Anatomy of CSS Rule.

```
<style>
    selector ⇒ tag to which style is to apply, here it is Paragraph, head lin

    P {
            Color       :  blue ;
            font-size : 200px ;
    }
              ↑                   ↑
          Properties        Values

    h1 {
            Color     ≠ gree ;
            text-align : center ;
    }

</style>
```

selector ⇒ tag to which style is to apply, here it is Paragraph, head lin

} Declarations which property and value seperated by semicolon. [;] is not compo

whole is stylesheet.

Selectors: Used to HTML element to which we want to style.

1) **Element selector** – selector which uses tag name as a selector for styling.

```
P {
    color : Blue;
}
```

<P> ---- <P>
<P> ------<P> ?
         ↑
every paragraph is bl

2) **Class selector** – We define selector as a class with `. Classname` selector and this class is assigned to required tags. many classes car assigned one element as `class = "Blue Red"` Blue clas

```
. Blue {
    colour : blue;
}
```

<P Class = "Blue" > ---
<P Class = "Blue" > ---<
<P>----<IP>   ↑
       ↑   Blue te
   unaffected.

3) **id selector**

```
# name {
    color : blue;
}
```

<P>----<IP>
<div id = "name"> -- <

Here id is given to div element from that id div element to which style is to be applied is identified and inside <style> we use `id with #` and assigned style to it.

Combining Selectors

1) Element with class selector.

```
P. big {
        font-size = 20px;
}
```

```
<P class = big  > ---- </P
<div Class = big  > -- </div
```

← space

only P elements with given class are affected others are unaffected.

2) child selector

every P of article which is direct child gets the style.

```
article > P {
        Color ; blue;
}
```

```
<article>
        <P> ---- </P>
        <P> ---- </P>
</article>
```

But P should be direct child.

3) Descendant selectors

```
article P {
        color = blue;
}
```

```
<article>
        <P> ---- </P>
        <div><P> --- </P></div>
</article>
```

Every P element inside article element irrespective of wheather it is direct element or not gets that style.

Various selector combinations

.Colored P {
     color : blue;
}

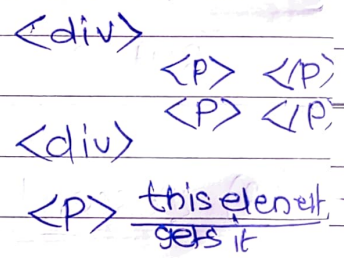every P element inside elements with class colored gets style

article > .colored {
     color : blue;
}

every element having class colored and inside article any direct child gets style.

Adjacent Sibling Selector
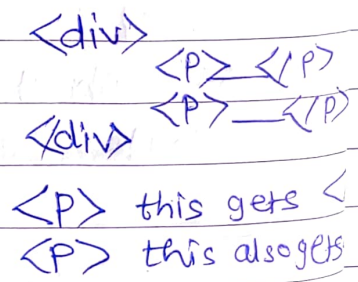
div+P {
     background-color : yellow;
}

adjacent sibling is element next to g element at same

```
<div>
   <P> </P>
   <P> </P>
<div>
<P> this element gets it
```

General Sibling Selector

div~P {
     background-color : yellow;
}

All the siblings given element.

```
<div>
   <P> </P>
   <P> </P>
<div>
<P> this gets
<P> this also gets
```

Pseudo-Class Selectors

```
selector : Pseudo-class {
                    Property : value;
}
```

selector are choosen according to rule studied and Pseudo-classes classes can be as follows-

1) link — If element is a link

2) visited — If element visited

3) hover — If mouse hovers over

4) active — If mouse click on it but not realesed

5) nth-child(x) — Perticular level of child's of element selected.

```
e.g. 1) a : hover {
              color : red;
}
```

hovering over link in a gives red colour to it.

```
2) P : hover, a : active {
              color : red;
}
```

hovering over paragraph and clicking on link changes them to red.

CSS Pseudo-Elements

CSS pseudo-element is used to style specified parts of an element.

for e.g.

1) Style the first letter, or letter of an element
2) Insert content before or after the content of element.

```
selector :: pseudo-element {
              Property : value;
}
```

1) **first-line** - It styles first line of element only.

```
P :: first-line {

}
```

2) **first-letter** - It styles first letter of element o

```
P :: first-letter {

}
```

3) **before** - It is used to insert something before e like emoji with [Content] property

```
P :: before {
       content : url (smiley.gif);
}
```

4) **after** - used insert after Content of eleme

5) **selection** - when element content is selected b user.

```
P :: selection {
       color : red;
       background : yellow;
}
```

Attribute Selectors

Used to style Elements having specific attribute.

Syntax

Element [attribute = "value"] {

}

e.g.

a [target = "_blank"] {
background-color : yellow);
}

It styles link having [blank] as [target] attributes with background color yellow.

Other formats

1) element [attribut = "value"]
Attribute with specific value.

