Eco Friendly

N. Nature™

YOUR CHILD IS OUR PRIORITY

BIG BEN
LONDON

#NOTEBOOKS

A4

student exercise

notebook

Full Stack Web Development :-

INDEX

Name: Rohit Kumar Yadav

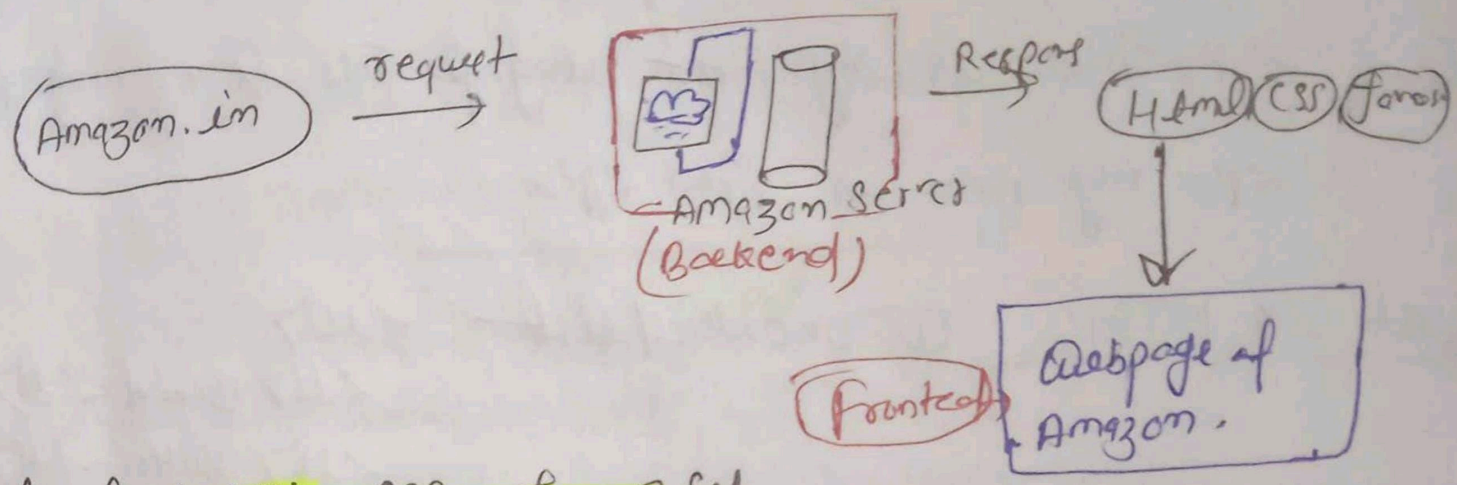
Class: Subject: Full Stack Web development. Roll No. 2023101014009

Serial No.	Date	Title	Page No.	Teacher's Sign/Remarks
1-		Intro + 5, Prequisite, HTML level Part A		
2-		HTML Level 1 Part B		
3-		HTML level 2		
4-		HTML level 3		
		CSS		
5		CSS getting started		
6		CSS Part 2		
7		CSS Part 3		
8		CSS Part 4		
9		CSS Part 5		
10		CSS part 6		
11		→ mini project (CSS)		
12		→ Bootstrap		
13.		→ Project (CSS)		
		Java Script		
14.		Java script Part 1		
15.		Java Script Part-2		
16		Java Sript Part-3		
17		Java Script Part 4		
18		Java Script Part-5		
19		Java Script Part-6		
20		Java Script Part-7		
21		Java Script Part 8		
22		Java Sript Part-9		
23		Java script Part-10		

Serial No.	Date	Title	Page No.	Teacher's Sign/Remarks
24		JavaScript Part Mini Project		
25		JavaScript Part 11		
26		Java Script Part-12		
27		JavaScript Part 13		
28		Terminal		
29.		Git & github.		
		Backend		
30		Backend 1 (Node JS)		
31		Backend 2 (Express)		
32		Backend 3 (Node EJS)		
33		Mischelenous		
34		Backend (Rest)		
35		Starting with SQL		
36		SQL part 2		
37		Backend 5 (Node with SQL)		
38		Mongo DB, installation Part 1		
39		mongo DB Part-2		
40		Mongo DB with express		
41	→	~~Project Part 1~~ (Part a)		
42	→	Project Part 1 (Part b)		
43.		Backend 6 (middleware)		
44.		Backend 7 (crm)		
45.	→	Project phase 1 (Part-c)		
46		Database Relationship		
47.	→	Project Phase 2 (Part a)		
48.	→	Project Phase 2 (Part b)		
49.	→	Project Phase 2 (Part c)		
50.	→	Project Phase 2 (Part D)		
51	→	Major Project Phase 2 (Part e)		
52	→	Major Project Phase 3 (Part a)		
55		" b, c, d		Sh Rearth

Web development

Build website for the Internet.



Role of HTML, CSS, Java Script

- HTML → Structure
- CSS → Style
- JS → Functionality

↳ what we learn?

Frontend :- HTML, CSS, Java Script, Bootstrap, Tailwind

Backend :- Express, Node JS

Database :- SQL, MongoDB.

React.

HTML

Hyper text markup language

↓

Structure & formative

Text → Formatting

Ex Ram, Syam, Rohit →
- Ram
- Syam
- Rohit

Paragraph Element:-

 Html tag used represent paragraph.

Ex This is my First paragraph. This is my paragraph

 my name is Rohit

List in HTML

① Unordered list

```
<ul>
<li> Bread </li>
<li> Butter </li>
<li> Jam </li>
</ul>
```

output:
- Bread
- Butter
- Jam.

Ordered list

1. a.
2. b.
3. c.
4. d.

```
<ol> <li> Ram </li>
<li> Shyam </li>
<li> Syam </li>
```

output
1. Ram
2. Shyam
3. Syam

```
<ol type= A>
```

A. Ram
B. Shyam
C. Syam
D. Rohit

HTML Attributes:

Attribute are used to add more information to the tag.

Ex. <ol type="a"> <Html lang="english">

attribute.

Anchor Element

> used to add links to your page.

<a href="https://google.com">Google</a>

(attribute)

(hypertext refrence)

Absolute links
- Same as Internet

Relative links
- file links

Br. Tag

used to add next line (Break) to your page.

Bold, Italic, Underline

 Bold

 Italic

<u> Underline </u>

Comment of HTML

↳ This is part of code that should not be parsed.

<!-- Poem -->

[Ctrl + /]

Forward (Backward)

HTML is Not Case Sensitive

Case Sensitive — a A ✗

Non Case Sensitive — a, A ✓

Practice Questions – ① Create a portfolio page
② Add you name & picture on top
• Add Section for your Education, Skill, Hobbies, Contacts.

#3

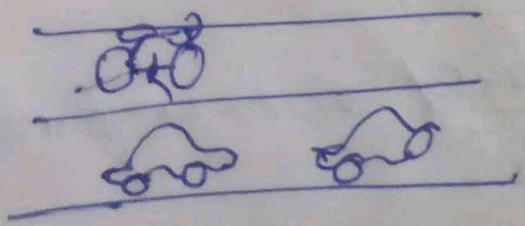
Inline VS Block

Level 2

Inline:

Takes up only necessary width.

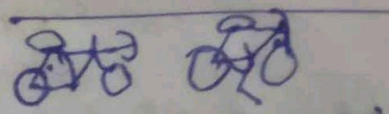
Don't start new line.



Block Element

• Takes up full-width available (whole block)

• Start from new.



Am I Inline or Block:-

- Heading element - Block element <H1> — </H1>
- Paragraph → Block element <P> — </P>
- Anchor tag → Inline element <a href="www.google.com">
- Image Element → Inline element.

Div Element

1) Div is a container used to hold other HTML element or group element together.
2) It is a block element.

{ Content division element }

Ex. <body>
<a href="www.google.com">Google</a>
<a href="www.facebook">Facebook</a>
<a href="www.youtube">Youtube</a>

① output → Google Youtube Facebook

② output Google Face book
Youtube

- Background <Div>

Span Element

Span is also generic container used to hold other Html element or group element together

It is an inline element

Ex

```
<span>
<a href="www.google.com">Google</a>
<a href="www.youtube.com">Youtube</a>
</span>
<a href="www.facebook.com">Facebook</a>
```

output: Google. Youtube Facebook Ex Quora.

HR Tag (Horizontal line)

Horizontal Rule Element.

<hr> Divide

Sub & Sup Tag

$a^2 = b^2 + c^2$ — superscript

H_2O → Subscript <H2> H <sub> 2 <O> </H2>

Q Print Pythagorus theorem says. $a^2+b^2=c^2$

Formula of glucose is $C_6H_{12}O_6$.

Sematic markup

↳ It is the markup that related to the meaning of Content.

Sematic	Non Sematic
<H>	<Div>
<p>	
<imp>	

① meaningfull / layout structured.
② SEO friendly (Search engine optimisation)
③ readablest screen reduced.
↓
UI

Semantic Markup:

① < Header > < /Header > → main part
② < main > < /main > (major part of website)
③ < footer > < /footer >
④ < nav > < /nav >

① < body >
< Header > Apna College < a href = "www.apnacollege.in" > < /a > < /Header >
footer part

② < main > < /main > Header { Apna College }
main { • It is education platform.
• Cont. No.
• Education helping people }

③ < footer > < /footer > Footer { Address :- Delhi

④ < nav > < /nav >
↳ (Navigation)
{ Website parts }

< Header > Apna College.
< nav >
< ul >
< li > About < /li >
< li > Contact < /li >
< li > a href = " About < /a >

(5) <article> </article>

(6) <Section> </Section> related Content (like)

(7) <Aside> </aside>

related/like Content

Practicle Q:6 Add semantic markup to your portfolio page

- Header
- Footer
- Skill, edu. (Section).

HTML Entities Space

- Html entitity is a piece of text ("string") that begin with an ampersand (&) and end with a semicolumn (;)
- used to display the reserve character (which would be interpreted as Html Code) and invisible character (like non breaking space).
- Can also use in place of Character that are difficult to type with a standard keyboard.
- Browser interprets them & renders correct character

  non breaking space.

< <

> >

<H1> &heart; </H1>

Character	Entity	Note:
&	&	Interepreted as the begining of entity or Character.
<	<	Interpred biginning of tag
>	>	ending of tag
"	"	begining and ending of an attribute's value.

Ex a < < ;
 a > > ;
 a & ; b
 a " ; b &quote

output: a<b
a>b
a & b
a "b"

Q5 Print the following on the screen using an <H1> heading elements and entities.

* Search for suitable cloud & rain entities.

The sky is cloudy ☁ It will rain today

Emmet

Emmet.io → Development process → Website

! → Boiler plate Code.

! + Enter → Boiler plate Code. (Abbribiation)

Syntax: Child :>

parent-child relation

 <img>

(Parent tag) (Child tag)

<main>

↓

main
↓
p
↓
ul
↓
li

 <ul> <li> <li>

</main>

UNDERStand HTML 5

The term HTML5 is a essential a buzzword that reffers to set of modern Web technologies. This include the HTML living Standard, along with Java Script API to enhance storage, multimedia, and hardware acces

① new features/ upgrades in Html + other thing/ fealnes not part of Html.

→ How Html works:

HTML → Standard

- Html standard is a document that tells to the browser how Html should works

Day 4 Level 3 Table in HTML

Tables are used to represent real life table data.

Name	Roll. No.
Aman	24
Shradha	86
Rajat	12

← Table

Table Header

→ Row

Column

Ex <table> (Food menue)
<caption> Table Caption </table>
<tr>
</tr>
</table>

Food menue ← (Title (caption))

<table>

<tr> used to display table row.

<td> used to display table data

<th> used to display the table header.

<table> or <table Border = "black">

<caption> Table Caption (Food Item) </caption>

<tr>
<th> Item </th>
<th> Price </th>
</tr>
<tr>
<td> Fruotie </td>
</tr>
</table>

Semantic Tag

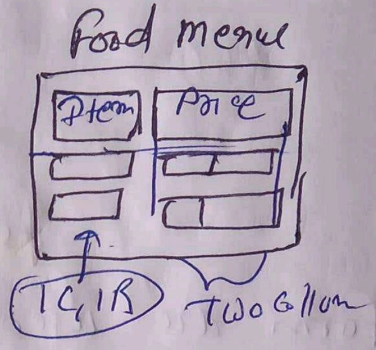
<thead> to wrap table header.
<tbody> to wrap table body.
<tfoot> to wrap table footer.

```
<table>
  <caption> <thead> <tr>
        <th> Item </th>
        <th> Price </th>
        </tr>
        </thead>
<tbody>
    <tr> <td> Frutti </td>
        <td> 10 </td>
        </td> <tr> <td> Samosa </td>
        </tr>
        <tbody>

<tfooter>

        </tfooter>

        </body>
        </html>
```

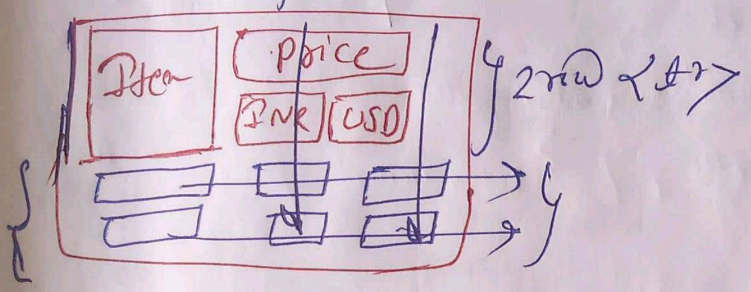
Colspan & rowspan Attribute

Used to create cell which spans over multiple rows or column.

rowspan = " "
Colspan = " "



<th Colspan="2"> price </th>
<th> rowspan="2"> Item </th>

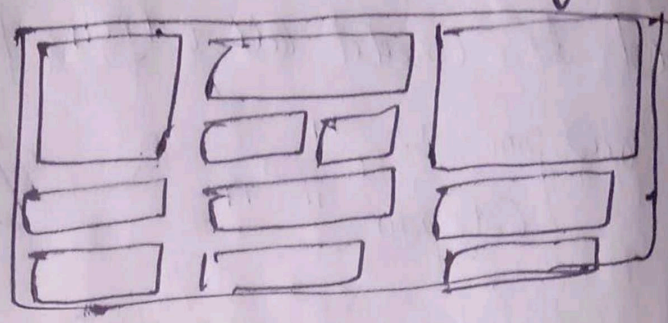


y <body> <table> or <table border="black">
<Caption> food menue </ Caption>
<thead> <tr> <th> Item </th>
<th Colspan="2"> price </th>
<th rowspan="2"> Item </th>
<tr>
<td> Frooti </td>
<td> Rs. </td>
<td> 0.4 </td>
</tr>
<tr>
<td> Samosa </td>
<td> 20 </td>
<td> 0.4 </td>
</tr>
</table>
</table>

Pratic Question:7

Recreate following:

A test table with merge cell



Form In Html

Form are used to collect data from the user.

```
<Form>
Form Content
</Form>
```

Action Attribute: Action attribute is used to define what action needs to be performed when a form submitted or where the form data should be sent.

```
<Form action = "/action.php">

<Form action = "/action">
```

Input:

- used to create multiple form control.
- there are multiple type of input that can be created using type attribute.

< input type= " " >

Cible:- < body >

< form action= " / action" >

< input >

output

Type Attribute:- < input >

< input type="text" >

< input type= " password" >

< input type= " Number" >

< input type= "time" >

< input type= " Color" >

< body > < form action= " / action" >

~~< input >~~ userName

< input type= " text" > < br > [a,b,c,d]

< input type= " Password" > < br > [* * * * *]

< input type= " Number" > < br > [1 2 3]

< input type= " time" > < br >

< input type= " Color" > [redacted]

Form Elements:

placeholder attribute:-

< input type= " text" - placeholder= " Enter Name" >

output. user name [Enter Name]

Label:

- Label element represent a caption for an item in a user interface.

```
<label>
  Enter your Name:
  <input type="text" placeholder="UserName">
  </label>
```

Label Classical Approach:-

```
<label for="username">Enter your username:</label>
<input type="text" id="username" placeholder="UserName">
```

id → unique Name
elements

Code:

```
<div>
<form action="/action">
  <label for="username">UserName:</label>
  <input type="text" placeholder="Enter username" id="username">
  <br>
  <label for="password">Password:</label>
  <input type="password" placeholder="Enter password" id="password">
  <br>
</form> </div>
</body> </html>
```

output: Username: [Enter User]
Password: [Enter Password]

Button element

<button> Submit </button>

Username: [Enter User]

Password: [Password]

Button: [Submit]

Type of Button: ① <button type="submit"> Submit </button>

② <button type="button"> do some thing </button>

③ <button type="reset"> do something </button>

<button type="reset"> reset </button>

[Submit] [Hello] [reset]

4 Button using input:-

<input type="submit">

<input type="submit" value="click me">

Name Attribute:-

Name of the form control. Submitted with the form as part of a name/value pair.

