VISION

DATE
PAGE 1

Web Dev

client <-- request --> Browser <-- request --> Server
client <-- feedback -- Browser <-- feedback -- Server

frontend
HTML
CSS
JS

tag :- Basic structure.
Attributes :- Basic Attributes to tags.

Basic structure

```
<!DOCTYPE html>
<html>
<head>
<meta ...>
<title> </title>
<Link ... >
</head>
<body> ---> Content
</body>
```

(left-4 se)

txt styling in html.

<tagname style="property : value;" >

properties

* background-color
* color
* font-family
* font-size
* text-align.

Favicon.
inside the <head> tag.

<Link rel="icon" type="image/x-icon" href="/images/favicon.ico" >

Linking html & CSS
using link tag.

<link rel="stylesheet" href="style.css">

linking html & JS

<script src="script.js"></script>

head tag <head></head>

includes meta data, title and ~~link~~ stylesheet

<title> title </title>

<meta> includes all meta data
like charset, viewport, content width, description, etc.

<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">

<body> tag.

includes all contents of the body.

meta description.

<meta name = "description" content = " This is the discription" >

<body> tag

includes all contents of the body.

heading ~~tano~~ tags.

<tagname style = "property: value" >

<h1> </h1>

We have h1 to h6 i.e. 6 types of headings.

<P> tag

paragraph tag.

<P> </P>

Attributes

① style = "background-color : color"

formatting tags in HTML (Pair type)

<b> → Bold	<small> smaller txt
→ Imp txt	<del> Deleted txt
<i> Italic	<ins> Inserted txt
emphasized	<sub> Subscript
<mark> Marked	<sup> Superscript

Anchor tag

Link.

→ Mandatory Attribute.

used to create links.

Attributes

① target = "_blank"

for opening the link in new tab.

<image> tag

used for image in web page.

<image src="" alt="" >
↓ ↳ name of image.
Source (alt txt.)
location.

Attributes
width = "230"
height = "230"
↳ px

tables ~~tag~~. in html

<table>
<tr> → table row
<th> Name </th> → table heading
<th> Phone </th>
</tr>
<tr>
<td> Atharva </td> → table data
<td> 940 xxx xxxx </td>
</tr>
</table>

for merging col or row
colspan = " " or rowspan = " " Attributes
are used in <td> tag.

Caption tag in table tag is used to give caption to the table.

→ <caption> Table name </caption>

More tags can be used to group the table:-

<thead> </thead> → heading or heading row
<tbody> </tbody> → body
<tfoot> </tfoot> → footer.

Lists in html.

types :-

① Unordered lists <ul>
② ordered lists <ol>
③ defination lists. <dl> </dl>

Unordered lists.

type = "*" or "square" or etc.

```
<ul>
  <li> a </li>
  <li> b </li>
  <li> c </li>
</ul>
```

ordered lists.

type = "1" or "I" or "a" or etc.

```
<ol>
  <li> a </li>
  <li> b </li>
  <li> c </li>
<ol>
```

defination lists.

```
<dl>
 <dt> D1 </dt>
 <dd> Content 1 </dd>
 <dt> D2 </dt>
 <dd> Content 2 </dd>
</dl>
```

let-b se.

SEO (web.dev)

Search engin optimization.

Developing a website in such a way, that search engin gives it more preferances over other similar content.

Core Web Vitals.

*1) CLI (cumulative layout shift)
→ CLS is measure of shifting of the layout of the web page while loading.
→ should be minimised.

*2) LCP (Largest Contentful Paint)
time required to load the largest element to load.

good LCP score → 2.5 Sec
needs Improvement → 2.5 to 4 Sec.
Poor > 4 Sec.

*3) FID (First Input delay)
* time taken by website to process the input (i.e text box or a button, etc) given by user is FID
* should be less than 100 ms.

** It is a good practice to give height & width in html to minimize CLT.

Lighthouse Report

this Report gives the information about optimizations, core web vitals & Scope of Improvements

This can be generated by
Right click > Inspect > Lighthouse > Analyze page load

Meta txt & Meta description.

* title should depict the contents of the page

* Meta description.
<meta name = "description" content = " This is the desc ">

Form.

<form> </form>
Attributes :-
action = " " (Get or Post)

<input> tage. * pattern = "+91xxx" use of patterns etc.
* type = " " define the type of Input.
* id = " " use same as ~~label~~ that of label (unique) to make label responsiv
* name = " " use to group or to define the type of the input
* value = " " use to define the value.
* placeholder = " " use to give txt at the place of the txt will be entered by the user.
* required for, if the input is mandatory.
* autofocus.

<label> tag

Attibutes

for = " " here the id of the related input should be entered so that label is also responsive

<textarea> tag.

for resizeable text box with rows & cols attributes

```
<textarea name="connect" rows="4" cols="34">
  Enter your comments.
</textarea>
```

select tage.

```
<select name="fruits">
    <option value="apples">Apple</option>
    <option value="Banana">Banana</option>
    <option value="cherry">cherry</option>
</select>
```

Inline or Block elements.

