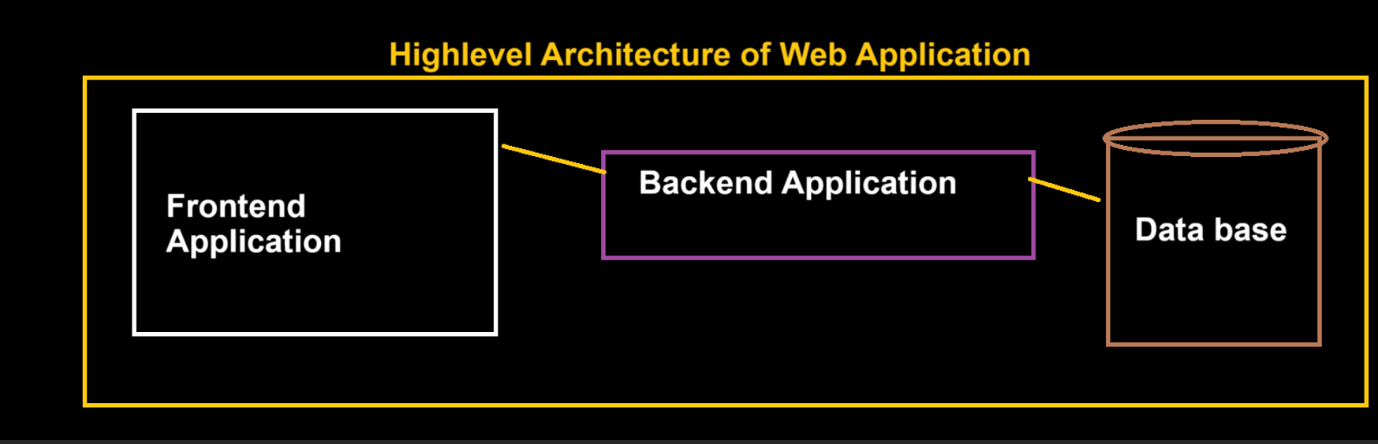
**What is mean by Web Application?**

Any application which is accessed by web is called web application.

