CSS

M T W T F S S
Page No.:
Date:
YOUVA

Anatomy of CSS Rule.

```
<style>
selector ⇒ tag to which style is to appy, here it is paragraph, headlin
  p {
      Color     : blue;
      font-size : 200px;
  }
      ↑             ↑
   Properties     Values

  h1 {
      Color      : Gree;
      text-align : center;
  }
</style>
```

} 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 can assigned one element as `class = "Blue Red"`

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

<p class = "Blue" >...
<P class = "Blue" >---</
<P>---</P>
↑ Blue te
Unaffected.

Blue
class

3) **id selector**

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

<P>----</P>
---</

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

M T W T F S S
Page No.:
Date:
YOUVA

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

selector are chosen 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.

Page No.:
Date:
YOUVA

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.

```
P :: first-letter {
        }
```

3) before - It is used to insert something before element like emoji with content property

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

4) after - used insert after content of element

5) selection - when element content is selected by 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 flowerofthis

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

when flower word occured somewhere like in ~= but may not be separated by space or (-) like myflowerisis 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;"> ------ </P>
```

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) Head styles - style with <style> tag (Internal)

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

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

CSS Colors

Properties and Values

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

2) Value

RGB & RGBA

1) rgb(R, G, b)
2) rgba(R, g, b, α)

R, g, b ranges {0-255}
α ranges {0-1}

B) EXADECIMAL-HEX

#000000
to
#ffffff

first, second and third "00" are for R, G, B respectively

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,α);
iv) hsl(h,sl) ;
v) hsla(h,s,l,α);
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; righ
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 sl
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);

- 1st 2px: Horizontal
- 2nd 2px: vertical
- 3rd 2px: Blurr
- rgba(R,G,B,α): Color

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 left.

Border Width

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

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

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

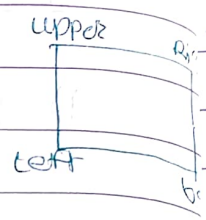
Border Radius

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

ii) border-radius : 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-offeset : x px ;

CSS Margins Padding & Height, Widths

Margins

```
margin : auto;      // browser calculates
         x px;      // length
         x em;      // length
         x %;       // % 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: xpx; xem; x%; auto; inherit; initial;
ii) max-width: xpx; xem; x%; auto; inherit; initial;
iii) min-height: xpx; xem; x%; auto; inherit; initial;
iv) max-height: xpx; xem; x%; auto; inherit; initial;

max-min property used when resizing is concern browsers.

CSS Text

Page No.: YOUVA
Date:

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(R,G,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 for
```

Font Google - 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";
```

Font Property Shorthand

```
font : italic;
       small caps;
       bold ;
       12px ;
       Georgia, serif;
```

Sequence

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 otherthan this we need of 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 ever
}
a:active {  // clicked li
    style for ev
}
```

CSS Lists

M T W T F S S
Page No.:
Date:
YOUVA

Unordered list

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

    list-style-image : url('name.png');
```

