DATE: / /
PAGE NO.

CSS (Cascading Style Sheets)

To connect html file to CSS file use tag <link> in the <head> section of the page.
eg> <link rel="stylesheet" type="text/css" href="style.css">
rel → Specifies relationship b/w current document & linked document required
type → specifies media type of linked document

Syntax in CSS: /* selector { property: value; } */
selector → for element
Comment

eg> h2 {
color: red; }
This will change the color of text in h2 element to red.
CSS is cascading so last statement is the active one.
eg> p { color: pink; }
p { color: green; }
First it will set color to pink then to green as it is written in order.
CSS can be also be implement without linking to separate CSS file. There are two ways.
1- Using inline style attribute
eg> <h2 style="color: blue"> About </h2>

Note → Use unsplash.com for free HQ images

DATE: / /
PAGE NO.

2- Using Head element <style>
eg> <style>
h2 { color: brown; }

CSS Properties

h2 { color: red;
text-align: center;
border: 5px dashed purple;
cursor: pointer
size of border, type of border, color of border
when the cursor hover it becomes the hand
HOME
the hand when on h2 element

body { background-image: url(backimg.jpg),
background-size: cover;
/* If the image size is smaller then the over will stretch to whole page */

li { list-style: none; }
This will remove bullets or numbers
display: inline-block; → Display lists in single line

Use paletton.com to choose color of different shades. Using this site after getting desirable shade we get its hexadecimal id. Paste this id followed by # in place of value of color.
eg> h2 { color: #5C1D96;
shade of purple; }

more properties: (for li specially)
list-style: none (remove bullets or numbers)
display: inline-block (displays list in single line)

DATE: __/__/__
PAGE NO. _____

Instead of using hex id we can use individual intensities of red green & blue.
eg → color: rgba(255, 176, 170, 0.5)
opacity (0 to 1)

CSS Selectors (some imp selectors)

1- class: It helps in graping of data
In html section: eg → ipsum
 latin
In CSS section:
.webtext { border: 5px dashed green; }
[All the selectors having class = 'webtext' will have its element these properties].
for multiple classes
eg → ipsum
In CSS section
.active { color: red; }
Now this p element will acquire both properties of class webtext & active.
Note: Due to cascading if in the last section active if border properties are not defined then these properties will be active not of webtext.

2- #id: like class id can also be used to grop them. element is used here. Unlike class id can be used only one time

In HTML section
eg →
 ipsum
 latin

In CSS section
#div1 {
background: rgba(0, 0, 255, 0.2); }
/* This is for blue for lesser opacity so that it don't effect the visuals of text inside.

Though cascading is there (last property is active) but there is a priority too.
eg →
In html

 latin

In css
p { color: red }
.text { color: blue }
#div1 { color: green }
It seems that final color is green but it is red as priority is p > text > div or

3- * → eg → * { text-align: right }
It defines the property of every element. It has lowest priority. So

DATE:__/__/__
PAGE NO.____

for h1 & p is aligned to center so they won't change. but list elements will.

4- element> Already seen eg> h2{ color: red};

5- element, element> To specify 2 elements to have common property
eg> h1, p{ color: green};

6- element1 element2> Two elements written in a space when defined will only be ~~apply~~ apply property if ~~both elements are used~~ element 1 is parent of element 2.
eg> In CSS
h2 p{ ~~color~~ text-align: center};
<h2> Home </h2>
<h2> homegrp </h2>
Output > Home ← not ~~affected~~ aligned to center
homegrp → aligned to center

7- element1 > element2> ~~Two elements~~ Two element written like this, property will only be applied if element 1 is direct parent of element 2
eg> ~~In html~~ In CSS: h2>p{ text-align: center;}
<h2>
 homegroup
</h2>
Output
homegroup
aligned to center

<h2>

 homegroup

</h2>
Output
homegroup > not aligned to center

• element1.element2> When both classes written in class attributes
no space b/w them

DATE:__/__/__
PAGE NO.____

8- element1 element2> Two elements with ' ' sign b/w, property will only be applied if element 2 is written just below element 1. property will applied to element 2 only.
~~Like~~ eg> <h2> ~~<h2>~~ Home </h2>
 homegroup
Output
Home
homegroup
After h2 if any other element is there then property won't apply.

9- :hover> eg> h2:hover{ color: green};
color will only change to green when the mouse pointer will hover it (touch it)

10- :last-child: It is used with class. Property is applied to the last member of class
eg> ~~lorem~~ ipsum
 latin
CSS: .webtext: last-child { text-align: center; }
Output> ipsum
latin

11- :first-child: like :last-child property is applied to 1st member of class.