<input type="text" placeholder="Enter Name" id="Username" name="Us

Code:

```
<form action="/action">
<div>
<label for="Username">Username:</label>
<input type="text"
        placeholder="Enter name"
        id="Username"
        name="Username" />
</div>
```

Username: [abcd]

Password: [abcd]

← Q File: /server/?username=abcd&pass=abcd

<form action="https://www.youtube.com/results?search_query=go

<input type="text" placeholder="Searching on tube" name

<button> search </button>

Practical Q. 2 Create a search option that redirects its search request google

17 Checkbox:

<input type="Checkbox" name="age" id="Age" checked/>

<label for="Age"> Don't</label>

Code: <form action="/server>

<input type="Checkbox" id="age" name="age" />

<label for="age"> Don't</label> Checked

Output: ☑ Don't

☑ Don't

Radio Button:- Yes/No, male/female

<input type="radio" name="fruit" id="apple" value="Apple

<label for="Apple"> Apple. </label>

Dropdown:-

<select name="profession" id="profession">

<option value="Student</option>

<option value="dev"> Developer</option>

</select>

Developer ☑

Rohit

Shyam

Range: <label for="Volume">Volume</label>

<input type="range" min="0" max="100" name="vol">

Text Area:

<label for="feedback">Please provide feedback:</label>

<textarea id="feedback">

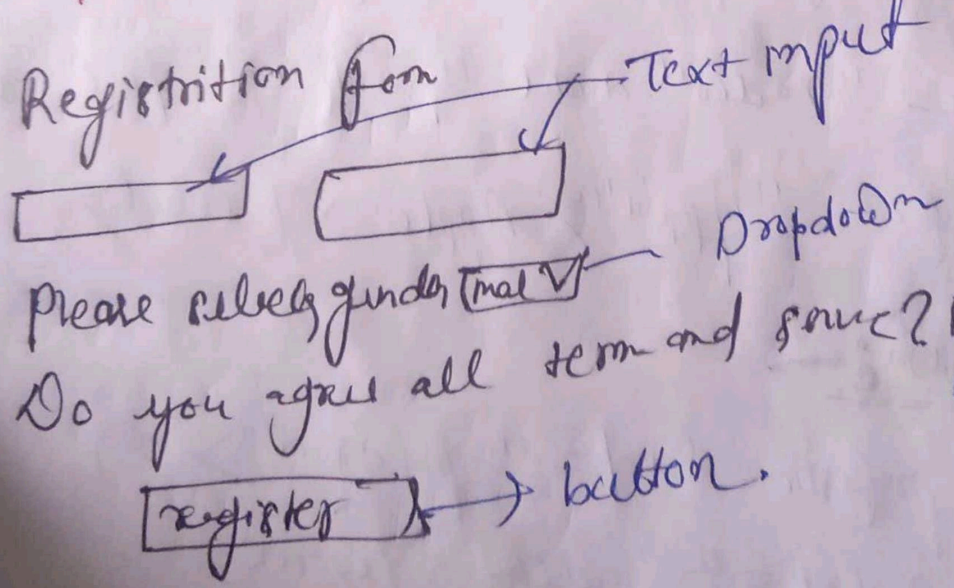
name="feedback" placeholder="Text Area"

rows="5" cols="100"

</textarea>

Q9 Practical

Registration form



Day. 5 CSS

Cascading style sheet:-

sheet add style in Document.

It is language that is used to describe the style of document.

Eg text color, bg color, button. } Properties.

Basic format of CSS:

Selector

```
h1 {
    color: red;
}
```

terminator a=2
b=3

Property value

How to include styles:-

① **Inline styles:-** writing style directly inline on each element.

```
<h1 style="color: Red">Appna College</H1> } inline
```

② **Using style tag**

Style is added using the <style> element in the same document

```
<style>
    h1 { colored: red;
    }
</style>
```

(multiline style)

```
h1 { color: red;
     Background-Red;
}
</H1>
```

- External stylesheet:

Writing CSS in a separate document & linking it with HTML file.

style.CSS + HTML linking

```
<Head>
    <link rel="stylesheet" href="style.CSS">
</Head>
```

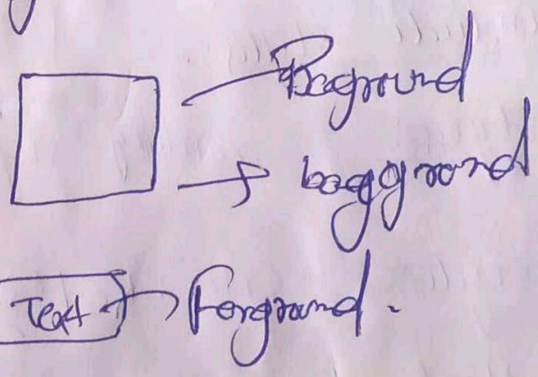
(relation)

path

Color Property

- used to set the color of background.

```
Color: purple;
Color: #ffffff;
```




```
Color: blue;
Color: #ff000;
Color: rgb (214, 124, 200);
```

Apply CSS:

```
h1 {  color: blue;
p {   color: red;
}
```

< button > click me </button >

< button > click me </button >

button{ color: gold;

}

Background Color Property:-

We to set the color of background.

background-color : purple;

{ background color : gray; p{ background color = gray;

} }

Practical Question

Hi there!

my name is Tom & I am there to help you.

need help

h2 : orange

button: blue background Color
white text

p : background color white text

Color System

(16) → 150
Color → color

Named Color

Color = black;

Each browser recognizes around 140 named color.
e.g. red, pink, purple etc.

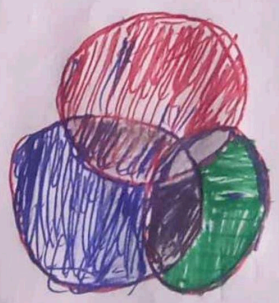
Colorful <H1> Rohit Yadav</

CSS H1 { color: Red; } output Rohit Yadav
}

Color System

RGB Red green blue (0 - 255)

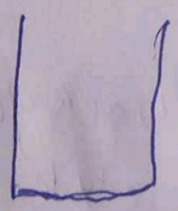
Color = rgb (0,0,0);



Red Green blue (0 - 255)
red is rgb (255, 0, 0)
black is rgb (0,0,0)
Yellow is rgb (255, 255, 0)

Red Channel: 0 — 255
Blue Channel: 0 — 255
Green Channel: 0 — 255

Red + green → yellow
green + blue → greyish
blue + red → pinkish



8. Color System

Hexadecimal or Hex → 16 color = #FFFFFF
R g b

Letter: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 (10) | A, B, C, D, E, F (6)

rgb (R, g, B)

Decimal

Digit: 0 to 9
used in daily math
Base 10

(0-9) (0-9) (0-9)
000 → 999

Binary

Digit 0 & 1
used in Computer
Base-2

01 01 01
000 (Black) , 111 (Light)

Hexadecimal
6 (0-9) 10 (A B C D E F) (10, 11, 12, 13, 14)

Range → 0 - 255
hexade (00 - ff)

ffa511

000000 Black
R g B

0000ff → Blue
R g b

ffff00 yellow

9. Practice Question:- Write the hexcode equivalent to rgb(255, 255, 255) and tell what colour is it

R g b

white

ffffff = white Color

Write the best possible rgb and hexCode for following Color:-

yellow rgb(225, 255, 0) = #ffff00

blue rgb(0, 0, 255) = #0000ff

10. Text Properties

- text-align
- font-weight
- text-decoration
- line-Height
- letter spacing
- font Size.

Text-align

```
h1 { text-align: Centre;
     border: Solid;
     width: 50%;
}
```

Centre

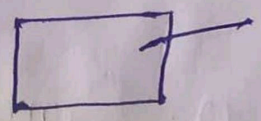
```
h3 { text-align: right;
p { text-align: Right;
  }
```

11. Font Weight

weight
bold

```
Font-weight: normal        //400
Font-weight: bold          //700
Font-weight: 100
h3 { Font-Weight: bold
     }
p { Font Weight: 700;
  }
```

Font-weight: bolder
Font-weight: lighter } parent



text-Decoration

Sets the appearance of decorative lines on text underline.

abc — underline

~~abc~~ — line through

abc (overline)

text-decoration: underline

text-decoration: overline

text-decoration: line-through

```
h1 {
    font-weight: bold;
    text-decoration: underline; or
    text-decoration: overline;
    text-decoration: line-through;
}
```

Color change of line: text-decoration: green underline;

abc

abc

abc

text-decoration: double blue overline; underline

① line through

② Another tag) line

```
a {
    text-decoration: None;
}
```

12. Line Height

Control the height of the line of text

line-height: normal

line-height: 100;

p{ text-align: justify;
line-height = 100;
}

a ⇄ b
line-high

2.5 Normal

letter-spacing: 10 px

pixel: px
unit of measurement

letter-spacing

Controls the horizontal spacing behavior between text character.

letter-spacing: normal;

letter-spacing: 10px px ⇒ pixel.

h1 {
letter-spacing = 5 px;
}

Fixed vs Scalable

Font-Size in CSS

Absolute	Relative
px	%
pt	em (d) current size
pc	rem parent } Relative

cm	ch
mm	vh
in	vw + many more.

1 inch = 96 px

72 pt = 1 inch

12 pt = 1 pc

Pixel: (px) most commonly used absolute unit

96 px = 1 inch.

not suitable for impressive website.

Font-size: 50 px;

5 px

100 px

P { font-size = 2 px;

}

14. Font family

specifies a prioritized list of one or more font family names.

Font-family: arial; [illegible]

Font-family: Montserrat; sans-serif;

Question practice.

Design the given Apna College logo using CSS instruction..

- Set the Font family to league spartan.
- Set the Color of text to #ffa511b
- Center align the text.
- make the text size 55px & Font weight 900
- Set spacing between letters to 2px
- Set the line hight to 1.5px the normal value.
- Add a teal double underline
- upper Case all the letter only using CSS. (use google)

Day 6 UNIVESAL Selector (Selector)

h1 { color : red ; }

Selector — property — Value.

* — universal selector

Syntax * {

To select every thing in element. }

<Header>
<Div>
<button>

Element Selection

Select all elements of same type.

h1, h2 {

}

Id Selector (Form)

Select an element based on the Value of the element id attribute.

id Selector

my id { property : Value ;

}

login { ~~bg~~ color : green;
green.
}

# signup { background-color: Red;
}

Class Selector

select an element based on their class attribute.

• my class { property: value;
}

class selector

Ex <h1>
<button>
<p> } text color : red (more than 3 element)

<h1 class="a">
<h1 ~~class~~ button class="b">
<p1 class="a">
<button class="upvote"> upvote </button> } class Selector

Practice Question Set the color paragraph inside divs with the id "myId" to green.

Descendant Selector

e.g Select all paragraph inside divs.

descent select div p { property: value
}

descendant Selector

p span ul li

grandfather → parent → child

p a { color: red

}

Adjacent Sibling Combinator

e.g. Heading 3 that comes immediately after any paragraph

p+h3{ property: value;

}

<p> <\/p>
<h3> <\/h3>
<h3> <\/h3>
 <h3><\/h3>
<\/div>

p ⇔ H3 sibling HTS-DET

elements + element

<p>
<h3> > sibling

p+h3{ text-decoration: underline;

}

Child Combinator

e.g. Select all button which are direct children of span.

child Combinator{ property: value;

}

<p>

span > button{ property : value
}

div > input
{ property : value;
}

Attribute Selector

- Select element based on the presence or value of a given attribute.

input [attr = "value"] { property : value;
}

input [type = "text"] {
color: red;
}

input [type = "password"] {
color: red;
}

Pseudo Class

A keyword added to a selector that specifies a special state of the selected element (s).

: hover
: active
: checked
: nth of type

↳ Special state.

Touch करने से change हो जाये,

→ Element : pseudo class

① Hover : element : hover

↳

eg button : hover {
background : color

↳ specificity h2 : hover {
text-decoration : Underline;
}

Active : click

button : Active {
background : Pink
Color-scheme : red
}

Pseudo Element

A keyword added to a selector that let you style a specific part of the selected element.

* First-letter → h1, p abc
* First-line ::
* Selection ::

① h1 :: first-letter { Color : green }

② p :: first-line { Color : Red }

③ p :: Selection { background-Color : Red }

CSS : Cascading Style sheet.

• what is cascade in CSS ?

The CSS cascade algorithms job is to select CSS declaration in order to determine the correct value for CSS property.

h2 { background-color : yellow; }

h2 { background-Color : blue; } ← Final (same element)

Selector specificity

What is specificity...

Specificity is a algorithm that calculate the weight that is applied to a given CSS declaration.

0	0	1
id	class attribute	element pseudo element

id > class > element

myId {

}

h2 { background-color: Red

}

0	0	1

$\boxed{001 = 1}$

.myClass:hover { color: red;

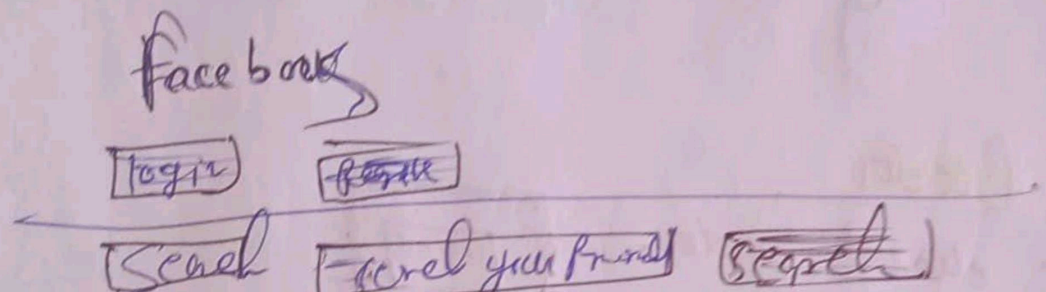
}

0	2	0

020 = 2

Practice Question:-

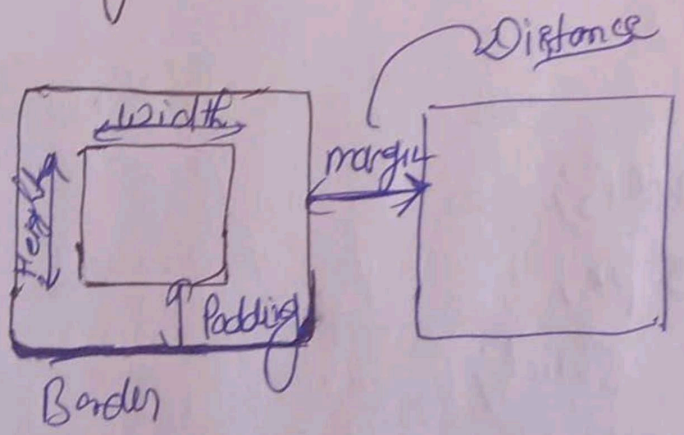
- Set font family of entire document to Courier New.
- Set text color for all H1 heading to #1b744c4.
- Set background color of each search button to black & text to white.
- Set background color of login & signup button to Aqua.



Day 7:-

Box Model of CSS

- Height
- Width
- Border
- Padding
- Margin

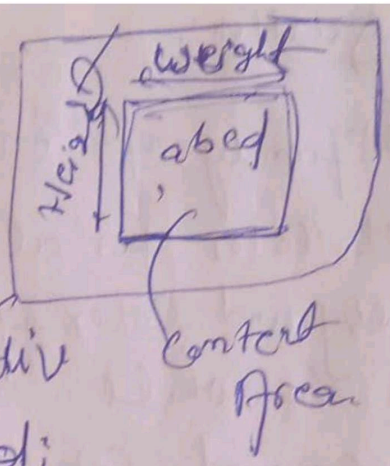


Height

By default, it sets the content area height of the element

```
div {
    height: 100 px;
}
```

```
<div>
abcd
</div>
```




```
div {
~~abcd~~ background-color: Red;
}
```

Border

- used to set element border.

