```
selector {  
    background-image: url("path/filename");  
}
```

In the example above, replace "path/filename" with the actual path to your image file.

You can also specify additional properties to control how the background image is displayed, such as **background-repeat**, **background-size**, and **background-position**. Here's an example that includes some of these properties:

Example

```
body {  
    background-image: url("image.png");  
}
```

Background-repeat Property CSS

- **repeat:** This is the default value. It repeats the background image both horizontally and vertically to cover the entire background area. If the background image is smaller than the element, it will be repeated to fill the space.
- **repeat-x:** The background image is repeated only horizontally. It will be displayed in a row, repeating horizontally to cover the width of the element. The image is not repeated vertically.
- **repeat-y:** The background image is repeated only vertically. It will be displayed in a column, repeating vertically to cover the height of the element. The image is not repeated horizontally.
- **no-repeat:** The background image is not repeated. It is displayed only once and will not be repeated in any direction. If the image is smaller than the element, it may be cropped or not fully displayed.
- **space:** This value is similar to repeat, but it adds space between the repeated images. The space is distributed evenly between the repeated images, creating gaps between them. If there is extra space left after distributing it evenly, it will be added at the ends.
- **round:** This value is also similar to repeat, but it resizes the repeated images to fit the background area without distorting them. It scales the images up or down proportionally so that they cover the entire background area. If the images cannot be resized evenly, they will be repeated with the remaining space distributed as evenly as possible.

Background-position Property CSS

- In CSS, the background-position property is used to control the positioning of a background image within its containing element. It specifies where the background image should be placed relative to the element's padding box.
- The background-position property accepts one or two values, which can be specified using keywords, percentages, or length values.
- Here are the different ways to specify the **background-position** values:

Keywords

- **left:** Positions the background image at the left side of the element.
- **center:** Positions the background image at the horizontal and vertical center of the element.
- **right:** Positions the background image at the right side of the element.
- **top:** Positions the background image at the top of the element.
- **bottom:** Positions the background image at the bottom of the element.

Background-position

Percentages

- The first value represents the horizontal position, where 0% is the left edge of the element and 100% is the right edge.
- The second value represents the vertical position, where 0% is the top edge of the element and 100% is the bottom edge.

Length values

- The first value represents the horizontal position, which can be specified using pixels (px), ems (em), or any other CSS length unit.
- The second value represents the vertical position, also specified using CSS length unit
- The **background-position** property can also be combined with the **background-repeat** property to control both the positioning and repetition of the background image.

Example

```
body {  
    background-image: url("img_tree.png");  
    background-repeat: no-repeat;  
    background-position: right top;  
}
```

Background-Attachment

- In CSS, the background-attachment property is used to control whether a background image scrolls with the content or remains fixed in place when the content is scrolled.
- The background-attachment property specifies whether the background image should scroll or be fixed (will not scroll with the rest of the page):

Example

```
body {  
background-image: url("img_tree.png");  
background-attachment: fixed;  
}
```

Background-Attachment

The **background-attachment** property accepts the following values:

- scroll: This is the default value. The background image scrolls along with the content as the user scrolls the element. When the element is scrolled, the background image moves accordingly.
- fixed: The background image remains fixed in its position within the element's viewport, regardless of scrolling. As the content is scrolled, the background image stays in place, creating a "fixed" effect. This can be useful for creating parallax effects or backgrounds that remain static while the content scrolls over them.
- local: The background image scrolls with the element's contents but is fixed relative to the element itself. This means that the background image scrolls along with the content within the element, but it does not scroll out of the element's boundaries. This is useful when you want the background image to be fixed within a specific area of an element, such as a scrolling panel.
- initial or inherit: These values are used to inherit the background-attachment property from the parent element or set it to its initial value.

By utilizing the background-attachment property, you can create different visual effects and control how background images behave in relation to scrolling content.

Self Study Task

Background Origin

Background Clip

Background Blend Mode

Background Shorthand Code: (Important)

HTML Table

- **Tables in HTML** are a collection of data organised in rows and columns, or a more complicated form tables are commonly used in data analysis, research, and communication.
- Tables can be used for a variety of purposes like showing text and numerical data. It is used in a tabular form layout to differentiate between two or more topics. **Databases are built using tables.**
- To create a Table in HTML <table> tag is used which is the main container of the table.

```
<table>
  <tr>
    <th> Heading </th>
  </tr>
  <tr>
    <td> Data </td>
  </tr>
</table>
```

HTML Table

```
<table>
  <tr>
    <th>Name</th>
    <th>Course</th>
    <th>Application Number</th>
  </tr>
  <tr>
    <td>Aliana</td>
    <td>B.Tech CSE</td>
    <td>17218</td>
  </tr>
  <tr>
    <td>Maria</td>
    <td>Fashion Tech</td>
    <td>17219</td>
  </tr>
  <tr>
    <td>Sarah</td>
    <td>Journalism</td>
    <td>17272</td>
  </tr>
  <tr>
    <td>Elena</td>
    <td>MBA</td>
    <td>17291</td>
  </tr>
</table>
```

Output

Name	Course	Application Number
Aliana	B.Tech CSE	17218
Maria	Fashion Tech	17219
Sarah	Journalism	17272
Elena	MBA	17291

Table Row

- To arrange data horizontally in different table cells are called table rows.
- To create a table row we use `<tr>` tag.

```
<tr>
  <th> Heading </th>
  <td> Content </td>
</tr>
```

```
<table>
  <tr>
    <th>Name</th>
    <th>Age</th>
    <th>Course</th>
  </tr>
  <tr>
    <td>Sarah</td>
    <td>20</td>
    <td>B.Tech</td>
  </tr>
</table>
```

Output

Name	Age	Course
Sarah	20	B.Tech

From the output, we can say that 2 rows are created with 3 columns using data cells i.e., the `<td>` tag.

Table Cell

- The <td> tag is defined as a cell in Tables in HTML which contain data.

Syntax

```
<tr>
  <td>.....</td>
  <td>.....</td>
</tr>
```

Output

Sarah	20	B.Tech
-------	----	--------

Example

```
<table style="width:100%">
  <tr>
    <td>Sarah</td>
    <td>20</td>
    <td>B.Tech</td>
  </tr>
</table>
```

From the following output, we can see that a table row is created containing three table cells which are created in a row using <td> tag.

Table Header

- To represent a table cell we use table headers. Tables in HTML can have headers horizontally as well as vertically.
- To create a table header `<th>` tag is used.

Syntax

```
<tr>
<th>.....</th>
<th>.....</th>
</tr>
```

Table Header Example

```
<table width="100%">
<caption>Timetable</caption>
<tr>
    <th>Time/Days</th>
    <th>Monday</th>
    <th>Tuesday</th>
</tr>
<tr>
    <th>8:00am</th>
    <td>Python</td>
    <td>Java</td>
</tr>
<tr>
    <th>12 noon</th>
    <td>Spanish</td>
    <td>English</td>
</tr>
<tr>
    <th>3:00pm</th>
    <td>Data Structures</td>
    <td>Networking</td>
</tr>
</table>
```

Output

Time/Days	Monday	Tuesday
8:00am	Python	Java
12 noon	Spanish	English
3:00 PM	Data Structures	Networking

As we can see table headers are added vertically as well as horizontally to do so we have added `<th>` tag in every `<tr>` tag.

Table Tags

Tags	Description
<table>	Used to create a table.
<tr>	Used to create table rows.
<td>	Used to create table data/cells.
<th>	Used to create table header.
<caption>	A table caption is defined.
<colgroup>	It is used to format a group of one or more columns in a table.
<col>	Used with the <colgroup> element to define column characteristics for each column.
<tbody>	The table body is grouped together using it.
<thead>	Table head is grouped together using this tag.
<tfooter>	Footer content is grouped together using this tag.

Table Attributes in HTML

Attributes and tags	Example
Cell spacing	<code><table cellspacing="">...</table></code>
Cell padding	<code><table cellpadding="">...</table></code>
Table border	<code><table border="">...</table></code>
Alignment	<code><table align=center/left/right>...</table></code>
colspan in table	<code><td colspan="">...</table></code>
rowspan in table	<code><td rowspan="">...</table></code>
Cell color	<code><table bgcolor="#\$\$\$\$\$\$">...</table></code>
No linebreaks	<code><table nowrap>...</table></code>

HTML tables with Rowspan

- Rowspan in table, works similar to the colspan for columns, but here, we assign **multiple rows** to a cell using an attribute rowspan="" .

Name	Jobs		Working Experience
John	Software Engineer	Data Analyst	5 Years
Ale	Senior Web developer		
Jack	Junior Tech Writer	Blogger	6 Months

A diagram illustrating the use of the rowspan attribute. An arrow points from the text "rowspan=2" to the empty cell in the "Working Experience" column for the "Ale" row. The cell contains the value "5 Years".

HTML tables with Colspan

- Colspan is an attribute which assigns multiple columns to a cell of a table. The number of columns depends on the value entered by you in colspan="" attribute.