Block

① elements which takes full width
ex div

② Inline.
elements which takes only width which is required by them.
ex span.

ID & class

ID & class are attributes given to the elements or tags.

ID -> Unique Identification
- * Any element can't have multiple IDs
- * Multiple elements can't have same ID

class -> groupping of elements for same properties.
- * Any element can have multiple class.
- * Multiple elements can have same class.

```
<lable id="firstlable" class="red bg-color" >
```

unique ID

multiple classes.

in css

```
class :-
. red {
}
. bg-color {
}
id(s)
# firstlable {
}
```

<Video> tag

<Video src="video.mp4" > </Video>

Attributes

#1) src:- specifies the path to the videos
#2) Controls:- Adds video controlees.
#3) autoplay:- Automatically starts playing the video when the page loads.
#4) loop:- Repeats the video once it ends
#5) muted:- Mutes the video by default
#6) posted:- Specifies an image to be displayed as thumbnail of the video.
#7) width & height.

<audio> tag.

All attributes same as Video.

Preload attribute

used to preload the video.

1) none:- This is the default value. It indicates that the browser should not preload the audio file.
2) metadeta:- This value tells the browser to preload only the metadata of the audio file, such as its duration & basic info about it. This is useful when you want to display duration of the audio without fully loading it)
3) auto:- This value tells the browser to preload the entire audio.

SVG (for logos & Icons)

example.

```
<svg height="100" width="100">
  <circle cx="50" cy="50" r="40" stroke="black"
  r strokewidth="3" fill="red" />
</svg>
```

or.

```
<img src="image.svg" alt="sample SVG">
```

or

with css

```
.background{
    background-image: url('image.svg');
}
```

<iframe> tag

used to bring or display any other webpage or youtube video in your webpage.

Attributes.

1. src :- location.
2. height & width :- resize.
3. frameborder :- Indicates weather to display border
4. ~~no~~name :- for targeting the iframe in JS.

Semantic tags.

Used to enhance SEO, improve accessibility, and make your code easier to read & maintain.

1) <header>: Used to represent the top section of a web page.
2) <nav>: Signifies a navigation menu on a web page.
3) <article>: Indicates a self-contained piece of content, such as a blog post or an article.
4) <section>: Represents a thematic grouping of content on a webpage.
5) <aside>: Typically used for sidebars or content that is tangentially related to the main content.
6) <footer>: Represents the footer of a webpage, usually containing copyright information & contact details.
7) <figure> and <figcaption>: used for embedding images, diagrams, or charts, along with a caption.
8) <main>: Signifies the main content area of a web page.
9) <time>: Used to represent time-related information like dates & times.

Q) Exercise → 1
you are given 6 audio & 6 video files. Design a webside using HTML only which shows these 12 files.

Misc.

<code> tag.
it is a sematic HTML tag used to display a code snippet. (it is used with <Pre>)
Syntext
<Pre><code> Multiline code snippet Here.
</code></Pre>

<canvas>
Container for graphics, which can be rendered via scripting.

HTML Entities
< for <
> for >
& for &
  for non-breaking space
© for ©

Quotation tag
<blockquote cite="source-url"> quotation </blockquote>
<q cite="source-url"> quotation </q>

DATE
PAGE 14

Character Sets

Why & do define charset ?

- Accurate Rendering :- To ensure that browser correctly display your txt.
- Multi-language Support :- To display txt in various ~~to~~ language and alphabets.
- Data Integrity :- To make sure the data sent and received remains consistent.

Common Character Sets.

- UTF-8 — Univalsal 8-bit char set
- ISO-8859-1 — Western Alphabe
- ASCII — American standard code for Info Interchange.

<meta charset = "UTF-8" >

Introduction to CSS

Cascading Style Sheet.

Basic Syntex

selector {
    color
}

div {
    color: red;
}

selector — Property — value

declaration

Multiple selectors can also be written.

```
div, span {
    color: red;
    background-color: yellow;
}
```

Linking css with html.

Inline CSS
* using style attribute in individual tag
* this is not considered as a good practice.

Internal CSS
* Using <style> tag inside <head> tag and making all css declaration in it
* better than Inline CSS, but not recommended

External css
* Making a .css file, then including it in html using <link> tag inside head tag.

```
<link rel="stylesheet" href="location of .css file">
```

* best Practice, Most used.

Selectors in CSS

1) Element selector (for a tag)

```
div {
    background-color: red;
}
```

2) Class selector (for a class)

```
.class {
    Property: value;
}
```

3) Id selector (for a particular element with an id)

```
#id {
    Property: value;
}
```

4) Child selector (for a tags inside a tag) Directly only.

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

* Here this is declared for all <P> tags inside all tags.
* When <P> is directly inside only then this will work.