2) element [attribute ~= "value"]
   . [title ~= "flower"]
when flower is one word inside title

3) element [attribute |= "value"]
   [title |= "flower"]
when title start with space separated / (-) separated flower word

4) element [attribut ^= "value"]
   [title ^= "flower!"]
when title start with flower word flower may not be separated like flowerofloish

5) element [title *= "flower"]
         [title *= "flower"]

when flower word occured somewhere like in [~=] but may not be separated by space or (—) like |myflower is| is title.

---

* Attribute selector is most useful for using CSS for forms.

Style placement.

1) style can placed in Elements. (Inline style)

<element Style = "Property : Value;">-----</P>

for e.g.:-

<P style ="text-align: center;">-----</IP>

this method is least used because it is least reusable.

2) External style

<head>

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

</head>

Style.css

```
body {
        background-color : grey;
        font size : 130%;
     }
```

In this method external styling sheet is used and its reference is given with link this method is used in real world application. It is usefull when there are lots of pages to style as perticular way.

3) <u>Head styles - style with <style> tag (Internal)</u>

```
<head>
<style>
        p {
            color : maroon;
        }
</style>
</head>
```

Head style is used while overriding the externa
styles in real world application.

CSS Colors

Properties and values

1) Property
i) color   iv) opacity
ii) background-color
iii) border

2) Value

RGB & RGBA

1) rgb (R, g, b)   R, g, b ranges {0 - 255}
2) rgba (R, g, b, α)   α ranges {0 - 1}

B) HEXADECIMAL - HEX

#000000.   first, second and third "00"
    to   are for R, G, B respectively
ffffff

HSL - HUE, SATURATION, LIGHT

1) hsl (h, S%, l%)

h - hue - colorwheel 0 is red, 120 is green, 240 is
                blue.

Saturation - Degree of grey shade   0% is grey
                100% is full color.

light   - Degree of light from 0% to 100%.

2) hsla (h, S%, l%, α)   - α ranges [0 - 1]

Background

I) Background color

i) background-color : i) "colorname" ;

ii) rgb(r,g,b) ;

iii) rgba(r,g,b,a);

iv) hsl (h,sl) ;

v) hsla(h,s,l,a);

vi) #000000;

ii) opacity   : (.0-1);

II) Background image

i) background-image : url ("url.format");

ii) background-repeat : repeat;

no-repeat;

repeat-x;

repeat-y;

iii) background-position : right top; right bottom; rig
     inherit;   left top; left bottom; lef
     (x% y%);   right center; left center;
     (x px, y px);   top center; bottom center.

III) Background Attachment

i) background-attachment : scroll;   background image si
                           fixed;   background image

IV) Background Shorthand – This property used to specify a
background property in sigle line.

i) background : rgb (R,G,B)
               url ("   .   ")
               repeat
               scroll
               50% 50% ;

Any property of shorthand sequence can be missing but sequence should be same.

     i) Color
     ii) image
     iii) repeat
     iv) attachment
     v) position

Opacity property can be used with images, text etc so that image transperancy can be controlled. It is also set with the help of background-color property with opacity attribute (α)

V) Background Shadow

i) text-shadow: 2px   2px   2px   rgba(R,G,B,α);

         Horizontal   Blurr   Color
             Vertical

Horizontal vertical
Can be negative for upward or left side

ii) box-shadow : 2px   2px   2px   rgba(R,G,B,α)

Used for shadow of whole box containing element.

CSS Borders

Border style

i) border-style : dotted; groove; inset; solid; dashed; ridge; outset; double;

ii) border-style : dotted solid ridge dashed;
             upper   Right   bottom   left,

iii) border-style ; dotted solid ridge ;
             upper   left Right   bottom

iv) border-style ; dotted solid ;
             upper bottom   right lef).

Border width

i) border-width : medium;
             thick ;
             x px ;

ii) border-width : x px y px z px medium/thick;
             top   Right   ~~lef~~   left.
                           bottom

iii) border-width : x px y px ;
             top   bottom

Border Color

i) border-color ; color ;   hsl(h, s:.,l:.) ;
             rgb(R,G,B);   rgba(R,G,B,A);
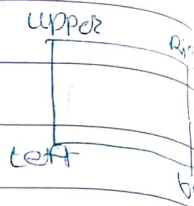   #ffffff;

Border Sides

border-top-style : styles
border-right-style ;-styles
border-bottom-style : styles
border-left-style : styles

Border Radius

i) border-radius : 15 Px 15 Px 15 Px 15 PX ;
    upper   Right   Bottom left

ii) border-raidus : 15 PX   15 PX   15 PX ;
    upper   Right left   bottom

iii) border-radius : 15 PX   15 PX ;
     upper bottom   Right left

iv) border-radius : 15 PX ;
    all

v) border-top-left-radius : 2 em ;

vi) border-top-Right-radius : 1 ;

vii) border-bottom-left-radius : 50% ;

viii) border-bottom-right-radius : 5 px ;

Border Shorthand Property

border : 6 px solid red ;

Order of values
i) width
ii) style (compulsory)
iii) color

CSS outlines

outline is used to making element standout by a
it outside borders with some spacing.