Name	Jobs		Working Experience
John	Software Engineer	Data Analyst	5 Years
Ale	Senior Web developer		2 Year
Jack	Junior Tech Writer	Blogger	6 Months

Colspan="2"

HTML tables with Rowspan

```
<table>
  <tr>
    <th>Name</th>
    <th colspan="2">Jobs</th>
    <th>Working Experience</th>
  </tr>
  <tr>
    <td>John</td>
    <td>Software Engineer</td>
    <td>Data Analyst</td>
    <td rowspan="2">5 Years</td>
  </tr>
  <tr>
    <td>Ale</td>
    <td colspan="2">Senior Web developer</td>
  </tr>
  <tr>
    <td>Jack</td>
    <td>Junior Tech Writer</td>
    <td>Blogger</td>
    <td>6 Months</td>
  </tr>
</table>
```

Name	Jobs	Working Experience	
John	Software Engineer	Data Analyst	5 Years
Ale	Senior Web developer		2 Year
Jack	Junior Tech Writer	Blogger	6 Months

Colspan="2"

HTML tables : task

Day	Seminar		
	Schedule		Topic
	Begin	End	
Monday	8:00 a.m.	5:00 p.m.	Introduction to XML Validity: DTD and Relax NG
Tuesday	8:00 a.m.	11:00 a.m.	XPath
	11:00 a.m.	2:00 p.m.	
	2:00 p.m.	5:00 p.m.	XSL Transformations
Wednesday	8:00 a.m.	12:00 p.m.	XSL Formatting Objects

	Monday	Tuesday	Wednesday	Thursday	Friday
8:00am					
9:00am					
10:00am	Office Hours		Office Hours		
11:00am	CS245A	CS301A	CS245A	CS301A	
12:00pm					
1:00pm	Lunch				
2:00pm					
3:00pm	Office Hours	Office Hours		Office Hours	
4:00pm					
5:30pm					
6:30pm	CS505A	CS556AH1	CS556BH2		
7:30pm					
8:00pm					
	CS556AH1 is a first term course				
	CS556BH2 is a second term course				

day	seminar		
	schedule		topic
	begin	end	
Monday	8:00 a.m.	5:00 p.m.	Introduction to XML Validity: DTD and Relax NG
Tuesday	8:00 a.m.	11:00 a.m.	XPath
	11:00 a.m.	2:00 p.m.	
	2:00 p.m.	5:00 p.m.	XSL Transformatics
Wednesday	8:00 a.m.	12:00 p.m.	XSL Formatting Objects

S.No	Name	Score
	Tina	100
2	Nina	99

HTML Frame

- Frames are distinctly different from any other HTML element you probably know how to do. If you plan on using frames think of it like sewing two web pages together into one page.
- Frames are the sections created on the window.
- Each section can load a separate HTML document.
- The window is divided into frames in a similar way the tables are organized into rows and columns.
- Frames provide the facility to display multiple HTML document inside one browser window at a time.
- To create frames, we need `<frameset>` and `<frame>` tag.

<frameset> tag

- Collection of frames is called as frameset.
- If you are using frames on a page then use <frameset> tag instead of <body> tag.
- The rows attribute of <frameset> tag indicates horizontal frames and cols attribute indicates vertical frames.

Syntax: <frameset>.....</frameset>

<frameset> tag attributes

Attributes	Description
cols (Columns)	It specifies how many columns are to be contained in the frameset and the size of each column.
rows	It works like the 'cols' attribute and takes the same values, but it is used to specify the number of rows in the frameset.
border	It specifies the width of the border of each frame in pixels. For example; border = “5”. A value of zero means no border.
frame	It specifies whether a three-dimensional border should be displayed between frames. It takes value either 1 (Yes) or 0 (No). For example; frameborder = “0” specifies no border.
framespacing	It specifies the amount of space between frames in a frameset. It can take any integer value. For example; framespacing = “10” means there should be 10 pixels spacing between each frame.

<frame> tag attributes

- **<frame> tag**<frame> tag is used to define a frame within a <frameset> tag.
- A frame should be used within a <frameset> tag.

Syntax:<frame src="frm1.htm">

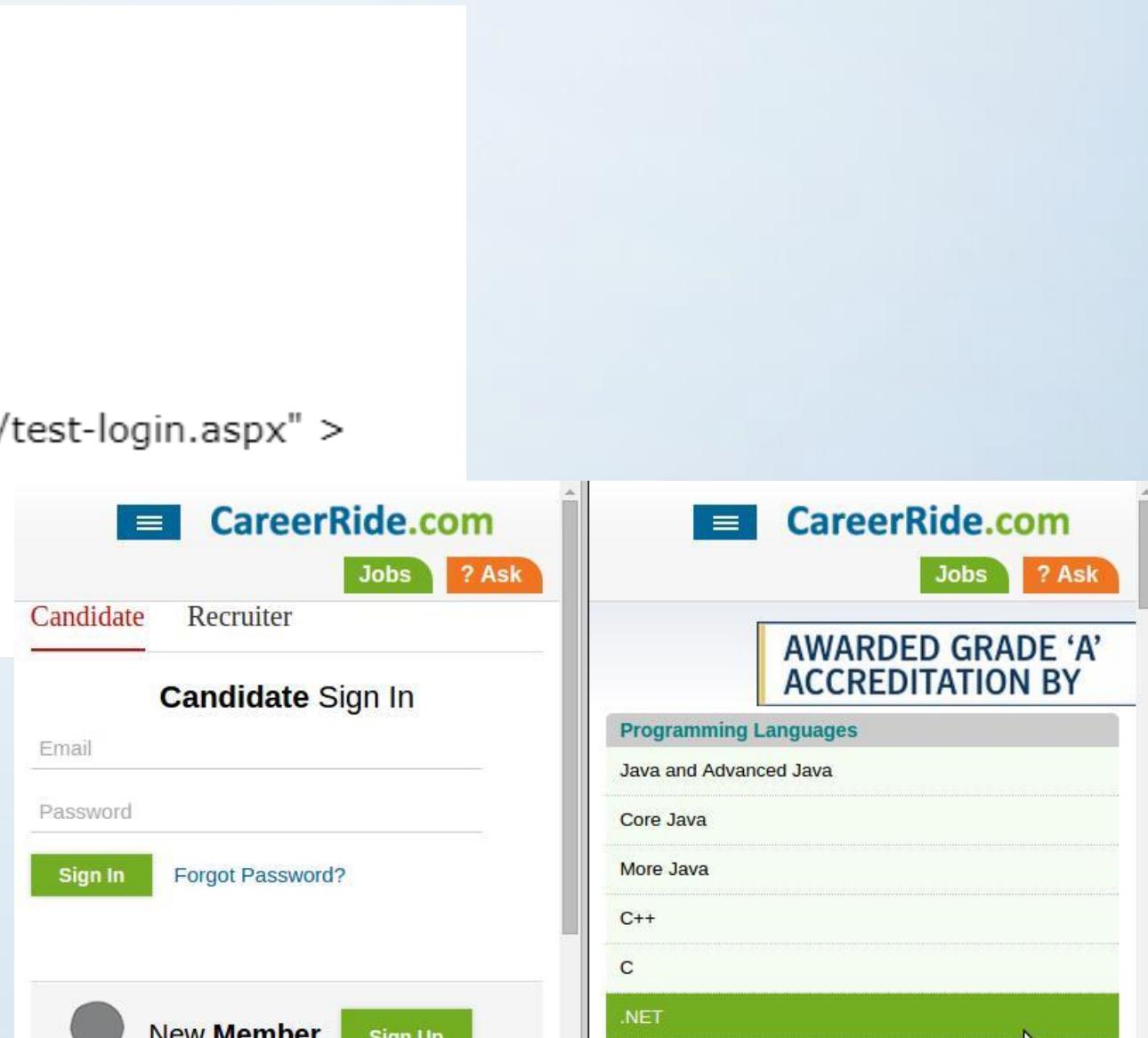
Hidden attribute is used to hide frame

<frame> tag attributes

Attributes	Description
src	It is used to give the file name that should be located in the frame. Its value can be any URL, for example, src= “/html/abc.html”
name	It allows to give a name to a frame. This attribute is used to indicate that a document should be loaded into a frame.
frameborder	It specifies whether or not the borders of that frame are shown. This attribute overrides the value given in the frameborder attribute on the <frameset> tag if one is given. This can take values either 1 (Yes) or 0 (No).
marginwidth	It allows to specify the width of the space between the left and right of the frame's border and the content. The value is given in pixels. For example; marginwidth = “10”.
marginheight	It allows to specify the height of the space between the top and bottom of the frame's borders and its contents. The value is given in pixels. For example; marginheight = “10”.
noresize	By default, a user can resize any frame by clicking and dragging on the borders of a frame. It prevents a user from being able to resize the frame. For example; noresize = “noresize”.
scrolling	It controls the appearance of the scrollbars that appear on the frame. It takes values either “Yes”, “No” or “Auto”. For example; scrolling = “no” means it should not have scroll bars.
longdesc	It allows to provide a link to another page which contains a long description of the contents of the frame. For example; longdesc = “framedescription.html”

HTML Frame Example

```
<!DOCTYPE html>
<html>
  <head>
    <title>HTML Frames Example</title>
  </head>
  <frameset cols="50%,50%">
    <frame src="http://www.careerride.com/test-login.aspx" >
    <frame src="http://careerride.com/" >
  </frameset>
</html>
```