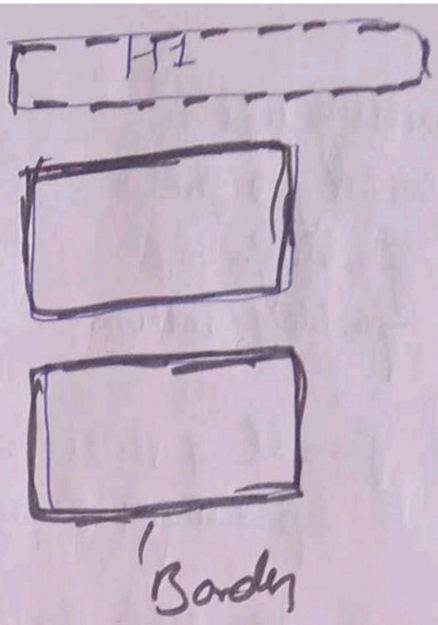
- Border-width
- border-style
- border-color

```
div{ Height: 200px;
     width : 500px;
     background-color: black;
     border-width: 5 px;
     border-style: Solid;
     border-color: black;
}

H1 } border-width: 2px;
     border-style: dashed;
     border-color: Pink; }
```

Border Shorthand: (One line)

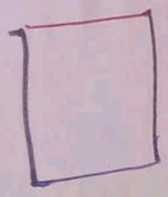
width | style | Color

```
div { border: 5px solid black;

H1 { border: 2px dashed green;
}
```



Border Side: To control on individual side of the box.

- Border-left
- border-right
- bordertop
- borderbottom



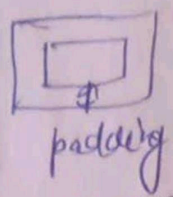
```
{ border-left-color: yellow;
  border-left-style: dashed;
}
```

Border Radius: used to round the corner of an element outer border edge.

```
div {
    border-radius: 15px;
}

div {
    border-radius: 50%;
}
```

Padding

- padding-left
- padding-right
- padding-top
- padding-bottom.




```
div{ padding-left = 50px;
}
```

Padding-Shorthand

for all 4 slide
padding: 50px;

#2 top & bottom / left & right
padding: 1px 2px;

#3 top / left & right / bottom
padding: 1px 2px 3px;

#4 top / right / bottom / left
padding: 1px, 2px, 3px, 4px

margin

to set spacing on the outside of border.

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

```
Div{ margin-bottom: 25px;
     margin-left: 20px;
     margin: 20 px;
}
```

Practical Question:

① Create a div 3 sibling divs inside it

② For the ~~div~~ outer div set following style.
height: 200px, width 75px and border as 2px solid black color

Inline v/s Block ① Display

It set whether an element is treated as a block or inline element and the layout used for its children.

display: inline → Inline element
display: Block → Block element

```
h2{ display: block;
    Background-color: red;
}

h2{ display: inline;
}
```

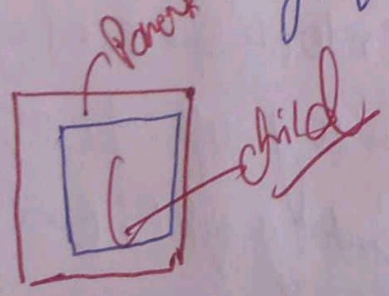
UNIT in CSS:

Absolute	Relative
px	%
pt	em
pc	rem
cm	ch

→ %

It is often used to define a size as relative to an element's parent object.

e.g. width: 33.33% // relative to the parent.
margin-left: 50% // relative to the parent size

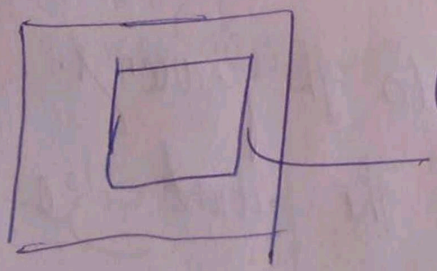


Em: ~~UNIT~~ ~~Relative to~~

Em font size of the parent, in the case of typographical properties like font-size, and : font size of the element itself, in the case of other properties like bredth. text → typographical properties

#outer { height: 400px;
width: 400px;
border: 2px solid
background-clr: Blue;
font size: 20px

#inner { height: 100px;
width: 100px;
background-color: cornflower;
font-size: 2em;



font size = 20 px
fontsize = 3 em
padding = 1em;

button { fontsize = 50
border-radius: 15 px;
border: 5px solid gre
padding-left: 1 em;
padding right: 1 em;

Rem (Root Root Em)

Font size of Root element.

```
section { font-size: 15px;
        }
div { font-size: 0.5 rem;
    }
```

Day 8 Level 4 CSS - next step

Alpha Channel: "Set the opacity for a Color Range from 0 to 1"

0 → hidden

Color: rgb
Red, green, blue

Red :- 0-255
Blue :- 0-255
green :- 0-255
Alpha 0-1
0 → hidden, 1 → visible

rgba (255, 255, 255, 0.3)

```
<body>
<div>
</div>
</body>
```

```
div{
    height: 200px;
    width: 200px;
    background-color: ~~red~~ rgba(255,0,0,0);
}
```

0 0 → 1 1

Completly hidden — Completly visible

Opacity

Set the opacity for the element Rang 0 - 1

opacity : 0.5

```
# opacity { background-Color : yellowish;
          opacity : 0;
          opacity : 0.5;
}
```

opacity element पर Apply होती है

CSS- Transition

Transition enable you to define the transition between two states of element.

(time सेट कर सकते है)

```
div { transition : 1s;
}

div : hover {
    background-Color : black;
    border-radius : 50%;
}
```

```
<div>

</div>

transition-Duration : 2s;
```

Transition Short Hand:-

(background-color: red

| property name | duration | timing function | delay

transition: margin-top 2s ease-in-out 0.2s;

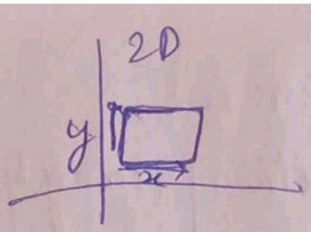
```
div{
    transition: delay: 2s;
}

<div>
red

</div>

div{ margin-top: 10px;
     margin: 0 auto;

}

div{ height: 100px;
     width: 100px;
     background-color: Green;
     margin: 0 auto;
     margin top: 10px;
     transition: margin-top 2s ease-in-2;
}
                ease-out 0.2

div:hover{
margin-top: 400px;
background-Color: Red;
```

CSS Transform

This property let you rotate, scale, skew, or translate an element

① Rotate transform: rotate(45deg);

Scale Transform



(ii) Scale

transform: scale (0.5);

transform: scale (1.5);

x y

transform: scale (0.5, 1)

Translate

transform 2D

: translate (50px, 50px);

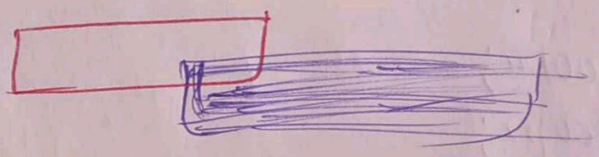
transform: translate(10px);

transform: translateY (10px);

third {

transform: translate (100px, 100px);

}



Skew

(iv) transform: skew (30deg);

fourth { transform: skew (45deg);

Five {

transform: rotate (30deg); translate: (45deg)

}

Box Shadow

- It adds shadow effect around an element frame.

x offset, y offset, blur radius, shadow color

box-shadow: 2px 2px 10px green;

Background Image

background-image: url("./somepath/img.jpg");

background-size: Contain / Cover / Auto;

Scale ✗ Repeat Crop ✗ — Crop ✗ Scale ✗ — Stretch

Splash website download image.

```
div {  width: 10px;
       height: 10px;
       margin: 0;
       background Color: Pink;
       background-image: url("img.jpg");
       background-size: Contain;
                        Cover
                        Stretch (Auto)
```

Creating a Card Hover

Position:- The position CSS property set how an element is positioned in a document.

The top, right, bottom and left properties determine the final location of positioned element.

- Static
- relative
- Absolute
- Fixed

Statics:- The top, right, bottom, left and z index property have no effect. This is the default value.

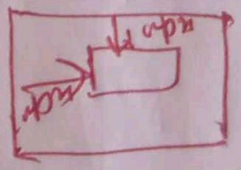
e.g.

```
<Section>
  <h2> Static Demo </h2>
  <div></div>
  <div id="static"></div>
  <fluttary
```

```
div{
  height: 100px;
  width: 100px;
  background-color: gray
}
#static{
  background-color: yellow
  position: static;
  top: 100px;
  left: 100px;
```

Relative: The top, right, bottom, left, and z index properties have no effect. This is the default value.

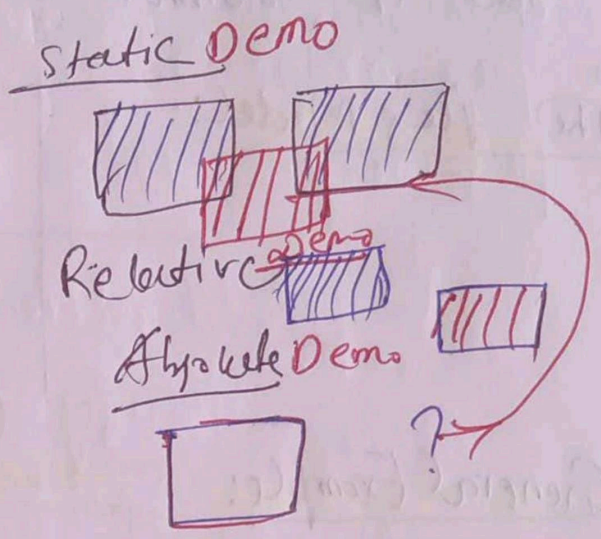
"The element is relative to itself based on the value of top, right, bottom and left"



Absolute:- The element is removed from the normal document flow, and no space is created for the element in the page layout.

It is positioned "relative to its closest position ancestor" if any; otherwise, it is placed relative to the initial containing block.

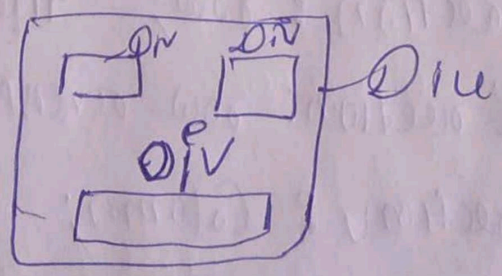
#Absolute { background-clr: yellow;
position: relative
top: 10px;
left: 100px;
}



Position-fixed: – The element is removed from the normal document flow, and no space is created for the element in the page layout. It is "positioned relative to the initial containing block. established by viewport

Practical Create a Smile Page.

Rough ① Outer Div
②

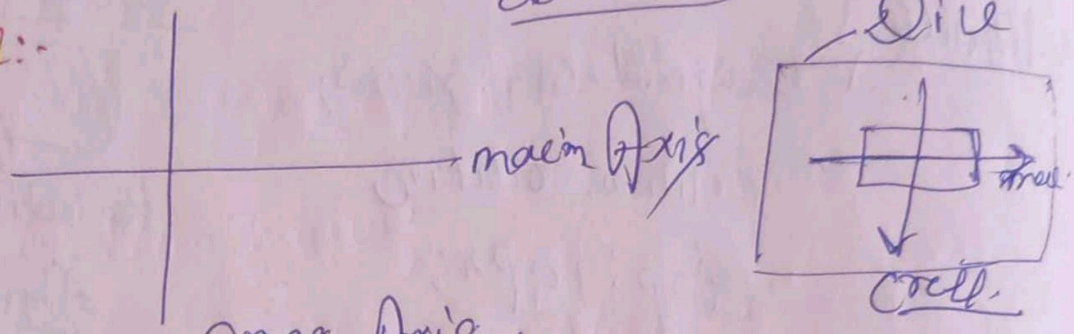


Day: 9 Flex Box → Responsive → Screen vertical Adaptable

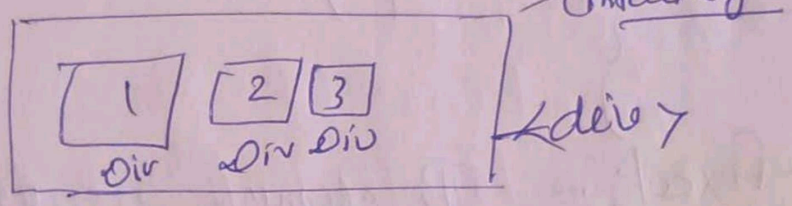
- What is flexBox:
- "It is a one-dimensional layout for arranging item in row or Column."

direct = row

The flex Model:-



General Example:




```
<div>
  <div> </div>
  <div> </div>
  <div> </div>
</div>
```

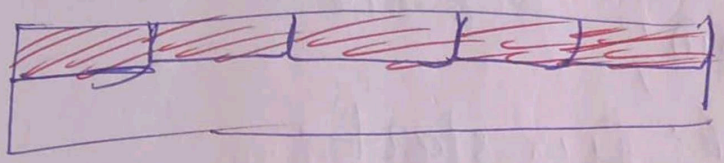
display: flex

flex box direction: It sets how flex item are placed in the flex container along which Axis and direction.

- flex-box-direction: row; main Axis, left to right
- flex box-direction: row-reverse; main Axis, right to left
- flex box-direction: Column; Cross Axis, top to bottom
- flex box-direction: Column-reverse Cross Axis, bottom to top.

```
container
# {  height: 300px;
   width: 80%;
   background-color: Azure;
   margin: 10px solid black;
   display: flex;
   flex-direction: row;
}
```

```
# Container div {
   Height: 100px;
   width: 300px;
   color: blue;
   font-size: 1.5em
}
```



Flex box direction

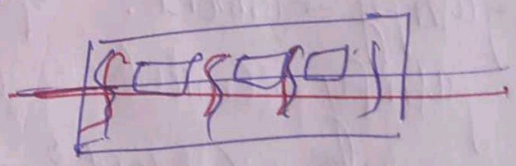
It set how flex item are placed in the flex container along which axis and direction.

- flexbox-direction: row: main Axis; left to right

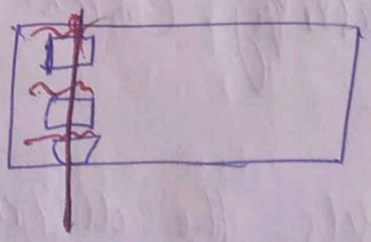
Justify Content

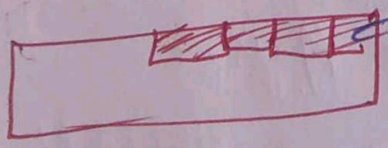
Dell now the browser distribute space b/w and around content item along the main Axis

- Justify-Content: flex-start;



- Justify-Content: flex end;
- Justify-Content: Center;





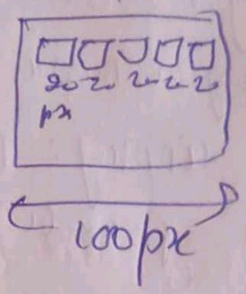
flex-end

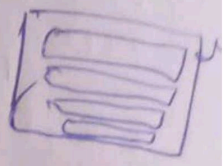
- Justify-Content: space-between;
- Justify-Content: Space-around;

Justify-Content: Space-evenly

Flex Wrap

Set whether flex item are forced onto one line or can wrap onto multiple lines.





- flex-wrap: nowrap;
- flex-wrap: wrap
- flex-wrap: wrap-reverse (will start range)

{ flex-direction: row;
justify-content: flex start

}

Align Item

Distribute our item along the Cross Axis:-

- justify-content: flex-start;
- justify-content: flex end;
- justify-content: Center
- justify-content: Base line

Align Content

it sets the distribution of space between and around content item along a flexbox's cross Axis.