Ex:-

 <article> <P> ABC </P> </article> ✗

 <P> ABC </P> ✓

Attribute selector.

```
[x=a]{
   Property: value;
}
```

<tag x="a">
↳ user defined attribute

DATE
PAGE 17

5) Descendant selector (for tags inside tags)

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

* Here, this is ~~doesn't~~ declared for all tags inside all tags.
* Unlike child selector, this will work even when is not directly inside .

 <article> abc </article> ✓
  abc   ✓

6) Universal selector (for all)

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

7) Psudo selector (Used for styling ~~an~~ elements like <a> ~~colors~~ @ any action like link visited, hovered, etc)

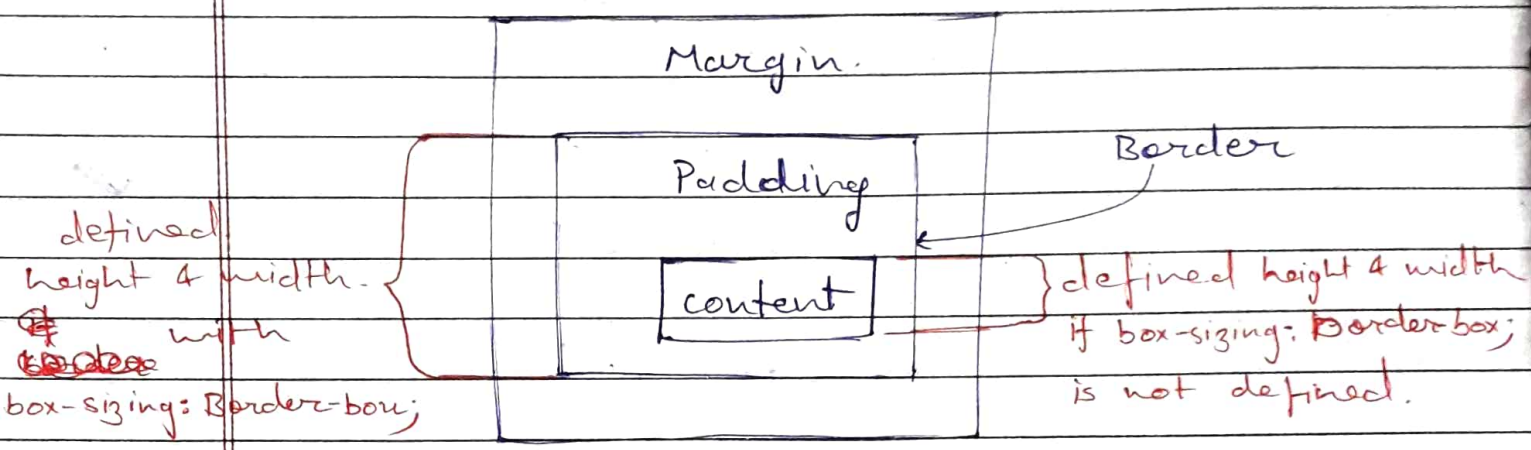
```
a:hover{
   property: value;
}
a: visited{
   property: value;
}
~~a: link~~
a:link{
   property: value;
}
```

```
div p:first-child{
} → 1st p in div

div p:nth-child{
} → nth p in div.
```

used when link is not visited.

```
a:active {                    // when the link is active.
    property: value;
}
```

let-18
CSS BOX Model



The CSS Box model defines how elements are rendered and how their dimentions are calculated. It describes the structure of an element as a rectangular box that has content, padding, a border, and a margin.

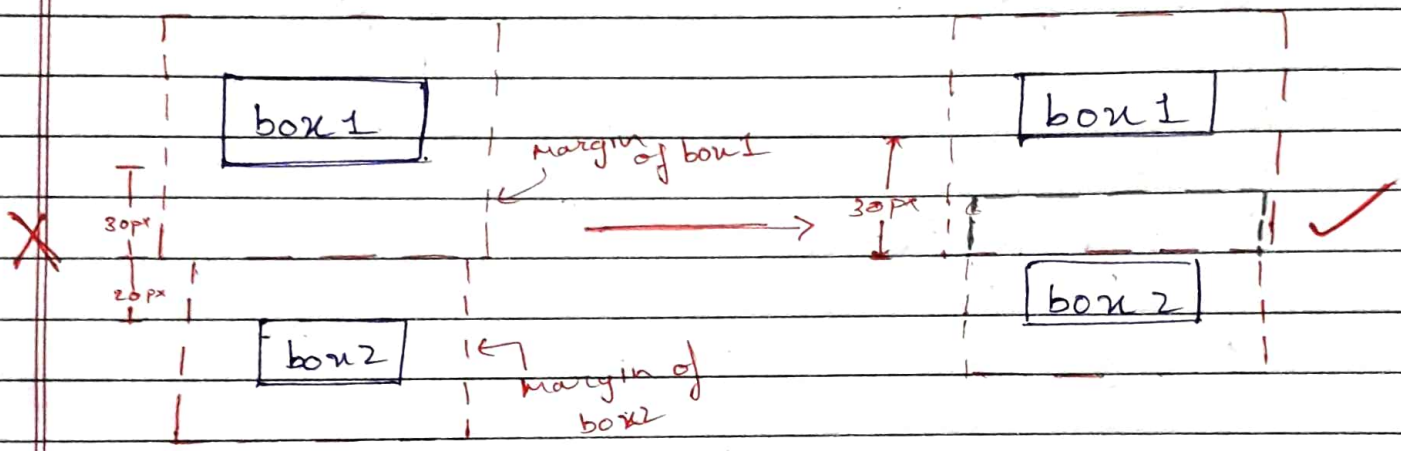
Everything in html is described as a rectangular box in CSS which has content (innermost main content), padding (space b/w content and border), border, margin (space outside margin).