12-

DATE:__/__/__
PAGE NO.____

12- !important (not recommended) → This is used along with the property.

It breaks the cascading rules and it locks the property. So if after this is used if anyone wants to change the property he/she can't it.

eg→ Hover

In CSS: ~~p{ color: blue }~~ p { text-align: center !important; }
p { text-align: left }

Output → Hover

Specificity → Cascading doesn't win everytime due to specifity. Higher the specifity of selector the more chances of it gets applied.

eg→ ~~p{ color: red }~~ #div1 p { color: red }
.webtext { color: green }
p { color: blue }

HTML

1- Latin Output → ~~blue~~ blue colour
2- <p ~~webtext~~ class="webtext"> Latin Output → green
3-   latin
Output → ired colour

~~Specificity~~ Specificity

Inline style > id > class, attribute & pseudo class > element & pseudo element

If the specifity is same then cascading is deciding factor.

 Hello

p.class#id > p#id > p.class > p

eg→ HTML section

 <u> Latin </u>

CSS

.div1 u {
color: red; }
.text u {
color: green; }

Output → green

.text u {
color: green; }
.div1 u {
color: red; }

Output → red

Inheritance: Properties can be inherited from parent element/class/id, etc. For this keywords like → inherit, initial & unset is used.

eg→

~~Inherit~~
 ~~class="my-class-1"~~
 Default link
 Inherit the link
 Reset the link
 Unset the link

class="my-class-1"
class="my-class-2"
class="my-class-3"

CSS → .class1 a { color: green; }
.my-class-1 a { color: ~~blue~~ inherit; }
.my-class-2 a { color: initial; }
.my-class-3 a { color: unset; }

DATE:__/__/__
PAGE NO.____

Default link → default blue color

Inherit link → green inherited from parent color element (which has class = "class1" having color: green)

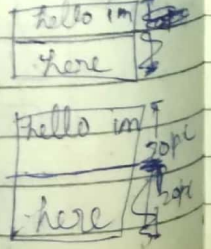
Reset the link → Default black color (This is default colour of browser and disabling all css properties)

Unset the link → it will inherit the properties which are naturally inheritable like → color, etc & reset the property like border. If in the parent tag border is there it won't inherit & reset to default border property which is none.

Text & Font

Property →

1- text-decoration: Values: underline, line-through(cross)
2- text-transform: uppercase
3- line-height: Space allotted to text (in pixels)
Values → 20px, 200px. eg →
It is the height of the line (not for paragraph)
eg → If it is multiline then →



DATE:__/__/__
PAGE NO.____

4- font-style: Values: - italic
5- font-weight: bold
6- font-size; 150%, 80%, large, etc
7- font-family: font types.
eg → font-family: "Times New Roman", Georgia;
If Times New Roman is supported by browser then it is displayed else Georgia if available is displayed.

If we want that all fonts are available whether it is not available in the system or not then we need to include a font file in ~~<th> <link> element = "or"~~ OR using google fonts.

- Go to google fonts & select the desired font.
2. Then there are different styles like regular, bold, lighter, etc. select regular as it's original & embed the font, by copying the HTML code [<link href = " ... ">] and CSS rules [font-family: " ... "] into your code. Done

This will make the website slower as it needs to visit the google font site first to display the font.

DATE: / /
PAGE NO.

Images in CSS

In HTML

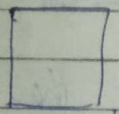
```
<h2> Home </h2>
<img src='img.jpg' width="50px" height="50px">
</img>
<p> This is nice </p>
```

To align images (left center or right) use text-align.

Instead of aligning if we want to float to left or right,

use float: left or float: right in CSS section in img selector

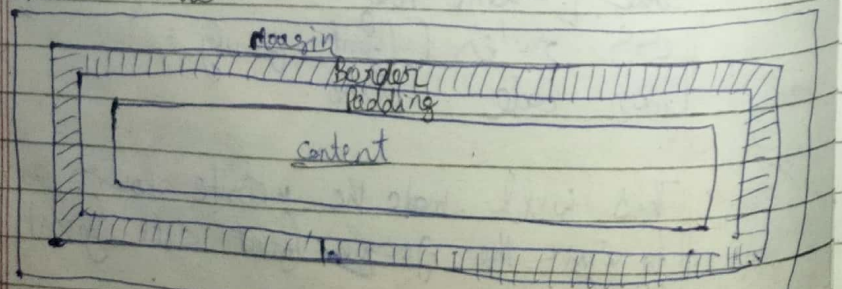
eg> img { float: left }



This is nice

↳ Here instead of displaying text bottom of the image, it is displayed towards right as image is floating towards left.