- Align-content: flex-start / flex-end / center
- Align-Content: space-between / space-around / evenly
- Align-Content: baseline;

Align-self (individual properties)

Aligns an item along the cross Axis.

- align-self: flex-start;
- align-self: flex-end;
- align-self: center;
- align-self: baseline;

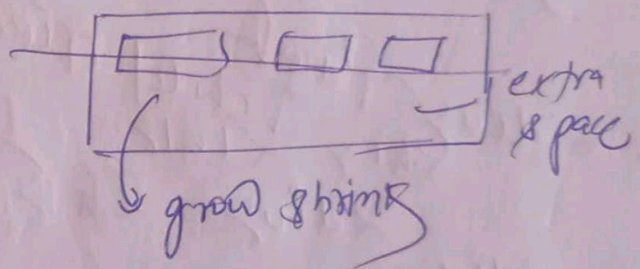
```
#org {
    align-self: flex-start;
}
```

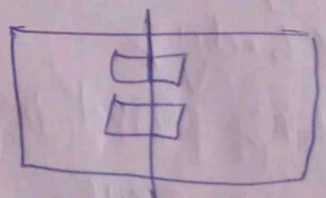
```
<div id="org">
```

flex Size

- flex-basis.
- it sets the initial main size of flex item.

flex-basis : 100px





main Axis.

flex shortHand

- flex-grow | flex-shrink | flex-basis

  flex: 2 2 100px;

- flex-grow | flex-basis

  flex: 2 100 px;

flex-grow (unitless)

flex: 2;

flex-basis

flex: 100px;

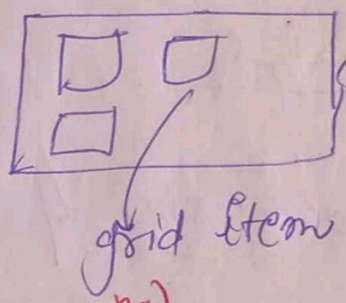
Day 10:- CSS Part 6 — CSS Grid

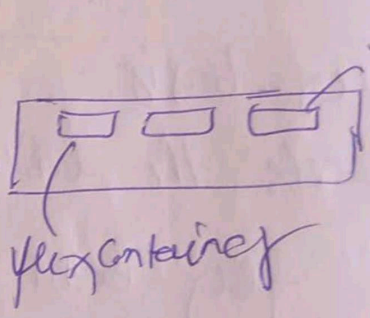
Setting a container's display to grid will make all children grid items.

Container{

display: grid

}





Parent

child (flex)

subchild

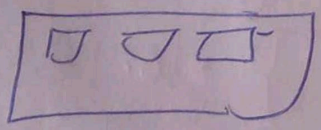
```
<div>
  <h1> <|h1>
  <div> <h1> <|h1> <|div>
<div>
```

eg

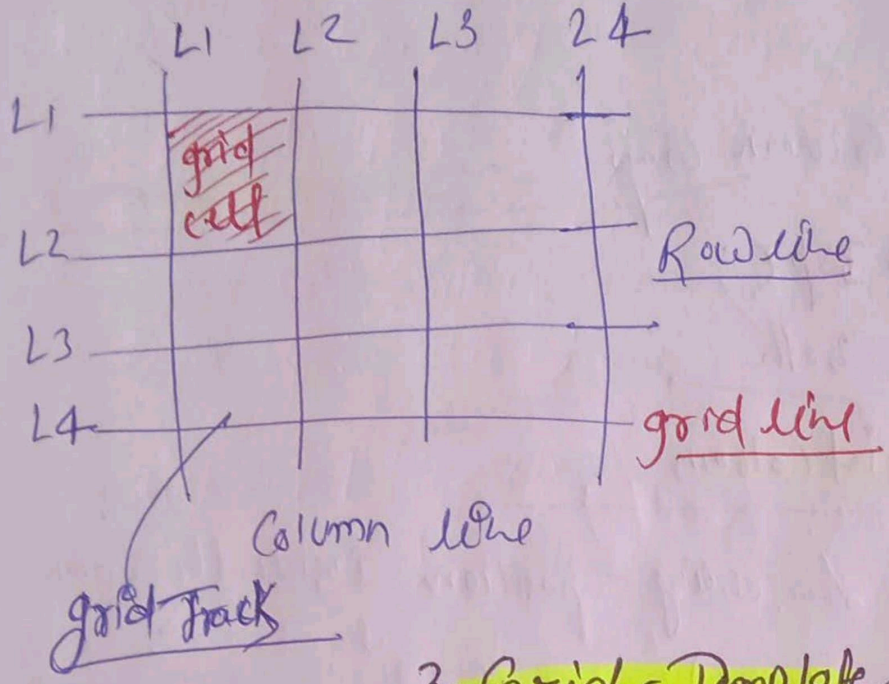
```
<body>
  <h1> Grid playground <|h1>
  <div class = "Container">
    <div class = "item"> 1 <|div>
    <div class = "item"> 2 <|div>
    <div class = "item"> 3 <|div>
    <div class = "item"> 4 <|div>
  <|div>
```

CSS

```
.Container{
  width: 600px;
  height: 300px;
  background-clr: gray;
  display: flex or grid;
}
.items{
  width: 100px
  height: 100px;
  background-clr: red
  border: 2px solid black
  text-alig: center;
}
```



2. Grid Model



Grid line:

Grid cell:

Grid track:

3. Grid - Template

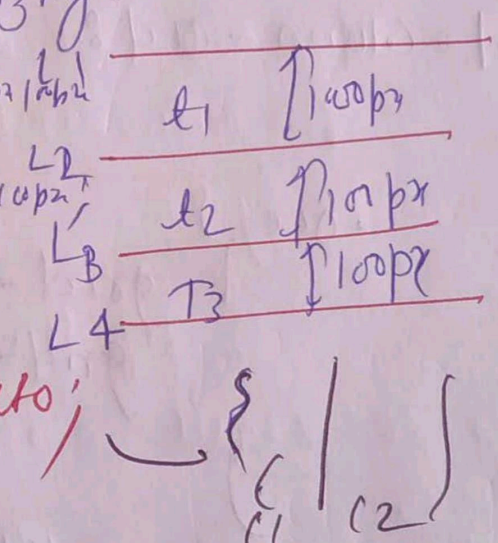
They define the lines & tracks sizing.

grid - template - rows; 100px 100px 100px

grid - template - Column: 100px 100px 100px;

grid - template - Colum: auto.

grid - template - Colum: auto auto;



4. Grid - template - Repeat

Repeat is used to divide all availiable space.

grid - template - columns: - repeat (Count, 1fr) 3, 1fr

grid - template - Row: repeat (Count, 1fr) (3, 1fr)

1 fraction of 100% width

5. Grid - gap

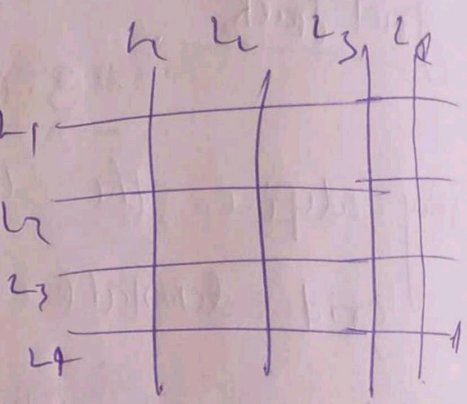
Define the gap between the line.

Row-gap: 10px;
Column-gap 10px;
grid-gap: row Gap Column Gap
10px 30px;
10px; both

6- grid Column

Define an item starting & ending position inside the Column.

grid-Column-start: line number (1)
grid-Column-end: line number (3)



.one {
grid-Colum-start: 1;
grid-Colum-end: 3;
}

grid-Colum: start_col / end_col
grid-Column: start_Cl. / span Number

.one {
grid-Colum: 1/4;
}

7. grid-Row

.one { grid-Row-start: 1;
grid-Row-end: 3;
}

{ grid-gap: 1/4 }

Common Properties :-

- justify-items (container)] Horizontal
- justify-self (item)] Horizontal
- align-items (container)] Vertical
- align-self (item)] Vertical
- place-items
- place-self

Animation CSS

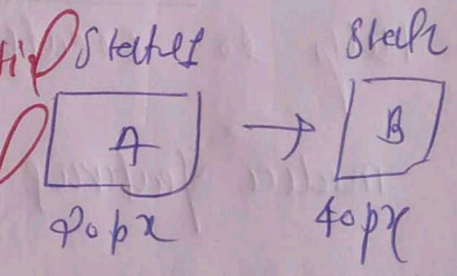
To Animate CSS elements:

(a) keyframe my name { anything like AB

from { Font-size: 20px; } initial state

to { Font-size: 40px; } final

}



(a) keyframe Rohit {

from { Font-size: 20px;

}

to { Font-size: 40px;

}}

- Animation - name
- Animation - duration
- Animation - timing-function
- Animation - delay
- Animation - iteration-count
- Animation - direction

div {
Animation name: Robot;
Animation-duration: 2s;
animation-timing-function: ease-one;
Animation-delay: 0;
Animation-iteration-count: 3;
Animation-direction: normal;
}

9 Media Queries

- Help to create a responsive website.

media features - width (of view port)

@ media (width: 400px)
div {
background-color: Red;
}
}

10 Animation & hard Hard

Animation: Robot 38 case-in OS infinite

11 Media Query orientation

orientation (of view port)

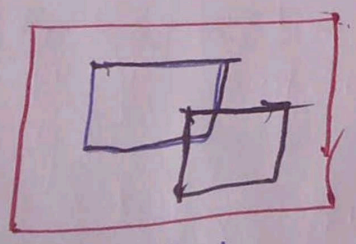
{ portrait
landscape }

@ media orientation : landscape){
div{ background-Color: Red;
}}

Create a responsive page.

Z index

It decide the stack level of elements:-
overlapping elements with a large z-index over those with a smaller one.



z-index : auto (0)
z-index : 1/2/
z-index : -1/-2/- —

BootStrap

FramWorks

Bootstrap

• understanding

pre-baill components.

" Bootstrap is a powerful, feature-packed frontend toolkit
Build anything from prototype, ~~feature-packed~~ to production - in minutes."

↓ css files

style.css.

Benefit of Bootstrap:

- Make the development faster & Easier
- Good choice of pre-built Components
- good browser Compatibility

2. Using of Bootstrap

Cdn → Content delivery networks (Server)

← Bootstraps

```
<title> Bootstrap </title> <link href="http://cdn"
rel="stylesheet"/>
```

use

```
<link href="style.sheet" href="style.CSS"/>
```

what we will Cover:-

Components

pre-built and reusable piece of of user Interface (UI)

eg button, form

Layouts

overall structure and arrangements of the elements

e.g Containers grid

Container Layout

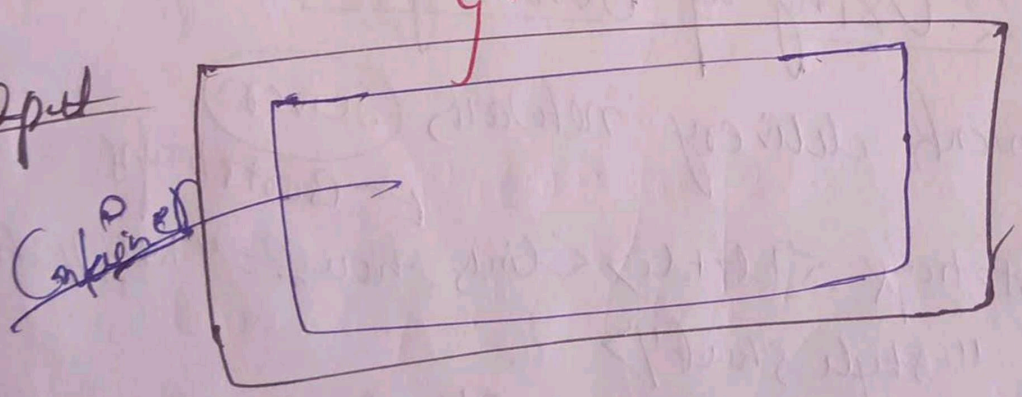
Containers are used to contain, pad, and (sometimes) center the contents within them.

```
<body>
<div class="Container">
  <H2> Bootstrap Demo </H2>
  <P> Rohit K- ———— ———— </P>
</div>
```

CSS

```
.Container {
    background-color: Red;
}
```

output



Button → class = btn

- Layout → Components → button.

<button type="button" class="btn">base class</button>

- <button class="btn">click me</button>

<button class="btn btn-dark">btn one.</button>

<button class="btn btn-color">btn two</button>

<button class="btn btn-success">btn two</button>

CSS. .btn-success { background-Color = Red;

<button class="btn btn-outline-success">btn two</button>

<button class="btn btn-outline-success btn-lg">btn</button>

<a href="#" class="btn btn-dark">google me</button>

Badge

- 10+

<h3> Notification 20+

output: Notification 20+

<button class="btn btn-primary">
Followers
1001

Followers 1001

Alert

Download failed try again

alert-primary
or
black.

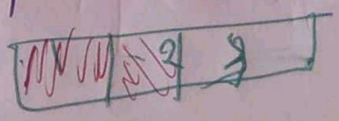
Congratulation

Button group

<button class="btn btn-success">one</button>
<button class="btn btn-danger">two</button>
<button class="btn btn-danger">3</button>

1 2 3

<button class="btn btn-success">one</button>



Java Script

Using the Console

Console (Java Script)

MCQ REPL

Read-Evaluate-Print loop → Repeat

Save: .js

Calculation

output

Mac: Cmd+K

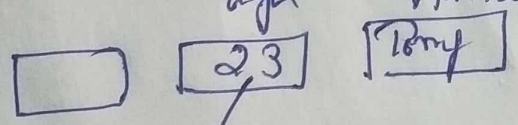
Window: Control+L

} Clear

Arrow Key: ↑ ↓

Variable

A variable is simply the name of storage location



a=5

name = "Rohit"

Data Type

Primitive:

1. Number
2. Boolean
3. String
4. Undefined
5. Null
- Bigint ✗
- Symbol ✗

a = 25 ↵

25

type of a

→ Number

Count (Number)

text (String)

follow → yes or no?

(Boolean)

Instagram page:

167 post 305K follower 4 following

Count (String)

Number

- Positive (14) & Negative (-14)

integer (45, -50)

floating number - with decimal (4.6, -8.9)

a = 5 b = 2

a = 1.5 a*b = 10

operation

a = 20

b = 10

sum = a+b // Addition

diff = a-b

mul = a*b

div = a/b

- Remainder

5/.2 = 1

Exponential (power operator)

2**3 = 8

a = 20
b = 10
a % b = 0
9 % 2 = 1

$\boxed{a \times b = ab}$

$2^3 = 2 * * 3$
a**b = 27

NAN in JS

- The NaN global property is value representing Not-A-number

0/0
→ NAN
Type of NAN
output: 'Number'

NAN+1
→ NAN
- NAN*NAN
→ (invalid)

operator Precedence

- This is the general order of solving an expression.

()
**
*, /, %
+, −

BODMAS

(5+2) ÷ 7 + 1 * 2
→ 7 ÷ 7 + 1 * 2
→ 3

Q (2+1) * 3
→ 9

3 / 1 + 2 * * 2
→ 7

4 + 1 * 6 / 2
→ 7

Practice

Q/ let age = 23;

age+2; // after 2 years

output → 25

Comment → //

let Hindi = 80;

let Math = 100;

let Eng = 90;

let avg = (hindi + eng + Math)/3.

→ ~~135~~ $\frac{270}{3}$ = 90

Assignment operator

age = age+1

age+=1

age = age-1

age-=1

age = 24

age* = 2

48

age = age*1

age*=1

age = age/2

age = age/.4

age% = 4

Unary Operator

age = age+1 | age = age-1
age+=1 | age-=1

age++ // increment | age-- // decrement

(use then change)

a++ Post increment | a-- Post Decrement
++a pre increment | --a → pre Decrement

(Change, then use)

let age = 10;
let new age = ++age
age
→ 11

Question Practicle

let num = 5; ← num = 5
let newNum = num++; ← NewNum = 5 num=6
newNum = ++Num; ← newNum = 7 num=7

Identifier

- All Java Script variable must be identified with unique names (identifier)
-

- Name can contain letter, digit, underscore, and dollar sign (no space).
- Name must begin with letter.
- Name can also begin with $ and _.
- Name are case sensitive (y and Y are different variable)
- Reserved word (like Javascript Keyword) (cannot be used as names.