box-sizing: border-box;

If this is used, then defined height or width will include border & Padding and without this defined height or width only include the dimentions of the ~~conten~~ content (~~with~~ padding & border exclusive).

Margin Collapse.

If Margins of two elements are overlapped, they will collapse. ~~Eg~~ i.e., only the grater margin will exist.



Fonts.

* font-family: (for selecting font)
* font-style: italic;
* font-weight: 500;
* text-decoration: underline; (none, if you want to remove by default txt decorations)
* font-size: ; (for line)
* line-height: ; (for spacing)
* letter-spacing: ; letter Spacing used for

also can be imported through google fonts.

to capitalize 1st letter :-

* text-transform: capitalize;
  or
  lowercase; (to lowercase)
  or
  upper case; (to uppercase)

* text-decoration-color; ;
  for color of underline.

* text-overflow: ; } to manage the
  overflow: ; overflow

Colors.

Ways of Represent

* Keywords
* hex code
* ~~rgb~~ RGB, or RGBa
* HSL or HSLa

Exercise 2

Write html & css code to style a paragraph inside a div which contains 5 other paragraphs. The first paragraph must have background color yellow and text color red. ~~The HTML is written below for~~ the other paragraphs must have background color blue & text color white.

Specificity

to solve any type of Conflict, css uses cascade algorithm

1) Position & order of appearanch.
(only if selector type is same)
the order in which your css rules appear.

2) Specificity Preference

Inline > ID Selector > class or Attribute > Element > Universal

3) origin
the order in which css appears and where it comes from, weather that is a browser style, css from a browser extension, or your ~~$~~ authored css.

4) Important
when used !important, it has ~~most~~ highest specificity score.

Specificity Score Calculation.

- Universal selector = 0
- Element selectors & pseudo-elements = 1
- class selector, attributes selector & pseudo-class = 10
- Inline styles : 1000.

if complex selector used, Specificity should be added ex is @ back.

Example :-

specificity = 1

h1 . box { property: value; }

class (specificity = 10)

total specificity = 1 + 10 = 11

Sizing Units.

Let -> zz

(1) PX → Pxel ie a fixed unit.

(2) VW → (for width) } Responsiv units.
VH → (for height)

(3) em (non-Responsiv)
'x' times of.

x em => x times of inherited value.

(4) rem (non-Responsiv)
x rem => x times of inherited value from root element

(5) Vmin Vmax . (Responsiv)

min- or max- height or -width (responsive)

* used to define maximum or minimum dimentions

: %

* used to set a % of parent width or height
Note :> dimentions of the parent is defined.

Display Properties.

display : "Property" ;

#

#1) Inline (display: inline;)
* to make any element inline
* Padding top is not respected.
* Margin top is not respected.
* width property is not respected.

#2) inline-block. (display: inline-block;)
* to make any element inline.
* top-(Margin, Padding) & width is respected, unlike display: inline.

#3) none (display: none;)
* to hide any element with all its space aquired

Visiblity: hidden;
to hide any element, but unlike (display: none;) it hides only the element while the space aquired by it remains intact.

Shadow

Syntax

box-shadow: h-offset v-offset blur spread color inset;

- h-offset, v-offset: Shadow's horizontal & vertical positions
- blur: larger this value more blur the shadow
- spread: Expands or shrinks the shadow size
- color: Color of the shadow
- inset: inset = Makes the shadow inner

text-shadow: h-offset v-offset blur color;

outline & Border

1) Position:- outlines don't take spaces, they're drawn around the element, outside of any border
2) offset: Using the outline-offset property, you can set the space b/w an outline & the edge or border of an element.
3) Width: Borders can have varying widths on different sides, outlines have a uniform width.
4) Rounded corners: Border ✓ (with border-radius)
   outline ✗

List styleing

~~List style~~

* list-style : → none (for removing the marker)
; to style the Marker of the List.
any language ~~" * " (special symbol) or a emoji~~
(by default marker isn't included in styling like background-color, border etc)

* List-style-position : inside;
used to include the marker in styling like background-color, border, etc.

* List-style-image : url(" ");
used to set any image as marker.

* list-style-type : " any emoji or special symbol ".
used to set any emoji or special symbol as marker.