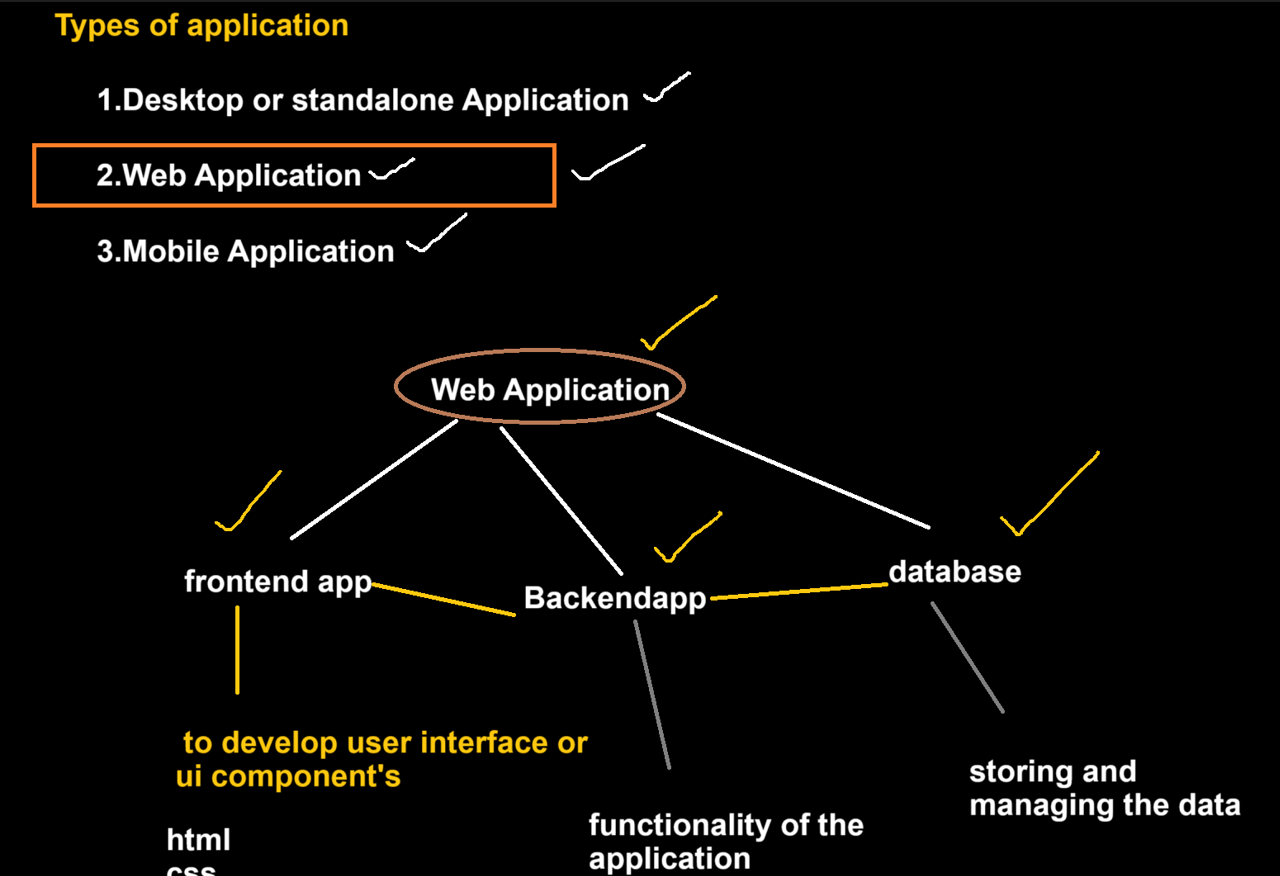


**FRONT END APP or client-side App.**

**BACK-END APP or Server-side App.**

**DATABASE**

**Types Of applications:**



**HTML**

**What is HTML?**

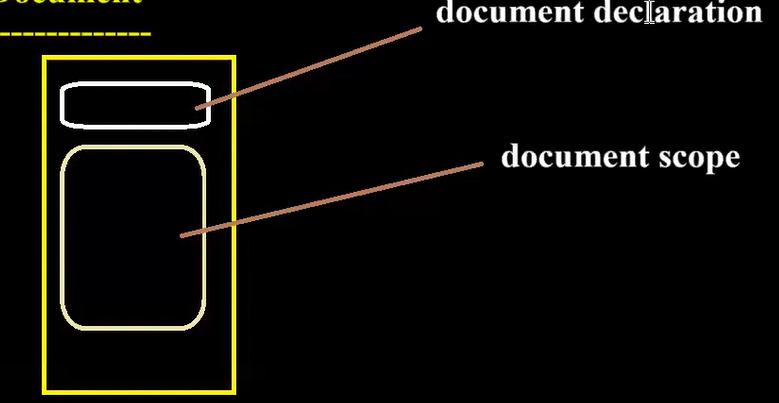
**HTML (HYPER TEXT MARKUP LANG)** is a markup language.

**Why to Learn HTML and its purpose?**

HTML is used to create User Interface components (collection of UI components).

**HTML Document Structure**

<!DOCTYPE html> (Document declaration: version of html)



<html>

<head>

    <title>Document Title</title> Title for the page

</head>

<body>

    <! -- Content goes here -->

</body>

</html>

**\*HTML is a language used to create UI components, html will provide elements and each and every element provides some format called as tags with help of these creating UI components.**

**HTML Meta Tags ®**

**○ Charset**

   <meta charset=“utf-8”> specifies page description, keywords, copyright, language, author of the documents, etc.

**○ Name**

**○ Content**

**○ Http-equiv (refresh)**

**HTML Element?**

HTML provide element to create UI components, using this element we can create UI Components.

Each and every element must have tags.

Building blocks of a web page.