HTML Frame Example

```
<html>
<body style="background-color:#ff9900;">
  <h2 align="center">First frame (frame_1.html)</h2>
</body>
</html>
```

frame_1.html

```
<html>
<body style="background-color:#ffcc00;">
  <h2 align="center">Second frame (frame_2.html)</h2>
</body>
</html>
```

frame_2.html

```
<html>
<head>
  <title>Frameset Example 1</title>
</head>
<frameset rows="35%, 65%">
  <frame src ="frame_1.html" />
  <frame src ="frame_2.html" />
</frameset>
</html>
```

frame_example1.html

First frame (frame_1.html) by

Second frame (frame_2.html)

HTML Frame Example

```
html>
<body style="background-color:#ff9900;">
  <h2 align="center">First frame (frame_1.html)</h2>
</body>
</html>
```

frame_1.html

```
html>
<body style="background-color:#a08029;">
  <h2 align="center">Second frame (frame_3.html)</h2>
</body>
</html>
```

frame_3.html

```
html>
<body style="background-color:#ffcc00;">
  <h2 align="center">Third frame (frame_4.html)</h2>
</body>
</html>
```

frame_4.html

```
<html>
<head>
  <title>Frameset Example 2</title>
</head>
<frameset rows="35%, 65%">
  <frameset cols="50%, 50%">
    <frame src ="frame_3.html" />
    <frame src ="frame_4.html" />
  </frameset>
</frameset>
</html>
```

frame_example2.html

First frame (frame_1.html)

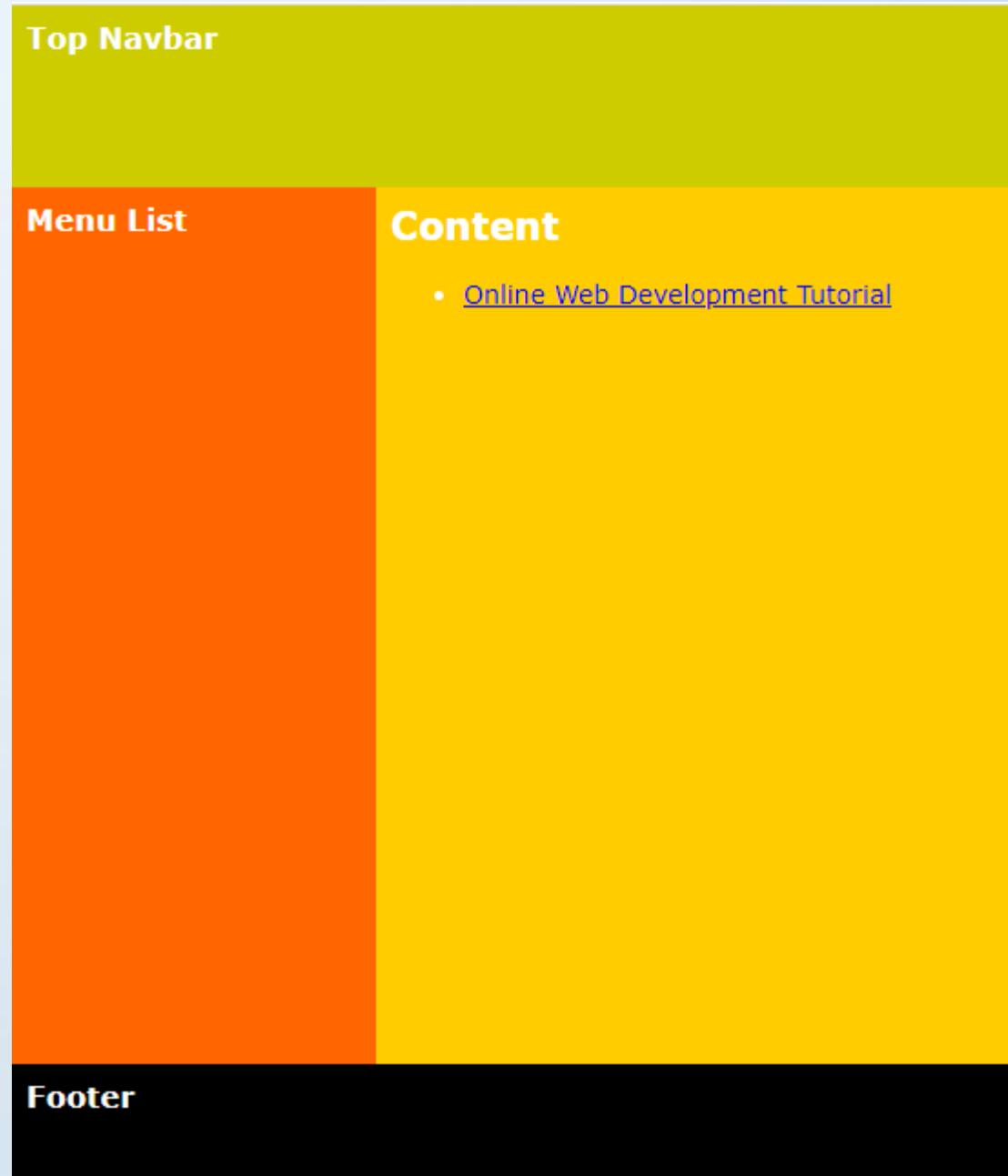
Second frame (frame_3.html)

Third frame (frame_4.html)

HTML Frame Example

Frame Example (Remove the frame border)

```
<html>
<head>
  <title>Frame Example 3</title>
</head>
<frameset rows="100,* ,75" frameborder="0" border="0" >
  <frame name="topNav" src="top_nav.html">
<frameset cols="200,*" frameborder="0" border="0">
  <frame name="menu" src="menu_list.html" scrolling="auto" noresize>
  <frame name="content" src="content.html" scrolling="auto" noresize>
</frameset>
  <frame name="footer" src="footer.html">
</frameset>
<noframes></noframes>
</html>
```



iframe- Inline Frame

- Iframe represents an HTML inline frame that contains another document.
- An iframe is used to display a web page within a web page.

Syntax: <iframe src="URL"></iframe>

Example

```
<!DOCTYPE html>
<html>
  <head>
    <title>HTML Iframes Example</title>
  </head>
  <body>
    <iframe src="http://careerride.com/" width="600" height="200"></iframe>
  </body>
</html>
```

iframe- Attributes

Property	Description
src	It indicates the location of the web page to be loaded into the frame.
align	It aligns the iframe. Its values can be left, right, top, middle, bottom.
name	It specifies the name to the frame.
frameborder	It specifies whether or not display a border of an iframe. Its values can be 0 or 1.
height	It sets the height in an iframe.
width	It sets or returns the value of the width attribute in an iframe.
marginheight	It sets the top and bottom margins of the content of an iframe. Its values can be in pixels.
marginwidth	It sets the left and right margin of the content of an iframe. Its values can be in pixels.
longdesc	It specifies the link of a page that contains a long description of the content of an iframe.
noresize	It disables the frame resizing capability.
scrolling	It controls the appearance of horizontal and vertical scrollbars in a frame. Its values can be 'yes', 'no' or 'auto'.

iframe- Example

```
<!DOCTYPE html>
<html>
  <body>
    <iframe width="600" height="200" name="iframe_a"></iframe>
    <p><a href="http://careerride.com/" target="iframe_a">CareerRide.com</a></p>
  </body>
</html>
```

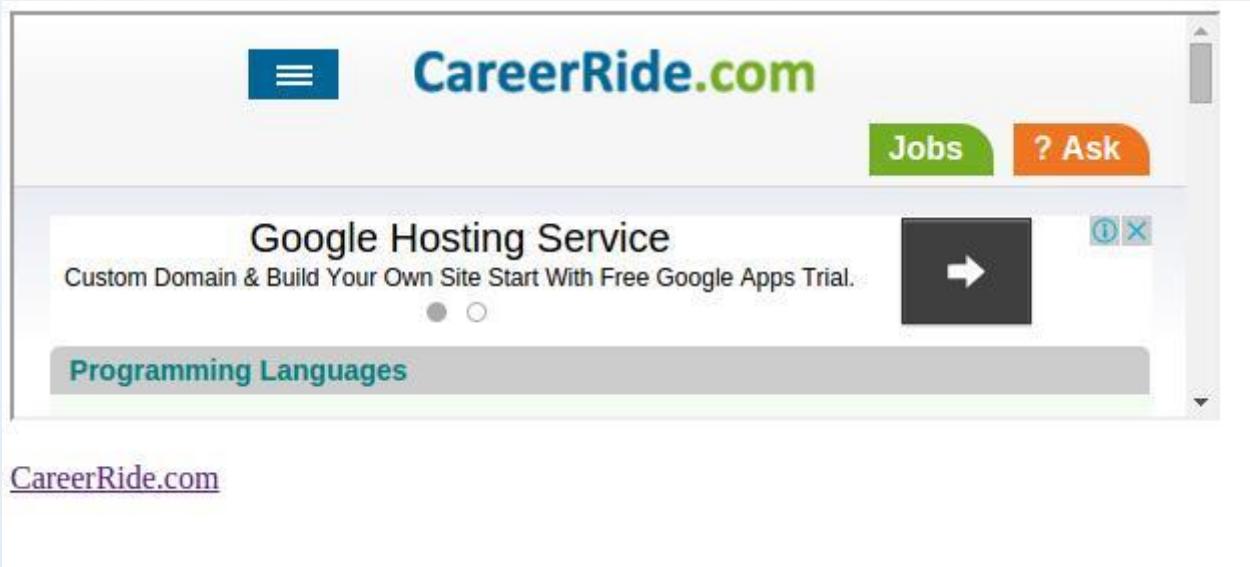


Image Maps in HTML

- We can use HTML image maps to make clickable areas on images.
- An image map comprises a picture with clickable areas, where you can click on the image and it will open to a new or already specified location.
- The `<map>` tag can include many `<area>` elements that define the coordinates and type of the area. You can simply link any area of a picture to other pages using the `<map>` tag without dividing the image.