Box Model



DATE: / /
PAGE NO.

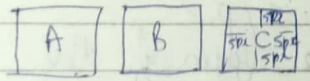
Using div element box model can be implemented. For defining box model class or id is used. Class is recommended as it can be used multiple time.

eg> A
 B
 C

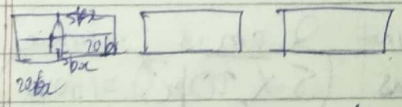
In CSS section

```
.boxmodel
{ border: 5px solid red;
  display: inline-block; (displaying in same line)
  padding: 5px; (from top right bottom & left 5px space will be given)
}
```

eg>



For different padding from different directions use padding: 5px 20px 5px 20px; (in clockwise direction)



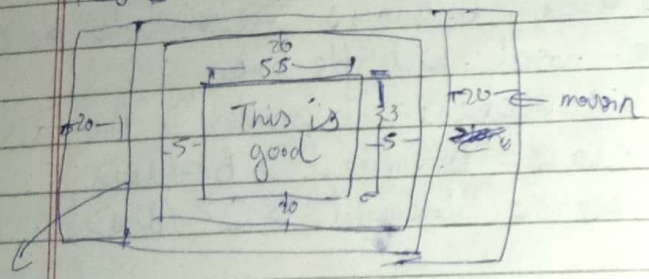
or use padding-left, padding-bottom, padding-top, padding-right

margin: 0px 20px; or (for same property margin: 10px;)

margin: 0px 20px 0px 20px

section

for content height & width use
height: 33px; width=55px; (in CSS section)



border

px vs em vs srem

In html section
  lorem ipsum is a nice style
span is inline form of div

In CSS p { font-size: 20px; }
span { font-size: 5em; }

px is absolute unit & em is relative unit
Here 5em means (5 x 20px) = 100px as in p the containing element (parent element) it is 20px

srem is also used when size is used with relative to root element ie html element. Here instead of 5em we right 5rem then

5rem will not change respect to font-size value in p element as it is only related to html element.

More about CSS

1- To remove underline from hyperlinks use
a { text-decoration: none; }
2- To capitalize a word (so that it starts from capital letter) use
p { text-transform: capitalize }
3- Negative values are not allowed in the padding property (length & percentages are allowed)
4- li { list-style : square outside none; }
or
li { list-style-type: square;
list-style-position: outside;
list-style-image: none;
Also instead of inbuilt user defined type is also allowed
li { list-style-type: "→";
5- position elements help to manipulate location of element
eg> In HTML
<h1> The position Property </h1>
<h2> Heading of </h2>
 Nice & good

In CSS

h2 { position: static
left: 100px;
top: 150px;
}

In static (which is by default) the element remains in flow of document & left, right top & bottom properties don't work

In relative the element in flow of document but left right top & bottom properties works.

In absolute element is removed from flow of document & left, right top & bottom properties works

In fixed element it works similar to absolute but here it is not relative to document (page) it is relative to the window (means scrolling won't work). After it is scrolled its position won't change

In sticky, the element is treated as relative until scroll location reaches a threshold & then after that point it works as fixed where it is told to stick.

Advanced CSS

Critical Render Path: A path that website take to render (display) the webpage to the user.
First a html is loaded then the linked & inline CSS then other things.
For eg: Google fonts API. If website may take more time to load until then font won't appear.
If the file (say CSS file) is big in size then it can be compressed by deleting white spaces (extra spaces) using the web tool 'CSS Minify'

Flexbox

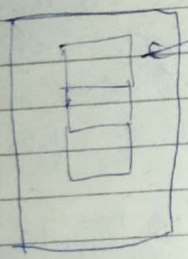
Also known as CSS Flexible box layout is a CSS 3 web layout model. It is a particular way to specify the layout of HTML pages.
In HTML section

```
<div class="container">
<img src="img1.jpg">
<img src="img2.jpg">
<img src="img3.jpg">
</div>
</body>
</html>
```