**HTML Tags**

**○ Container tag (opening and closing tag <\_> </\_>)**

**○ Self Tag (only opening tag <\_/>)**

**HTML Attributes**

**“COMMON ATTRIBUTES FOR MOST OF ELEMENTS: (Id, Class, name)”**

**“SPECIFIC ATTRIBUTES MENTIONED TO THE CORRESPONDING ELEMENTS: (EX: src--**Source**, href—**hypertext reference**, rel--** relationship**, target, type, reverse, height, width so on…”)**

**HTML Comments(<!----- 🡪)**

**HTML Style Guide**

**HTML Button**

**<button>** Button name **</button>**

**HTML Headings**

* **Heading 1(<h1></h1>)**
* **Heading 2(<h12></h2>)**
* **Heading 3 (<h3></h3>)**
* **Heading 4(<h4></h4>)**
* **Heading 5(<h5></h5>)**
* **Heading 6(<h6></h6>)**

**HTML Paragraphs:**

Using this element, we can add paragraph to the document.

Navigation link can achieve using this anchor element we can connect multiple web pages.

Only one tag that is **“P”** tag, tis paired tag.

**<p> Parah </p>**

**HTML Line Breaks**

**<br/> giving next line.**

**HTML Text Formatting Elements**

* **<b> and <strong> Elements: Bold and strong are like same, for whole parah we can go with strong for particular things we can go with bold**
* **<i> and <em> Elements: Bold and strong are like same, for whole paragraph we can go with strong for particular things we can go with bold**
* **<small> Element: Parah we can go with small for particular things.**
* **<mark> Element: highlight the Parah by default in yellow colour.**
* **<del> Element: Parah will be strike out.**
* **<ins> Element: Parah we can go with small for particular things.**
* **<sub> Element: Parah will be above particular things.**
* **<sup> Element: Parah will be below particular things.**

**HTML Preformatter**

**HTML Links & Anchor Tag**

Using this element, we can create links on web page.

Navigation link can achieve using this anchor element we can connect multiple web pages.

Only one tag that is **“A”** tag, its paired tag

1.We can navigate to from one page to another page. —**INTRA NAVIGATION**

2. We can navigate from one section to other within in this same page. **INTER NAVGATION.**

**Attributes of Anchor**

* **Target:** where our new page needs to open here whether in same tab or new tab.

**\_blank: Opens in new tab.**

**\_self: By default, it is self only so no need to mention target for anchor**

**\_parent:**

**\_top: Opens in new tab.**

* **href: The path of HTML page / WEB page where we want to navigate.**
* **id**
* **class**
* **name:**
* **title: (Link will show the title message as hint for the action we are performing)**

**Difference between Anchor and Link tag.**

**Anchor:** Representation of clickable link.

**Link:** linking of external sheets, but not any clickable link

**HTML DIV**

Used to create section/layout on the web page.

Used to group multiple elements, it is a plain element.

It provides only one tag called <div> </div> and it is paired tag.

We don’t have any explicit attributes for div elements.

By default no styles will be added.

* **Section:**
* **Layout: It uses rows and columns, which can be achieved using FLEX or GRID from CSS.**

**HTML Images**

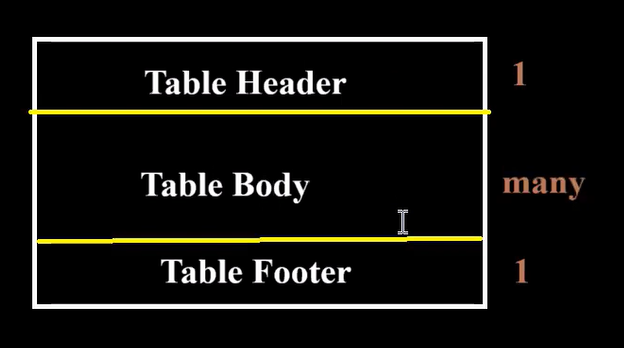
**It’s a self-closing tag <img />**

* **Src**
* **Alt**
* **Width**
* **height**

**HTML Image Links**

**HTML Tables**

Tables are often used to organize data or display information in a tabular format.



