21/3/23

WEB TECHNOLOGIES

* 3-Tier Architecture

We have 3 layers in 3-Tier Architecture.

1. Front End
2. Middle Layer
3. Back End

① FRONT END

Whatever we see in the UI [User Interface] is called FrontEnd
To develop FrontEnd we need technologies like HTML, CSS, JavaScript, React JS, Angular JS, Vue JS

② MIDDLE LAYER

In this the main logic to run an web application is present
To develop Middle layer we need technologies like Java, Spring, Servlets, Hybernate, NodeJS, Express JS

③

NOTE What is Server?
A Server is a place where Software is installed

④ BACK END

In this layer database is required to store some information
To develop BackEnd we need technologies like SQL, MySQL, PL SQL, Hadoop, MongoDB

HTML

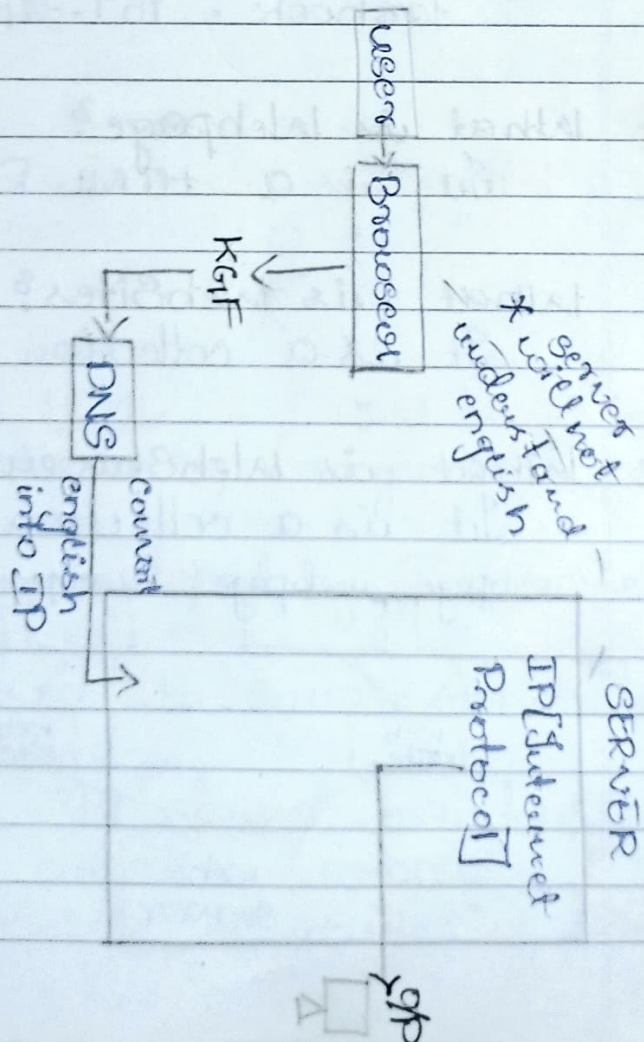
* Why is HTML used?
  HTML [Hypertext Markup Language] is used to create and give structure to the webpages

* Why is CSS used?
  CSS [Cascading Style Sheets] is used to style and layout webpages

* Why is JavaScript used?
  JavaScript is used to give action and special effects to the webpages

23/03/23

* How the web works



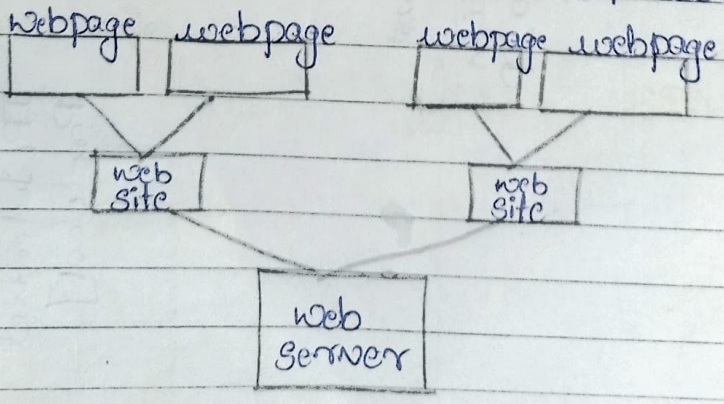
* What is Browser?
It is an application to load webpage, websites, webservers
Eg: Google chrome, FireFox, Opera, Safari, UC Browser etc

* What is DNS?
DNS [Domain Name System] which will take user input in human understandable format & convert that into Server understandable formate i.e IP address [Internet Protocol]
Eg:
Twitter : 199.16.156.0/22
Facebook : 157.240.16.0

* What is Webpage?
It is a HTML Document

* What is WebSites?
It is a collection of webpages

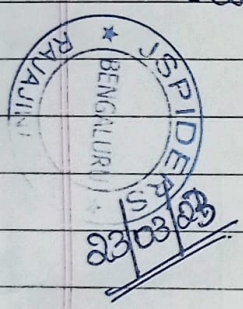
* What is WebServer?
It is a collection of websites



* We have 2 Types of webpages
  * Static webpages
  * Dynamic webpages

* Static Webpages
  - ✓ These webpages are developed using HTML & CSS
  - ✓ These webpages will not have any functionality

* Dynamic Webpages
  - ✓ These webpages are developed using HTML, CSS, JS
  - ✓ These webpages will have some functionality



24/03/23

HTML

Hyper Text Markup Language

1) HTML was developed by Tim-Berneslee in 1993
2) HTML is used to create the structure of webpages
3) HTML is standard markup lang for creating webpages
4) HTML consists of series of elements

5) HTML element tells the browser how to display contents

* Structure of HTML:

<!DOCTYPE HTML> → version of HTML5
<html> → Root Element
<head> → meta info / browser info
<meta>
<title> </title> → Browser Tab info
</head>
<body> → Displayed on the UI
|
|
</body>
</html>

```
<html>
  <head>
    <title> PageTitle</title>
  </head>

  <body>
    <h1> heading </h1>
    <p> paragraph </p>
    <p> another paragraph </p>
  </body>
</html>
```

→ DOCTYPE

Here we declare the version of HTML
The current version of HTML is HTML5

→ HTML

This element is the root HTML tag

→ head

This head contains meta information i.e., browser information

→ title

This contains the information which is present in browser tab

→ body.