~~Overflow~~ overflow

overflow : scroll;
to give a (mandatoryly) scroll bar at both x & y.

overflow : auto;
to give a scroll bar only where it required.

text-overflow: ellipsis;
to hide overflow data, like this

data of...

overflow-x: hidden;
overflow-y: scroll;
} Controlling overflow individually.

overflow: hidden auto;
↳ x ↳ y.

Position Property.
→ to controll the z-index, "z-index = ;" can be used.
by default → static (doesn't Allow top, bottom etc)

position: relative;
* Elements are positioned relative to their normal position in the document.
* you can use top, right, bottom, & left properties to move the element from its original position.

position: absolute;
* Elements are positioned relative to the positioned ancestor (parent), which means we need to have a parent element with a position positioning other than 'static'.
* An absolutely positioned element is removed from the normal flow

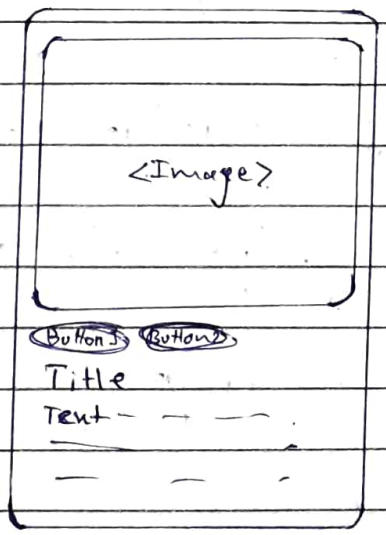
transform, filter or perspective properties can also make an element appear as ~~positioned~~ positioned.

position: fixed;
Elements are positioned relative to viewport (screen) and do not move when the page is scrolled.

position: sticky;
(hybrid of 'relative' & 'fixed')
* It allows an element to become "stuck" to the top or bottom of its container when scrolling, but it ~~beha~~ behaves like relative positioning within the container until it reaches a specified offset.

position: floting;
(More details after)

Exercise 3
Write html & css code to design this card.



CSS variables.

We can define global & local varibles in css which can hold various properties and then we can use the varibles in the place of the properties. So that we can change many properties of many selectors by just changing the varible values.

Ex use case

Multiple themes of a webpage

Syntax:-

```
:root {
  --varible : value;
}
```

} all the global variables are defined here.

Media Queries

it is used to make your website work efficiently across various devices (different sizes of screens) it makes your website adapt different screen & responsive to different screen sizes

Syntax:-

~~@media not/only mediatype and (expressions) {css-code}~~

~~@media not/only mediatype and (expressions) {css-code}~~

```
@media not/only mediatype and (expressions) {
  css-code;
}
```

Example :

@ media only screen and (max-width : 455px) {

css properties, etc.

}

@ media only screen and (orientation : portrait / or / landscape) {

css properties, etc.

}

Exercise 4

design a ~~color~~ multi-color nav bar colored & themed using css variables.

Float & Clear (let → 34) (not so used in modern websites)

Float → Just like how "wrap through" works in MS Word, the float property allows the balance b/w txt & the image shown beside it.

float : right; / or / left / or center or none.

use display: flow-root; with floated elements to resolve any layout issues

→ When we want to remove it,

clear : right; / or left or both or none

DATE
PAGE

More of CSS Selectors

1st child selector:

```
.box:first-child {
    property: value;
}
```

follow up.

```
.box + p {
    property: value;
}
```

to select the next of a element

```
<div class="box"></div>
<p></p>
```

1st line selector:

```
.box:first-line {
    property: value;
}
```

```
:not(  )
```

↳ any id or class or element

select all the elements which don't match.

Univalsal selector in a element

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

Attribute selector.

```
[attribute="value"] {
    property: value;
}
```

nth child selector.

This can be 1, 2, 3, etc or even, odd. for every even or every odd children

```
.box:nth-child(<nth child>) {
    property: value;
}
```

content can be

before (dynamically inserted b4 any element)..

```
.box :: before {
    content : " Abc ";
}
```

this will place the "Abc" b4 the box

```
.box :: after {
    content : "Abc";
}
```

Same as b4 but content is inserted after the box.

```
.box:: selection {

}
```

for, when user selects any txt.

```
input :: placeholder {
}
```

to style placeholder in <input>

Exercise 5

Q) Design layout (see the video 37 for details)

Flexbox (AKA Flexible box)

display: flex;
~~it makes~~
it makes it easier to layout, align & style items in the container while maintaining the responsiveness of the website.

Properties (container)

① justify-content: value;
★ works with main-axis ★
↓ flex-end, flex-start ~~left, right~~, center, space-around, space-between
★ used to align elements horizontically. ★

② align-items: ~~con~~ value; ★ works with cross-axis ★
(used to align elements vertically)
↓
~~Top, Bottom~~ flex-start, flex-end, center, etc.

③ align-content: value; ★ works with cross-axis ★
↓
flex-end, flex-start, centre
(used to align the content inside the elements present insid the flexbox
Used to align ~~to~~ the elements when multiple ~~item~~ lines (wrap).