* **thead**
* **tbody**
* **tfoot**
* **tr**
* **th**
* **td**
* **Colspan**
* **Rowspan**
* **Table fully styles**
* **Nested Table in HTML**

**Attributes of table:**

Frame:

**above, below, box, lhs, rhs**

Rules:

**all, groups, rows, cols**

border**: This attribute specifies the width of the border around the table. The default value is 0, which means no border.**

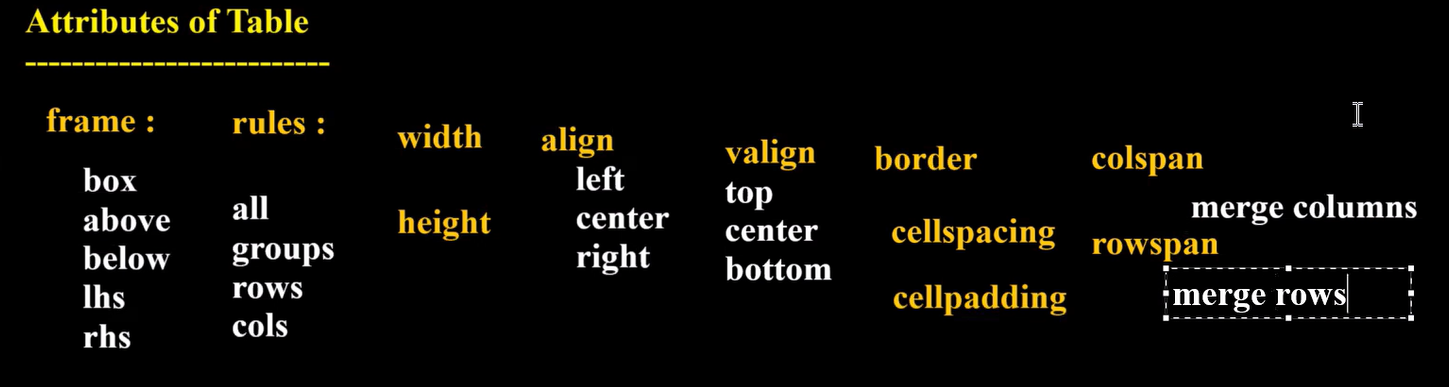
cellpadding**: This attribute specifies the padding within each table cell. The value is measured in pixels.**

cellspacing**: This attribute specifies the space between table cells. The value is measured in pixels.**

width**: This attribute specifies the width of the table. The value can be a percentage or a pixel value.**

height**: This attribute specifies the height of the table. The value can be a percentage or a pixel value.**

align**: This attribute specifies the horizontal alignment of the table within its containing element. The possible values are "left", "center", or "right".**



**HTML Lists**

* **Unordered HTML List**
* **Unordered Nested**
* **Ordered HTML List:** Used to display content in list format

**<ol>**

**<li></li>--- can have any number of li inside this ol**

**</ol>**

* **Ordered Nested**
* **Description Lists**

**HTML Block and Inline Elements:**

**Block**

**Inline:** span, strong, image, input are inline element, it will print in the same line and unable to change width and height except image.

**HTML class Attribute**

**HTML id Attribute**

**HTML Iframes :**

* **it is paired tag**
* **A way to embed another HTML document into the current document. It is commonly used to include content from other websites, such as a video, a map, or a social media widget**

**list of attributes that can be used with the element:**

**src:** This attribute specifies the URL of the document to be embedded.

**<iframe src ="http://www.example.com"></iframe>**

**width:** This attribute specifies the width of the iframe. The value can be specified as a positive integer followed by a unit of measurement, such as "px" or "em".

**<iframe src="http://www.example.com" width="500px"></iframe>**

**height:** This attribute specifies the height of the iframe. The value can be specified as a positive integer followed by a unit of measurement, such as "px" or "em".