DATE:__/__/__
PAGE NO.____

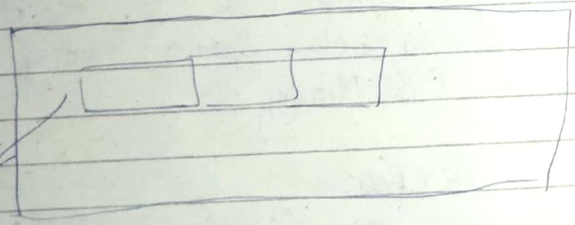
In CSS section

.container {
display: flex → enables flexbox layout
flex-wrap: wrap;
justify-content: center; // to align items horizontally
(location to center)



content)

when extended (window is extended towards right)
then it will unwrap as it is wrapped



content)

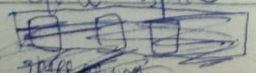
* Play the game flexbox froggy to improve flexbox skills.

img { width: 450px;
height: 300px;
margin: 10px; }

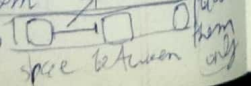
space-between: items are evenly distributed in line: first item on first line, last item on end line.

space-around: items are evenly distributed in line with equal space around them.

eg →



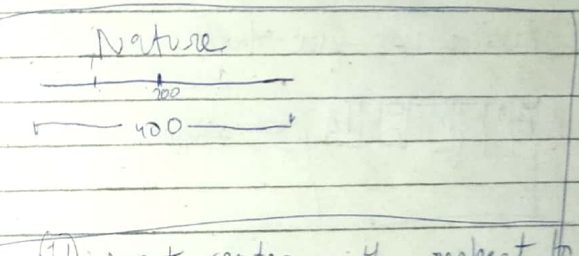
eg →



DATE:__/__/__
PAGE NO.____

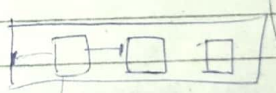
h1 { font-family: fantasy;
border-bottom: 2px solid pink;
width: 400px
text-align: center
}
here the h1 element will be in center with respective to width not respect to the page

eg:



This h1 is not center with respect to the page.

eg → space around



also in justify-content properties like flex-start & flex-end for aligning to left & right respectively

equal spaces around them not just between them

More about flexbox

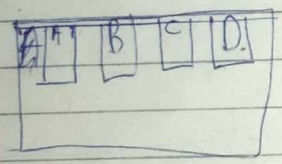
1- align-items: This CSS property align items vertically

DATE: __/__/__
PAGE NO. _____

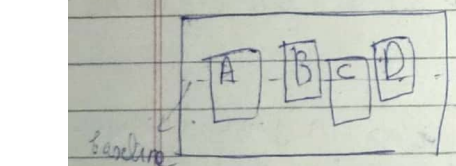
Values:

1- flex-start: Align to top of container
2- flex-end: Align to bottom
3- center: " to vertical center
4- baseline: Align to baseline
5- stretch: Default value. Stretch to the container

Baseline vs flex-start



← align-items: flex-start



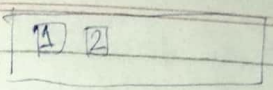
← align-items: baseline

2- flex-direction: This property defines the direction in which items are placed in the container & accepts the following values:

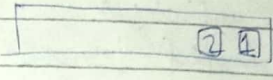
1- row: items are placed same as text direction
2- row-reverse: Placed opp to text dirⁿ
3- column: Placed top to bottom
4- column-reverse: Placed bottom to top

To prevent flex item go out of screen, we can use margin: auto in the item (not container)

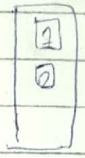
DATE: __/__/__
PAGE NO. _____



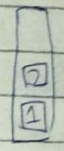
← row



← row-reverse



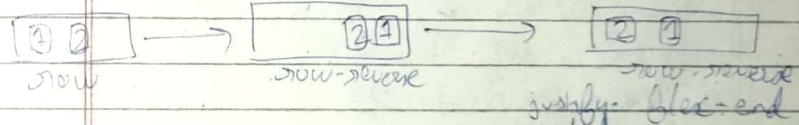
← column



← column-reverse

i) If the row-reverse or column-reverse is used then start & end is also reversed.

eg:



ii) If flex-direction is column then justify-content changes to vertical & align-items to horizontal.