It has all the properties same as borders.

only one diffent property as follows :

outline-offset : x px ;

CSS Margins,Padding & Height,Widths

Margins

margin :   auto;   // browser calculates

           x px;   // length

           x em;   // length

           x %;   // o/o width

margin - top   : auto;

margin - bottom : x px;

margin - right   : x em;

margin - left   : x % ;

margin : x px , y px , z px , a px ;

          upper   right   bottom   left

margin : x px , y px , z px ;

          upper   right left   bottom

margin : x px y px

          upper bottom left right

margin : x px

          all .

Padding

Same as that of margin.

Height & width

i) height : auto; x px; x em; x %; inherit; initial;

ii) width : auto; x px; x em; x %; inherit; initial;

auto - default value, browser calculate height width

x px - length

x em - length

x % - % of box containing it.

inherit - sets value to default.

initial - parent's value.

i).min-width : x px, xem; x/.; auto; inherit; initial;
ii)max-width; x px; xem; x/.; auto; inherit; initial;
iii)min-height; x px; xem; x.y.; auto; inherit; initial;
iv)max-height; x px; xem; x/.; auto; inherit; initial;

max-min property used when resizing is concern
browsers.

CSS Text

Text have Properties like -

Text color

color   : "color" ; rgb(R,G,B) ; rgba(R,G,B,α); #000000, hsl( )

background - color : "color"; rgb(R,G,B) , rgba(RG,B,A); #000000, hsl( )

Text Alignment

text - align : Justify ;   // every line stretched to same width.

right ; left; center

vertical - align : top;
                   middle;
                   bottom;

Text decoration

text - decoration : overline ; underline; line - through;
text which is not link should not be underline it
creates confusion.

Text transform

text - transform : uppercase;
                   lowercase;
                   capitalize;

Text spacing

text - spacing
letter - spacing
word - spacing
line - height

Text shadow

text - shadow : 2px 4px 2px "colour",
                Horizontal vertical blur colour.

CSS Fonts

Font family – It used for multiple fonts when first font is not supported then next font selected as per compatibility of browser.

i) If font name is more than one word then ["
commas are used to quote them.

ii) To start with font we want we make it a class name.

e.g.

```
.serif {
        font-family : "Times New Roman", Times, se
}
```

Font Style

```
font-style  : normal;
              italic;
              oblique;
```

Font Weight

```
font-weight : normal;
              bold;
              900;          Boldness level [100-9
                            in multiple of hundre
```

Font Size

```
font-size : x px;        // x px
            x em;        // x is relative to parent
            x vw;        // x is default and vw fon
```

<u>Font Google</u> — The link of google fonts inserted and then font family is used.

```
<link rel="stylesheet" href="https://fonts.googleapis.com
      /css? family = Sofia">
```

font-family : "Sofia";

<u>Font Property Shorthand</u>

```
font : italic;
       small caps;
       bold ;
       12px;
       Georgia, serif;
```
<u>sequence</u>
font-style
font-variant
font-weight
font-size   (Required)
font-family   (Required)

CSS links

CSS properties for links are same as we applied text or blocks but other than this we need properties according to activities of mouse.

a: link { // unvisited link

      style for event

}

a: visited { // visited link

      style for event
      text-decoration : none;
}

a: hover { // hover ov.

      style for even

}

a: active { // clicked li

      style for ev

}

CSS Lists

Undrdered list

```
ul {
    list-style-type: circle;
                     square;
}
```

```
list-style-image: url('mname.fu');
```

Ordered list

```
ol {
    list-style-type : upper-roman;
                      lower-roman;
                      upper-alpha;
                      lower-alpha;
}
```

Style position

```
li {
    list-style-position: outside;
                         inside;
}
```

list have certain margin/paddings.
we can remove bullet point by setting style to none
```
    list-style-type: none;
    margin: 1px;
    padding: 2px;
```

CSS Tables

we can define borders, background colors tex col
events of mouse etc to table and tr, th, td etc-

```
table, th, td {
      border-collapse : collapse
      border   : 1px, solid black;
}
```

Properties for table, th, td, tr

i) border : 1px, solid, black;
ii) border-bottom : 1px, ridge, red;
iii) border-collapse : collapse; no-collapse;
iv) width : xpx;
v) height : x px;
vi) text-align : left; Right;
vii) padding : xpx;
viii) vertical align : upper; bottom;
ix) color : "color"
x) background-color : "colour"

CSS Layout - Overflow — It decides how content that unfits into box is handled.