**<iframe src="http://www.example.com" height="300px"></iframe>**

**name:** This attribute specifies a name for the iframe. This name can be used as the target of links, form submissions, and JavaScript functions.

**<iframe src="http://www.example.com" name="myframe"></iframe>**

**frameborder:** This attribute specifies whether or not the iframe should have a border. The value can be "0" or "1".

**<iframe src="http://www.example.com" frameborder="0"></iframe>**

**scrolling:** This attribute specifies whether or not the iframe should have scrollbars. The value can be "yes", "no", or "auto".

**<iframe src="http://www.example.com" scrolling="auto"></iframe>**

**allowfullscreen:** This attribute specifies whether or not the iframe should be allowed to enter fullscreen mode. The value can be "true" or "false".

**<iframe src="http://www.example.com" allowfullscreen="true"></iframe>**

**allow:** This attribute specifies a space-separated list of features that the iframe is allowed to use. Example:

**<iframe src="http://www.example.com" allow="camera; microphone"></iframe>**

**HTML Computer Code Elements ®**

* **<code> for Code Elements**
* **<kbd> For Keyboard Input**
* **<samp> For Program Output**

**HTML Semantic Elements / Tags**

* **<header> Element**
* **<footer> Element**
* **<nav> Element**
* **<section> Element**
* **<article> Element**
* **<aside> Element**

**HTML Marquees**

***it is paired tag***

* **Direction**
* **Behavior**
* **Scrolldelay**
* **Scrollamount**
* **Loop etc..**

**HTML Forms**

**It is a paired tag, using form with sub elements can create form in UI.**

**LOGIN form, SIGNIN form, QUIZ form, FEEDBACK form.**

**Form are used to take the data from users.**

* **Form Elements**
* **Form Attributes**
* **Input Types: self-closing tag, can accept lot of attributes.**
* **Placeholder**
* **Text Input Controls**
* **Checkboxes Controls**
* **Radio Box Controls**
* **Select Box Controls**
* **File Select boxes**
* **Hidden Controls**
* **Textarea**
* **Submit and Reset Button**
* **Autocomplete**
* **Required**
* **Disabled**
* **minlength and maxlength etc..**

**HTML New Input Type**

* **color**
* **Number**
* **Range**
* **Date**
* **Datetime-local**
* **Month**
* **Week**
* **Time**
* **Email**
* **Search**
* **Url**
* **Image**

**HTML Progress Tag:**

**Completion progress length**

**HTML Media**

**Video**

* **Source**
* **Autoplay**
* **Controls**
* **Loop**
* **Muted**
* **Src**
* **Poster**

**Audio**



* **Source**
* **Autoplay**
* **Controls**
* **Loop**
* **Muted**
* **Src**

**HTML CSS**



**ELEMENTS PROPERTY**



**TAGS Key:Value**



**UI Components Style to UI component**

**CSS**

**What is CSS?**

CSS(Cascading Style Sheet) is a styling language.

**Why to Learn CSS or Use of CSS?**

Designing our UI components or styling our UI components.

Used to styling the UI component or HTML elements.

**Advantages of CSS**

Provides set of or collections of Properties, using this we can style up UI components.

Every properties will give style/designs.

**CSS Selectors**

* **Simple selectors**
* **Combinator selectors:** Here elements are selected using relationship.
* **Descendant: parent element( parent space Element)**

**Example: container p{ color:red }**

**All p tag avail under that selector will apply .**

* **Adjacent relationship**

**Example: div + p{ color:red }**

**Only for the first p tag avail under that div will apply .**

* **Child: only direct child element will get its style.**

**Example: container > p{ color:red }**

**All p tag avail under that selector will apply .**

* **General sibling**

**Example: container ~ p{ color:red }**

**Only for all siblings of p tag avail under that div will apply .**

**Note: not for grand siblings**

* **Attribute selectors**

**STEPS To Add CSS:**

Select the element which you need to style it, we can select element in two ways.

|  |
| --- |
| **SYNTAX**  **Tag Name or Id Value {**  **CSS PROPERTIES**  **}** |

**How To Add CSS**

* **External CSS**
* **Internal CSS**
* **Inline CSS**

**CSS Colors**

* **RGB Colors**
* **HEX Colors**
* **HSL Colors**

**CSS Backgrounds:**

This property sets the background color of an element. You can specify the colour using a colour keyword, a hexadecimal value, hsl.