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: 0px;
padding: 0px;
```

CSS Tables

We can define borders, background colors text 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 : xpx;
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.

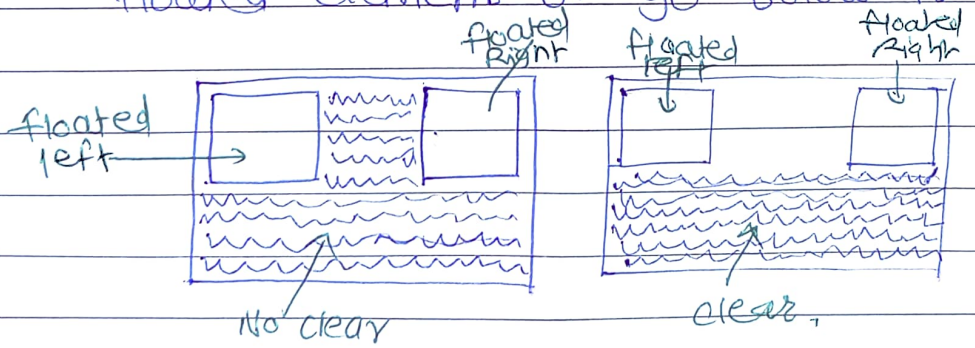
i) overflow : visible - overflow not clipped
hidden - overflow hidden
scroll - overflow handled with scrollbar compulsorily
auto - ~~or~~ scrollbar only when necessary.

ii) overflow-x
iii) overflow-y
// To handle overflow in x or y directions only.

CSS Layout - float & clear

float - specify how element should float, i.e. the property 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 finally 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 positioning method used for an element
(static, relative, fixed, absolute or sticky)

Static

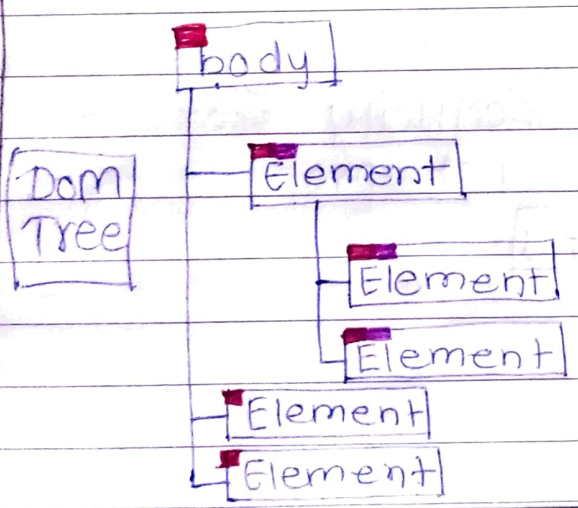
Conflict Resolution

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"**

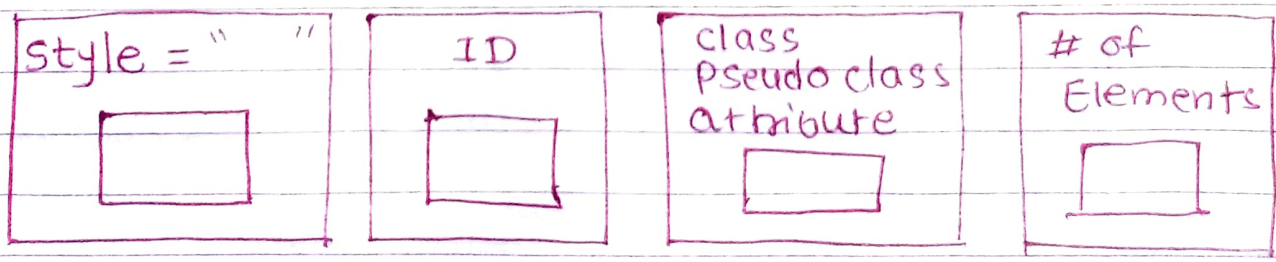
Inheritance



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 :-



```
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 rgba(0,0,0,0)
a for transferency.

e.g. color : #00 00ff ;
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 100, 200, 300 - - - 900

e.g. font-weight: 900;

font-size - The default is 16px for most of browsers we can set it with px declaration or % of default.

eg font-size : 24PX;