i) overflow : visible - overflow not clipped
           hidden - overflow hidden
           scroll - overflow handled with scrollbar compuls
           auto - ~~a~~ Scrollbar only when necessary.
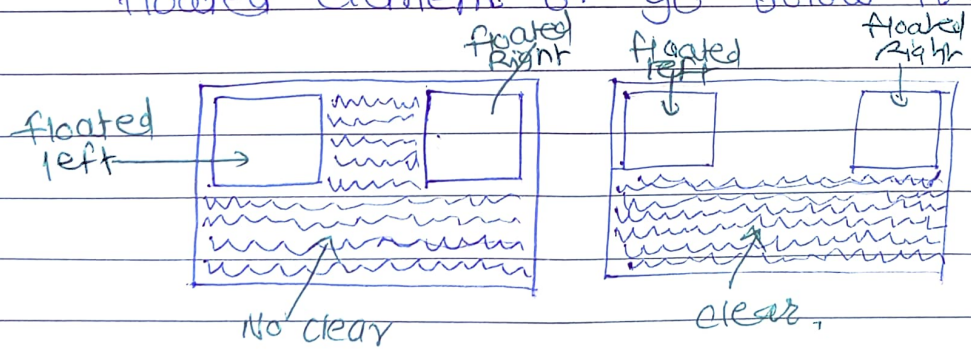
ii) overflow-x   // To handle overflow in
iii) overflow-y   // x or y directions only.

CSS Layout - float & clear

float - specify how element should float, i.e. the proper for positioning and formatting content inside of container.

Clear — It is used after the float property to decide wheather element should be on next to finitially floated element or go below it.

i) float : right;
          left;
          none;
          inherit;

ii) clear : right;   // No floating element on right side
           left;   // No floating element on left side
           both;   // No floating element on either sides
           none;   // floating element on either side allowed
           inherit; // Inherit.

CSS Display - override element's inline or block property.

Display : inline;
        block;
        inline -block;

visibility : hidden;   // element affects layout but is hidden.

CSS Position

The [position] property specifies the type of posit method used for an element
(static, relative, fixed, absolute or sticky)

Static

Conflict Resolution.

1) When there is conflict between two declaration we follow the rule - ["Last Declaration wins"] because HTML follows [top to bottom approach.]

When external style declarations have conflicts they follow the same rule as per the position of declaration as most of time <link> declared at head (top).

2) When there is no conflict it follows rule - ["Both declaration merged"]

Inheritance

[body]

[Dom Tree]

├─[Element]

└─[Element]

└─[Element]

├─[Element]

└─[Element]

Any property given to parent element is also inherited by child element like properties 🟥 & 🟪 inherited.
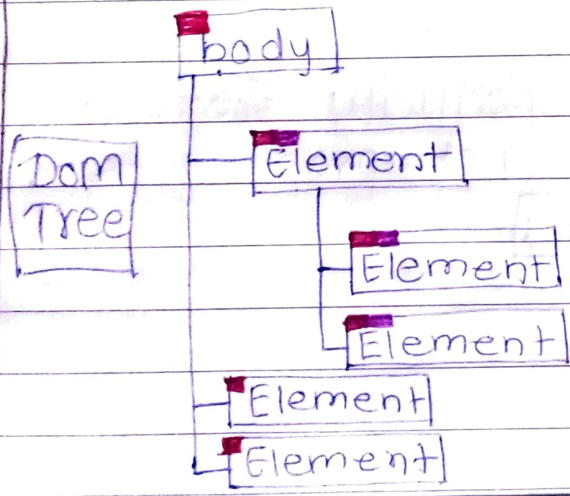
3) Specificity - [Most specific Selector Combination win]

To find the specificity we use specificity ....

style = " "	ID	class Pseudo class attribute	# of Elements
☐	☐	☐	☐

```
div #my para{
    Color: blue;
}
```

```
div.big p{
    color: green;
}
```

Style=""	ID	Class	# Elem
0	1	0	1

Score = 101

Style=	ID	Class	#Ele
0	0	1	2

Score =12

according to scores Paragraph gets blue color.

Overriding the specificity rules.

When declare the style with ! important tag it overrides all of rules of cascading and use the same property declared with !important tag eg. above example will show green color irrespective of their specificity score if we add important tag after property as-

Color: gree ,important ;

CSS Text Styling -

font-family — It gives different font choices for user's browser from which browser selected font which it supports.

e.g. font-family : Ariel, Helvetica, sans-serif;

Color — It allows to set colors either by name or by RGB formats in hexadecimal no. followed after #, or $\boxed{rgba(0,0,0,0)}$

a for transferency.

e.g. color : #0000ff;

   Red green Blue

font-style — It gives options like
   i) Normal   iii) italic
   ii) Oblique   iv) inherit

e.g. font-style : italic;

font-weight — It gives boldness by assigning the word value **Bold** or numbers from $\boxed{100, 200, 300 \text{---} 900}$

e.g. font-weight :900;

font-size — The default is $\boxed{16\,px}$ for most of browsers we can set it with $\boxed{px}$ declaration or $\boxed{\%}$ of default.

eg font-size : 24px;

But font sizes changes according to user make it zoom in or zoom out so keep everything relative to each other we style font size with relative styling method as follow.

```
body {
        Font-Size : 120% ;
}
```

Now when we want to change the style of font size
we redeclare without overriding original but
relative to it. as.