Ex let price$;

Camel Case

- Way of writing identifier.
- Camel Case (J.S. naming Convention) → let fullname;

Snake_case let full_name;

Pascal Case let FullName;

Boolean in J.S

- Boolean represent a truth value → true or false / yes, no

let age = 13;

let isAdult = false;

age = 21;

isAdult = ~~25~~ true;

output true

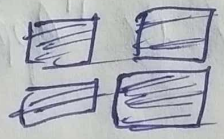
is adult = True; ✗ All small letter

let a = 5
a = true boolean

let a = 5;
typeof a
output: 'Number'
a = true;
typeof a
output: 'boolean'

What is type Script

Static typed, where J.S. is dynamic typed.
Designed by Microsoft.



a = 5 ✓
a = true ✗
a = 6 ✓

Question Practice:

(9) Find Error:

let 1age = 5;
let 2age = 10;

let age1 = 6;
let age2 = 10;

let marks = 75;
let is pay = True;

let is pay = true;

String on J.S

String are text or sequence of characters

```
let name = "Tony stark";
let role = 'iron man';
let char = 'a';
let num = '23';
let empty = " ";
```

'this is "Apple"'

```
let sentence = 'this is "apple"';
```

output:- This is "Apple"

String Indicy

```
let name = "Tonny stark";
```

T	O	N	N	Y	_	S	T	A	R	K
0	1	2	3	4	5	6	7	8	9	10

index:-

name [2] > N index = position

Code:

```
let name = "Tonny stark";
name
```

→ 'Tonny stark'

name[1]

→ 'o'

name.lenght

⇒ 10

"Sharadha".lenght

→ 7

"Sharadha"[6]

→ 'a'

"Sharadha"[0]

→ 'S'

let name = "Tonny stark",

name[name.lenght];

name[name.lenght - 1];

(-2)

(-3)

⟹ "K"

Catatenation (Adding string together)

"Tonny" + " " + "stark"

→ "Tony stark;

let first name = "Tony".

let last name = "stark".

let full name = first name + last name

full name

"Tony stark"

→ "tonny" + 1

tonny1

null & undefined in J.S

Undefined: A variable that has not been assigned a value is of type undefined.

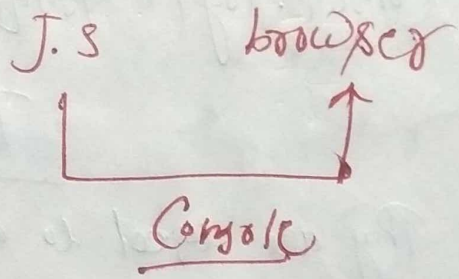
null: The null value represent the intentional (intentionally) absence of any object value.
To be explicitly assigned.

	null:
let a;	let a = null;
> a	a

#2 Console.log() — JavaScript

To write (log) a message on the Console.

```
Console.log ("Apna College");
Console.log (1234);
Console.log (2+2);
Console.log ("Apna", "College", 123);
```



Code: `Console.log("Hello!");`

output → Hello!

`Console.log(1+2);`

output → 3

- `Console.log("Hello", "world", num, (1+5));`

→ Hello world 123 6

Linking J.S file (Add function)

```
<Script src="App.js"></Script>
```

Template Literal

They are used to add embedded expression in string

```
let a=5;
let b=6;
Console.log (`your pay $(a+b) rupees`);
```

let pencil price = 50;
let eraser price = 50;
Console.log(" Total price is : ", pencil price + eraser price, "Rupees")

Template literal

They are used to add embedded expressions in a string

```
let a = 5;
let b = 10;
Console.log ('your pay $ (a+b) rupees');
// Console.log ("Price is", a+b, "rupees");
let output = 'The total price is : $' + pencil + eraser + 'Rupees.';
Console.log( output);
```

operator in Java Operator

- Arithmetic (+, -, *, / %, **)
- Unary (++, --)
- Assignment (=, +=, -=, *=, /=, %= etc)

Comparision

- Logical.

a + b
operand operator

Comparison Operator

Comparision operator to compare 2 values.

> greter than
>= greater than or equal to
<
<=
==
!= (Not equal to)

let age = 23.
age >= 18.
true false
Comparision operator

5 == 5
→ true,
5 != 6
→ true

* == Compares value, not type.

"123" == 123
→ true
1 == '1'
→ true
0 == ' '
→ true
0 == false
→ true
null == undefined
→ true.

• ===
Compare type & value

"123" === 123
→ false
1 === '1'
→ false
0 === ' '
→ false
0 === false
false
null === undefined
false

Comparision of non-number

'a' > 'A'

→ true

'a' > 'b'

→ False

'b' < 'c'

true

'B' < 'C'

true

'*' < '$'

False

'a' -> 61, 'b' -> 62

'A' -> 41, 'B' -> 42

'p' > 'p'

True

unicode

60 > 50

Conditional Statement

- If - else
- Nested if-else
- Switch

Condition

True

False

If statement

```
// Some Code before if
if (Some Condition){
    Do something
}
// Some Code after if
```

```
if (age>5)
print ("Son")
else (Shayan)
```

Comparision of non-number

'a' > 'A'

→ true

'a' > 'b'

→ False

'b' < 'c'

true

'B' < 'C'

true

'x' < 'y'

False

'a' -> 61, 'b' -> 62

'A' -> 41, 'B' -> 42

'p' > 'p'

True

unicode

50 > 50

Conditional Statement

- If - else
- Nested if-else
- Switch

Condition

True

False

If statement

```
// Some Code before if
if (some Condition){
    Do something
}
// Some code after if
```

```
if (age>5)
print ("Son")
else (Shayan)
```

Q Practicle Qc

· Create a traffic light system that show what do based on color.

color red → stop
yellow → slow down
green → go

let color = "red"
log("stop")

Code: let color = "green";
if (color === "red"){
console.log("stop");}
if (color === "green"){
console.log("go slow");
}
if (color === "yellow"){
console.log("go");
}

output: go

Else if statement

if

```
if (condition1){
    // Do something
}
else if (condition2){
    // Do Something Else
}
else if (condition3){
    // Do something else.
}
```

Q

```
let age = 2;
if (age >= 18){
console.log ("Adult");
} elseif (age <= 18){
    console.log ("teen");
}
    elseif (age = 2){
    Console.log ("good boy");
}
```

Output teen

Else:

```
let age=25;
if (age>=18){
 Console.log("You Can Vote");
} else { Conlole.log("You Cant Vote");
}
```

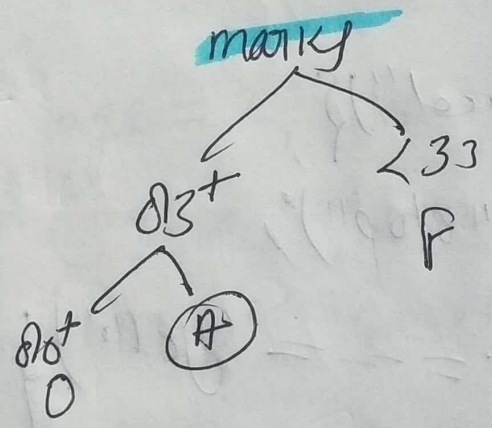
Code:

```
let color="red";
 if (color===="red"){
    Console.log("stop");
 } else if (color===="yellow"){
    Conyole.log(" go");
 } else if (color===="green"){
    Conyole.log("Slow go");
 } else {
    Console.log(" traffic signal broken");
 }
```

Nested if-else

Nesting is writting if-else inside if-else statements. It can have many level.

```
if marks >= 33
    if marks >= 80
        print "O"
    else print "A"
else
    print "better next time"
```



Ex

```
let marks = 45;
if (marks >= 33){
Console.log("Pass");
} if (marks > 33){
Console.log("Grade A");
}
else {
Console.log("better luck next time");
```

Logical operator

Logical operator to combine expression

→ && Logical AND (exp 1) && (exp 2) — Ampersand

(a<3) (b<c)

false, true true false.

→ || (pipe) logical or operator

→ ! logical not operator

&& logical AND

(5>3) && (3>1)

true true

→ true.

(5>3) && (5>6)

false

true && true

→ true

true || false

→ true

false || false

false.

or ||

if (marks >= 33 || marks >= 80)

Let marks = 99;

if (marks >= 33 && marks >= 80)

Console.log("Pass");

Console.log("A grade");

}

}

Not !

a>b

if (!marks >= 33)

Truthy and Falsy

Everything works if true or false (in boolean context)
This doesnt mean their value itself is false or true but they are treated as false or true if taken in boolean context.

False Value: False 0, -0, 0n (~~Bight~~ Bigint value) "" (empty string) null, undefined NaN.

Truthy value:
Everything else.

```
if (true){
    Console.log(" It has true value");
else{
    Console.log(" It has false value);
}
```

Switch

used when we have some fixed value that we need to compare to

```
let color = "red";
switch (color){
case "red":
  console.log("stop");
    break;
case "yellow":
```

Alert & Prompt

- Alert display an alert message on the page

```
alert("Something is wrong");
```

Prompt display a dialog box that asks users for some input.

```
prompt("Please enter your roll no.");

alert("Something is wrong");
```

String Method

Method – Action that can be performed on objects

Formate: String Name. method() . console.log()

```
let str = "apna College";
str.abc()
```

Str. trim() Trimes whitespaces from bothend of string and return a new one

```
let msg = "Hello ";
msg.trim();
→ 'Hello'
```

```
let password = prompt("Set your Password");
console.log(Password.trim());
```

String are immutable in java:-

No changes can be made to strings whenever we do try to make a change, a new string is created and old one remain same.

```
let msg = "Apna ";
```

let str = "Random string";

- str.toUpperCase() → Random string

str.toLowerCase() → Random string

- name.toUpperCase() → RANDOM STRING.

String method with Argument

Argument is a name ~~is~~ value that we pass to the method.

Format: String name.method(arg)

let name = "Rohit";

To find O

Index of Return the first index of occurence of some value in string or give if not found.

let str = "love Coding"
0 1 2 3 4 5 6 7 8 9

str.indexOf("love") // 1

str.indexOf("j") // -1 (not found)

str.indexOf("o") // 2 only

Method Chaining

Using one method after another. Order of execution will be left to right.

⇒ str. toUpperCase(). trim()

```
let msg = "        Hello.  ";
let newmsg = msg.trim();
Console.log(newmsg);
newmsg = newmsg.toUppercase();
Console.log(newmsg);
let newmsg = msg.trim().toUpperCase();
```

output: HELLO

Slice

Return a part of the original string as a new string

let str = "Love Coding"
0 1 2 3

str.slice(5) "Coding"

str.slice(1,4) ()"love"

Str.slice(num) = str.slice(length - num)

Replace method

specify a value in the string & returns a new string with the value replaced.

```
let str = "I Lae Coding";
str.slice("Lae", "do")        // 'I do Coding'
str.slice("o", "x")           // 'IlxvCoding'
```

Array (Data Structure)

linear collection of things.

```
let student = "Aryan";
let students = "Aman";
let student2 = "Rohit";
```

```
let students = ["aman", "Aryan", "Rohit"]
```

visudisation of Arrays

2	4	6	8
0	1	2	3

```
let name = "Shredha";
name[0];
name[6];
nam.lenght
typeof name
```

```
let nums = [2,4,6,8];
num1
output [4] (2,4,6,8);
nums(2);
nums(3);
```

Creating Array

let info = ["shradha", 23, 89.9]; // mixed Array

info

output: 'shradha' 23, 89.9

Empty Array: arr = [];

arr.

arr.length

Array are Mutable (Change हो जाते हैं)

⇒ let fruit = ["mango", "Apple", "litchi"];

fruit[0] = "banana";

Banana

Array Method

① Push: add to end

Unshift: add to start

Pop: - delete from end & return it

Shift: - Delete from start & return it.

Code:

car = ["Bolero", "Swift", "Scarpio"];

car.push("Ferrari");

output: 'Bolero', 'Swift', Scarpio, 'Ferrari'

car.pop(); Remove करने के लिए

⇒ 'Bolero', 'Suzuki', 'Scorpio'

Shift - starting से Remove करने के लिए

car.shift();

car.unshift('Fin');

⇒ let followers = ["a", "b", "c"];

let blocked = followers.shift()

output:- 'b', 'c'

blocked

'a'

Practicle Question

Start : ['January', 'July', 'March', 'August']

→ ['July', 'June', 'March', 'August']

Array method

indexof : returns index of something let primary = ['red', 'yellow', 'blue'];

primary.indexOf("yellow");

primary.index of ("green");

→ -1

primary.index of ("yellow");

→ -1

Include

Array method

Let primary = ["Red", "blue"] let secondry = [green, voilet, yellow]

Concat :- merge 2 Arrays → Concatenate

primary.Concat (secondry);

output: (6) ['red', 'yellow', 'blue', 'orang', 'green', 'voilet'

Reverse :- Reverse an Array

- primary.reverse();

output: [blue, yellow, red]

Slice

Slice : Copies a portion of an arrays.

let colors = [red, yellow, blue]
0 1 2

color.slice()

red, yellow, blue

Slice (start, end) exclusive

optimal.

Color.slice(1,2);

[blue]

Color . Slice(2, 3);

blue

Splice Method

Splice: remove / replace / add element in place.

Splice(start, delete Count, Item0, Item N)

Color. splice(4);

let Color = ["red", "yellow", "blue", "orange", "pink", "white"];

output: [pink, white]

Color:

⇒ [red, yellow, blue, orange]

Color . splice(0,1);

⇒ [red] o/p

Color

o/p → [yellow, blue, orange]

color.splice(0, 1, black, grey);

o/p → yelow

Color; [black, grey, blue orange

Array Method

Sort: Sorts an Array

let days = ["Sunday", Tuesday, "Friday"];

days.sort()

Chars = ['a', 'b', 'd', 'c'];

a, b, d, c

Practice Question

Q. start = [January, July, March, August]
final = [July, June, March, August]

Q. Return the index of Javascript from the given array if it was reversed.

['c', CSS, HTML, Javascript, Python, Java, C#, SQL]

Array References (address in memory)

"name" == "name"

true

[1] == [1]

false

[1] == [1]

false

let num = [1, 2, 3]

ref variable

1	2	3
0	1	2

⇒ let arr1 = [1]

let arr2 = [1]

arr1 (1,2,3)

arr2 456

let arr = [a, b, c];

let arr copy = arr;

arr == arr copy;

o/p true

Constant Arrays

address

Const arr = [1, 2, 3, 4];

or

let a = 5;

Const a = 5;

let arr = [1, 2, 3, 4]

*arr.push(5);

arr

o/p ⇒ (1, 2, 3, 4, 5)

* Const g = 10;

g

o/p → 10

g = 9.8;

o/p → false error

Nested Array (Multidimensional Array)

array of array.

let num = [[2,4], [3,6] ,[4,8]];

num

o/p :- (3) [Array(2), Array(2) , Array (3)]

0 : (2) [2,4]
1 : (2) [3,6]
2 : (2) [4,8]

- num.lenght

o/p 3

- num [0]

o/p [2,4]

* num[0].length

2

* num[0][0]

2

	0,0	0,1
1,0	2	4
2,0	3	6
	4	8

1,1

2,1

3 rows = No. Array

Colum = Element

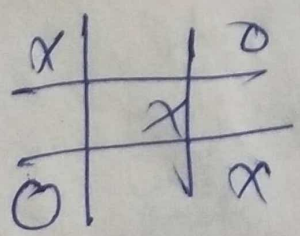
num.[row][colum]

num[0][0]

num[0,1][1]

num [0][0] (1|2)

num[0][1]



Practice Quest.

Create a nested array to show the following tic-tac toe game state.

Loop (Repeat)

```
for (initialization, condition ; updation) {
    do something
}
```

Start end update.

```
for ( let i=1 ; i<=5 ; i++){
    Console. log (i);
}
```

Dry Run (Paper Pen)

- Print all odd number (1 to 15)

```
for( let i = 1 ; i <= 15 ; i = i+2){
    console. log (i)
```

start end

→ o/p :- 1, 3, 5, 7, 9, 11, 13, 15

Even Reverse

```
for. (let i=10; i>=2 ; i=i-2){
    Console. log(i);
}
```

Output: 10, 8, 6, 4, 2

Infinite loop

```
for (let i = 1; i >= 0; i++){
}
for (let i = 1; i <= 5; i--){
}
for (let i = 1; . ; i++){
}
```

ending condition

missing condition.