But font sizes changes according to user make it zoom in or zoomout 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>
```

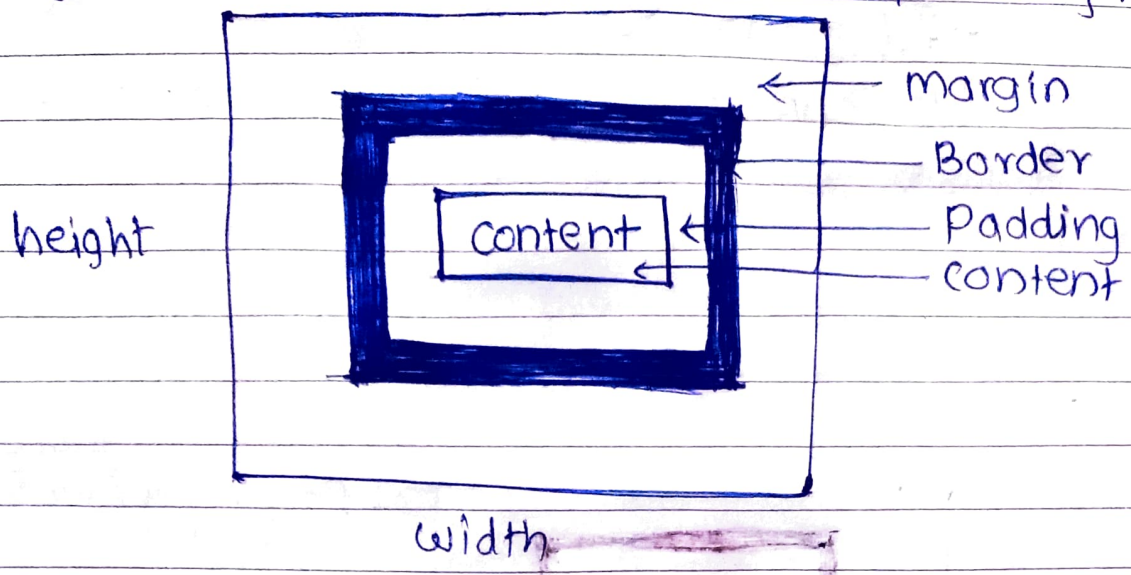
2em, 3em, 4em or 0.1em, 0.2em gives font size of double, triple - - - - respective to parent font size which we set to 120% initially. As body is direct parent of <P>

Text Effects

```
P {
    text-overflow : clip ;      ⇒ Shubham saw
                    ellipsis;   ⇒ Shubham sa...

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

    word-wrap     : break-word ⇒ It break word if 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

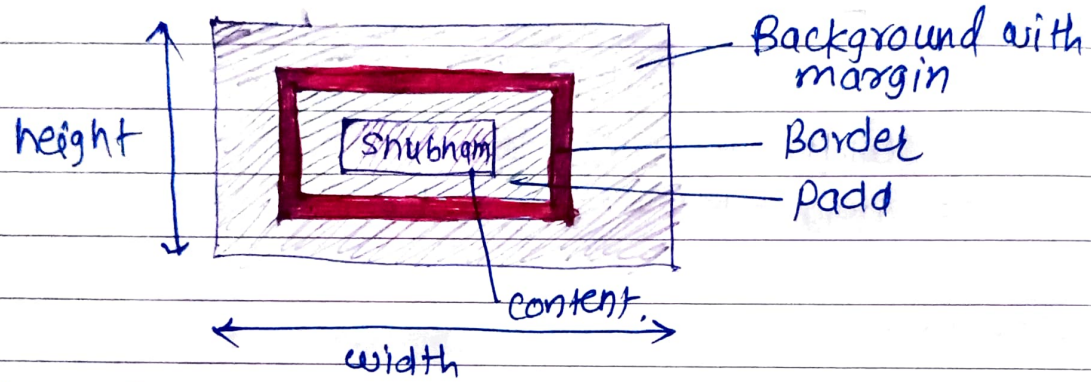


width = width + 2x Border + 2x Padding ~~+2xmargine~~
height = height + 2x Border + 2x padding ~~+2xmargine~~.

```
<Style>
 body {  background-color : grey;
         margin : 0;
         Padding : 0;
      }

 #box {
        background color : green;
            padding  : 10 px . 10px , 10 px , 10 px //(T,R,B,L
            border   : 5 px  solid black ;
            margin   : 40px ;
            width    : 30px ;
            height   : 10 px ;
 # content {
            background-color : Violet;
          }
</Style>
```