3- Order: When reversing row or column order isn't enough then order property is applied to individual items. Its default value is 0, but it can be set to +ve or -ve integer like {-2, -1, 0, 1, 2}

DATE:__/__/__
PAGE NO.____

Then

eg→ [1] [3] [2]

• three (flex container containing number 3)
{ order: 1; } // shifted towards right (as it is +ve) ~~2 times~~

Now

[1] [2] [3] → 3 is shifted towards right

Also

Then [1] [2] [3] [4]

• one { order: 2; }
• two { order: 4; }
• three { order: 1; }
• four { order: 3; }

Now

[3] [1] [4] [2]

4- align-self: It accepts same values as align-items but is for specific item only.

eg→ [1] [2]

[1] [2]

• one { align-self: flex-end; }

[2]
[1]

flex-grow: How much item will grow relative to other items
#item_1 { flex-grow: 2; } → here 1st item occupies double space than 2nd.
#item_2 { flex-grow: 1; }

5- flex-wrap: nowrap | wrap | wrap-reverse

i) wrap-reverse → eg:

[1] [2] [3]
[4] [5]

wrap

[4] [5]
[3] [2] [1]

wrap-reverse

[1] [5]
[1] [2] [3]

for flex-direction: column

[1]
[2]
[3]
[4]
[5]

no-wrap

[1] [4]
[2] [5]
[3]

6- flex-flow: Combination of two properties: flex-direction & flex-wrap.

eg→ flex-flow: row wrap

flex-direction flex-wrap

7- align-content: To set how multiple lines are spaced apart from each other

Values:

• flex-start: Lines are packed at top of container
• flex-end: packed at bottom
• center: packed at vertical center
• space-between: Lines display with equal spacing b/w them
• space-around: equal spacing around them
• stretch: Default value. Lines stretch to take up the remaining space

DATE: / /
PAGE NO.

Like align-items align-content will be applied horizontally if flex-direction is column or column-reverse

CSS-3

In CSS 3 there are some new properties like transition, transform, etc.

eg: img { transition: all 1s; }

img:hover { transform: scale(1.1); }

hover means the properties defined in the curly braces will take effect when mouse is pointed on the element (here ie img).

Now when hover image will be grow by 10% in 1 sec as transition time is 1 sec.

There is another property: box-shadow which is used to create a shadow effect.

for eg: . { }

h1 { box-shadow: 4px 4px 5px #888888 }

horizontal offset, vertical offset, blur radius (higher no higher blur shadow is), spread value

After blur there is a spread radius (how far shadow is casted)

DATE: / /
PAGE NO.

For previous version of browser this feature is experimental feature. So we may need to use prefixes for all popular browsers.

eg:

h1 {
-moz-box-shadow: 4px 4px 5px #888888
for mozilla firefox
-ms-box-shadow: 4px " " "
for edge & internet explorer
-webkit-box-shadow: " " " "
for google chrome & safari
-o-box-shadow: " " " "
for opera browser
box-shadow: " " " "
}

To check which version of browser support the CSS 3 property use the site: caniuse.com

Syntax of transition: transition: property duration
property to be changed in given duration

For multiple properties:

eg: transition: width 2s, height 4s

transform other values: rotate(20deg), skew(xangle, yangle), translate(in pixels)
to move by pixels

DATE:__/__/__
PAGE NO.____

Responsive UI

Responsive here means whether there is a big screen or small screen like mobile it should be responsive no matter what the device is.
Responsive UI is a priority

Simple Robot Animation

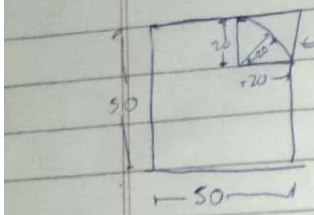
Before doing any animation we need to make a robot using CSS & HTML.
To make a robot we need to define classes for various parts of robot like head, eyes, torso, ~~hands~~ arms & legs.
In html div element is used to implement the classes(parts).
We may need a robot icon(icon for tab).
so we need to use
<link rel="icon" type="image/ico" href="robot.ico">
We need classes & some classes are nested like this.
android
- head
  - eyes
    - left_eye
    - right_eye
- upper body
  - left_arm
  - torso
  - right_arm
- lower body
  - left_leg
  - right_leg

DATE:__/__/__
PAGE NO.____

In CSS
Except eye first we need to give color to all the body parts. lets give color blue. We will use the box method to create a robot
As box is section of page so to color them we need to use
background-color: blue (This element should be inside of every classes separated by commas (class1, class2 & so on) except eyes.

Let's define head
Box model is in rectangular shape by default and android head is not rectangleler.
For circular shape
width & height should be same & ~~border~~
border-radius: 50%
/* 50% means the border radius is equal to 50% of width & height
Also length can also be used. Unit: px, etc.
Note: If box is rectangular & if we need circular border(not elliptical) then use length not % as % takes horizontal radius as % of width & vertical takes % of height.

How border-radius works?