`<P style = "font-size = 2em">- - - -</P>`

2em, 3em, 4em or 0.1em, 0.2em gives font
size of double, triple - - - - respective to parent
font size which we set to 120% initially,
As body is direct parent of `<P>`

Text Effects

```
P {
        text-overflow : clip ;     ⇒ [Shubham saw]
                       ellipsis;  ⇒ [Shubham Sa...]

        word-break : keep-all;  ⇒ Break only at space
                     break-all; ⇒ Break at any charrac

        word-wrap : break-word ⇒ It break word u
                                not fitted in box

        writing-mode : horizontal-tb; ⇒ write hor
                       vertical-rl ;  ⇒ write upside
}
```

Box Model

The Box is the wrapper that wraps every html element
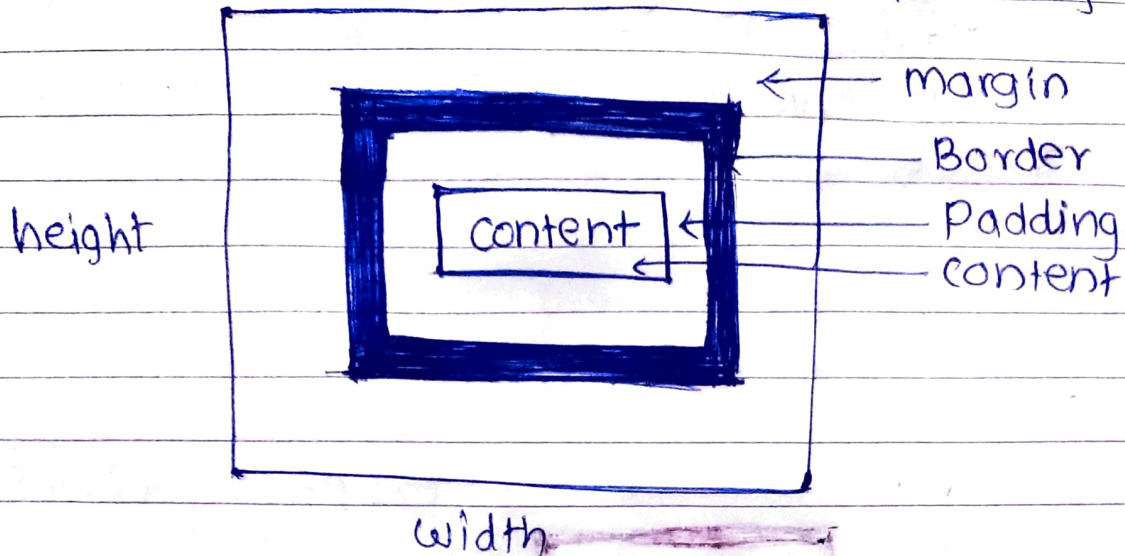
width = width + 2× Border + 2× Padding + 2×margine

height = height + 2× Border + 2× padding + 2×margine.

```
<style>
    body {
            background-color : grey;
            margin : 0;
        }   padding : 0;


    # box {

            background color : green;
                padding   : 10 PX . 10PX , 10PX , 10PX   // (T,R,B,
                border    :  5 px   solid black ;                    2
                margin : 40px ;
                width     : 30 PX ;
                height    : 10 PX ;

    # content {

            background-color : violet;

        }

</style>
```
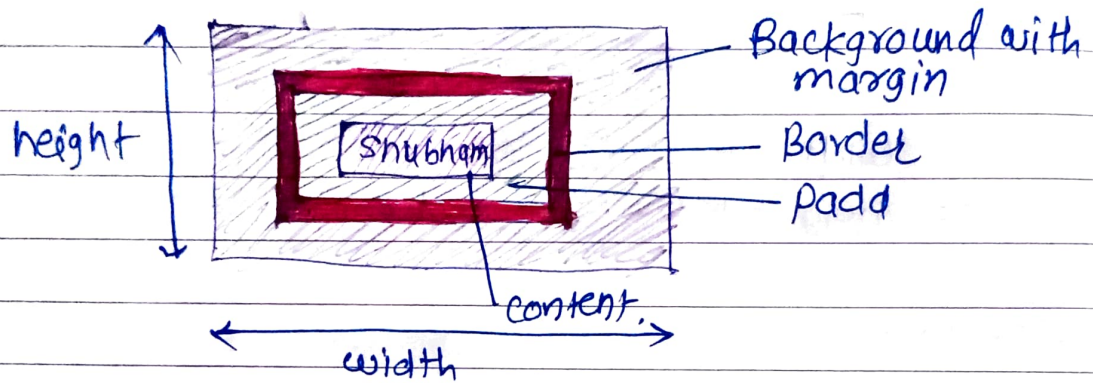
```
<div  id = "Box">
    <div  id ="content">  Shubham </div>
</div>
```

output

Background with margin
Border
Padd
content
height
width

$$width = 30 + 5 + 5 + 10 + 10 = 60 \text{ px}$$
$$height = 10 + 5 + 5 + 10 + 10 = 40 \text{ px}$$