Create Image Map

- Image map is created using a `<map>` element. The `<map>` element is used to generate an image map and is linked to the image by using the name attribute.

```
<map name="deskmap">
```

The name attribute must have the identical value as the images `usemap` attribute then only it will work.

The Areas

After creating the map element we need to create clickable areas. A clickable area is marked using an `<area>` element.

```
<area shape="rect" coords="175,242,420,358" alt="Keyboard"
target="_blank" href="https://en.wikipedia.org/wiki/Computer keyboard">
```

The area can be a **rectangle**, **circle**, **polygon**, or a whole region; we can select any of them depending on our requirements. Within the area element, we must define the shape we are using as well as the coordinates of the area we want to make clickable.

Create Image Map

```
<!DOCTYPE html>
<html>
<body>

<h2>Image Maps</h2>



<map name="deskmap">

  <area shape="rect" coords="175,242,420,358" alt="Keyboard" target="_blank"
  href="https://en.wikipedia.org/wiki/Computer_keyboard">

  <area shape="rect" coords="444,251,481,357" alt="Mouse" target="_blank"
  href="https://en.wikipedia.org/wiki/Computer_mouse">

  <area shape="rect" coords="375,14,481,357" alt="Diary" target="_blank"
  href="https://en.wikipedia.org/wiki/Book">

</map>
</body>
</html>
```

Image map-Rectangle

- Rectangle coords have the values x1,y1,x2,y2. The coordinates of the rectangle's top-left and bottom-right corners are specified by this value.

coords=" 60,40,391,195 "

- The coordinates are 60, 40, and 391,195 specifying the top-left and bottom-right corners of the rectangle respectively.

```

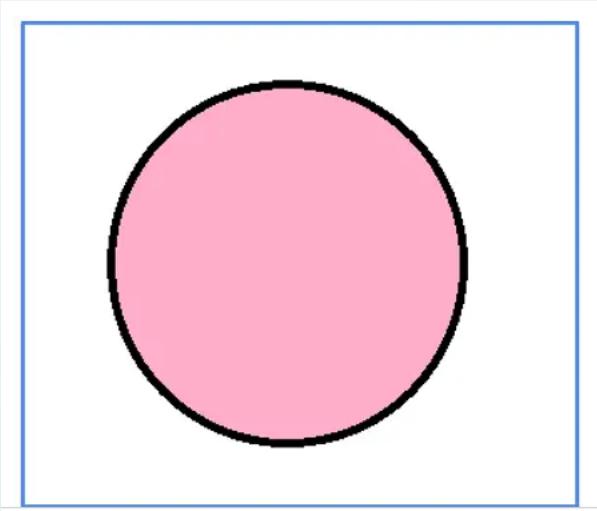

<map name="rectmap">
  <area shape="rect" coords="60,40,391,195" alt="rectangle"
  href="https://en.wikipedia.org/wiki/Rectangle">
</map>
```



Image map- Circle

- Circle coords have the value x,y, radius. The coordinates of the circle center and the radius are specified by this value.

coords=" 151,147,120 "



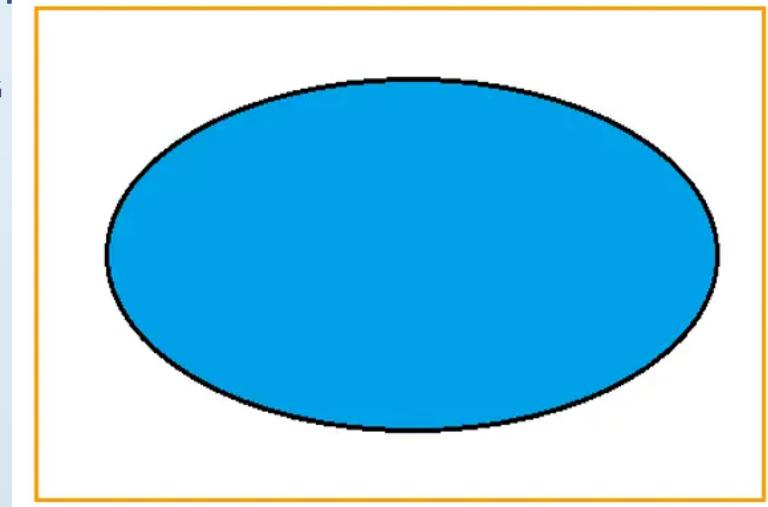
```


<map name="circlemap">
  <area shape="circle" coords="151,147,120" alt="circle" href="https://en.wikipedia.org/wiki/Circle">
</map>
```

Image map- polygon

- Polygon coords have the values x1,y1,x2,y2,...,xn,yn. The coordinates of the edges of the polygon are specified by this value. The browser will add the last coordinate pair to close the polygon if the first and last coordinate pairs are not the same

coords=" 48,163,260,56,465,166,267,270,48,163 "



```
  
  
<map name="polymap">  
  <area shape="poly" coords="48,163,260,56,465,166,267,270,48,163" alt="polygon" href="https://en.wikipedia.org/wiki/Polygon">  
</map>
```

Server-Side Image Maps

- An image map file is created and stored on the web server. The map file can be either in map format (.map), CGI, or PHP.
- The mapped image is displayed in a browser and linked to the map file.
- When a visitor clicks on a portion of the image, the clicked coordinates are sent to the server.
- The server processes the coordinates based on the instructions contained within the map file, identifies the appropriate hyperlink, and sends the visitor to the linked destination.
- The **ismap** image attribute is used to identify an image as part of a server-side image map, and the img tag is wrapped in an anchor element which points toward the map file.
- ` `

<form> tag in HTML

- The form tag in HTML is used to create a form for taking user input on a webpage.
- This form could contain multiple fields such as Name, Address, and Phone Number that can be used to collect information about the user or for collecting feedback from the user about a particular commodity.
- The data collected into these forms are passed to the specified URL in HTML.
- These form tags can be used to implement Log-in features on a website or as registration forms on a website etc.

<form> tag in HTML

- The form tag in HTML is used to create a form for taking user input on a webpage.
- Forms in HTML help the user to enter certain details asked and send them further to the server for processing.
- This form could contain multiple fields such as Name, Address, and Phone Number that can be used to collect information about the user or for collecting feedback from the user about a particular commodity.
- The data collected into these forms are passed to the specified URL in HTML.
- These form tags can be used to implement Log-in features on a website or as registration forms on a website etc.

What is the use of forms in HTML?

- **Forms in HTML are required to collect information or data from the user.** For example, registration form, form for user's order details, inquiry forms, etc.

Let's say you want to make an account on any social media platform; there will be an option for Sign-up, which redirects the user to a page where the user has to input certain details like Name, Email, Date of Birth, Passwords etc., that can be sent to the server.

Syntax Of The Form Tag

The syntax of the form tag in HTML is as follows:

```
<form> <!-- statements --> </form>
```

Forms Attribute

Attributes	value	Description
action	URL	This attribute tells us where to send the data that the user will submit.
autocomplete	ON/OFF	This attribute tells us whether the form should have the autocomplete feature or not.
method	get/post	This attribute specifies which HTTP method to use while sending the form's data.
name	text	This attribute tells us the name of the form.
target	_blank,_self, _parent, _top	This attribute specifies where the form data has to be displayed after the form is submitted.
novalidate	novalidate	This attribute tells whether the form should be validated or not when submitted.
accept-charset	charset	This attribute is used to specify the character encoding that is to be used for the form submission.
rel	help, prev, external,nofollow	This attribute is used to show the relationship between the current document with the linked resource.
enctype	Multipart/form-data	Only for method="post" , This attribute specifies how the form data should be encoded while submitting it to the server.

Forms <input> tag

<input>

- It defines a data input field for the user. It plays important part in Html Forms.
- These are the following input types -

<input type = "radio">

<input type = "password">

<input type = "number">

<input type = "checkbox">

<input type = "date">

<input type = "time">

<input type = "range">

<input type = "email">

<input type = "reset">

<input type = "submit">

Forms <textarea> & <button> tag

<textarea>

If you want a user to enter a paragraph like tell me about yourself, then we use textarea; in short, if you want multi-line input, we use textarea.

Syntax : <textarea id= "value" rows = "2" cols = "10"> </textarea>

<button>

<button> tag is used to create clickable button which can take tags like ,<i>,
 etc. It is necessary to specify the type attribute for the button to tell the browser what kind of button it is.