Syntax

For 4 values: border-radius: top-left, top-right, bottom-right, bottom-left (in clockwise dirn)

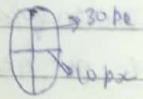
For 2 values: top-left & bottom-right; top-right & bottom-left

For 3 values: 2nd value → top-right & bottom-left

For 1 value → to all corners

As in the figure horizontal & vertical radius are same so we can make borders elliptical using different horizontal & vertical radius

border-radius: 10px/30px /* hori rad / ver rad */



For robot head we don't needed rounded border at bottom & only on top.

So let's take

border-radius: 200px 200px 0 0;

As we need to construct robot in center so use margin: 0 auto;

DATE: / /
PAGE NO.

Here auto will set the margin of left & right as such as head is horizontally center.

As head is used for animation so use transition: 1s ease-in-out /* ease-in-out will make sure that at start & end animation speed is reduced */

For eyes class we need to use

display: flex

so that eyes are aligned.

give width, height & use border-radius to make circular, also background color must be different from blue to make it visible (say white).

For classes left-eye & right-eye.

Similarly define class for upper body. left-arm, torso & right-arm have space b/w them so use margin.

For eg: left-arm { margin-right: 10px; }

Similarly define for lower-body, left-leg & right-leg.

If the left-leg or right-leg is not in correct position then use margin to properly space them.

For left & right eye use position: relative to use top & left property & space them properly in head.

Animation:- For the parts to be animated classes are defined for hover like this
.head:hover
In head we want to change width alternately & infinitely as long as it is hovered so we need to define @keyframes rule

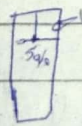
→ @keyframes changewidth {
    from { width: 200px; }
    to { width: 300px; }
}

and in head:hover {
animation-duration: 1s
animation-name: changewidth
animation-iteration-count: infinite;
animation-direction: alternate; }

We need to rotate left arm & right-arm anti-clockwise & clockwise respectively at 90° when hovered.

. . .right_arm:hover { transform: rotate(-90deg)
transform-origin: 50% 10%;

50% → horizontally center
10% → 10% below from top



rotate(-90deg) → clockwise rotation 90°.

Bootstrap

It is a front-end framework which contains CSS & Javascript based design templates

We can either download it (CSS & Javascript files) or use CDN (Content Delivery Network) using <link> to link the files without including them in the project.

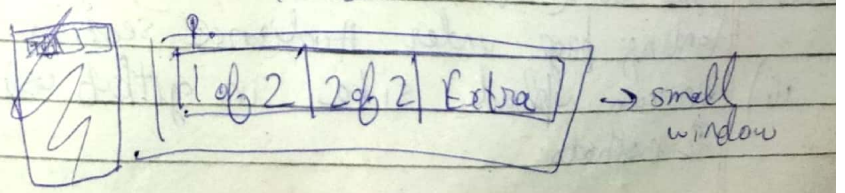
BootStrap Grid

Before release of flexbox it is used for layout

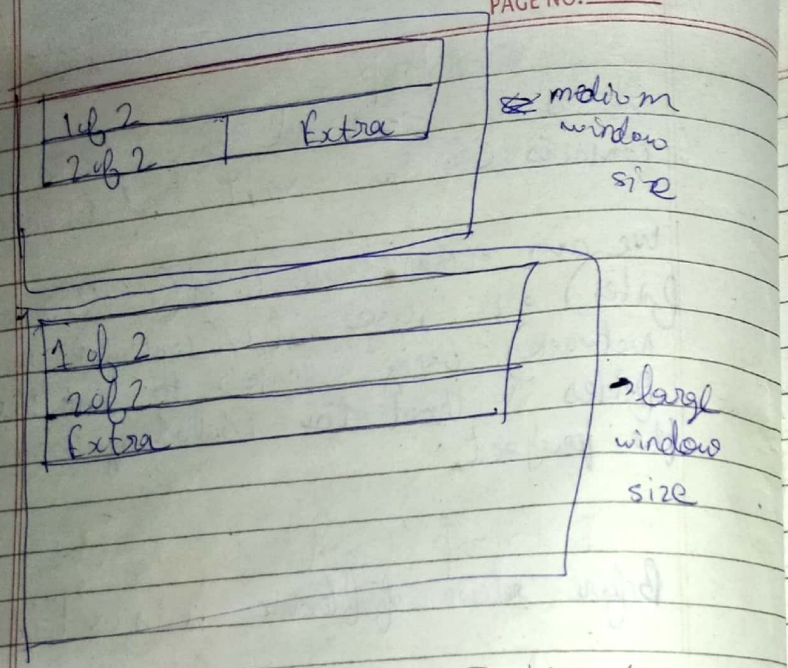
/* col-sm-6 means space given to this cell when window size is small
col-md-12 → medium window size
col-lg-12 → large window size */
1 of 2

2 of 2

Extra



DATE:__/__/__
PAGE NO.____



More about CSS & HTML

i) If we want a vertical gap b/w two containers then we can use buffer (empty) which has some height.
eg.
CSS: .buffer { height: 10rem; }

ii) We can use a site mailchimp which is email marketting service which provides database & templates for the site to record the data (like email address) by creating landing page under Audience section

iii) We can upload site in github using 2 methods.