This contains all elements which should be displayed on UI

NOTE: Hypem → Links | Text → Format | Markup → Tags | Language → Programming lang

* TAGS

Tags are predefined reserved keywords which are enclosed with angular brackets < >

eg: <Tagname>

NOTE "Fundamental blocks in html is called html tag"

→ Assignment 1

1) Write an come 40 Tags of HTML

1) <a> : anchor	14) <form>	27) <code>
2) <b> : bold	15) <hr>	28) <dd>
3) <big> : big (text)	16) <html>	29) <del>
4) <body>	17) <i>	30) <fieldset>
5) 	18) <img>	31) <figure>
6)	19) <applet>	32) <form>
7) <dd>	20) <b>	33) <img>
8) <dl>	21) <button>	34) <label>
9) <dt>	22)	35) <link>
10)	23) <blockquote>	36) <main>
11) <embed>	24) <code>	37) <meta>
12) <font>	25) <caption>	38) <nav>
13) <head>	26) <data>	39) <object>
		40) <output>

* Tags

- Paired Tags [opening & closing tag]
  - → Heading Tags
  - → Formatting Tags
  - → Other Tags
- Unpaired Tags [Self closing tag]

24/03/23

25/03/23

* Paired Tags:

1) Heading Tags

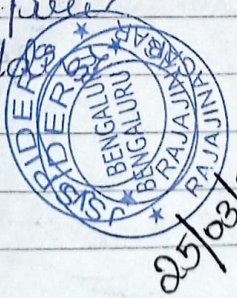
1) <h1> </h1> ⟶ large
2) <h2> </h2>
3) <h3> </h3>
4) <h4> </h4>
5) <h5> </h5>
6) <h6> </h6> ⟶ small

2) Formatting Tags

Bold   ] → same
Strong   ]
Italic   ] → same.
Emphasis   ]
Underline <u> </u>
Marker <mark> </mark>
Subscript
Superscript

3) Other Tags

Paragraph
Templet <pre> </pre>
Division Tag



25/03/23

* Unpaired Tag:

break
horizontal line <hr>
image <img>
input

No element is inserted after this tags so it called "void element"

** [Emite Observation]

NOTE: (*) What is BoileuPlate?

Seations of code that are repeated in multiplae places can be written with only one keyward or one line code

Eg: Instead of typing the HTML format we can simply type ! where it displays the full format

(*) We can save the html document by giving the extensions as .html or .htm

(*) When only opening tag is declared and the closing tag is missed the output is printed, as the first preference is given to the opening tag.

eg: i) <p> Hello World

o/p: Hello World.

2) <h1> Hi Everyone </h6>

o/p: Hi Everyone.

27/03/23

* Attributes In HTML

1) Which gives extra information about the tags
2) All elements can have attributes

eg: Youtube

→ href ——→ (hyper reference) which create links to another pages.
→ link

eg: YOUTUBE
(name to be printed as a link)

→ Target : Which specifies where to load the page, whether in the same page or different page.

3) Attributes are written in form of name — value pairs
eg: ____
↳ attribute

4) Attributes are written in opening tag of an element

eg:
```
<!-- anchor tag -->
<a href = "https://www.youtube.com" target = "_blank" >
    Youtube </a>
```

→ <a> (link tag) specifies anchor tag in html
href, target are attributes for

anchour tag.

* img: To insert any image

Syntax: `<img src="" alt="" height="" width="" >`

① Absolute Path URL
② Relative Path ./

→ Absolute Path: Where the image address is copied from the browser and the url is pasted at src

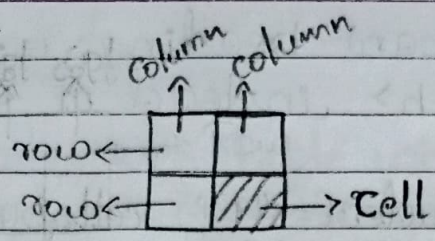
eg: `<img src="www.youtube.com" >`
↳ image address

→ Relative Path: Where the image is saved in the PC and then the image path is pasted in the src. It is given as a pwd eg as ./

eg: `<img src="./www5td68" >`
↳ image path

08/03

* TABLE



→ Tables are collection of Row and column

→ Row are horizontal representation of table

→ Columns are vertical representation of table

→ Cells are intersection of Rows & columns.

* Creating Simple 2X2 table

```
<table>
    <tr>
        <td> 1 </td>
        <td> 2 </td>
    </tr>
    <tr>
        <td> 3 </td>
        <td> 4 </td>
    </tr>
</table>
```

**

<tr> : Table Row
<td> : Table Data
<th> : Table heading.

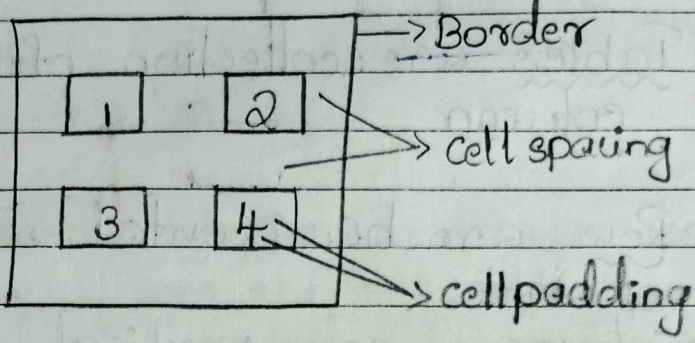
O/P:

1 2
3 4

NOTE: If we want heading we need to give <th> instead of <td>

* <table border="1" cellspacing="15px" cellpadding="10px">

o/p:

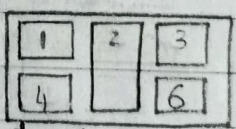


* Rowspan and Colspan

→ Rowspan: To merge the 2 or more rows we use Rowspan

Eg:
```
<table>
    <tr>
        <td> 1 </td>
        <td rowspan="2"> 2 </td>
    </tr> <td> 3 </td>
    <tr>
        <td> 4 </td>
        <td> 5 </td>
        <td> 6 </td>
    </tr>
```

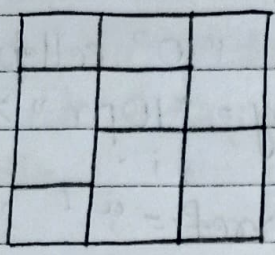
o/p:



2nd & 5th row is merged.