Multiplication Table of 5

```
for(let i = 5; i <= 50; i = i + 5){
    console.log(i);
}
let n = prompt("write your number");
    n = parseInt(n);
for (let i = n, i <= n*10; i = n + i){
    console.log(i);
}
```

Nested loop

```
for (let i = 1; i <= 3; i++){
    for (let j = 1; j <= 3; j++){
        Console.log(j);
    }
}
```

output:

```
outer
1
2
3
outer
1
2
3
outer
1
2
3
```

while loop

- while ((condition) {
  // do something

```
let i = 1;
while( i <= 5){
    Console.log(i);
    i++;
}
```

Dry run:

		o/p
i = 1	T	→ 1
i = 2	T	2
i = 3	T	3
i = 4	T	4
i = 5	T	5
i = 6	F	

```
1
2
3
4
5
```

favorite movie

2: Condition

```
let fav.movi= "Avtar";
let guess= prompt(" ");
while( guess != fav.movi) && (guess != 'quit') {
  Console.log("wrong");
guess=prompt(" ");
}
```

Loop with Array

- let fruit= ["mango, "appli", "banana", "litchi", "orange"];

```
for(let i=0; i<fruit.length; i++){
    Console.log(i, fruit(i))
}
```

O/p: mango
apple
banana
litchi
orange..

Loop with Array

Nested loop with Nested Array:-

```
let heroes = [["ironman", "spider man", "thor"], ["superman", wonder woman",
                                                  "flash"]];
for (let i=0 ; i < heroes.lenght ; i++){
    Console.log("my # {i}")
for (let j=0 ; j < heroes[i].lenght ; j++){
    Console.log(heroes[i][j]);
}
}
```

0 1

o/p: 0:(3) [ironman, spider man, thor]
1(3) [superman, wonder woman, flash]
lenght: 2

For of loop

```
for (element of Collection){ Array, string
    // do something
}
```

Code:

```
let fruits = ["mango", "apple", "banana", "litchi", "orange"];
for ( fruit of fruits){
    Console.log(fruit);
}
```

o/p:- Mango
Apple
Banana
Orange
Litchi

```
for (char of "Apna College") {
    console.log(char);
}
```

Nested for of loop

(1)
```
let hero = [["Superman", "Batman", "Wonder Woman"],
            ["Spiderman", "ironman", "thor"]];

for (list of heroes) {
    console.log(list);
}
```

(2)
```
for (list of hero) {
    for (name of list) {
        console.log(name);
    }
}
```

o/p: Superman
Batman
Wonderwoman
Spiderman
Iron man
thor

To Do App

- List - to show all today
- add - to add to todo
- delete - to delete a task
- quit - to exit the todo.

Code:

18 JS Object literal (Data Structure)

* used to store keyed collection & complex entities.

property => (key, value) pair

object are collection of properties.

Console: let arr = [];
typeof arr.
o/p: object.

object

	student	Shradha
property	name →	Shredha
p2	age	23
p3	marks	94.4
	key	value

```
let delhi = {
    lattitude : "28.704° N",
    longitude: "77.1025° E",
};
```

Ex

```
const student = { name: "Shradha",
                  age: 23,
                  marks : 94.4,
                  city : Delhi
};
```

Thread / Twitter Post

- Create an object literal for the properties of thread / twitter post which include:
  - user name
  - Content
  - Like
  - Repost

Get value

```
let student = {
    name: "shradha",
    marks: 94.4
};
```

```
post["content"];
post.content
```

```
student["name"]
```

or

post.content or object.content

> Javascript automatically convert objects key to string
Even if we made the number as key, the number will be converted to string.

Code:

```
Const ob = {
    1: "a",
    2: "b",
    3: "c",
    4: "d",
};
```

O/p →

```
1 'a'
2 'b'
3 'c'
4 'd'
```

Add / upadate Value

- Change the city to "mubai"
- Add new property gender: "Female
- Change the marks to "A1"

```
Const student = {
    Name: "shradh",
    age: 20,
    mark: 20.5,
    city: Delhi,
}
```

O/p:

```
Name: "Shradh"
age: 20
marks = 20.5
city = Delhi
```

Student.city

O/p → Delhi

Student.city = "Mumbai";

O/p → Mumbai

Object of object

String information of multiple student.

```
Const classinfo = {
    aman: {
        grade: "A+",
        city: "Delhi".
    },
    Shradha: {
        grade: "A",
        City: "Pune",
    },
    Raman: {
        grade: "O",
        city: "Mumbai".
    };
```

Array of object

String information multiple students.

```
Const info = [
    { name: "Aman",
      grade: "A",
      City: "Mumbai"
```

{ name: "Rohan"
city: "Mumbai"
}
};

Math object

Property	Method
Math.PI	Math.abs(n)
Math.E	Math.pow(a,b) ✓ a^b
	Math.floor(n) ✓ num rounded (nearest)
	Math.ceil(n) ✓ ceiling (larger nearest)
	Math.random() ✓ Different value new.

Math.random()
0.1019
Math.random()
0.1419

Random Integer

From 1 to 10

Step 1 let num = Math.random(); 0.467

Step 2 num = num*10; 4.674774

Step 3 num = Math.floor(num);
4

Step 4 num = num+1;
5

let random = Math.floor(Math.random()*10)+1;

random
o/p → 4

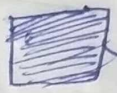
Question

① generate a random number b/w 1 and 100.

math.floor(Math.random()* 100)+1

generate a random number b/w 1 to 5.

Function: 6 J.S



- Block of code to perform specific tasks.

console.log(). arr.push()
pop().

Syntax: ① function funcName(){
// do something
}

function hello(){
console.log("Hello")
}

② function calling (using the function)

funcName(); hello();

Q function print1to5(){
for(let i=1; i<=5; i++){
console.log(i);
}
}
print1to5();

output:
1
2
3
4
5

Practice Question 1

① Create a function that print a poem.

Code:
```
function printpoem(){
    Console.log("Twinkle twinkle little star");
    Console.log("how I wonder what you are");
}

printpoem();
```

Practice Q. 2

• Create a function to roll a dice & always display the value of dice (1 to 6)

$\underbrace{\text{Math.floor(Math.random()*6)}}_{0 \text{ to } 5} + \underbrace{1}_{6}$

```
let rand = Math.floor(Math.random()*6)+1;
    Console.log(rand);
```

Function:-
```
function rollDice(){
    let rand = Math.floor(Math.random()*6)+1;
    Console.log(rand);
}

rollDice();
```

Function with Argument

- Value we pass to the function

```
function function( Arg1 , Arg2 ){
    // do anything
}
```

```
function printName(name){
    Console.log(name);
}
printName("Shradh");
```

o/p Shradha

```
function printInfo(name, age){
    Console.log('${name}'s age is ${age}.');
}
printName("Shradha", age 23);
printName("Rohit", 24);
```

o/p Shradha's age is 23.
Rohit's age is 24.

- Order are important in function.

```
function sum(a, b){
    Console.log(a+b);
}
sum(1, 2);
sum(4, 5);
sum(9, 6);
```

o/p 3
9
15

Practice

· Create a function that gives us the average of 3 num.

```
function CalcAvg(a,b,c){
    avg = (a+b+c)/3
    Conole.log(avg);
}
CalAvg();
```

Create a function to create multiplication table.

```
function printTable(n){
    for (let i=n; i<=n*10; i=i+10){
        Conole.log(i);
    }
    printtable();
```

Return Keyword

Return keyword is used to return some value from the function

input → [Function] → output

```
function funcName (arg1, arg2){
    //do something
    return val;
}
```

Code: function sum(a,b){
return a+b;
}
console.log(sum(3,4));

sum((1,2)+3);
or console.log();

Question

(i) Create a function that return the sum of number from the 1 to n (Argument)

(ii) create a function that return the concatenation of all string in array.

Scope

Scope determine the accessibility of variable, object, and function from different part of the code

- function
- Block
- Lexical

Function Scope:

Variable define inside a function are not accessible (visible) from outside the function.

Code:

```
function calsum(a,b){
    let sum = a+b;
}
calsum();
X console.log(sum);
```

O/p: not sum

global scope → outside
function scope → within the scope

Block scope

Variable declared inside a {} block cannot be accessed from outside the block.

```
{
  ═══
}
```

```
{ let a = 21;
}
console.log(a);
```

O/p: Error.

Lexical Scope (nested function)

- A variable define outside a function can be accessed inside another function defined after the variable declaration.

The opposite is not true

Code:

```
function outerfun(){
    let x = 5;
    let y = 6;
    function innerfun(){
        console.log(x);
    }
    innerfun();
}
```

Question

```
let greet = "hello";
function changeGreet(){
    let greet = "namaste";
    Console.log( greet);
function innerGreet(){
    Console.log(greet);
}}
    Console.log(greet);
    Change greet();
```

Function Expression (name less)

```
const variable = function (arg1, arg2){
    // do or return
}
```

Code:

```
const sum= function (a,b){
    return a+b;
}
    sum (2,3);
let name = "strach";
let sum= function (a,b){
    return a+b;
}
```

Higher order function

A function that does one or both.
- Takes one or multiples function as Arguements.
- returns a function.

Code:

```
function multipleGreet( func, n){
    for( let i=1; i<=n; i++){
        func();
    }
}

let greet = function(){
    Console.log("Hello");
}

multipleGreet( greet, 2);
```

Return a function:- odd / even Number को Detect कर सके.

```
function oddevenTest (request){
    if (request == "odd"){
        Console.log (!(n%.2==0));
    }
    else if (request == "even"){
        return function (n){
        Console.log (n%.2==0);
        }
    else{
        Console.log(" Wrong request");
    }}
```

Method

Action that can be performed on an object.

```
Const Calculator = {
    add: function (a,b){
        return a+b;
    },
    sub: function (a,b){
        return a-b;
    },
    mul: function(a,b){
        return a*b;
    },
};
```

20. (This) keyword

"This" keyword refers to an object that is executing the current piece of code.

code:

```
Const student = {
    name: "shreeh"
    eng: 95,
    math: 60,
    phy: 76
}
student.name
```

o/p > shreeha.

```
const student = {
    name: "xhaath",
    age: 23,
    eng: 95,
    math: 27,
    Phy: 97
    get avg() {
        let avg = (this.eng + this.math + this.Phy) / 3;
        Console.log (avg);
    }
}
```

Accly. (calling object)

Try And Catch:

The try statement allows you to define a block of code to be tested for error while it is being executed

The catch statement allow you to define a block of code to executed, if an error occurs in the try block.

```
try {
    Console.log (a);
} Catch {
    Console.log ("variable dynot");
```

Ex

```
Console.log ("Hellow");
Console.log ("Hellow");
console.log (a);
console.log ("Hellow")
```

o/p: Hellow
Hellow
Error

Code:

```
console.log("Hello");
Conyole.lg("Hello");
try{
    conyole.log(a);
} catch{
    Conyole.log("Caught an error --- a is not define")
}
    Conyole.log("Hello2");
    Conyole.log("Hello2");
```

O/p:

Hello
Hello
Caught an error -- a is not define.
Hello 2
Hello 2

→

```
catch(err){
    Conyole.log("Caught the error");
    Conyole.log(err);
}
```

Miscellaneous Topic

Arrow function:

⇒ `Const funct = (arg1, arg2) => { function define }`

Arrow function

```
Const sum = (a,b) => {
    Conyole.log(a+b);
}
```

```
const sum = (a,b) => {console.log(a+b); };
```

O/P `sum(2, 3);`

→ 5

```
const cube = (n) => {
    return n*n*n;
};

cube(3);
```

O/P → 9

Implicit Return. (Automatic return)

```
const mul = (a,b) => (a*b);
```

Set Time out

Set Time out(function, timeout)

↑ Callback ↓ time (milliseconds)

```
console.log("Hi there");
Set Timeout( () => {
    console.log("Apna College");
},4000);
console.log("Welcome to");
```

4 second

Set Interval

time
↓
execute
↓
excute
↓
execute

```
SetInterval (function, timeout)

SetInterval( () => {
    Console.log ("Apple is good");
  } ; 2000)
,

let id = SetInterval(() => {
    Console.log("Apple is good");
  }, 2000);
  console.log(id);
```

This with Arrow function

Arrow

1) lexical Scope
   parent → Call
   (scope)

function

1) Scope → this → calling object

```
const student = {
    name: "Aman",
    mark: 95,
    prop: this,
}
```

Code:
```
const student = {
    name: "Aman",
    marks: 95,
    prop: this,   // global declaration
    getName: function () {
        console.log(this);
        return this.name;
    },
};
```

fun

obj.fun

student.getName

this = student

arrow func.

obj.func

this

Window

Practice question

- Write an array function that return the square of number 'n'.
- Write a function that print "Hello world" string at interval 2s each.

91. J.S Part-8

Array Method

- forEach
- map
- filter
- Some
- every
- reduce

forEach

```
arr.forEach(some function definition or name);
let arr = [1,2,3,4,5];
function print(el){
    console.log(el);
}
arr.forEach(print);
// or
arr.forEach(function(el){
    console.log(el);
});
```

or

```
arr.forEach(function){
    console.log(function);
});
```

Map

```
let new arr = arr.map(some function definition or name);
let num = [1,2,3,4];
let double = num.map(function(el){
    return el*2;
});
```

Every

Return true if every element of array gives the true for some function. Else return false.

arr. every(some function definition or name),
true false

Code: [2,4,6] . every((el) => el%2 == 0),
true

[2,4,6] . every((el) => el% == 0
true

Reduce

Reduced the array to single value.

arr. reduce(reducer function with 2 variable for (Accumulator, element)),

[1,2,3,4] . reduce((res, el) => (res + el)),

10

Element

(1) accumulator, element → return
(2) acc el → return
(3) acc el → return (Reduced)

let num = [1,2,3,4].
let finalval = num.reduce((res, el) => (res + el))
console.log(finalval);

Spread with object literal

```
const
~~let~~ data = {
    email: "xwhite@gmail.com",
    password: "abcd",
};

let const datcopy = {...data};
    3 {...data, City: "mubai"};
```

Rest

Allow a function to take an indefinite number of arguments and bundle them in an array.

```
function sum(...args){
    return arg reduce ((add, cl) => add+cl);
}
```

Destructuring

Storing value of array into multiple variable.

```
let name = ["tony", "bruce", "steve", "peter", "stan"];
let winner = names[0];
let runnerup = names[1];
let secondRunnerup = names[2];
```

or

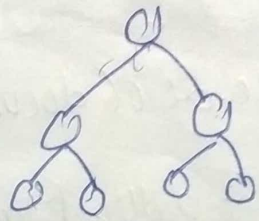
```
let winner, runnerup, secondrunnerup = name;
```

Destructuring object

```
Const student = {
  name: "Rohit",
  age: 14,
  class: 9,
  Sub: ["Hindi", English, "math"],
  pass: "abcd"
};

let username = student.username; or
let pass = student.pass;

let {username, password} = student;
```

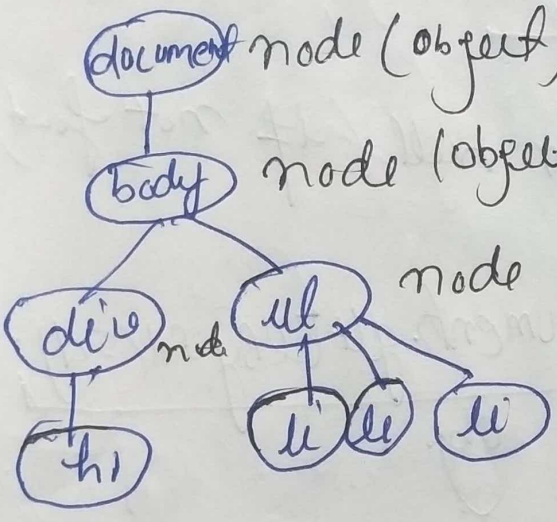
JavaScript 9 part

DOM (Document object Model)



- The DOM represent a document with a logical tree.
- It allows us to manipulates/Change webpage content (Html element)

DOM (Document Object Model)




```
Console.log(document);
```

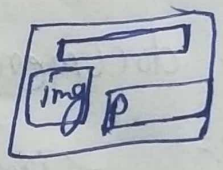
o/p:- # document

```
Console.dir(document);
```

o/p:- property of document

Download Starter Code: There are two step:-

1. Select (img)
2. Changes / manipulation




```
Console.dir(document);
```

o/p:- Property

Selecting Element by ID

```
Console.dir(document.all);
```

For Access:

```
Console.dir(document.all[0]);
```

o/p: h1

Console.dir(documents.all[0].innerText);

o/p:- Spider Man.

documents.all[0].innerText = "Peter Parker";

o/p: "Peter Parker"

Selecting Elements

Return the elements as an object or null (if not

getElementsById. Code: documents.getElBy Id(I

obj.