Uploading site in GitHub

DATE:__/__/__
PAGE NO.____

a) By cloning existing empty repository by adding web files (index.html & other files) using github desktop by providing a readme file & publish using desktop client.

Note: Url of site must contain username

b) Without using desktop client, a new repository with initializing a readme file then upload files & under setting providing source, → main/root/master, etc.

To use animations we can use the tool animate.css using CDN like this (using link elem

<h1 class="animated animated animate_bounce">

animated → prefix
animate → double underscore

An animated element </h1>

Now this will make h1 element bounce 1 time.

For infinite bounce (not recommended as it is annoyi

<h1 class="animated animated animate_bounce animate_infinite">infinite bounce </h1>

DATE:__/__/__
PAGE NO.____

iv) There is a tool "creative tim" which uses bootstrap tool ~~for animate~~ which provides themes & templates

~~CSS Grid~~

v) There is a property linear-gradient which is used for gradient effect

Syntax:

background-image : linear-gradient(direction, color stop1, color stop2, ...)

For repeating linear gradient

repeating-

" : linear-gradient(top, red 0%, yellow 10%, green 20%)

↓ this pattern (red → yellow → green) will repeat 5 times to reach 100%

Eg : linear-gradient to right, red 0%, yellow 50%, green (100%)

CSS- Grid

Like flexbox CSS grid is used for the layout.

flexbox is suitable for one-direction but CSS grid is for two dimension

Before flexbox & CSS grid, bootstrap grid was used for layout.

How to use?

HTML:

// content

· container

display: grid / inline-grid

grid-template-columns: 300px 250px

Now there are 2 columns of width 300px & 250px respectively.

or for more columns

" : 300px 400px 500px

1st col → 300px 2nd col → 400px

3rd col → 500px

grid-gap: 20px (gap b/w grid is 20px from all dirn.

(Also grid-column-gap & grid-row-gap is also available)

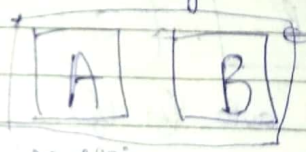
But if we squeeze the page (viewport) the items in grid will not squeez & needed to be scrolled horizontally so we can use fr.

grid-template-columns:

" : 25% 25% 25% 25%

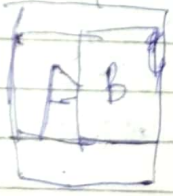
But in case of % squeezing is not restricted & may destroy integrity of items

eg →



viewport

After squeezing



& scroll down is still needed too.

But using fr

" : 1fr 1fr 2fr ← proportion

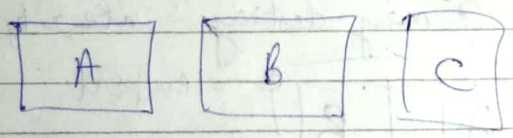
Now after scrolling squeezing the viewport comes a point at which it is more squeezed than 1st & 2nd col. (3rd column is twice large so it is more squeezed & destroy integrity) these scrolling is not needed but

template

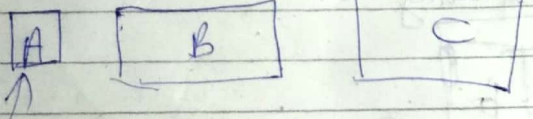
PAGE NO.