* Assignment

1. Create 3X3 table with images clicked
2. Template: Get this pattern printed.

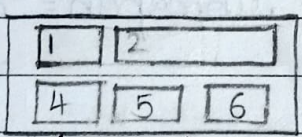


28/03/23

→ Colspan: To merge two or more columns we use colspan.

Eg:
```
<table>
    <tr>
        <td> 1 </td>
        <td colspan="2"> 2 </td>
    </tr>
    <tr>
        <td> 4 </td>
        <td> 5 </td>
        <td> 6 </td>
    </tr>
</table>
```

o/p:



2nd & 3rd column is merged

Assgn① Create a 3X3 table with image clicked it should re-direct another link.

```
<body>
<table border="10" cellspacing="5px"
    cellpadding="10px">
<tr>
    <td><a href="" target="_blank"
    ><img src="image address" height="100px"
    width="100px"></a></td>
```

↑ link URL

→ ... same for other 2 column

```
</tr>
</table>
</body>
```

Assgn② Get this pattern painted.

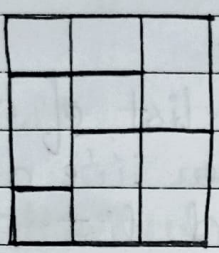
```
<table border="5" cellspacing="0.5px"
    cellpadding="10px">
    <tr>
        <td>1</td>
        <td>2</td>
        <td rowspan="2">3</td>
    </tr>

    <tr>
        <td rowspan="2">4</td>
        <td>5</td>
    </tr>
```

```
<tr>
    <td colspan="2">8</td>
</tr>

<tr>
    <td>10</td>
    <td colspan="2">11</td>
</tr>
</table>
```

op:



29/03/23

* LIST in html

If we want to store set of items in a list we need to go for list in HTML

We have 3 types of List

1> Ordered List
2> Unordered List
3> Description / Descriptive List

① Ordered List:

If we want list of items in a particular order we need to go for ordered list

 → ordered list
 → list

** By default ordered list will give output in Number Type

Eg:
```
<ol>
    <li> onion </li>
    <li> Tomato </li>
    <li> Garlic </li>
</ol>
```

output:
1. Onion
2. Tomato
3. Garlic

Type attribute

type = "A"
type = "a"
type = "i"
type = "I"

Eg:
```
<ol type="a">
    <li> Onion </li>
    <li> Tomato </li>
    <li> Ginger </li>
</ol>
```

output:
a. Onion
b. Tomato
c. Ginger

Start attribute: If we want to start the order from a particular value
start = "3", It takes only Numbers

Eg:
```
<ol type="A" start="3">
    <li> Tree </li>
    <li> Plant </li>
    <li> Root </li>
</ol>
```

output:
C. Tree
D. Plant
E. Root

NOTE: - Ole have an property called reversed, if we want to inverse the current order of list

Eg:

② Unordered List

If we want to show set of list items in an unordered way we need unordered list

**By default the output will be disc.

 → unordered list

Type attribute

type = "circle"

type = "square"

type = "null"

Eg:
```
<ul type="square">
    <li> Tom </li>
    <li> Anny </li>
    <li> Jerry </li>
</ul>
```

Output:

- Tom
- Anny
- Jerry.

③ Description / Descriptive List

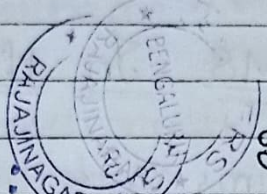
If we want to give any heading & sentence format we need to go for Description list

<dl> → description list

<dt> → description Term

<dd> → description Data

Eg:
```
<dl>
    <dt>Subject:</dt>
    <dd> Asking permission
         To 2days leave
    <dd>
</dl>
```



output: Subject:
Asking permission To 2days leave

30/03/23

* Nested List

A list inside another list is called Nested list

NOTE: For Unordered list inside a nested list will give an output circle type by default

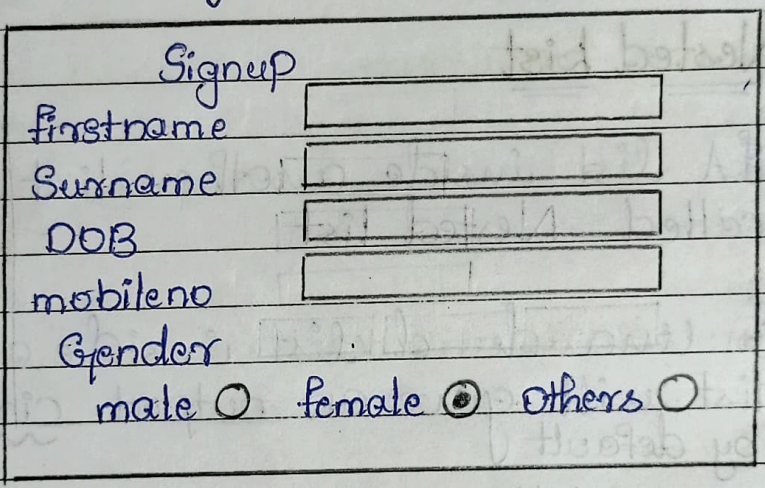
eg:
```
<ol>
    <li> Rajesh </li>
    <li> Subjects </li>
    <li>
        <ul>
            <li> Maths </li>
            <li> kannada </li>
        </ul>
    </li>
<ol>
```

output:
1. Rajesh
2. Bubject
3. ○ Maths
   ○ kannada

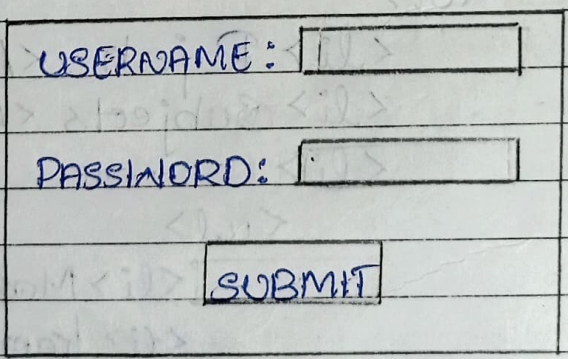
* FORMS In HTML

Forms are used to collect user information and send it to the server

Eg: Signup Form In Facebook



Eg: 1



code ⟶

```
<form action=" ">
    <label for="username">USERNAME:
                </label>
    <input type="text" name=" "
        id = "username "> <br> <br>

    <label for= "password "> PASSWORD:
                </label>
    <input type = "password " name=" "
        id= "password "> <br>
    <button> SUBMIT </button>
</form>
```