* **body {background-color: coral;}**
* **body {background-color: #92a8d1;}**
* **body {background-color: rgb(201, 76, 76);}**
* **body {background-color: hsl(89, 43%, 51%);}**

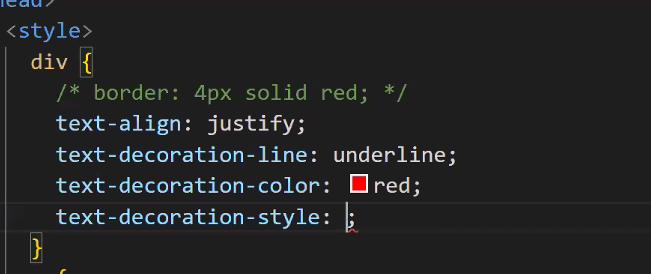
**background-repeat: no-repeat;**

**background-position:12px, 7px(denotes x-axis, y-axis)**

**background-position:12px, 7px(have top, bottom, right, left options also)**

**background-size:**

**background-attachment: (scrolling, fixed)**



**CSS Fonts Style ®**

* **font-family**
* **font-style**
* **font-weight**
* **font-size**
* **font**

**CSS Text:**

* **color**

Used to color the HTML element.

P {

Color: blue;

}

* **direction**

Usually converts text into alternate direction.

Right to left and right to left

Ex: style = “direction: ltr”

Ex: style = “direction: rtl”

* **text-align**
* **text-decoration**
* **text-transform:**

Usually converts text into upper case or lower case or capitalise format.

**Values are:**

* **uppercase**
* **lowercase**
* **capitalize**

**Ex: text-transform: uppercase**

* **word-space**

Used to generate space between the word.

**Ex: word-spacing: 20px;**

* **letter-spacing**

Used to generate space between the letter.

**Ex: letter-spacing: 10px;**

* **white-spacing**
* **text-shadow**

Used to add Shadow to the text.

**Values are:**

* **h-space v-space blur color**

**Ex: text-shadow: 50px 60px 2px red;**

* **text-indent**

Used to generate indentation for the first line of your paragraph

**Ex: text-indent: 50px;**

**CSS Box Model**

**CSS Borders**

**If use pixel it won’t depend on anything.**

**If use % div will depend on body.**

* **border-style**
* **border-width**
* **border-color**
* **border(property)**

**CSS Margins:**

**Margin property is used to create space around an HTML element.**

**Margins are the empty spaces between the element's border and the adjacent elements or space outside the border.**

**Values are applied on clock wise direction.**

**Margin: auto (equal space from left and right, exactly center)**

* **margin-top**
* **margin-right**
* **margin-bottom**
* **margin-left**
* **margin**

**CSS Padding: it use to generate space between border and content of the element (or) used to generate space inside the border.**

* **padding-left**
* **padding-right**
* **padding-bottom**
* **padding-top**
* **padding**

**CSS Width**

* **max-width**
* **min-width**

**CSS Height**

* **max-height**
* **min-height**

**CSS Links Style ®**

* **:link**
* **:visited**
* **:hover**
* **:active**

**CSS Outline**

* **outline-width**
* **outline-style**
* **outline-color**
* **outline-offset**
* **Outline**

**CSS Display Property**

* **inline**
* **inline-block**
* **block**

**CSS Float Property**

**CSS Clear Property**

**CSS Cursors Style**

**CSS Overflow Property**

* **overflow-x**
* **overflow-y**
* **overflow**

**CSS Position Property**

* **Static**
* **Absolute**
* **Relative**
* **Fixed**
* **Sticky**

**Difference between position layout none and position absolute?**

**Absolute:** related to nearest positioned ancestor or initial containing block.

**Relative:** related to normal position in the document.

**CSS Z-index Property**

**CSS Combinators**

* **descendant selector (space)**
* **child selector (>)**
* **adjacent sibling selector (+)**
* **general sibling selector (~)**