The width is always addition of all element inside element content, border and pad as box is defined to content but it can be changed. using property [box-sizing: border-box;] inside that element styling & to parent like body, or *Selector for all the element.

```
* {
    box-sizing: border-box;       // for all elements
}
```

```
body {
    box-sizing: border-box;       // for child element
}
```
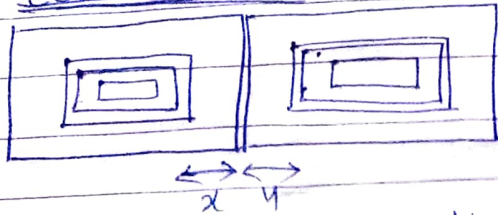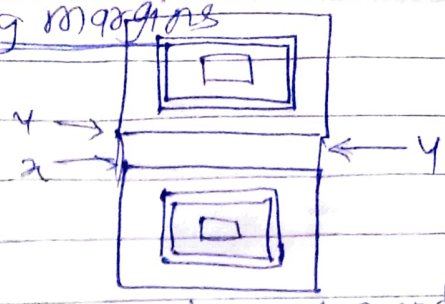
```
p {
    box-sizing: border-box;       // for specific element
}
```

Commulative and collapsing margins

margins on same line
get commulated hence
[Net margin = $x+y$]

margin on top and below
boxes are merged with
biggest margin ∵ $y > x$
[margin = y]

overflow property

overflow — when box size is set to very low with height and width and texts inside are overflowing outside the boxes then we use this command overflow to handle the situation.

overflow: visible — Defaut it cause text to overflow out
hidden — It hides the text that don't fit inside
auto — It provide scrollbar whenever necess.
scroll) - It always provides two scrollbar even when not required.

shubham Ashok sawant
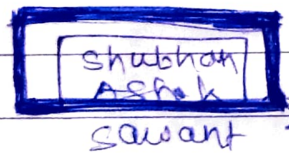
overflow

CSS forms

We use CSS forms to Style the form element like input with different formats.

1) ## Input Field

```
input {
        width    : 50%;
        padding : 12px 20px;
        margin : 8px 0;
        box - sizing: border-box;
        border : 2px solid red;
        border - radius: 4px;
        color : black;
        background color : white;
```

2) ## Input field with specific type
To style specific type of input.
```
input [type = text] {
input [type = Password] {
input [type = number] {
```
also button, submit, reset
   etc.

3) ## Focused inputs
To change the style when it gets focused (clicked)
we use focuse selector for doing so.

```
input [type= text] : focus {


                              background-color : lightblue;

}
```

4) Input with icon/image

If we want icon inside the input use the [background] properties.

```
input [type=text] {

        background-color : white;
        background-image : url ('searchicon.png');
        background-position : 10 px 10px;
        background-repeat : no-repeat;
        padding -left : 40px;

    }
```
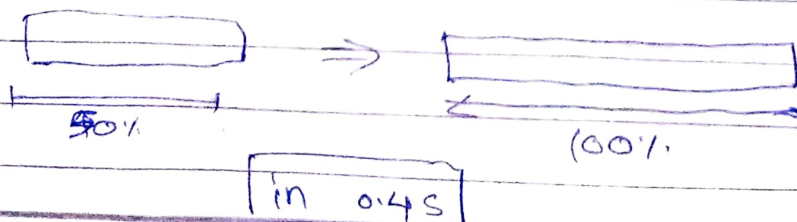
5) Animated search input

use [transition] [Focused] properties to animate input when focused.

```
input [type=text] {

            transition: width 0.4s ease-in-cut:
            width: 50% ;
            padding: 4px;
            background-color: white;

    }
```

```
input[type=text] : focus {

            width : 100% ;
    }
```

Output

50%   (00%

[in 0.4s]

6) Text areas input - use resize property to define wheather text areas should be risizable or not

text area {
  resize: none;
  width: 100%;
  height: 50%;
}

7) Select menus

select {
  width: 100%;
  height: 20px;
  border: none;
}

CSS Animation

1) <u>CSS @ key frames Rule</u> — @keyframes defines animation from changing one CSS style to other CSS style gradually with defined [% selectors] or [from to] selectors as follows —

0% start
10% :
50% :
100% end

from (same as 0%)
:
:
to (same as 100%)

<u>Syntax</u>

@key frames animationname {

   keyframe {CSS-style}
   selector

}

2) <u>animation-name</u> — used to apply animation with @ key frames to CSS block in which its specified.

animation -name : name of animation ;

3) <u>animation duration</u> — duration in seconds.

animation-duration : 4s ;

4) <u>animation-delay</u> — delay to start animation in seconds.

animation-delay : 2s ;

5) <u>animation -iteration-count</u> — number of time a repeats.

animation- iteration- count : 3 ;

we can set to infinite for forever run of an

6) <u>animation - direction</u> — The direction in which animation to place.