***

1. action attribute

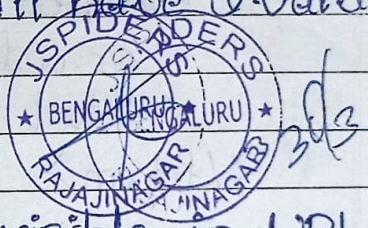
Once the form is submitted the action attribute is used to give the server URL

2. method attribute

This method will have 2 value

i) Get

ii) Post



Get: The data is visible in URL

Post: The data is secured not visible in URL

NOTE: By default get method will be there when we create a form

31/3/23

Input Types

1. Text
2. Password
3. Date
4. Month
5. Colour
6. file
7. Number
8. URL
9. Range
10. Radio
11. Checkbox
12. Submit
13. Hidden

→ Radio Button.

Eg:
```
<form action = " ">
    <label for = " ">GENDER</label>
    <label for = "male">MALE:</label>
    <input type = "radio" name = "
    gender" id = "male"><br>
    <label for = "female">FEMALE:
        </label>
    <input type = "radio" name = "
    gender" id = "female"><br>
    <label for = "
</form>
```

→ CHECK BOX

Eg:
```
<form action = " ">
    <label for = " ">SUBJECTS:</label>
        <br>
    <label for = "kannada">kannada
        </label>
    <input type = "checkbox" id = "kanna
        a"><br>
```

```
<label for="english">ENGLISH</label>
<input type="checkbox" id="english"
</form>
```

* Text Area

```
<form>
<label for="ABOUT">ABOUT US:</label><b
<textarea name="" id="ABOUT" cols="30"
        rows="30"> </textarea>
</form>
```

* Drop Down

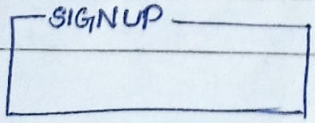
```
<form>
  <label for="city">City:</label>
  <selected name="" id="city">
    <option value="India">India
        </option>
    <option value="Swizz" selected>
        Swizz </option>
    <option value="USA">USA
        </option>
</form>
```

* Fieldset

```
<form>
<fieldset>
      <legend>SIGN UP</legend>

</fieldset>
</form>
```

op:



Input

* ~~Type~~ Attributes

① Value → By default given value
② Read only → cannot edit just seen
③ Disabled → cannot select
④ Size → width
⑤ Maxlength → validation only for text
⑥ Place Holder → [...]
⑦ Required → Mandatory to fill
⑧ Autofocus → cursor will be automatically pointed to input
⑨ Autocomplete [form] → to get suggestion

* Tool Tip: to give popups when clicked on letter

<h1 title = "GOOD BOY">ಒಳ್ಳೆಯ ಹುಡುಗ</h1>

ಒಳ್ಳೆಯ ಹುಡುಗ
[GOOD BOY]

* HTML Elements

The content which is enclosed within or written within < >

1) Block level element : eg - <table>, <blockquote> <p>, <section>, , <form>
2) Inline element : eg - <a>, <img>, , <input> <label>, ,

1) Block element: always takes up the full width available

<input>

2) Inline element: It takes up as much width as necessary.

<input>

1/04/23

* Assignment
  i) Create a form using all the Fields, TextBox, Button, Dropdown etc
     City in dropdown
     Qualification not required.
     field set

* Media Tags

1. image tag
2. audio tag
3. vedio tag
4. iframe

1) image tag: to insert an image

Syn: <img src=" " alt=" " height=" " width=" " >

2) audio : If we want to insert one audio to a webpage we need to add audio tag.

Syn: <audio src="./ " controls></audio>

Attributes:

src : This is the source we need to pass for audio

controls: if we want to have, play, mute, pause, we need to give controls attribute

muted: if we want to have the audio in mute by default when we load the page

3) Video tag: if we want to insert video inside webpage we need video tag

** NOTE: [if we want use autoplay mandatory use muted attribute]

It has attributes.
a) src
b) controls
c) muted
d) autoplay
e) height
d) width
poster="

Syn: <video src="" controls autoplay muted></video>

4) iframe: this iframe is used to integrate one webpage inside another webpage

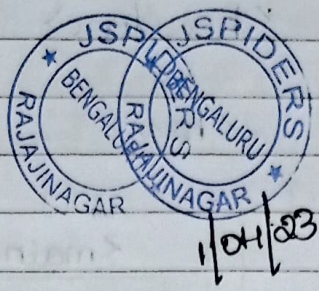
Attributes:
src
frameborder

step1: open youtube socket-veolio, click share button

step2: copy uniqueid, then paste it in src

Poster :-

Syn: <iframe src = " https://www.youtube.com/embed/_____ " frameborder = " ">



1/04/23

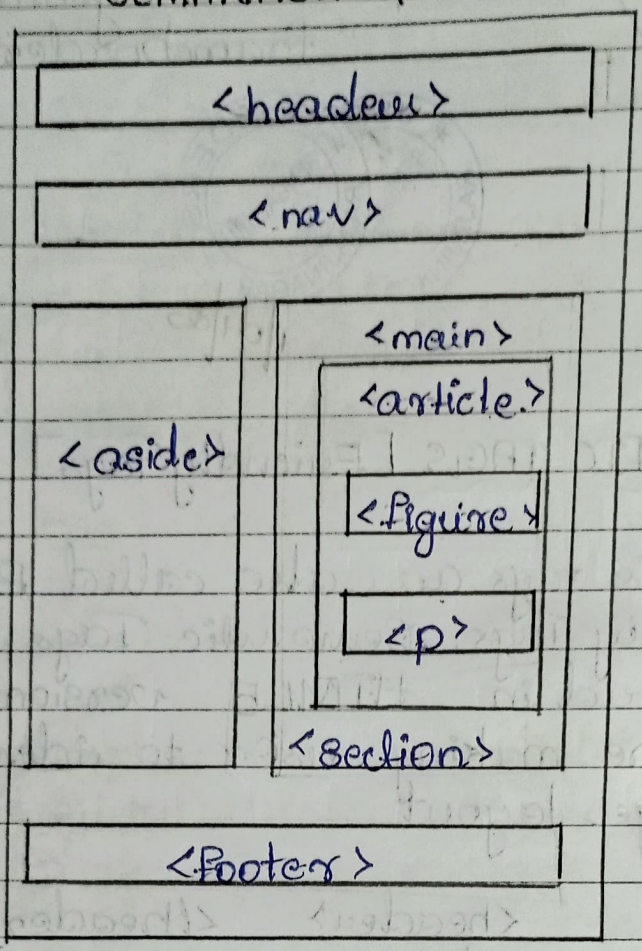
3/04/23

SEMANTIC TAGS [Friendly Tags]