**CSS !important Rule**

**CSS Units ®**

* **(em,rem,vh, vw,vmax,vmin ,px,%)**

**CSS Pseudo Classes ®**

* **:link**
* **:visited**
* **:hover**
* **:active**
* **:focus**
* **:focus-within**
* **:enabled**
* **:disabled**
* **:checked**
* **:required**
* **:read-only / :read-write**
* **:root**
* **:first-child**
* **:last-child**
* **:nth-child()**
* **:nth-of-type()**
* **:first-of-type**
* **:last-of-type**
* **:nth-last-of-type()**
* **:nth-last-child()**
* **:only-of-type**
* **:empty**
* **::first-letter**
* **::first-line**
* **::marker (UL/OL List)**
* **::selection**

**Difference between display none and visibility hidden**

**CSS Rounded Corners**

* **border-top-left-radius**
* **border-top-right-radius**
* **border-bottom-right-radius**
* **border-bottom-left-radius**
* **border-radius**

**CSS Gradients**

**CSS Accent-Color**

**CSS filter Property**

**CSS Media Types (Responsive)**

* **@media print**
* **@media screen**

**CSS Shadow Effects**

* **Text Shadow**
* **Box Shadow**

**CSS 2D Transforms:**

* **translate() Method**
* **rotate() Method**
* **scale() Method**
* **skew() Method**

**CSS 3D Transforms**

* **rotateX() Method**
* **rotateY() Method**
* **rotateZ() Method**

**CSS Transitions**

* **transition**
* **transition-delay**
* **transition-duration**
* **transition-property**
* **transition-timing-function**

**CSS Animations**

* **@keyframes**
* **animation-name**
* **animation-duration**
* **animation-delay**
* **animation-iteration-count**
* **animation-direction**
* **animation-timing-function**
* **animation-fill-mode**
* **Animation**

**CSS Flexbox**

Flex is used to create layout for the web page in one direction.

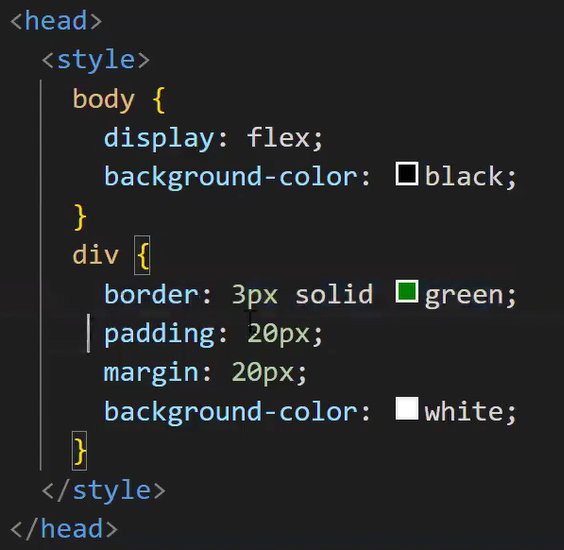
We can create layout in horizontal or vertical direction, we can create only one direction at a time.

**LAYOUT**: arranging the elements in a row or column is called layout.

Body or parent element as flex container.

Body{ display:flex; }-🡪Direct child will act as flex container of that element.

Example:



Default direction of the flex in flex container is row so the flex items will display left to right.

|  |  |
| --- | --- |
| **row-direction** | **column-direction** |
| Flex items will display in horizontal direction or x-axis line. | Flex items will display in vertical direction or y-axis line. |

**flex-direction**

It specifies the direction of flex item.