> animation - direction : normal
> : reverse
> : alternate — forwar backu
> : alternate- reverse

7) <u>animation timing function</u> — defines animatio speed curve

animation-timing-function : ease // Start and end s
: linear // same constant s
: ease-in // slow start
: ease - out // slow end
: ease-in-out // slow start &e
: cubic-bezier (n,n,n,n)// define
defined

8) <u>animation - fill - mode</u> — style of element when element when the animation is not playing

animation — fill — made : none //no style before aft
: forwards// keeps last keufram
: backwards//keeps first keufram
: both . // extend property in
forward & backward

```css
div {
        width : 100px ;
        height : 100px ;
        background-color ; red ;
        position : relative ;
        animation-name ; example ;
        animation-duration : 4s ;
        animation - iteration-count : 3 ;
        animation - direction ; alternate ,
        animation - delay ; 2s ;
        animation - timing-function : ease
        animation-fill-mode : forwards

}


@ keyframes example {

        0% { background-color : red ; left : 0px ; top : 0px ;}
        50% { background-color : blue ; left : 200px ; top : 0px ;}
        100% { background-color : green ; left : 0px ; top : 0px ;}

}
```
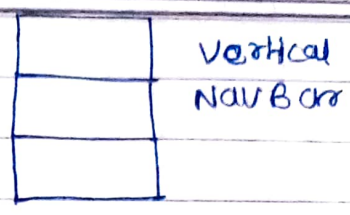
Output
___

1) 0%.   ▨ →

                  ↓

2) 50%.   ← ▨   } 3 times

                  ↓

3) 100%.   ▨
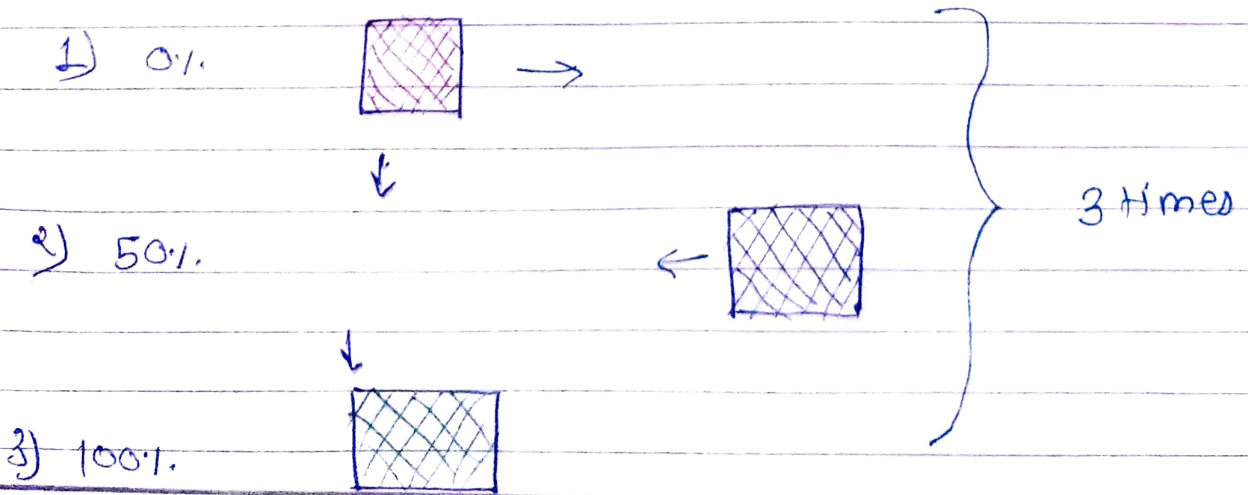
CSS Navigation bar

Navigation bar is a kind of list of url's.
It has base of html list elements and their are
two kinds of Navigation bars:

1) ## Vertical Navigation bar

Vertical Nav Bar

```
ul {
        list-style-type : none;          // order style of list
        width    : 50%;                  // width of Navigation bar
        background-color : #f1f1f1;      // background color
        height : 50%;                    // height of Navigation bar
        position : fixed;                // Position of Navigation
        overflow : auto;                 bar with respect to
                                         webpage wheather it
                                         should be fixed or move
                                         with page scroll.
    }                                    // overflow defines what
                                         should happen with unfitted
                                         elements of block.
                                         (Tabs of Nav bar)

li a {
        display : block;                 // to make block elements
        color : #00000;                  // color of text of tab

    }

li   { text-align : center;             // text aligns in tab

    }

li a:hover{background-color ; #555;     // background color
           color : white;               when hovered over
                                         tab.
    }                                    // color of text when
                                         hover.

li a.active {
            background-color : #CAE50;
        color : white;
    }
```