Syntax : <button type="button">Click here</button>

Forms <select> -Combo Box

<select>

The select tag is used to take user input and create a drop-down list. The two required attributes are <name> and <id>. The <name> attribute is required to refer to the form data after the submission of the form.

The select tag has a <option> tag, which defines the options available in the drop-down menu.

Example

```
<select name="list" id="list">  
    <option value="About">About</option>  
    <option value="Contact">Contact Us</option>  
    <option value="team">Team</option>  
</select>
```

Forms <select> -Combo Box

<option> : We use the <option> tag to make choices in the drop-down menu.

```
<select name="select" id="menu">  
    <option value="about">About</option>  
    <option value="contact">Contact Us</option>  
</select>
```

<optgroup> : It is used to group related alternatives within the drop-down menu.

```
<select name="file" id="file">  
    <optgroup label="Food">  
        <option value="Chinese">Chinese</option>  
        <option value="Italian">Italian</option>  
    </optgroup>  
</select>
```

Forms <fieldset>,<label>,<output> tag

<fieldset>

The `fieldset` tag is utilised to gather together related items in the form and create a box around the related items. For example, to gather information like personal details of the user, we can group the details required separately.

<label>

For the `<input>` tag, the `<label>` tag specifies a text label. The label is a regular text that allows the user to pick a form element by clicking it.

<output>

It specifies the final result of the calculation performed by input data. It is recently introduced in HMTL5.

<legend>

A legend tag defines the caption or title for the `<field set>` tag, and it is the element's first child. Because it is a caption, it normally sits on top of the frame.

Example : code

```
<form id="survey-form">
  <h2> Survey Form</h2>
  <div>Thank you for taking the time to help us improve the platform.</div>
  <div class="form-group">
    <label id="name-label" for="name">Name</label>
    <input type="text" name="name" id="name" class="form-control" placeholder="Enter your name" required/>
    <label id="email-label" for="email">Email</label>
    <input type="email" name="email" id="email" class="form-control" placeholder="Enter your Email" required/>
    <label id="number-label" for="number">Age<span class="clue">(optional)</span></label>
    <input type="number" name="age" id="number" min="10" max="99" class="form-control" placeholder="Age"/>
  <p>Which option best describes your current role?</p>
  <select id="dropdown" name="role" class="form-control" required>
    <option disabled selected value>Select current role</option>
    <option value="student">Student</option>
    <option value="job">Full time Job</option>
    <option value="learner">Full time Learner</option>
    <option value="preferNo">Prefer not to say</option>
    <option value="other">Other</option>
  </select>
  <button type="submit" id="submit" class="submit-button">Submit</button>
</label>
</form>
```

Example: output

The screenshot shows a survey form titled "Survey Form" with a light gray background and a white content area. The title is in large, bold, black font. Below it is a message: "Thank you for taking the time to help us improve the platform". There are three input fields: "Name" with placeholder "Enter your name", "Email" with placeholder "Enter your Email", and "Age(optional)" with placeholder "Age". Below these fields is a question: "Which option best describes your current role?". Underneath the question are two buttons: "Select current role" with a dropdown arrow icon and "Submit".

Survey Form

Thank you for taking the time to help us improve the platform

Name

Email

Age(optional)

Which option best describes your current role?

Forms Example

```
<form>
  <label for="name">Name:</label>
  <input type="text" id="name" name="name">
  <br>
  <br>
  <label for="email">Email:</label>
  <input type="email" id="email" name="email">
  <br>
  <br>
  <label for="phone">Phone:</label>
  <input type="tel" id="phone" name="phone">
  <br>
  <br>
  <input type="submit" value="Submit">
</form>
```

HTML Form

Name:

Email:

Phone:

HTML forms with checkboxes

- Checkboxes are square boxes that let the user select one or more options from a list of limited options. These checkboxes are selected when clicked on the square box or their label.
- For example, these checkboxes are used when you have to select multiple subjects to study in your semester from a list of options provided to you.
- It is created using the input element of the form tag along with the attribute **type="checkbox"**.
- If you want a checkbox to be by default selected, then you can either write **checked** or **checked="true"** as an attribute in the input element of the form tag. Both of them will, by default, select the particular checkbox.

HTML forms with checkboxes

```
<form>
  <input type="checkbox" id="Artificial Intelligence" name="Artificial Intelligence" value="subject1">
  <label for="subject1">Artificial Intelligence</label><br>

  <input type="checkbox" id="Operating System" name="Operating System" value="subject2">
  <label for="subject2"> Operating System</label><br>

  <input type="checkbox" id="Computer Networks" name="Computer Networks" value="subject3">
  <label for="subject3"> Computer Networks</label><br>

  <input type="checkbox" id="Compiler Design" name="Compiler Design" value="subject4">
  <label for="subject4"> Compiler Design</label><br>

  <input type="checkbox" id="Computer Graphics" name="Computer Graphics" value="subject5">
  <label for="subject5"> Computer Graphics</label><br><br>

  <input type="submit" value="Submit">
</form>
```

Select the subjects you want to opt for this semester.

- Artificial Intelligence
- Operating System
- Computer Networks
- Compiler Design
- Computer Graphics

Submit

HTML forms with radiobuttons

- Radio buttons are also used to make selections from a list of options, just like a checkbox. However, the **difference** between a checkbox and a radio button is that **we can select multiple options in a checkbox, whereas the radio button lets you select only one option from a list of options.**
- The radio buttons are represented in radio groups. **Radio groups** are a collection of radio buttons which has a set of related options. Also, in this collection of radio buttons, only one of them has to be selected at a particular time.
- All the radio buttons in a particular group must have the same name attribute in the **input** tag to be treated as a group. Once a particular option is selected, then all other options of that group will be automatically deselected.
- However, the **value** attribute of the input tag has to be different because it specifies the unique value with which it is identified and is associated with each option that is sent to the server.

HTML forms with radio buttons

```
<form>
<input type="radio" id="CSE" name="Branch" value="CSE">
<label for="CSE">Computer Science and Engineering</label>
<br>
<input type="radio" id="IT" name="Branch" value="IT">
<label for="IT">Information Technology</label>
<br>
<input type="radio" id="ECE" name="Branch" value="ECE">
<label for="ECE">Electronics and Communication Engineering</label>
<br>
<h2>Select the semester in which you are:</h2>
<input type="radio" id="I" name="sem" value="I">
<label for="I">First Semester</label><br>
<input type="radio" id="II" name="sem" value="II">
<label for="II">Second Semester</label><br>
<input type="radio" id="III" name="sem" value="III">
<label for="III">Third Semester</label><br>
<input type="radio" id="IV" name="sem" value="IV">
<label for="IV">Fourth Semester</label><br>
<input type="radio" id="V" name="sem" value="V">
<label for="V">Fifth Semester</label><br>
<input type="radio" id="VI" name="sem" value="VI">
<label for="VI">Sixth Semester</label><br>
<br> |
<input type="submit" value="Submit">
</form>
```

Select the branch in which you are:

- Computer Science and Engineering
- Information Technology
- Electronics and Communication Engineering

Select the semester in which you are:

- First Semester
- Second Semester
- Third Semester
- Fourth Semester
- Fifth Semester
- Sixth Semester

Submit

CSS float Property

- The CSS float property allows a developer to incorporate table-like columns in an HTML layout without the use of tables.
- If it were not for the CSS float property, CSS layouts would not be possible except using absolute and relative positioning — which would be messy and would make the layout unmaintainable.
- The purpose of the CSS float property is, generally speaking, to push a block-level element to the left or right, taking it out of the flow in relation to other block elements.
- This allows naturally-flowing content to wrap around the floated element.

CSS float Property :Layout Example

<http://web.simmons.edu/~grovesd/comm328/modules/layout/floats>



CSS float Property

- The float CSS property can accept one of 4 values: **left**, **right**, **none**, and **inherit**. It is declared as shown in the code below.

```
#sidebar {  
    float: left;  
}
```

- The most commonly-used values are left and right. A value of none is the default, or initial float value for any element in an HTML page. The value inherit, which can be applied to nearly any CSS property, does not work in Internet Explorer versions up to and including 7.
- The float property does not require the application of any other property on a CSS element for float to function correctly, however, float will work more effectively under specific circumstances.

CSS clear Property

- The CSS clear property is a complementary property to float. You can use it when you want some elements to be free from the influence of floated elements. You can set an element to be “cleared” on one side, or both sides. The cleared element will be moved below any floating elements that precede it.