* **flex-direction: column**

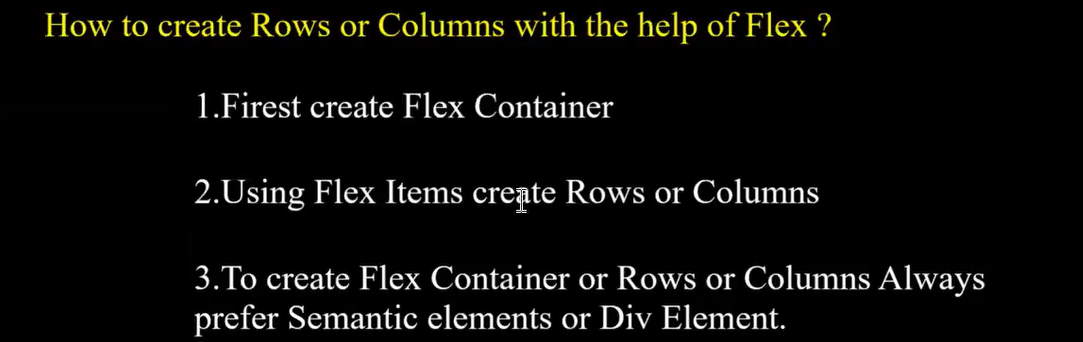
The display will be in column direction after flex-direction property is used.

* **flex-direction: row-reverse**

The flex items will display in row but from right to left.

* **flex-direction: column-reverse**

The flex items will display in row but from bottom to top.



**CREATING A FLEX IN COLUMN DIRECTION:**

**In style tag**

**<style>**

**display: flex;**

**flex-direction: column;**

**</style>**

* **flex-wrap**
* **flex-flow**
* **justify-content**
* **align-items**
* **align-content, order**
* **Grid**
* **Grid Template Columns**
* **Grid Template Rows**

**Difference between CSS FLEX and CSS GRID.?**

**FLEX:** One dimensional layout, ideal for single row or column.

**GRID:** Two-dimensional, complex layout.

**BOOTSTRAP**

**Introduction to Bootstrap? and its Features**

**Bootstrap Installation**

**Setting Development Environment**

**Grid System**

**Components**

* **Basic Typography**
* **Text Alignment & Display**
* **Colors Background:**
* **Buttons**
* **Navbars**
* **List groups**
* **Badges**
* **Forms**
* **Carousel**
* **Modal**
* **Table**
* **Cards**
* **Pagination**

**JAVASCRIPT**

**Javascript?**

Javascript is synchronous it is line by line execution.

**Debugging in Javascript**

1. console.log()

console.assert(exp,message); //only error occurs it will be printed

2. debugger; // keyword used to stop execution

3. Browser breakpoint

4. Conditional Breakpoint

Right Click => Add a conditional break point

**Destructuring?**

**Optional Chaining?**

If trying to access a property that is not existing as a part of object it will throw an error in order to overcome we hav optional chaining.

Let data ={};

Console.log(data.address)=🡺undefined.

Console.log(data.address.city)=🡺Error.

In order to avoid such error, we have an optional chaining.

Console.log(data?.address?.city)=🡺it says undefined.

**Arrow Function?**

arrow function is a simplified code, or reduce of lines.

***let addingNumber = (a,b) => { return a+b };***

***or***

**let addingNumber = (*a*,*b*) => *a*+*b*;**

**let doubleNumber = *n* => *n*\*2**

***console*.log('doubleNumber',doubleNumber(2))**

**Map?**

When you want to iterate over each and every element perform some computation or some transformation.

map always return new array by performing action on original array.

Example: **const** *number* = [2,4,6,8];

Double numbers using map.

*let doubleNumber = number.map(dum=>dum\*2)*

*or*

*let doubleNumber = number.map(function(element,index){*

*})*

***let*** *increaseSala = userObj.map(obje* ***=>***

*{obje.salary  = obje.salary +2000;*

*return obje});*

*console.log(****'increaseSala'****,increaseSala)*

*Note:if you return more values it must have return keyword in flower braces.*

**Filter?**

*Need to filter based on Conditional based.*

*Act on Each and every element with condition.*