This tags are also called Browser Friendly Tags. Semantic Tags are introduced in HTML5 version This tags are making user to identify webpage layout

<header>	</header>
<details>	</details>
<article>	</article>
<main>	</main>
<nav>	</nav>
<section>	</section>
<time>	</time>
<aside>	</aside>
<figure>	</figure>
<footer>	</footer>

SEMANTIC TAG FORMAT



3/04/23

CSS [Cascading Style Sheet]

* Introduction

Cascading Style Sheet [CSS]

falling design, colors

It was introduced by "HAYKON WIUM" in the year 1996

→ We can write CSS in 3 ways

1. Inline method
2. Internal method
3. External method

1. Inline Method:

If we want to pass CSS to the current element itself we need to go for Inline CSS

Eg: `<h1 style = "color: red" >HELLO This Is H1 Tag </h1>`

[Always 1st priority is given for Inline]

2. Internal Method:

In Internal method we want to use style tag and write this inside <head> tag.

Syntax:

```
<head>
<style>
    h1 {
        colour: blue;
    }
</style>
</head>
```

3. External Method:

S1: Create one new file index.css
S2: Target the element give styling.
S3: Late need to link css file to the html page.
S4: <link rel="stylesheet" href="./index.css"> → inside <head> tag

NOTE Always the first priority is given to inline
Second priority is based on sequence or order of the internal or external, the last line will be executed.

1/4/23

* CSS Selectors.

If we want to target some element in CSS we should selectors.

→ We have 5 types of selectors.

1) Simple Selector
2) Combinator Selector
3) Pseudo class Selector
4) Pseudo Element Selector.
5) Attribute Selector

① Simple Selector.

a) Id ⟶ #
b) class ⟶ .
c) Tag name ⟶ tag name
d) Grouping selector ⟶ ,
e) Universal Selector ⟶ *

a) Id (#)

→ highlighted

→ `<h1 id="h1tag">This is an h1 </h1>`
`<h2 id="h2tag">This is an h2 </h2>`

→
```
#h1tag {
    colour: red;
}
```

b) Class (.)

→ `<h2 class="para">This is h2 </h2>`
`<p class="para">Hello </p>`

→ `.para {`

```
    colour : red;
}
```

c) Tag name (tags)

→ This is bold

```
→ b{
    colour: 2px solid red;
  }
```

d) Grouping Selector (,)

→ <h1> This is an h1 </h1>
 This is an span
 This is an div

```
→ h1, span, div {
        colour: red;
    }
```

e) Universal selector (*)

If we want to select all the element in the webpage we need to go for universal selector.

→ <h1> This is an h1 </h1>
 This an span
 This is an div

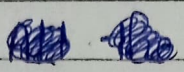
```
→ * {
    color: red
  }
```

5/04/23

② Combinator Selector

1) Descendant Selector (space)
2) Child Selector (>)
3) Adjasent Sibiling Selector (+)
4) General Sibiling Selector (~)

1) Descendant Selector (space)



eg:→
```
<div>
    <p> This is para one </p>
    <p> This is para 2 </p>
    <section>
        <p> one </p>
        <p> two </p>
    </section>
</div>
```

→
```
div p {
    colour : blue
}
```

[All the <p> tags will be highlighted or colour]

2) Child Selector (>)

eg:
```
<div>
    <h1> This is H1 </h1>
    <section>
        <h1> This is H3 </h1>
```

</section>

-> div > h1 {
    colour: red
}

[all the h1 tags of only child will be highlighted]

3) Adjacent Sibling Selector (+)

->
    <h1> This is h1 </h1>

<h1> This is h2 </h1> -> HIGHLIGHTED
<h1> This is h3 </h1>

-> div + h1 {
    colour: blue
}

4) General Sibling Selector (~)

->
    <h1> This is h1 </h1>
    <h1> This is h2 </h1>

<h1> This is h3 </h1> } HIGHLIGHTED
<h1> This is h4 </h1> }

-> div ~ h1 {
    colour: pink
}

6/4/23

③ Pseudo element selectors.

1) ::after
2) ::before
3) ::first-letter
4) ::first-line
5) ::marker
6) ::selection

1) ::after

Here the content is added after the current content

-> This is
<h2>This is</h2>

-> p::after {
content: " AN PARAGRAPH";
}
h2::after {
content: "AN h2";
}

o/p: This is AN PARAGRAPH
This is AN h2

2) ::before

Here the content is added before the current content

-> PARAGRAPH

-> p::before {
content: "This is";

color: red;
}

o/p: This is PARAGRAPH

3) ::first-letter.

If we want to give font effect for only starting letter we need to use ::first-letter

→ PARAGRAPH

→ p::first-letter {
colour: blue;
font-size: 1.5rem;
}

o/p: PARAGRAPH

4) ::first-line [It can be applied only to block-level elements]

only the first-line is highlighted which is targeted

→ Lorem
 Lorem

→ p::first-line {
background-color:
aqua;
}

5) ::marker

→
 MCA
 BCA
 ME

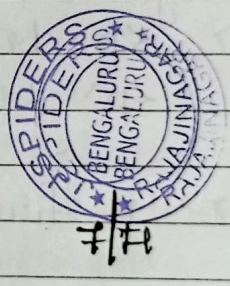
→ li::marker {
content: "☺"
}

o/p: ☺ MCA
☺ BCA
☺ ME

6) ::Selection

→ <h1> This is h1 </h1>
<h2> This is h2 </h2>

→ h1::selection {
background-color: yellow;
colour: red
}



7/11

* Pseudo Class Selector

i) Dynamic Pseudo Class (anchor)
a) link
b) visited
c) hover
d) active

eg2: hover using button

```
button:hover{
    bouder: 2px solid brown;
    border-radius: 25px;
      colour: aqua
    background-color: brown;
      cursor: pointer;
}
```