> documents.getElementsbyId(mainImg);

o/p:- null

> documents.getElementsbyId("mainImg");

o/p:- <img scr = "assets/spidermon" id = "main img">

Now changing the image:-

let imgobj = documents.getElementsbyId("mainImg");

img obj

o/p:- <img src = "assets/ spidermon img.jpg" id = "main

Console.~~dir~~ log(imgobj);

o/p:- <img src = "assets/spidermon" id = "main img

console.dir(imgobj);
→ img # main img
o/p: Property

imgobj.src

o/p:- ———

imgobj.Id

o/p:- 'mainId'

Image को Change करने के लिए;

imgobj.src = "assets/creation.t.png";

document.getElementById ("description");

o/p:

console.dir (document.getElementById("description"));

p# description:

o/p: property:-

document.dir("abc");

o/p: null

Selecting elements

getElementByClassName

Return the elements as an Html Collection or empty Collection

similar to array but not array array

document.getElementByClassName("#.id img");

> HTML Collection (3) [img.oldimg, img.oldImg, Img.oldImg]

→ document.getElementsByClassName("oldimg")[0];

<img src="assets/creation2.png" class="oldimg">

[1];

<img src="" >

[2];

<img src="assets/creation class="oldimg">

App. JS

let smallImgry = document.getElementsByClassName("oldImg

① for (let i=0; i<smallImgry.length; i++){

console.dir (smallImages[i]);

}

Changing all Img:- or

① {

smallImages[i].src = "assets/spidermon_img.png";

console.log(`value of img no. ${i} is changed.`);

o/p:

All small image changed

Selecting El

Selecting elements by ~~class~~ tag name

Return the elements as an Html Collection or empty collection.

document.getElementsByTagName("p");

> Html Collection

0: P

1: p#discription

length: 2

document.getElementsByTagName("p")[1].innerHTML="ab

'abc'

(Single element) Query Selector (method)

Allow us to use any Selector.

document.querySelector('p'); // select first p element

document.querySelector('#myId'); // select first element with Id = myId

document.querySelector('.myclass');

document.querySelectorAll("p") // select all p element

class name & classname: nth-of-type(2n+1)

App J.S

```
Console.dir (document.querySelector("#7"));
Console.dir (document.querySelector("#description"));
Console.dir(document.querySelector("oldPage"));
```

Setting Contents in Objects

Using Properties & methods:-

1) **Inner Text**
Show the visible text contained in a node

2) **text Content** show all full text (Actual .HTML फाइल में लिखा है)

3) **Inner HTML**:- show the full markup.

```
let para = document.querySelector('p');
para.innerText;      // screen पर जो दिखता
para.textContent;    // Actual Html में लिखा है
```

Manipulation

```
para.innerText = "abc";
'abc'
para.innerText = "Hi, I am peter!";
```

→ o/p Hi I am peter

let heading = document.querySelector('h1');
heading.innerText;
'spiderman'

heading.innerHtml = "<u> Spiderman </u>";
'<u> spiderman </u>'

Manipulating Attribute

getAttribute (attr) → getters
setAttribute (att, val) → setters

let img = document.querySelector('img');
img
<img src="assets/spiderman" id="main-img">
img.getAttribute('id');
'main-img'
img.setAttribute('id', 'spiderman');
img.setAttribute('src', 'assets/creation.jpg');

Manipulating styling

style property
obj.style

Console: let img = document.querySelector('img');
console.dir(img);

o/p:- img # mainimg

Property:- CSS:- background-image
J.S:- (Camel case) backgroundImage

let heading = document.querySelector('heading');
heading.style;
heading.style.color = 'purple';
'purple'
heading.style.color = 'purple';

App.JS: let links = document.querySelector('All box a');
for (let i = 0; i < links.lenght; i++){
links[i].style.color = "yellow";
}

Console: let box = document.querySelector('.box');
box.style;

Property:-

Manipulating style

Using class list

obj. classlist

Example: let img = document.querySelector('img');

img.classlist; or heading.classlist.

O/p: length: 0

Value: " ".

heading.classlist.add("abc");

heading.classlist;

heading.classlist.add("green");

heading.classlist

O/p:- 0: "green"

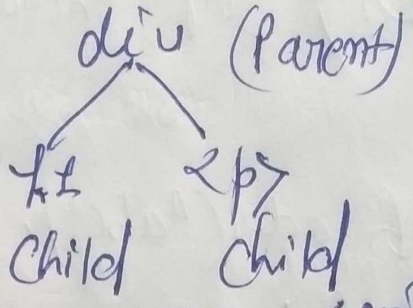
length: 1

value: "green"

Navigation

- Parent Element (single)
- Children
- previousElementSibling / nextElementSibling

```
<div>
  <h1>   <|h1>
    <p>   <|p>
  <|div>
```



Sibling (भाई बहन)

Console:
```
let h4 = document.querySelector('h4');
h4.parentElement
```

O/p :- ---<|div>

Children

```
let box = document.querySelector('.box');
box.children;
```

O/p :- HTMLCollection [h4, ul]

Code Console :- (2)

```
let ul = document.querySelector('ul');
ul.parentElement.
```

O/p :- div class="box">---<|div>

```
ul.childElementCount;
```

nextElementSibling

ul

p:- <ul> — </ul>

ul.children

p:- HTML Collection [li, li, li]

ul.children[0];

p: <li> — </li>

ul.children[2];

p.. <li> — </li>

ul.children[2].previousElementSibling;

p <li> — </li>

ul.children[3].nextElementSibling;

<li> — </li>

ading में Style को Add करना;

let img = document.querySelector('img');

img.previousElementSibling;

p <h1> spider man </h1>

img.previousElementSibling.style;

p CSS property;

img.previousElementSibling.style.Color = "green"

Adding Element

document.createElement('p')

1) appendChild(element)
2) append(element)
3) prepend(element)
4) insertAdjacent(where, element)

Console. document.createElement('p');

o/p: —

Let new p = document.createElement('p');

Console.dir(new p);

o/p:- paragraph property :-

newP.innerText = "Hi, Dom new p";

Console.dir(newp);

o/p: p

let body = document.querySelector('body');

body.appendChild('Hi my self rohit');

o/p: , Hi my self rohit

let box = document.querySelector('.box');

box.appendchild(newp);

To create button:-

let btn = document.createElement('button');

Console.dir('btn');

p. btn property

btn.innerText="Click me";

'Click me'

box.appendChild(btn);

o/p <button> Click me </button>

Append (Element | Text | String

new P.append("this is new text");

new p.append(btn);

Prepend (Element) First/start

Insert Adjacent(where, element)

Remove Element

- remove Child (Element)
- remove (Element)

btn. remove();

o/p: btn removed.

DOM Events

- Event are signal that something has occured. (user interaction)

HTML Editor: <button onclick = "console.log("button was clicked")
</button>

Dry Program: <button

DOM Event

onclick (when an element is clicked)

onmouseenter: (when mouse enters an element)

HTML Editor: <button> click me </button>

J.S
```
let btn = document.querySelector("button");
console.dir(btn);
```

O/p: button property

Event property: on ⇔ start होगा

J.S
```
function sayhello(){
    alert("Hello");
}

btn.onclick = sayhello;
```

J.S

```
let btn = document.querySelectorAll("button");

function SayHello(){
  alert("Hello");
}
btn.onclick = sayhello;
```

Apply loop:

```
for (btn of btn){
    btn.onclick = sayHello;
function sayhello(){
  alert("Hello");
}
```

onmouse Center (Automatically click)

J.S

```
btn onmouse center = function(){
  Console.log("you entered the button");
}
```

Event listner

addEventlistener

```
element.addEventListener(event, Callback){
    Console.log("button clicked");
});
```

```
> let btn = document.querySelectorAll("button");
  for (btn of btns) *
    btn.onclick = sayHello;
  btn.onmouseenter = function(){
        console.log("you entered a button");
    };
        console.dir(btn);
  }
(1)  function sayHello(){
        alert("Hello");
    }
    function sayName(){
(2)     alert("Aparna College");
    }
```

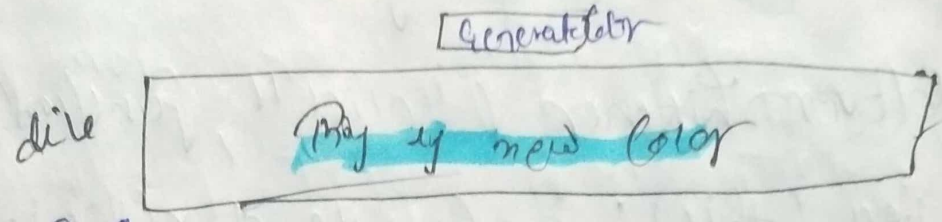
Event Listner

addEventListner

element.addEventListner('event', callback)

```
btn.addEventListner("click", function(){
    console.log("button");
});

* btn.addEventListner("click", sayHello);
  btn.addEventListner("click", sayName);
```

Generate a random Color



Process:
1. ~~h1~~
2. button
3. Div

Html:
```
<p>This</p>   <div class="box"></div>
```

J.S/
```
let p = document.querySelector("p");
p.addEventListner("click", function(){
    console.log("clicked");
});
```

"This" in Event listner

when 'this' is used in a Callback of event handler of something it refer to that something

obj → Event listner handler (event, callback)

```
<Html>
<button> click me </button>
```

```
let btn = document.querySelector("button");
btn.addEventListener("click", function(){
Console.log(this); or Console.dir(this.innerText)
});
this.style.backgroundColor = "blue";
```

Keyboard Events

```
<button> click me </button>
```

Ap-11

```
let btn = document.querySelector("button");
btn.addeventlistener("click", function(){
    Console.log("button clicked");
});
```

```
<input placeholder = " Type something </input>
```

Form Event

```
<form action="action">
<input place holder="username">
  <button> Register </button>
```

JS

```
let form = document.querySelector("form");
form.addEventlistener("submit",function(){
  Console.log("form submitted");
});

* Alert("Form submitted");
}
event.preventDefault()
* event.preventDefault();
```

Extracting form Data

```
let form = document.querySelector("form");
form.addEventListener("submit", function(event){
    event.preventDefault();
    let input = document.querySelector("input");
    console.log(inp);
});
```

input Regmts

Data store. के लिए: console.log(inp.value);

More Event

Change Event:- The change event occurs when the value of an element has been changed (only work on (<input>, <textarea> and <select> element)

Input event:- The input event fires when the value of an <input>, <select>, or <textarea> element has been changed.

```
I.S (et user = document.querySelector("#user");
     user.addEventListner("changed", function).
        Console.log(" input changed").
        Console.log(" final value = ", this.value);
                                user.value);
});
& event.preventDefault();
```

25. Javascript part 11

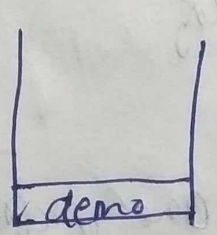
J.S Call Stack — Data structure

Call Stack :-

```
function hello (){
  Console.log("Hello");
}
hello();
```

Last in First out : 4, 3, 2, 1 — o/p : Hello

```
function demo(){
  hello();
}
demo();
console.log("Calling demo fnx");
console.log("done, bye!");
```



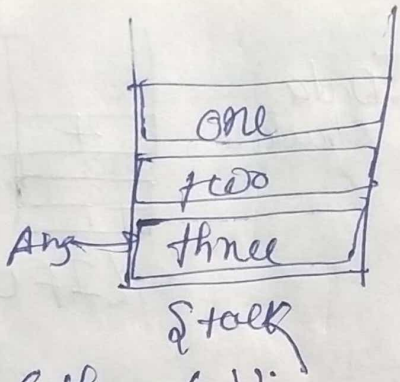
Ex

```
Hello (){
}
```

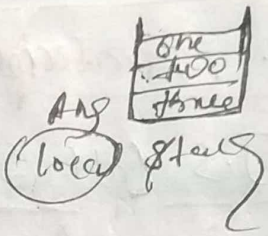
```
demo (){
  (Hello();
}
demo();
```

visualization the Call Stack

```
function one(){
  return 2;
}
function two(){
  return one() + one();
}
function three(){
  let ans = two() + one();
  console.log(ans);
}
```



- fnx Call



first Call

op: 3 (three());

Break point → Browser के through track कर सकते है।

Debuging के लिए use होता है

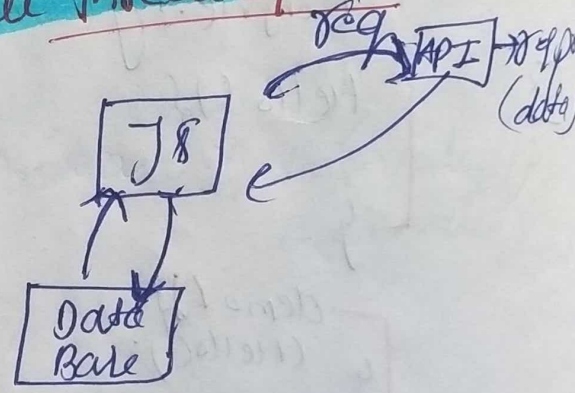
* Break point add करते है

* Local variable * global variable

> Call stack.

JS is Single threaded

one time one program excution.



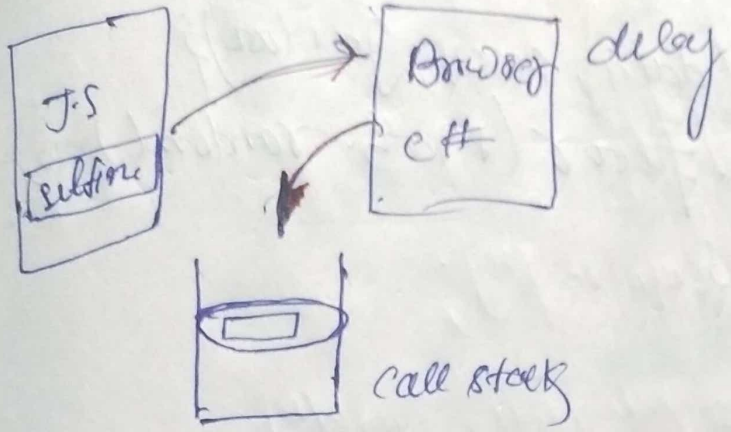

```
SetTimeout (function (){
    Console.log ("Apna College");
    }, 2000);
    Console.log ("Hello ----");
```

Call back

SetTimeout

Browser

JS:
```
SetTimeout(() => {
    Console.log ("apna College");
    }, 2000);
    Console.log("Hello ---")
```



Callback hell

Code:
```
h1 = document.querySelector("h1");
function changeColor(color, delay, nextColor){
    setTimeout(() => {
        h1.style.color = color;
        nextColor();
    }, delay);
}
changeColor("red", 1000, () => {
    changeColor("yellow", 1000, () => {
        changeColor("green", 1000, ()
        });
    });
```

Promises

The promises object represents the eventual completion (or failure) of an asychronous operation and its resulting value.

Code: function saveToDb (data, success, failure){

```
let internetSpeed = Math.floor(Math.random()*10)+1;
if (internetSpeed > 4){
    success();
} else {
    failure();
}
} saveToDb("Apna College", id) => {
    Console.log("your data was saved!");
}, () => {
    Console.log("weak connection, data not saved")
});
```

7. Refractoring with Promises

- The promising object represents the ~~vertical~~ eventual completion or failure of an asynchronous operation and its resulting value.