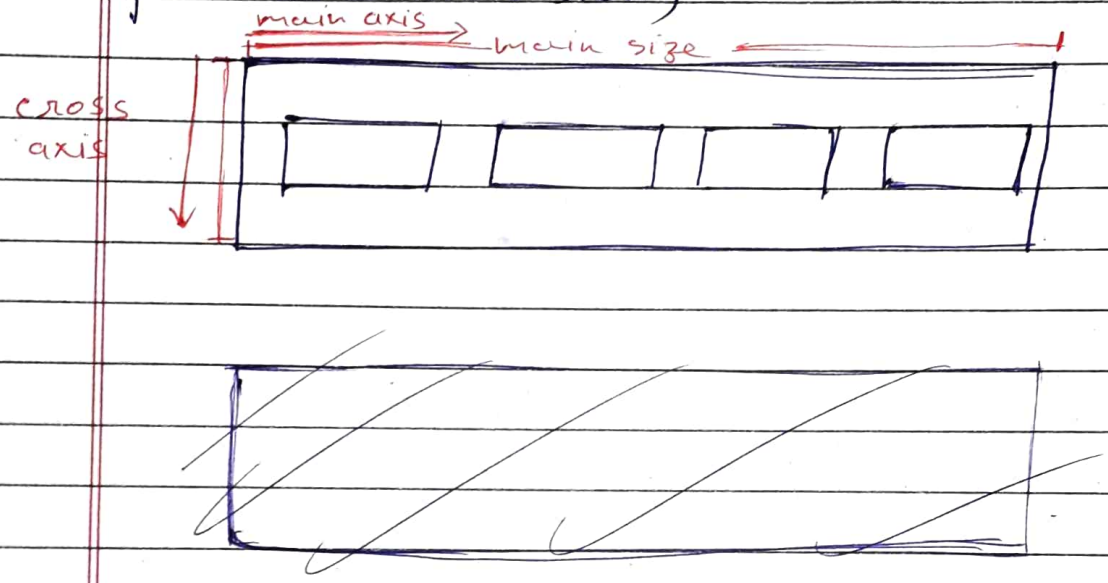
(4) flex-direction : value;

↓

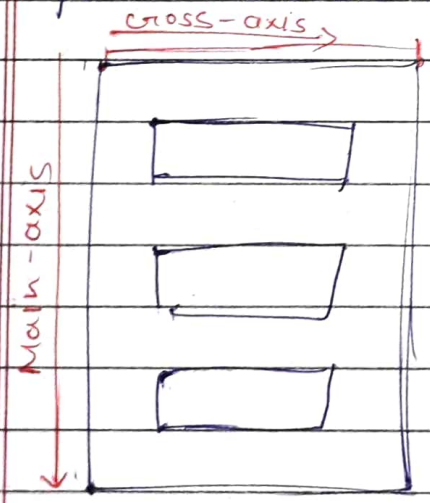
row , row-reverse, column, column-reverse.
~~column-reverse~~.

It defines in which direction the flex elements would be displayed. i.e. row or column, etc.

flex-direction : row;



flex-direction : column;



(5) flex-wrap: wrap;
(used to wrap elements in case of overflow)

flex-wrap: wrap-reverse;
to reverse-wrap.

flex-wrap: no-wrap; (default)

(6) flex-flow: row wrap-reverse;
row → flex direct^{n}
wrap-reverse → wrap.

This property is used to set two properties @ a time.

(7) gap: 30px;

used to make gap b/w all the elements horizontically & vertically both.

column-gap: value;
(for only vertically)

or gap: 10px 20px;
row column

row-gap: value;
(for only horizontically)

It is highly recomanded to use gap instead of margin / padding inside the flexbox or grid.

Flex & item properties.

① order : 100;
used to set order, ~~more the order later~~ of the elements in which they are displayed.

② flex-grow : value; → (1, 2, etc).
(by default → 0)

③ flex-shrink : value; (1, 2, etc).
ability of a flex item to shrink if ~~necessary~~ necessary.

④ align-self : value;
↓
flex-start, flex-end, centre, etc.
(used to align a perticular element in a flex box)

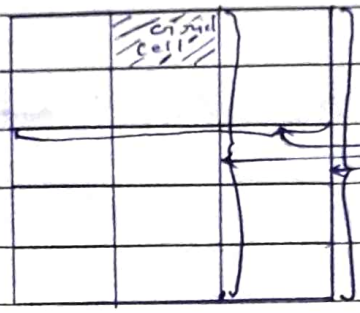
Grid.