The clear property has a similar set of values as float:

- none: the element is not moved down to clear past floating elements
- left: the element is moved down to clear past left floats
- right: the element is moved down to clear past right floats
- both: the element is moved down to clear past both left and right floats
- inline-start: the element is moved down to clear floats on the start side of its containing block; that is, the left floats on left-to-right scripts and the right floats on right-to-left scripts
- inline-end: the element is moved down to clear floats on the end side of its containing block; that is, the right floats on left-to-right scripts and the left floats on right-to-left scripts

CSS float,clear Property Example

```
<!doctype html>
<html lang="en">
<head>
<title>CSS Layout</title>
<meta charset="utf-8">
<style>

  *{ margin:0; box-sizing:border-box; }

body{
  font:16px/1.3 sans-serif;
}

.container{
  width:1200px;
  margin:auto;
}

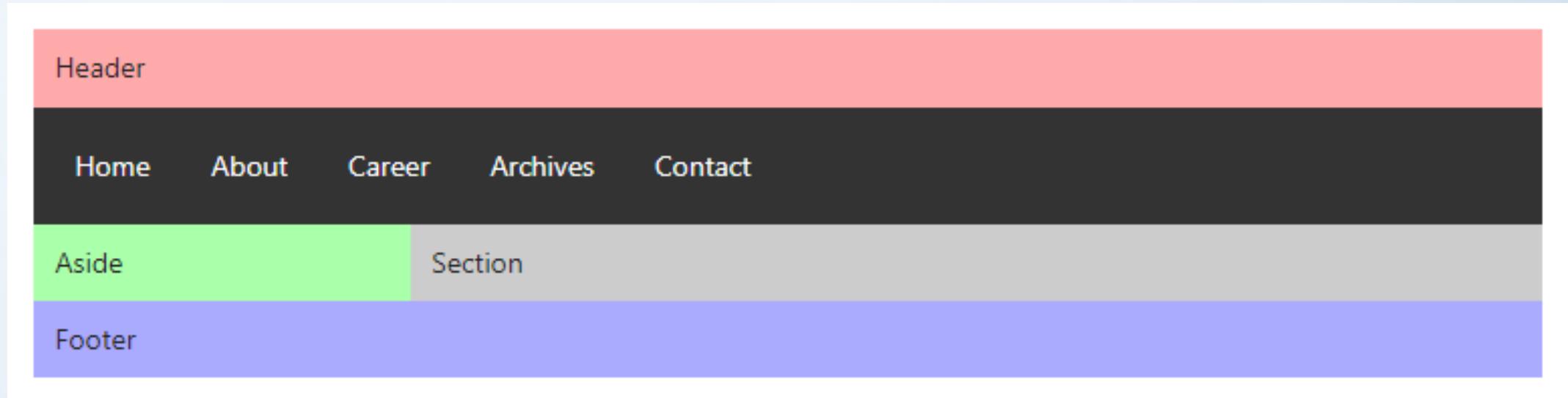
.header{
  background:#faa;
  padding:10px;
}

.nav{
  background:#333;
  padding:10px;
  color:#fff;
}
```

```
.nav ul{
  list-style:none;
  padding:0;
}
.nav ul li{
  float:left;
  margin-right:10px;
}
.nav ul li a{
  color:inherit;
  display:block;
  padding:8px 12px;
}
.main{
  background:#afa;
}
.aside{
  width: 25%;
  float:left;
  padding:10px;
  background:#afa;
}
.section{
  width: 75%;
  float:left;
  padding:10px;
  background:#ccc;
}
```

```
.section{
  width: 75%;
  float:left;
  padding:10px;
  background:#ccc;
}
.clear{
  clear:both;
}
.footer{
  background:#aaf;
  padding:10px;
}
</style>
</head>
<body>
  <div class="container">
    <div class="header"> header </div>
    <div class="nav"> nav </div>
    <div class="main">
      <div class="aside">aside</div>
      <div class="section">section</div>
      <div class="clear"></div>
    </div>
    <div class="footer"> footer </div>
  </div>
</body>
</html>
```

CSS float,clear Property Example



<https://tutorial.techaltum.com/css-layout-using-float.html>

CSS Position Property

- The position property specifies that an element is positioned.
- Positioned elements do not participate in the standard HTML document flow.
- Instead they specify their position relative to a container element.
- Valid position values include fixed, absolute, sticky, and others.
- The position property indicates what positioning method the element will follow.
- Valid values include: **static**, **relative**, **fixed**, **absolute**, and **sticky**.
- A positioned element uses the top, right, bottom, and left properties to align itself inside its container.

Position: relative

- Any element with position: relative is positioned relative to its normal position.
- The top, right, bottom, and left properties move the element from its original position.

```
<style>
  .position-relative {
    position: relative;
    max-width: 250px;
    left: 50px;
    border: 3px solid #6266f1;
    padding: 15px;
  }
</style>

<div class="position-relative">
  Position: relative
</div>
```

Position: absolute

- An element with position: absolute is positioned relative to the nearest positioned parent.
- The top, right, bottom, and left properties align the element relative to its nearest positioned container.

```
<style>
    div.relative {
        position: relative;
        max-width: 400px;
        height: 200px;
        border: 3px solid #6266f1;
        padding: 15px;
    }

    div.absolute {
        position: absolute;
        bottom: 10px;
        right: 40px;
        width: 200px;
        height: 100px;
        border: 3px solid #c6d2fe;
        padding: 15px;
    }
</style>

<div class="relative">
    Position: relative
    <div class="absolute">Position: absolute</div>
</div>
```

Position: sticky

- An element with position: sticky is positioned based on the page's scroll position.
- A sticky element toggles between relative and fixed depending on scroll position.
- The element is positioned relative, until a given offset position is met in the viewport.
- Once the offset is met, the element sticks in place like a fixed positioned element.

```
<style>
  div.sticky {
    position: sticky;
    top: 0;
    padding: 5px;
    background-color: #ff7461;
    border: 2px solid #F45D48;
    color: white;
  }
</style>

<div class="sticky">I am sticky!</div>

<div style="padding-bottom: 400px">
  <p>
    The sticky element sticks to the top
    of the page (top: 0), when reaching
    its scroll position.
  </p>
  <p>Scroll to see stickyness.</p>
</div>
```

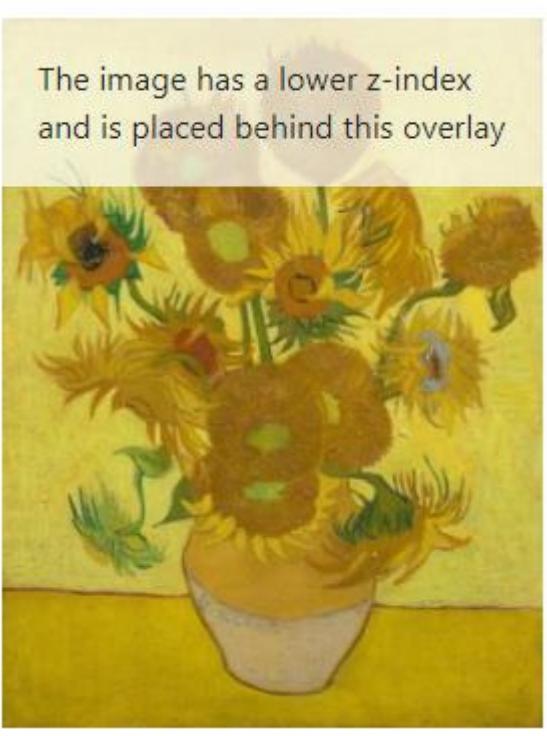
Z-index : Overlapping Elements

- Positioned elements can overlap other elements.
- Use the z-index property to specify the stack order of an element.
- Higher z-index values are in front, while lower value elements are behind.

```
<style>
.overlay-img {
  position: absolute;
  z-index: -1;
}

.overlay-text {
  background: rgba(255,255,255,0.7);
  padding: 20px;
  font-size: 18px;
  width: 300px;
}
</style>


<div class="overlay-text">
  The image has a lower z-index and is
  placed behind this overlay
</div>
```



Z-index Example

```
<style>
.art { position: relative; }

.overlay {
  position: absolute;
  bottom: 0;
  left: 0;
  background: rgba(255,255,255,0.8);
  padding: 10px;
  width: 300px;
  text-align: center;
  font-weight: bold;
  color: #765;
  z-index: 100;
}
</style>

<div class="art">
  
  <div class="overlay">The Card Players, by Paul Cézanne</div>
</div>
```



The Card Players, by Paul
Cézanne

Flex based layout

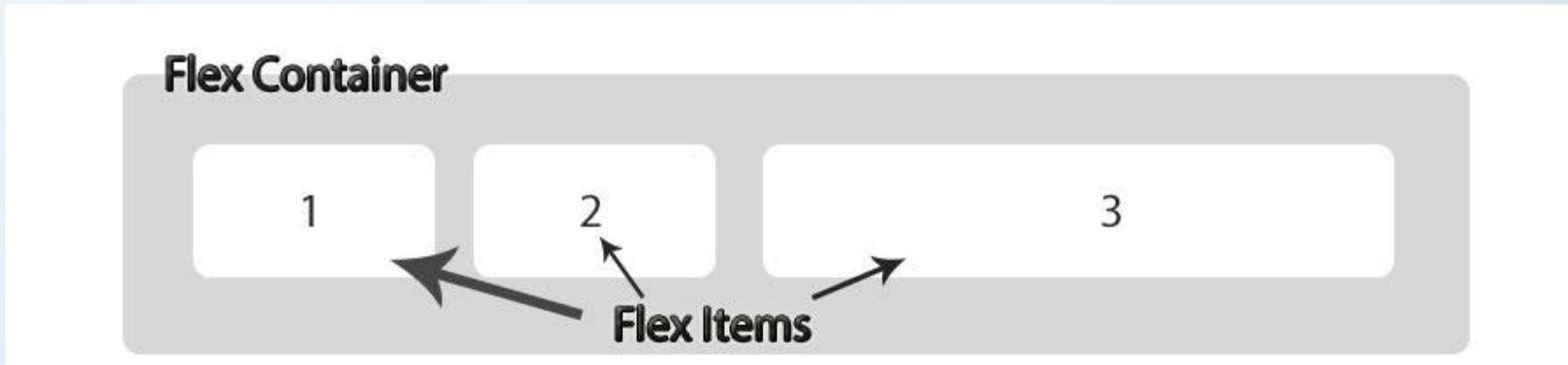
- The flex box is CSS Layout Design build using display:flex property. Flexbox is used to build one-dimensional layout in css. One dimensional means flexbox can build layout in one dimension (either row or column) at one time.
- Display flex or inline-flex is used to build flexbox. Flex can build one dimension layout which is better than float based layout.