Promises Resolve & reject function (asynch
success failure

State → pending
rejected (error)
fullfilled (resolved)

8. Promise then() & catch()

```
let request = saveToDBPromise("Apnacollege");
request.then(() => {
    Console.log("promise resolved");
});
.catch(() => {
    Console.log("promise rejected");
});
```

Promise — fullfilled (.then)
— rejected → error
.catch

9. Promises Chaining (Improved version)

```
SaveToDBPromise ("Apna College")
  .then(() => {
  Console.log("promise 1 resolved");
  return SaveToDBPromise("Hello World");
});
  .then (() => {
  Console.log("promise 2 resolved");
});
      .Catch(() => {
      Console.log("Some promises rejected");
  });
```

10 Promises

promises are rejected and resolved with some data (valid result or error)

de:

```
Save toDB Promise ("Apna College")
  .then ((result) => {
  Console.log ("result:", result);
   Console.log( "promise resolved");
  return save TO DB Promise ("Hello world"),
  })
  .then ((result) => {
   Console.log( " result :", result);
   Console.log( " promise2 resolved);
  })
  .catch((error) => {
  Console.log( " error :", error);
  Console.log( "some promise rejected");
  });
```

11. Refracting old Code

26 JavaScript Part 12

Async function:- async & await keywords → Promise

→ Creates an Async function:→

```
async function greet(){
    return "hello world!";   // return a promise
}
let hello = async () => { };   // return a promise
```

Function
- value (none) → P: fulfilled
- error → Rejected

+ return

Throw:- Random error

Await keyword

```
function getNum(){
    console.log(5);
}
async function demo(){
    getNum();
}
```

Code:

```
function getNum(){
return new promise((resolve, reject)=>{
  setTimeout(()=>{ *1
let num= Math.floor(Math.random()*10)+1;
  console.log(num);
   resolve();
},  1000);
});
   async function demo(){
    await getNum();
    await getNum();
```

3. Handling rejection

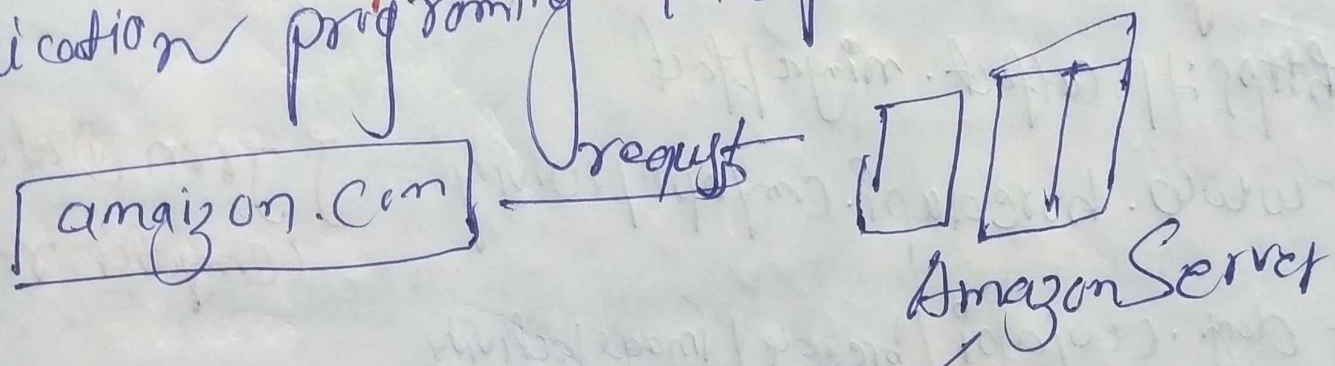
```
* async function demo(){
   await changeColor("red", 1000);
   await changeColor("orange", 1000);
   await changeColor("pink", 100);
   let a=5;
   console.log(a);
   console.log("new number=", a+3);
}
```

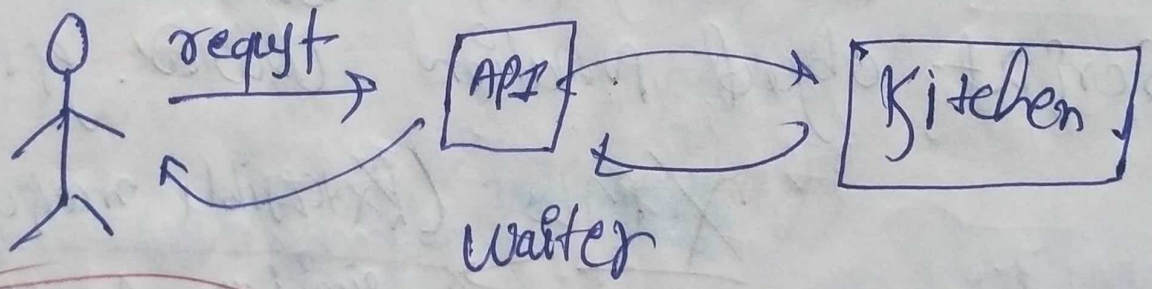
#1

```
let number = Math.floor(Math.random()*25);
if (num>3){
    resolve("promise resolved");
}
```

4. API

Application programming interface





User client

API
↓
data

JSON "Format में Data होता है"

User → request [URL] → API (link, endpoint)
client ← response (data)

Some Random Api:- Twitter API

Paid and free.
(Key use करनी पड़ती है)

https://catfact.ninja/fact
> www.boredapi.com/api/activity
> dog.ceo/api/breeds/image/activity
> google.map api

JSON (Format) Data
(Computer readable)

JSON (Format)

JavaScript object Notation www.jsm.org

XML (Extensible markup language)

similar to Html

XML
<Employeedata>
<employ id="345">
<
</

JSON
{
"string": "Hi",
"number": 255,
"boolean": true

7. Accessing JSON data

Access data from JSON

- JSON.parse(data) Method
  To parse a string data into a J.S object (JSON data → J.S object)

JSON.stringify(JSON) Method
To parse a J.S object data into JSON. (J.S object → JSON)

req → JSON data (string format)
Resp.

Ex let cf = {"fact": "Aproximatly 1/3 of cat owners think their.", "length": 78};

* Console.log(cf);

o/p. error

(एक Data दूसरे from में convert करना)

let validRef = JSON.parse(cf);
Console.log(validref.fact);

o/p :- Object.

8. Testing API request

Tools

① Hoppscoth

② Postman → (download करना पड़ेगा)

website: hoppscotch.io

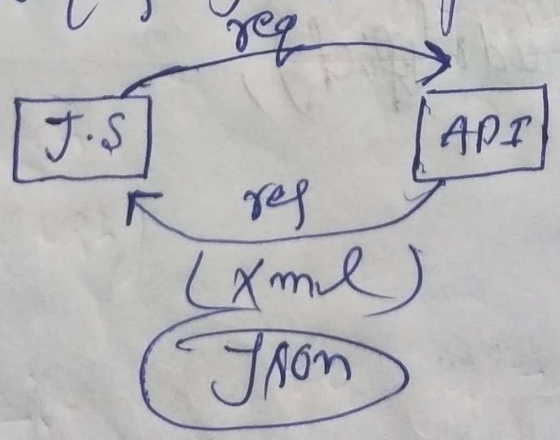
Codeforgeek : parle learning

JSON:{

}

Ajax

Asynchronous Javascript and XML



asynchronous process.

Http Verbs

- Get
- put
- Delete
- Head
- Connect
- Patch
- Parameters

Status Codes

- 200-OK → सब कुछ ठीक है
- 404- Not found
- 400 - Bad Request
- 500- Internal serval error.

Adding Information in URLs

Query Strings → Value.

https://www.google.com/search ?q=harry+potter.

Key Value

?q = applet.

?q = Harry + Potter

URL? name = Shradha & mark = 95

Hopscotch.io

https://api.potterdb.com/v1/movie

HTTP Header

- header : value

→ Inspect → Network tab

response header
request header

api response formate:

- text/html
- Application/Json
- text/plain

→ **meta data** :- data about data.

Our first Request

using Fetch

Fetch (URL)

XML http request ob

App.JS

let url = " https://catfact.ninja/fa

fetch (url)*;

```
*.then ((response) => {
    Console.log (response);
  })  *2 Console.log (res.json()); // promise
  *3                                  return करेगा।
  .catch ((err) => {
  Console.log ("Error -", err);
  });
```

o/p:-

```
*2  return response.json();
*3  .then ((data) => {
      Console.log (data);
    })
```

```
*.then ((data.fact);
  return fetch(url);
  })
  .then ((res) => {
    return res.json();
    })
  .then ((data2) => {
    Console.log (data2);
    })
```

using fetch using async

- using fetch with async/await

```
try {
    let res1 = awaiit fetch(url);
    let data1 = awaiit res1.json();
    Console.log("data1 - ", data1);
    let res2 = awaiit fetch(url);
    let data2 = awaiit res2.json();
    Console.log("data2 - ", data2);
} catch (e){
    Console.log("error :", e);
}
```

27. JavaScript (13)

Using Axios

Axios:- Library to make HTTP requests

```
async function getFacts(){
try{
    let res= await axios.get(url);
    console.log(res);
}catch(e){
    Console.log("ERROR-",e);
}}
```

Code: To Connect with API:

```
let url = "https://catfact.ninja/fact"}
try{
    let res= await. fetch(url);
    let data= await. res.json();
    Console. log(data.fact);
    let res2 = awaitfetch(url);
    let data2= await res2.json();
    Console.log(data2.fact);
}catch(e){
    Consolc.log("error-",e);
}
Consolc. log("bye");
```

→ Axio.github.com

CDN link paste in Html code: for linking.

Dog Picture API

```
let btn = document.querySelector("button");
let url2 = "https://dog.ceo/api/breeds/image/random"

btn.addEventListner("click", async ()=>{
        // Console.log(link);
});      let img = document.querySelector("#request");
         img.setAttribute("src", link);

async function getImage(){
    try{
        let req = awaiting axios.get(url2);
            Console.log(req);
    }
    catch(e){
    Console.log("error", e);
    return "No fact find";
    }
```

Axiosy

Sending Header:

```
Const. Config = { headers: { Accept: "application/ton" } };
let res = awaiting axiosy.get (url2, Config);
Consol.log (res), data);
```

Axiosy

Updating Query strings:

→ Hapicatch.io

Get: https://universities.hipolabs.com/search?name=nepal

Code:

```
let url = "http://universities.hipolabs.com/
let btn = document.querySelector("button") search?name=";
btn.addEventListner ("click" async ()=>{
let Country = " nepal ";
async function getCollegeData ()
try{
```

```
let res = await axios.get(url+country);
    console.log(res);
}   catch(e){
        console.log("error:", e);
    }
}
```

Convert: get CollegeID;

Property:

Html:
```
<Input placeholder="enter country">
<button>Search</button>
<p id="result"></p>
```

The Terminal 20 J.

A text input and output environment.
1) help in package eg (node J.S.)
2) increase speed.

Difference Term

Command line: Any interface that is used by entering textual command (gen. window centric)

Terminal: This is a type of command line (gen. Mac centric)

Console: A command-line interface used to work with your computer.

Shell: A program running on terminal.

Bash: - A popular shell on MacOS Linux.

Z-shell: Another shell default

Command

ls :- list file (show my file)

Delta
- CS
  - Ch.0
  - Ch.2
- JS

pwd print working directory (where am I ?) Hierarchy

clear clear screen

pwd (Home directory)

herero@Rohit Kumar yadav ~
$

Navigation

Provide Outside Directories.

① cd - Change directory

cd .. → Back button (one step back)

Path

Absolute & Relative

cd Desktop/Delta (relative path) — current location

cd /user/shreeth khetan/Desktop (Absolute path)

/ = root directory

~ = home directory

Making directory

Mkdir :- make directory

cd desktop/Delta

mkdir New dir

ls

mkdir New dir 2

mkdir. Hello

cd . / cd ..

file से अलग folder.

mkdir / user / shraddhashapra / Desktop / delhi / Nerder2 / Hello2 / abc

Flags

Flags are Character that we pass with the command to modify their behavior.

Manual Command:

Man ls:- give info about ls command

man mkdir:- give info about mkdir command

with flags

ls - l

ls - a

ls - la

manual → ls, clear, pwd, cd, mkdir

For Quit: Press q button.

Touch Command

touch index.html

Delete File/or Folder

rm → remove file

rmdir → remove empty folder

rm - rf → remove any folders

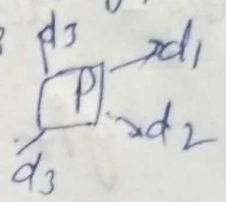
29. Git & github

Git

Free & open source Version Control System → tool that help to track changes in Code

web project: Frontend ① Html ② CSS ③ J.S
Backend

step1 → step2 → step3



① Track history
② help to Collaborate

github: website where we host repositories online.

Using Github

- Create a new Account
- Create repository
- **README.md** → Project, markdown
- **Commit**

changes: Commit → जो change हुआ है
add → Commit (छाप दिया)

Using git

- Command line (most popular)
- IDE / Code Editor like VSCode
- Graphical user interface (like gitkraken)
- **gitbase**

git Configuring

```
git config --global user.name "my name"
git config --global user.email "Someone@email.com"
```

Basic Command

- Clone – Cloning a repository on your local machine
- status – display the state of the code

(1) git clone <-some dirty->

(2) git status

ls Comand

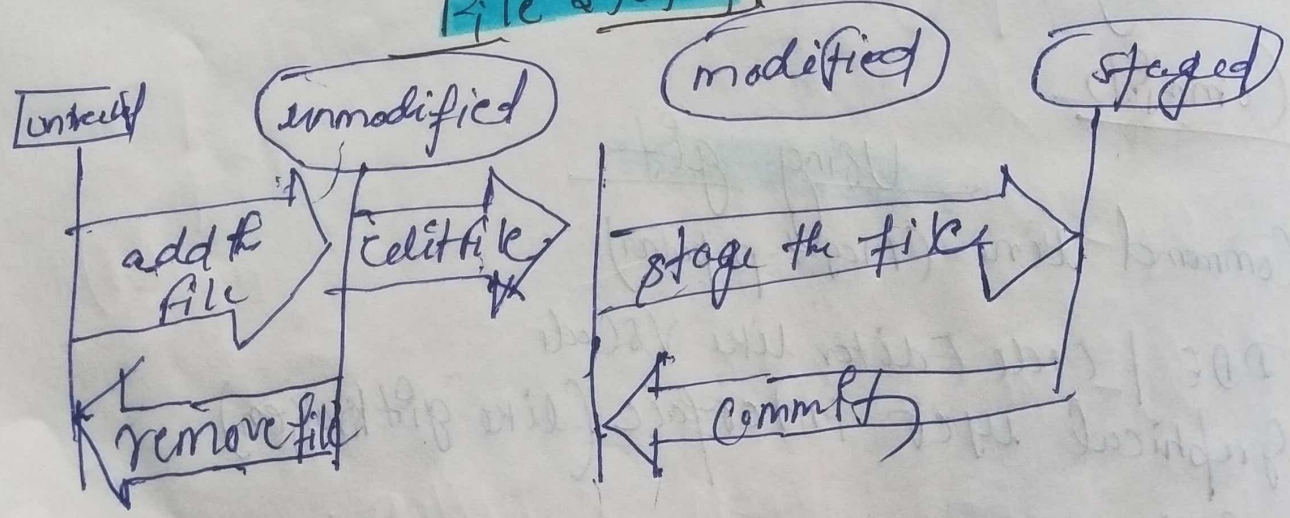
cd → Comand → Delta 1 मे चले जाएंगे

ls -a Hidden folder

Status Command

- display the states of Code.

File Status



Basic Command

add – adds new or changed files in your working directory to the git stagging area.

git add <-filename->

Commit It is the record of change

git commit - m "some message"

push - upload local repo content to remote repo.

git push origin name.

add → Commit

modified staged

final change

Working directory → staging area (add) → final state (Commit)

Add all file :- git add.

Commit :- git commit - m "Add new file"

github पर नई change करने के लिए

push: upload local repo content to remote repo.

git push origin main

{local file में project बनाते हैं} Init - used to create a new git repo.

Cmd - git init

ls - la → Folder: MKdir Project2