Just like flexbox, CSS Grid with the use of rows and ~~columns~~ columns, make it easier to style website elements

display : grid;

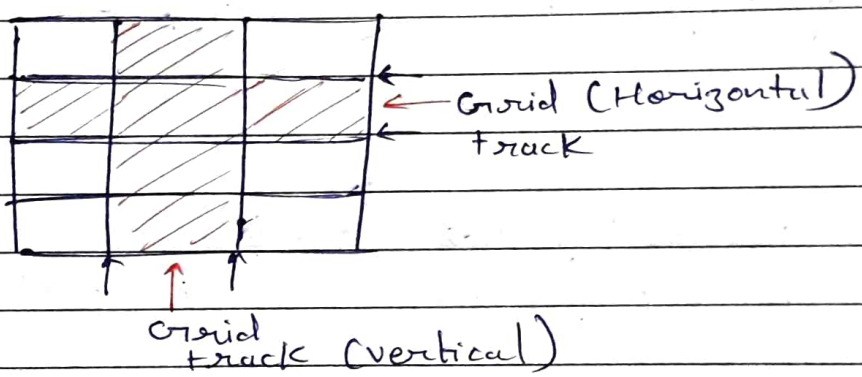
grid-template-columns: 120px 100px auto;
↑
no. of columns, size of columns can be defined.

Grid Line & Grid Cell

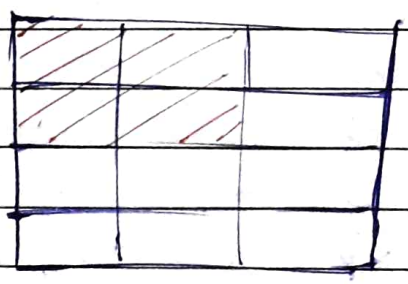


Grid track & Grid Area.

Grid track
space b/w two grid lines.



Grid area.
any area in grid.

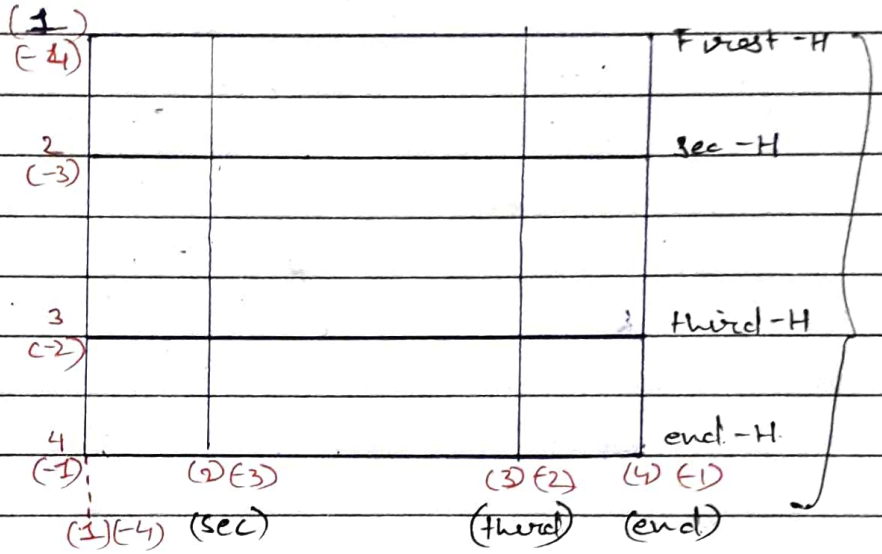


Grid Templete Rows & Columns.

* This is container property.

grid-templete-columns: 100px 200px auto;
grid-templete-rows: 200px 200px 200px auto;

by this property, we can define no. of cols & Rows and their size as well.



col & Row Default Numbering

This type of names (any names) can be given through by

grid-templete-columns: First 100px sec 300px third 100px end;

grid-templete-rows: First-H 100px Sec-H 300px Third-H 100px end-H;

they can be used in positioning an item inside grid.

example → grid-column: sec/third;

Positioning any element in the grid.
* this is item property.

```
grid-row: 1/2;
    or
grid-row-start: 1;
grid-row-end: 2;
```

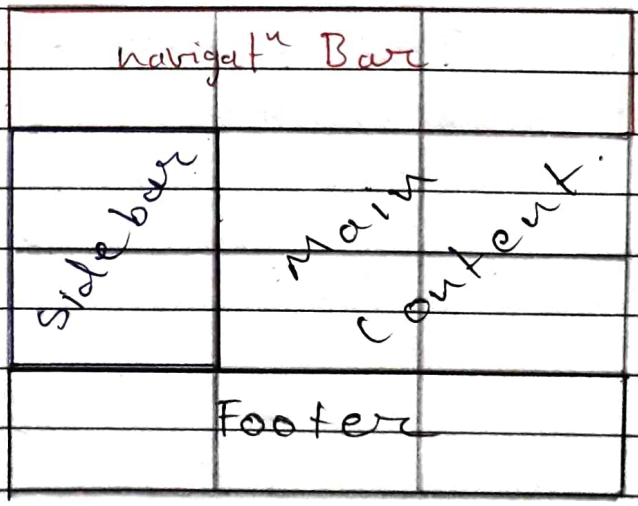
This means the item will occupie space of rows from 1 to 2. in grid.

```
grid-column: 1/3;
    or
grid-column-start: 1;
grid-column-end: 3;
```

This means the item will occupie space of columns from 1 to 3 in grid.