2)* Pseudo Structural Pseudo Class

a) first-child

html: <body>

 This is para1 —#1
 This is para2

 This is para3 —#1
 This is para4

</body>

css: p : first-child {
color : red
}

eg:
 This is para1
 This is para2

 This is para10

 This is para11
 This is para5

 This is para

div : first-child
{
color : blue
}

b) First-of-type.

```
<body>
    <p> This is para </p>
    <p> This is para </p>
    <p> This is para </p>
</body>
```

```
p:first-of-type
  {
    color: green;
  }
```

c) Last-of-type.

```
<body>
  <div>
    <p> This </p>
    <p> This </p>
  </div>
<div> <p> This </p>
    <p> This </p>
  </div>
```

```
p:last-of-type
  {
   color: blue;
  }
```

d) nth-child

```
<div>
   <p> This </p>
   <p> This </p>
   <p> This </p>
   <p> This </p>
</div>
```

```
p:nth-child(2)
  {
   color: red;
  }
```

5. Attribute Selector

If we want to target elements using attribute we need to go for attribute selector

1) [attribute]
2) [attribute = "value"]
3) [attribute ~= "value"]
4) [attribute |= "value"]
5) [attribute ^= "value"]
6) [attribute *= "value"]
7) [attribute $= "value"]

1) [attribute]

-> html

```
<div>
  <a href=" ">FACEBOOK</a>
  <a href=" " target="_blank">GOOGLE</a>
</div>
```

css
-> [target]

-> css

```
[target] {
    background: red
}
```

2) [attribute = "value"]

-> html

```
<a href=" " target="_blank">G</a>
<a href=" " target="_self">F</a>
```

-> css

```
[target="_self"] {
    background-color: red;
}
```

3> [attribute ~ = "value"]

any attribute value which matches the word will be selected, either their should be space.

-> html

```
<h2 class = "sample"> This h1</h2>
<h2 class = "test" > This h2 </h2>
<h2 class = "sampletest"> This is h3 </h2>
```

-> css

```
[class ~= "test"]
{
    background-color : red
}.
```

4> [attribute | = "value"]

-> html

```
<h2 class = "sample" > This </h2>
<h2 class = "test" > This </h2>
<h2 class = "test-sample" > This </h2>
```

-> css

```
[class | = "test"]
{
    background-color : red
}
```

5. 5> [attribute ^ = "value"]

-> html

<h2 class = "sample"> This </h2>

<h2 class = "test"> This </h2>

<h2 class = "testsample"> This </h2>

<h2 class = "jio"> This </h2>

-> css

```
[class ^ = "test"]
{
    background-color: red;
}
```

6> [attribute | = "value"]

-> html

<h2 class = "sample"> This </h2>

<h2 class = "test"> This </h2>

<h2 class = "sample-test-value"> This </h2>

<h2 class = "index"> This </h2>

-> css

```
[class | = "test"]
{
    background-color: red;
}
```

7> [attribute $= "value"]

-> html

<h2 class = "sample"> This </h2>

<h2 class = "test"> This </h2>

<h2 class = "testsample"> This </h2>

<h2 class = "jio"> This </h2>

```
[class$="test"]
{
    background-colour: red;
}
```

11/11/23

* COLOURS

In css we can apply colours for following properties

1) Background-color: [If we want to change background colour of particular element]
2) Border: [If we want to give border for particular element we need to pass value 3px solid pink]
   - 3px → Border thickness
   - solid → Type of border
   - pink → Color name
3) Colour: [If we want to change some content colour]

Example:

```
<body>
<div class="div-block">
        This is a DIV Block
</div>
</body>
```

```
.div-block {
    height: 300px;
    width : 300px;
    background-colour: red;
    border: 2px solid red;
     color : white;
  }
```

* Values In Colours

There are several ways we can initialize values to a element

1) Colour Names
2) RGB ( )
3) RGBA ( )
4) HSL ( )
5) HSLA ( )
6) HEXA CODE

1. Colour Names

Eg: tomato
deeppink
forestgreen
magenta
navy.

2. RGB( )
(red, green, blue)

we can pass values from 0-255
eg: (255, 0, 0) //red

rgb (0,255,0) // green
rgb (0,0,255) // blue
rgb (0,0,0) // black
rgb (255,255,255) // white

3) RGBA ( )
(Red, Green, Blue, Alpha)

NOTE: Alpha : this is the opacity of a colour
opacity means transperancy

Alpha range blw 0-1

eg: rgba (255, 0, 0, 0.6)

4) HSL ( )
hsl ( ) [Hue Saturation Lightness]
↓ Degree ↓ Percentage ↓ Percentage

red - 0°
yellow - 115°
green - 100°

100%
20%
amount of color

grey scale

eg: background-colour: hsl (0, 70%, 50%)

5) HSLA ( )
Hue Saturation Lightness alpha
↓ Degree ↓ Percentage ↓ Percentage ↓ 0-1

eg: hsla (0, 70%, 50%, 0.5)

6) HEXA CODE

HEXA means 0 - 15 but we can pass only 0-9 then continue with ABC

0 a
1 b
2 c
3 d
4 e
f
5
6
7
8
9

- eg: #FF0000
  R G B

OR

#F00
R G B

CIRAP
↓ ↓ ↓
color r

12/04/23

Date
Page

* Background - Image Properties

1) Background-Image : url(" ");

2) Background-repeat:
   repeat (default)
   no-repeat
   repeat-x
   repeat-y.

[*** by default it will be repeat]

3) Background-Attachment.

[*** by default it will be scroll]
   fixed
   to scroll.

If we want to scroll the image or move the image with web page content we require background attachment scroll.

4) Background-Position.

[*** By default the image will appear in top-left of the web-page]

top-left
top-right
bottom-right
bottom-left
center.

5) Background-Size : 100px 300px
width ↓ width, height ↓ height
height
contain
cover.