Like grid-columns there is a property grid-template-rows
eg: grid-template-rows: 2fr 1fr
If there are more than 2 rows (say totally 3 rows) then after two rows all are treated as default.
Instead of repeating for multiple times we can use repeat(4, 1fr) which is equal to 1fr 1fr 1fr 1fr
grid-template-rows: repeat(4, 1fr)

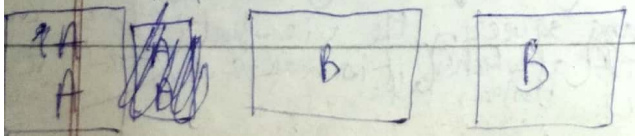
Also for 1fr 1fr 2fr can be written as repeat(2, 1fr) 2fr;
Also we can use auto instead of fr
eg:
grid-template-columns: auto 1fr 1fr
Without auto ie 1fr 1fr 1fr



With auto ie auto 1fr 1fr

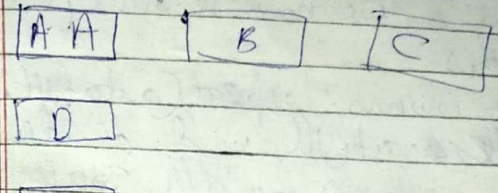


auto gives the width just required to fill the content.
If there are more than one A in box then without auto



DATE: / /
PAGE NO.

With auto



Here D & E is under auto but extra space given as we are using CSS grid so width must be fixed in whole column.
There are two more properties
justify-items → Align items in cell horizontally
align-items → Align in cell vertically
eg →
justify-items: end (other values → start, center, stretch (by default))



→ aligned at end horizontally

center is not same as stretch.
as center does not alter with width & height to fill cell but stretch do.

Using just fr is nice is also nice but it is not that responsive bcz if the viewport size is decreased there

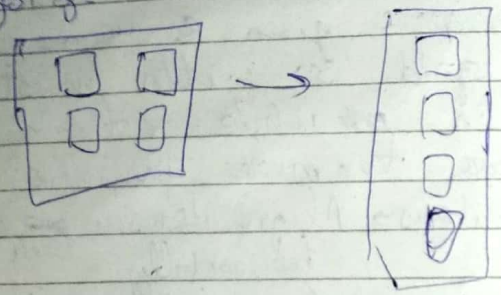
PAGE NO.

at a point comes that there is no more space left & we have to still scroll so to avoid this

grid-template-columns: repeat(auto-fill, 300px)

Now it auto-fill will decide how many columns of width 300 px are needed according to the width of viewport

for eg:



But here width is fixed so 1fr can be used instead of 300px as it is not responsive

because if viewport size is 600px then only 2 columns will be there of 300px & 240px width space is left blank.

so 1fr is recommended here instead of 300px

We can't use 1fr directly we need to use

minmax(200px, 1fr)

minimum threshold whose width is to be limited

eg: grid-template-columns: repeat(auto-fill, minmax(200px, 1fr));

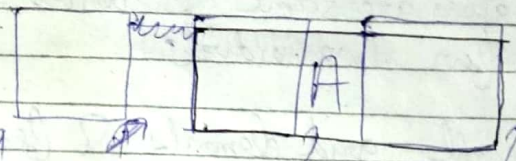
Now min size is 200px and it will grow to some size & and it will adjust accordingly.

DATE: / /
PAGE NO.

Now discussing about individual items

let's take a individual cell say cell of green class.

```
.green
{ grid-column-start: 2;
  grid-column-end: 4; }
```



Now our A of green class will use cell 2 & 3 as they contain lines 2,3 & 4 as starting line is 2 & ending line is 4.

If there are more contents of green class then in another row they will only be placed from line 2 to line 4.

The above code can be written as

grid-column: 2/4 → from 2 to 4

This same thing can be done for row to be

grid-row: 1/3

grid-column: 1/-1 from 1 to end of viewport

-1 represents end

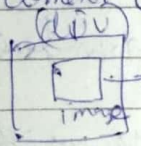
If we don't want to change position only to span then
grid-column: span 2 → span 2 columns
grid-row: span 2 → span 2 rows.

like justify-items we can use justify-self for individual and its values are same & works same only for individual.

To recall grid elements & feature there is a site : grid.malven.co. which is cheat-sheet for recalling.

Imp Points

1- for eg:- width: 100% means ~~size~~ taking full size with respective to its parent element (box).



→ if img{ width: 100%; }

and
img is under div tag which is box so it will use full width of box & it will not exceed it.

2- We can use media queries when we want changes in CSS properties according to the viewport size.
for eg:
~~@ media~~
@media only screen and (max-width: 600px)
{ .class
{ font-size: 0.5em; }
}
Here if ~~width~~ viewport width is in range 0 to 600px then CSS properties of selector defined under it will be applied.
After 600px default CSS properties (by user or HTML) will be applied.

DATE: / /
PAGE NO.

Imp Points for Developer & Career
2 out of 4 is necessary in Portfolio/resume
1- Open source Project
2- Building personal website
3- 1-2 Big Projects (Smartface recognition, ~~online~~ shopping)
4- Blog