* if the grid cell where u place an item is already occupied by another item, the existing item will shift and the desired item will be placed without overlapping unless both items have the same defined position.

Alternet way to use a Grid.
Grid Template Areas.
defines & name areas so that they can be used efficiently.



example.

Syntax :- (for above example)

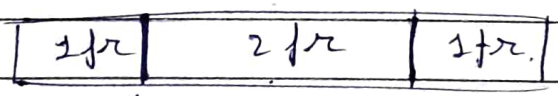
```
.container {
    display: grid;
    grid-template-areas: "nav nav nav"
                         "side main main"
                         "side main main"
                         "foot foot foot";
}

.nav-item {
    grid-area: nav;
}
```

--- and so on.

Some imp misc properties of grid.

① fr unit (fraction unit)

grid-template-columns: 1fr 2fr 1fr;



also can be used with rows.

② gap.

```
# gap: 10px;
  row-gap: 10px;
  column-gap: 10px;
```

} this creates gap b/w items.
it is ~~recom~~ highly recomended ~~not~~ to use this insted of margin, padding inside the grid or flexbox.

alignment :

Same as flex-box

justify-items : center ; → Horizontal } container Properties
align-items : center ; → vertical }
used to align items in their cell.

vertical. align-self : start ; } item properties
Horizontal justify-self : end ; } used to align any particular item.

justify-content : ; } container properties
align-content : ; } used when grid size & container size mismatch, to align the whole grid wrt the container.

→ place-content : ;

using some functions

example syntax :-

functions.

grid-template-columns : repeat(4, minmax(100px, 1fr));

repeat this 4 times'.

Exercise 6 :

Design a navbar using flexbox (like ultraedit website)

Transformers

This property allow you to move, rotate, scale, and skew elements.

transform: rotate( );

used to rotate the element # 90deg or 0.25turn.

1 turn is 360°

transform: rotateX( );
used to rotate the element about x-axis.
similarly:-
: rotateY( );
: rotateZ( );

transform: scale(n);
transform: scale(x, y);
used to resize the element n times the original size

transform: scaleX(n);
used to resize the elements about x-axis, n times the original size
similarly:-
: scaleY( );
; scaleZ( );

multiple transform properties.

transform: scaleX( ) rotateX( );

transform: skew(x, y);
↳ deg.

it skews an element along the x & Y-axis by given angles.
for seperate skew for X & Y.
:skewX();
:skewY();

transform: translate(x, y);

it moves an element from its current position ~~accord~~ (according to the parameters given for the x-axis & Y-axis)
similarly:-
:translateX();
:translateY();

transform-origin: x y;
used to set origin co-ordinates about which other transformations are applicable.

Q) Exercise 7
develop a grid like UltraEdit website.

Transition Property (used to add smooth Animation)

transition : Properties duration function delay;

Example :-

transition: transform 3s ease-in-out 1s;

Animation Property

Syntax :-

@Keyframes variable name { }

@Key Defination

```
@Keyframes any-name{
   from{
      property: value;
   }
   to{
      property: value;
   }
}
```

```
@Keyframes any-name{
0%{}
10%{}
50%{}
:
and so on.
}
```

call.

animation: name duration timing-fxn delay itrat^n count direction fill-mode;

Multiple calls :-

animation: name duration fxn, name2 duration2 fxn2.

Exercise 8

Q) Refer let 48

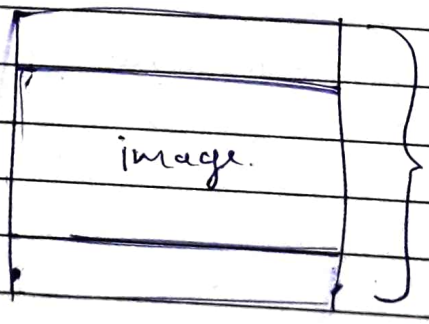
object-fit & object-Position.

* used to fit & position any object in certain dimentions.
* this property applies on dimentions declared in the same selector in which these properties are dectered.

object-fit : cover;
to cover up dimentions defined in the selector, crops certain part of image to fit it.

object-position: top right;
top left;
bottom right;
bottom left;
used to position the part of image to display when object-fit: cover; is used.

object-fit : contain; original
used to display the image in fit with the dimentions without crop or streching the image. Keep space to display image.



background-image

background-image : url(elements);
→ Set image as background of any container

background-position : top right;
to set which part of image to display as background and the position of background image.

background-repeat : no-repeat;
repeat-x;
repeat-y;
to manage the repeatation of the image in case of image is shorter than the container.

background-clip : border-box;
padding-box;
content-box;

Filter Properties.

filter : Blur(10px);
→ to blur any element

filter : brightness(200%) contrast(9.5);
→ to set brightness & contrast of elements.

filter : opacity(1);
(1 full, 0 none)

: invert(1); → to invert colors of images.