Eg: [** short-hand :
CIRAPS
C ↑ I ↑ R ↑
background : red url("") no-repeat
scroll ↓ A, cover ↓ D]

* Gradients In Css [pass to Background-Image]

1) Linear-Gradient
[** By default it is top-bottom

background-image: linear-gradient(red, yellow);
OR
: linear-gradient(to bottomleft, red, yellow);
OR
: linear-gradient(red 50%, yellow 50%);

2) Radial-Gradient

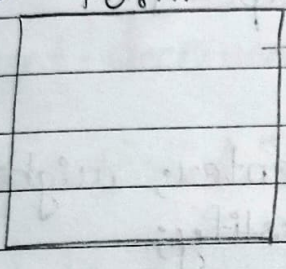
background-image: radial-gradient(circle, red, yellow);
background-image: radial-gradient(red 30%, blue);

3) Conic - Gradient

background-image : conic-gradient
(red 0deg, red 90deg, blue 90deg
blue 180deg, green 180deg, green 270deg
yellow 270deg, yellow 360deg)
border-radius : 50%;

Assgn.

Form

1) 

2) Create one rainbow output using gradients.

3) Create a text and give gradient for the content

Create Simple resume syntax

13/4/23

* Text Alignment

* TEXT PROPERTIES.

1. text align:
   - center [should be block level]
   - right
   - left
   - start
   - justify [newspaper]
   - end

2. text-align-last : center; right; left; justify;

3. direction : rtl; ltr

4. color:

* text-decoration:

1) text-decoration-line:
   - underline TEXT
   - over-line TEXT
   - line-through TEXT

eg: text-decoration-line : underline overline linethrough;
TEXT

2) text-decoration-colour: powderblue

TEXT

3) text-decoration-style:
dotted
wavy
double
dashed
solid

4) text-decoration-thickness: 5px;

→ Short-Hand Property [LCST]

eg: text-decoration: underline red wavy 2px;

* Other Properties

1) text-indent
2) text-transform
3) letter-spacing
4) line-height
5) word-spacing
6) white-space
7) text-shadow

1) text-transform: capitalize;
lowercase;
uppercase;

2) text-indent: 100%
[Space in the beginning of para]

3) letter-spacing: 2px

4) line-height: 5.2
[spacing b/w each-line]

5) word-spacing: 20px

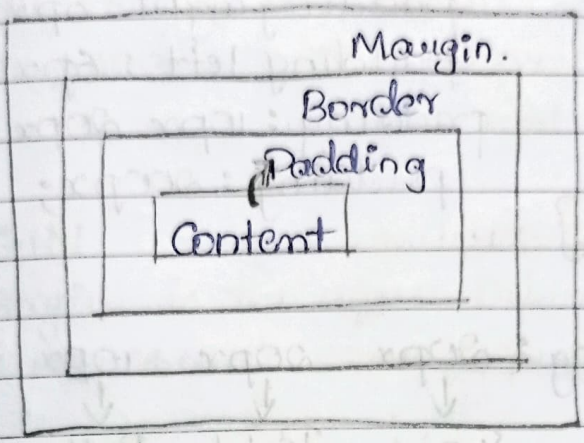
6) white-spacing:

7) text-shadow: green 2px 1px 2px;

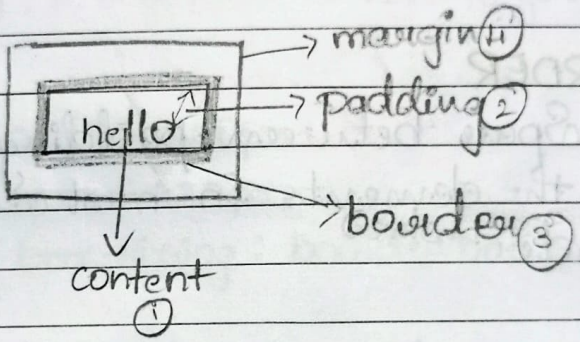
11/4/23

* CSS Box Model

All the elements which is written in HTML will contain CSS Box Model.



Eg.



* **Content**

This is nothing but information which is displayed by HTML.

eg:

```
<h1>Heading</h1>
<p>Today is holiday</p>
```

* **Padding.**

It is the space between content & border.

[NOTE Padding is invisible space among box model]

NOTE: CSS outline: After the margin outline is pass

eg: h1{
border: 2px solid blue;
padding top: 20px
padding bottom: 10px
padding right: 5px
padding left: 6px.
padding: 10px 20px 25px 50px;
padding: 200px;
}

padding: 20px 20px 10px 5px
↓ ↓ ↓ ↓
Top Right Bottom Left

* BORDER
→ Space between padding and margin
→ All the elements in html can have border

eg: bouder: 2px solid red;
bouder-bottom: 2px dashed blue;
bouder-top: 10px wavy blue;
bouder-left: 2px dotted red;
bouder-right: 2px dotted blue;

Bouder-style: dashed solid
dotted
double
groove
inset
outset
ridge

border-top-style : dashed
border-bottom-style: dotted
border-right-style : dashed
border-left-style : dotted
border-bottom-left-radius: 2px;
border-radius: 0px 20px 30px 40px;
top right bottom left

* MARGIN
margin is the space between border and browser

[NOTE Global-Resetting: by default browser will have some spacing, to overcome we need to go for Global-resetting.

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

Assign: Is that is top, bottom, left, right

margin-top
margin-bottom
margin-right
margin-left
margin: 20px 20px 10px 5px;
Top Right Bottom Left

17/11/22

* CSS Position

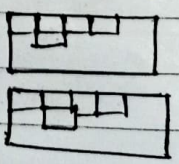
1) Static [By default it will be static]
2) Relative
3) Absolute
4) Fixed
5) Sticky

1) Static: [Position: Static]
-> By default position will be in Static
-> We cannot give top-20px because static will never change the position of the element

eg: .div {
Position : static;
top : 10px → will not works
}

2) Relative: [position: Relative]
Here the element will move or takes any change from the current position (or) default position of that element

eg: #item-3 {
background-colour : darkgold;
position : relative;
① → top : 20px;
② → left : 20px;
}



from current position.
it will take.

3) Absolute: [position : Absolute]
(first parent)
→ If absolute is given to an element that element will move to its parent top left corner.

→ if we performm top: 20px it will take from the viewpoint of the webpage.

NOTE: If we want to make the element to take top fuom its pavent, we need to give Position: Relative to its pavent.

4) Fixed: [position : fixed] (chat bot)
(first parent)
if we give position fixed the element will not scroll with the content it is fixed in the same position.