Advantage of using flexbox?

- Better layout designs
- Supports IE 10 and above browsers.
- Auto height and width option available.
- Easy to change direction of flex columns.

Flexbox properties

- Flex is the value of css **display** . By using display **flex** in parent element, child elements automatically align like column or row with auto width and auto height.

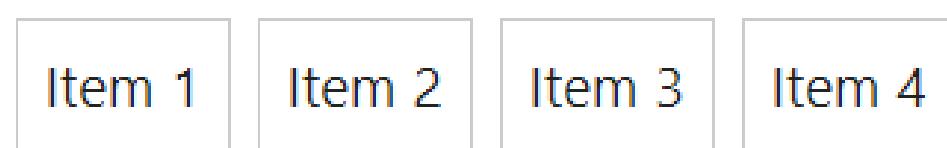


Display:Flex

- **Display flex** is the property of flex container to use flexbox. **CSS Display** property can have value **flex** or **inline-flex**. By using display flex or inline-flex to parent container, the children automatically enable flex content.

```
<style>
*{ margin:0; box-sizing:border-box}
.flex-container{
    display: flex;
}
.flex-item{
    padding:10px;
    border: 1px solid #ccc;
    margin: 5px
}
</style>
```

```
<div class="flex-container">
    <div class="flex-item">Item 1</div>
    <div class="flex-item">Item 2</div>
    <div class="flex-item">Item 3</div>
    <div class="flex-item">Item 4</div>
</div>
```



Flex Direction

- **Flex Direction** property is given to **flex container** to **change direction of flex items**. By default, flex direction is row.

Flex Directions value

- row (default)
- row-reverse
- column
- column-reverse

```
<style>
*{ margin:0; box-sizing:border-box}
.flex-container{
    display:flex;
    flex-direction: row;
}
.flex-item{
    padding:0 10px;
    border: 1px solid #ccc;
    margin: 5px
}
</style>
```

```
<div class="flex-container">
    <div class="flex-item">Item 1</div>
    <div class="flex-item">Item 2</div>
    <div class="flex-item">Item 3</div>
    <div class="flex-item">Item 4</div>
</div>
```

Justify Content

- Justify-content property is used to justify content in main axis. This can adjust items to left, center, right or add space in between.

justify-content values

- flex-start (default)
- flex-end
- center
- space-between
- space-around
- space-evenly

```
nav.container{  
    display: flex;  
    justify-content: center;  
}  
  
<nav class="container">  
    <div>Home</div>  
    <div>Search</div>  
    <div>Logout</div>  
</nav>
```

The flex property

- By default we have fixed-width items. But what if we want them to be responsive or custom width or similar width? To achieve that we have a property called **flex**. It makes it a lot easier than the old way of using percentages.
- Simply target all the items and give them a **flex** value of **1**.

```
.container > div {  
    flex: 1;  
}
```

Simply target specific item value and give them a **flex** value of **2**.

```
.container > div.home {  
    flex: 2;  
}
```

Flex wrap

- Flex items will always fit in one row even if content is more. flex wrap is used to allow items to wrap in next line or wrap in reverse direction.

Flex Wrap values

- nowrap (default)
- wrap (wrap to next line)
- wrap-reverse (multiple line but in reverse from bottom to top)

Flex Flow

- flex flow is the shorthand for flex-direction and flex-wrap properties. The default value is row nowrap, column nowrap.

```
<style>
```

```
*{ margin:0; box-sizing:border-box}
```

```
.flex-container{
```

```
    display:flex;
```

```
    flex-flow: column wrap;
```

```
}
```

```
</style>
```

Align items

- Align-items property define the behavior of how flex item laid out across horizontal axis. By-default, the value is stretch.

Values of align-items

- stretch
- flex-start
- flex-end
- center
- baseline

Align Self

- **align-self** is the property of flex items. It is used to align an individual flex item on y-axis.

Align Self Values

- auto
- flex-start
- center
- flex-end
- baseline

```
<style>
    .flex-container{
        display:flex;
        flex-flow:row wrap;
        height:150px;
    }
    .flex-item{
        padding:0 10px;
        border: 1px solid #ccc;
        margin: 5px;
        flex:1 0 auto;
    }
    .flex-item:nth-child(1){align-self:flex-start}
    .flex-item:nth-child(2){align-self:center}
    .flex-item:nth-child(3){align-self:flex-end}
</style>
```

```
<div class="flex-container">
    <div class="flex-item"> 1 </div>
    <div class="flex-item"> 2 </div>
    <div class="flex-item"> 3 </div>
</div>
```

More on flexbox?

- <https://torquemag.io/2020/02/flexbox-tutorial/>
- <https://medium.com/@jillplatts/6-minute-beginners-guide-to-css-flexbox-527b3ff3480b>

What is Semantic HTML?

- Semantic HTML means that your HTML tags convey the actual meaning of what they are used for.
- Semantics has been an integral part of HTML since its inception in the early 90s. But it never gained particular relevance until the late 90s when CSS started working in most browsers.
- With semantic HTML, semantically-neutral tags such as `<div>` and `` are frowned upon since semantically more descriptive tags such as `<header>`, `<nav>`, `<main>`, `<section>`, `<footer>` and `<article>` can do the same thing they do.
- A noticeable advantage of using semantic tags is that web crawlers are able to index the web page or website easily, improving SEO in return.
- In addition, a website that uses semantics becomes more informative, adaptable, and accessible to those who use screen readers to access websites.

Important Semantic Tags and What they Do

- <header>: The <header> element defines the introductory section of a web page. It contains items such as the logo, navigation, theme switcher, and search bar.
- <nav>: The <nav> element specifies the navigation items of the page such as home, contact, about, FAQs, and so on.
- <main>: The <main> element is conventionally treated as the immediate descendant of the tag. It contains the main sections of the HTML document apart from <header> and <footer>. Ideally, there should be just one of these in the whole HTML document.

Important Semantic Tags and What they Do

- <section>: The <section> element defines a particular section of the web page. This may be the showcase section, about section, contact section, or others. You can use numerous sections in a single HTML document.
- <article>: The <article> element represents a certain part of a web page that conveys some particular information. Such information could be a combination of text, images, videos, and embeds. Look at this element as a standalone blog post on a page containing excerpts about other blog posts.
- <aside>: As the name implies, this represents a sidebar on a web page. It is usually a part of the web page that is not directly related to the main content.

Important Semantic Tags and What they Do

- <footer>: The <footer> element accommodates items such as quick links, copyright information, or any other data related to the entire website or web page.

Note that since semantic elements convey actual meaning and what some particular content actually does (such as nav for navigation, aside for a sidebar, and so on), these elements are not automatically positioned where they are supposed to be.