2) Horizontal Navigation bar

use | display: block | to make whole area clickable
| float; left | to float blocks
| Position: fixed | to fix Navigation bar on page.

```
li {
        float : left;
        position ; fixed ;
}


li a {
        display : block;
        color : white;
        text - align : center ;
        Padding : 14 PX 16 PX ;
        text - decoration : none ;

}


li  a: hover {

        background - color : #111 ;
        }


.active {

        background - color : # 4CAF50 ;

        }
```

To make tabs divider in Navigation bar use border property   border-right, border bottom etc.

CSS Grid

CSS grid used for defining elements like keypad. when [Div] elements are declared inside [div] element then outer [div] is grid-container and inside [div] are grid elements/items.

```
.grid-container {
            display : grid;
            grid-column-gap : 2 PX;
            grid-row-gap : 3 PX;
         // grid-gap : 2PX 3 PX;
            grid-template-column : auto 30px auto,
            grid-template-rows : auto 30px auto,
```

no of element defined defines column, row numbers auto is undefined size and size can be given. here 3X3 created.

```
.grid-container div {
            background color : green ;
            text-align : center ;
            font-size : 20 px :

         }
```

```
<div class = " grid-container
            <div> 1 </div>
            <div> 2 </div>
            <div> 3 </div>
            <div> 4 </div>
            <div> 5 </div>
            <div> 6 </div>
            <div> 7 </div>
            <div> 8 </div>
            <div> 9 </div>
</div>
```
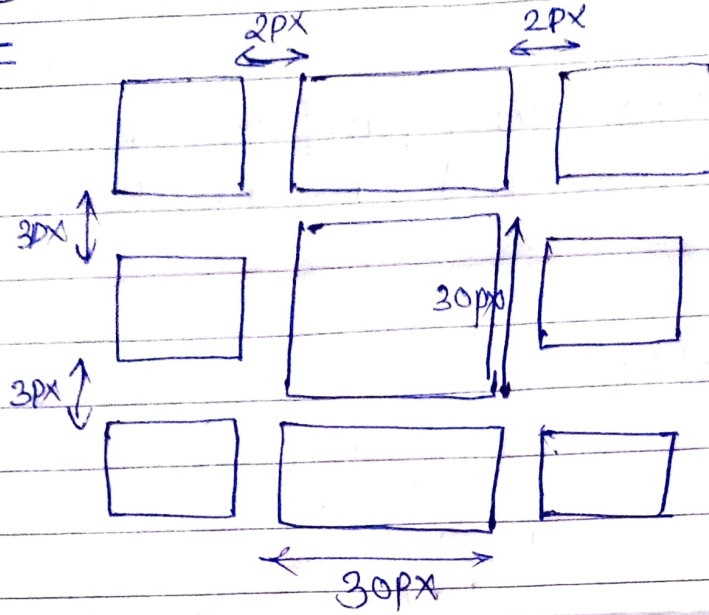
CSS transitions

css transition allows you to change property value smoothly over given time. duration.

We define transitions with two selector blocks css one have property [transition] and one have pseudo-selector and properties to change as mentioned in [transition name].

```
div {
        width : 100px;
        height : 100 px;
        background -color : Red;

        transition - property; width height;
        transition - delay : 2s;
        transition - duration :4s;
        transition - timing -function; ease;

}


div : hover {
            width : 200px;
            height: 200px;

        }
```

As here hover over div may change the css property as per selectors and css blocks but transition only helps us to transform css properties with defined ways.

[transition- property] - properties to apply transitions
[transition - timing - function] : ease ;   lineor;
                            ease -in;   cubic-bezier(nnm
                            ease-out;
                            ease-in-out;

CSS Units

Absolute lengths

cm - centimeter	in - inch	Pt - Points
mm - millimeter	px - pexels	Pc - Picas

e.g.   20px , 20cm , 20mm

pixels are relative to viewing device screen
for low dpi-device   one pixle is 1 device pixels
for high-dpi-device   one pixel is multiple pixels.

Absolute lengths are not recommended to use.

Relative lengths — It specify length relative to another length property. It is recommended method.

em - (2em means 2 times current font)
ex - relative to x-height of current font.
ch - relative to 0 width (zero's width)
rem - relative to font size of root element
vw - relative to 1% of width of viewport
vh - reative to 1% of height of viewport
vmin - relative to 1% veiwport's smaller dimension
vmax - relative to 1% viewport's longer dimension
% - retivabeto parent element