5) Sticky: [position : sticky] (nav bar)
→ if we want to move the element & stop at a certain point of top viewport

→ mandatory we need to mention top value

```
{
    position : sticky
    top : 0
}
```

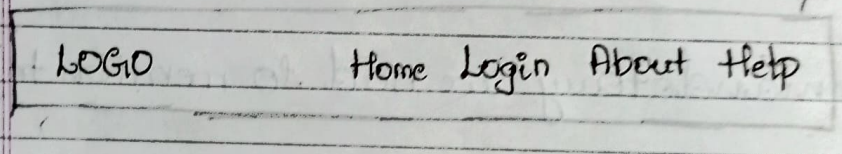
14/4/23

* Z-index
→ If we have many elements overlapped on each other we can set position for all the elements using z-index
→ We want to give position property mandatory
→ z-index : value;

eg: #img1 {
position : relative;
z-index : 1;
}

NOTE: 1. Display : flex [to convert block-level to inline]
2. Justify Content : space around [↓ to →]
3. Align-Items : center [□ □ □]

* Layout



Assgn : 1) Clone the same nav bar.
2) Brand Home About Products Search Search

NOTE: Pesticide for clone.
1. What font
CSS Viewer.

19/04/23

* Chat Bot:

Tawk Tawk.

<script

S1: open Tawk.com
S2: Signup
S3: Select lang
S4: Copy JS code & paste inside the body anywhere.

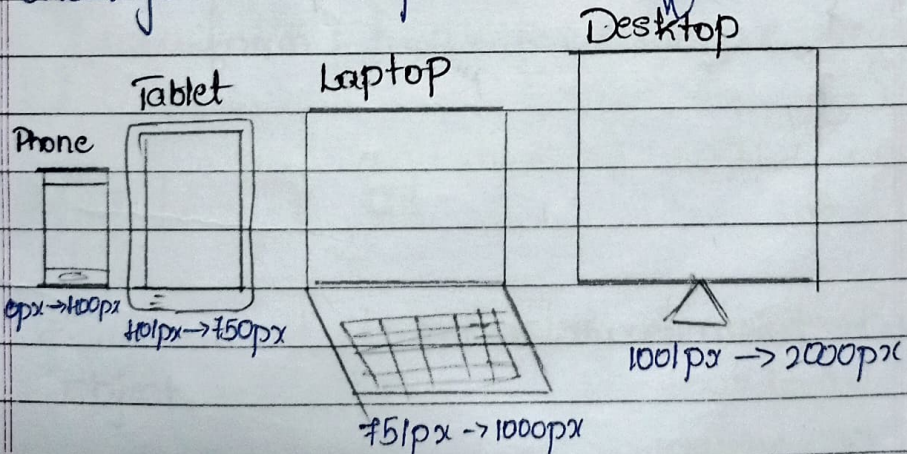
Asgn

① Activate our chatbot and change all the default setting & set our default msg.
② Create RWD for some images (H)

* Media Queries

RWD [Responsive WEB Design]

If we develop an appl^{n} in an laptop then run that appl^{n} on some other devices like phone-tablet / computer the alignment our UI of elements will be changed and produce different output



PHONE DISPLAY

```
@media screen and (max width : 1100px)
{
    div {

        }
}
```

TABLET DISPLAY

```
@ media screen and (min width: 101px)
        and (max width : 750px)
    {
    }
```

LAPTOP DISPLAY

```
@media screen and (min width: 751px) and
        (max width : 1000px)
    {
    }
```

DESKTOP DISPLAY

```
@media screen and (min width : 1001px) and
        (max width : 2000px)
    {
    }
```

Profiles

Note in the head give another link for getting tabicon

<link rel="icon" href="./ ">
↓ relationship.

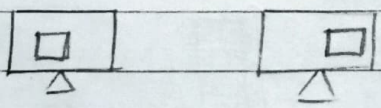
* transform

This property is used whenever if we wants to make some changes of particular element

1) translate( )
2) translate X( )
3) translate Y( )
4) Scale( )
5) Scale X( )
6) Scale Y( )
7) skew( )
8) skew X( )
9) Skew Y( )
10) matrix( )
11) Rotate( )

1) translate( )

transform : transform (100px, 200px)
x y

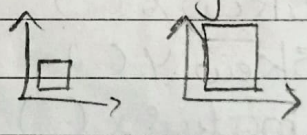


If we want to move the element or object.

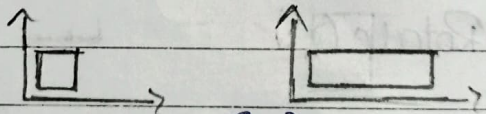
2 translate X()
{
transform : transform (100px);
}
To move along x-axis →

3. translate Y()
{
transform: transform (300px)
}
To move along Y-axis ↑

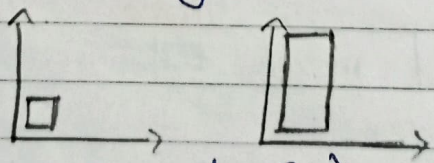
4. Scale ()
{
transform: scale (2, 3);
x y
}
change the size.



5. scaleX()
{
transform : scale (2);
x
}
change the size along x-axis



6. scale Y()
{
transform : scaly (5);
}
change the size along y-axis

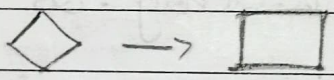


7. skew( ) in degree
{
skew(e) transform : skew (40deg, 90deg);
}

8. skewX() skewX()
{
transform : skewX(80deg);
}

9. skewY()
{
transform : skewY(200deg);
}

10. rotate
{
transform : rotate (20 degree);
}



if we want to rotate anticlockwise
-20degree.

11. Matrix → It is a short-hand-property.
one
Matrix(scaleX(), skewY(), skewX(),
scaleY(), translateX(), translateY())

21/4/23

* Transition Property
→ Here we need to mention for which property transition has to be applied
→ By default transition property will be applied for all
"transition-property: width, background-color;"

Transition-duration
If we want to give time for the transition
eg: "transition-duration: 5s"

Transition-delay
In this property the time to begin the transition is mentioned
we need to pass the time in seconds
eg: "transition-delay: 2s"

NOTE: By default transition-timing-function will be in case

Transition-timing-function
1. Linear
2. Case
3. Case-in
4. Case-in-out
5. case-out
6. Cubic-bezier()

Transition

transition : property duration transition timing delay ;

transition: width 2s ease 5s

* Transition + Transform.

```
eg: div {
        background-colour:
        height :
        width :
        transition-property: width, transform;
        transition-duration: 2s;
    }
    div: hover {
        transform : rotate (180 deg);
        height : 100vh;
        width: 100vw;
    }
```