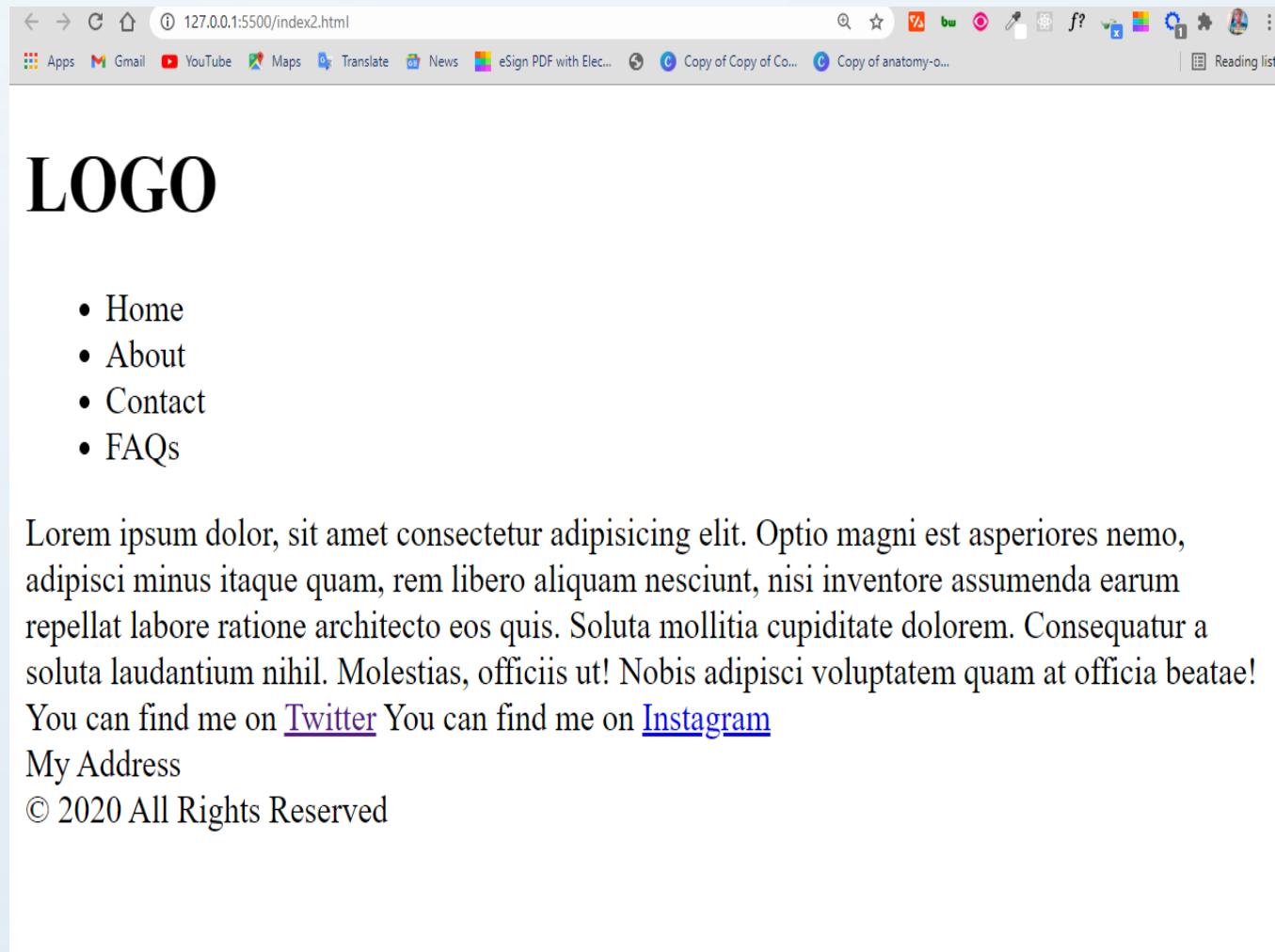
A super simple semantic HTML document

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Definition of HTML</title>
  </head>

  <body>
    <header>
      <h1 class="logo">LOGO</h1>
      <nav>
        <ul>
          <li>Home</li>
          <li>About</li>
          <li>Contact</li>
          <li>FAQs</li>
        </ul>
      </nav>
    </header>
```

```
<main>
  <section class="about-me">
    <article>
      Lorem ipsum dolor, sit amet consectetur adipisicing elit. Optio magni
      est asperiores nemo, adipisci minus itaque quam, rem libero aliquam
      nesciunt, nisi inventore assumenda earum repellat labore ratione
      architecto eos quis. Soluta mollitia cupiditate dolorem. Consequatur a
      soluta laudantium nihil. Molestias, officiis ut! Nobis adipisci
      voluptatem quam at officia beatae!
    </article>
  </section>
  <section class="contact-me">
    You can find me on
    <a href="https://twitter.com/koladechris">Twitter</a> You can find me on
    <a href="https://Instagram.com/koladechris">Instagram</a>
  </section>
  <aside class="address">My Address</aside>
</main>
<footer>© 2020 All Rights Reserved</footer>
</body>
</html>
```

A super simple semantic HTML document-output



The screenshot shows a web browser window with the URL 127.0.0.1:5500/index2.html. The page content is as follows:

LOGO

- Home
- About
- Contact
- FAQs

Text

• Lorem ipsum dolor, sit amet consectetur adipisicing elit. Optio magni est asperiores nemo, adipisci minus itaque quam, rem libero aliquam nesciunt, nisi inventore assumenda earum repellat labore ratione architecto eos quis. Soluta mollitia cupiditate dolorem. Consequatur a soluta laudantium nihil. Molestias, officiis ut! Nobis adipisci voluptatem quam at officia beatae!

You can find me on [Twitter](#) You can find me on [Instagram](#)

My Address

© 2020 All Rights Reserved

You can see that the content inside the `<aside>` tag isn't in the sidebar and the content inside the `<nav>` tag is not automatically available as the navigation bar. This is why you still have to make them look the way they are supposed to look with CSS.

Responsive Website

- Responsive Design is the practice of making sure your content looks good on all screen sizes.
- Everything in the website including layouts, fonts and images should automatically adapt to the user's device.
- In the early 2000's, developers focused on making sure their websites looked good on larger screen sizes like laptops and desktop computers, but In today's world, you have to consider devices like mobile phones, tablets, and even watches.

Media Query?

- In CSS, a media query is used to apply a set of styles based on the browser's characteristics including width, height, or screen resolution.

Basic syntax of a media query

```
@media media-type (media-feature){  
/*Styles go here*/  
}
```

Media Query?

- Break down syntax of media query
 - The @media is a type of At-rule in CSS. These rules will dictate what the CSS will look like based on certain conditions.
 - The **media type** refers to the category of media for the device. The different media types include all, print, screen and speech.
 - all - works for all devices
 - print - works for devices where the media is in print preview mode
 - screen - works for devices with screens
 - speech - works for devices like screen readers where the content is read out loud to the user
 - Except when using the **not** or **only** logical operators, the media type is optional and the all type is implied.

Media-feature

- Break down syntax of media query
 - You can choose to **omit** the media type and use this syntax instead.

```
@media (media-feature){  
/*Styles go here*/  
}
```

- The **media feature** refers to the characteristics of the browser which include height and width of the viewport, orientation or aspect-ratio

Media Query?

- Break down syntax of media query
 - If you wanted to create more complex media queries, then you can use logical operators.
 - **and** - This operator is used to join multiple media features. If all of the media features are true then the styles inside the curly braces will be applied to the page.
 - **not** - This operator reverses a true query into a false and a false query into a true.
 - **, (comma)** - This operator will separate multiple media features by commas and apply the styles inside the curly brace if one of the conditions is true.

Media Query Example

- If we want the background color to change to blue when the width of the device is 600px or less.
- In the CSS, we want to add a (**max-width: 600px**) for the media query which tells the computer to target devices with a screen width of 600px and less.

```
@media (max-width: 600px) {  
    body {  
        background-color: #87ceeb;  
    }  
}
```

Media Query Example

- If we want to change the background color from **blue** to **red** if the device has a width between 600 and 768px. We can use the **and** operator to accomplish this.

```
@media (min-width: 600px) and (max-width: 768px) {  
    body {  
        background-color: #de3163;  
    }  
}
```

Media Query Example

- If we want to change the background color from **blue** to **red** if the device has a width between 600 and 768px. We can use the `and` operator to accomplish this.

```
@media (min-width: 600px) and (max-width: 768px) {  
    body {  
        background-color: #de3163;  
    }  
}
```

Device Breakpoint

- Some common breakpoints used for media queries.

320px—480px: Mobile devices

481px—768px: iPads, Tablets

769px—1024px: Small screens, laptops

1025px—1200px: Desktops, large screens

1201px and more— Extra large screens, TV

Practice Theory Question.

- What is wireframe? Explain advantage of wireframe with example
- Explain web development process with major steps.
- Difference between webpage and website
- Difference between Web Client and Web Server
- Difference between website and web application
- Difference between Static and Dynamic Website
- What is wireframe? Explain advantage of wireframe with example
- Explain web development process with major steps.
- Difference between webpage and website
- Difference between Web Client and Web Server
- Difference between website and web application
- Difference between Static and Dynamic Website

Practice Theory Question.

- Write Basic HTML program structure.
- What is HTML Attribute? Explain different type of HTML attribute with some example.
- Explain different types of HTML element with example.
- Difference between paired and empty tag.
- What is HTML entities? List out commonly used HTML entity references.
- What is CSS? Explain different types of CSS with suitable example
- Explain advantage and disadvantage of using CSS.
- Why CSS is important for website? Explain different version of CSS with its major features.
- What is uses of CSS selector? Explain different CSS selector with example
- What is anchor tag? Explain different html attribute associated with anchor tag.

Practice Theory Question.

- What is CSS Box model? Explain different properties associated with box model
- Explain img tag with it's related attribute.
- What is HTML list? Explain different types of HTML list with example
- What is semantic tag in HTML? Why semantic tags are important for web development.
- Explain different types of link used into HTML ?
- What is Image Map? Difference between client side and server side image map.
- Explain different types of client side imagemap shape with proper example.
- What is HTML table? Explain importance of html table with pros and cons.
- List and explain html table tags and attributes with example.
- Explain various html 5 table tags and attribute.
- What is HTML frame? Why frame layout is important for web site.
- Differentiate between frameset and frame tag.

Practice Theory Question.

- How to define nested frame? Explain with rows and cols attribute.
- Difference between frame and iframe.
- What is HTML Form? Explain importance of form for web application.
- Explain different form elements and attributes with example.
- Explain html form elements and attributes related to html5.
- How to group form elements in standard way for better user experience.
- Explain html5 related form attributes for form validation.
- Explain importance of hidden element for html form.
- Explain different form related events with example.

Any Questions?

Thank you.