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

Output

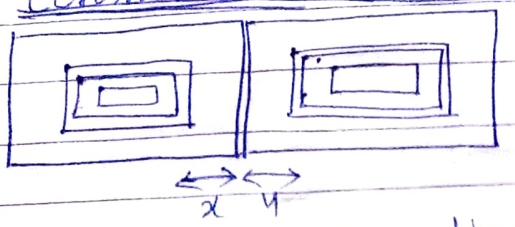


width = 30 + 5 + 5 + 10 + 10 = 60 px
height = 10 + 5 + 5 + 10 + 10 = 40 px

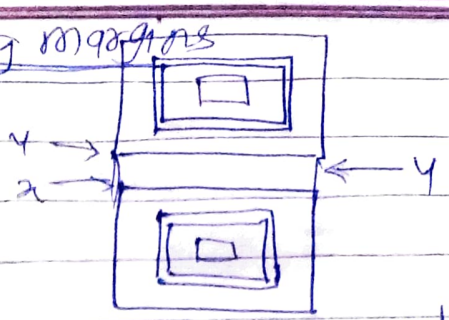
The width is always addition of all element inside element content, bordee and pad as box is defined to content but it can be changed. using property `box-sizing: border-box;` inside that element styling or to parent like body, or * selector for all the element.

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

body {
    box-sizing: border-box;        // for child elements
}

P {
    box-sizing: border-box;        // for specific element
}
```

Commulative and collapsing margins



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

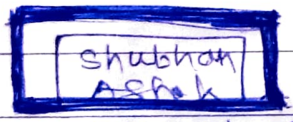


margin on top and below boxes are merged with biggest margin ∴ y>x
margin = y

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 - Default it cause text to overflow out
hidden - It hides the text that don't fit inside
auto - It provide scrollbar whenever neccess.
scroll - It always provides two scrollbar even when not required.



overflow

CSS forms

M T W T F S S
Page No.:
Date:
YOUVA

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 [focus] 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.

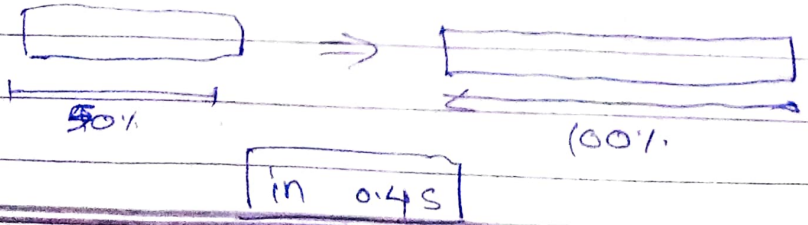
```
input [type=text] {
        background-color : white;
        background-image : url ('searchicon.png');
        background-position : 10px 10px;
        background-repeat : no-repeat;
        padding-left : 40px;
    }
```

5) **Animated search input**

use transition focused properties to animate input when focused.

```
input [type=text] {
        transition: width 0.4s ease-in-out;
        width: 50%;
        padding: 4px;
        background-color: white;
    }

input[type=text] : focus {
        width : 100%;
    }
```

output



6) **Textareas input** - use resize property to decide wheather textareas should be resizable or not.

```
textarea {
        resize: none;
        width: 100%;
        height: 50%;
        }
```

7) Select menus

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

CSS Animation

1) CSS @keyframes Rule - @keyframes defines animation from changing one CSS style to other CSS style gradually with defined [% selectors] or [from to] selectors as follows -

0% start	from (same as 0%)
10% :	:
50% :	:
100% end	to (same as 100%)

Syntax

```
@keyframes animationname {
    keyframe selector {CSS-style}
}
```

2) animation-name - used to apply animation with @keyframes to CSS block in which its specified.

```
animation-name : name of animation ;
```

3) animation duration - duration in seconds.

```
animation-duration : 4s ;
```

4) animation-delay - delay to start animation in seconds.

```
animation-delay : 2s ;
```

5) animation-iteration-count – number of time a repeats.

```
animation-iteration-count: 3;
```

we can set to infinite for forever run of an

6) animation-direction - The direction in which animation to play.

```
animation-direction: normal
                   : reverse
                   : alternate – forward backward
                   : alternate-reverse
```

7) animation timing function - defines animation speed curve

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

8) animation-fill-mode - Style of element when element when the animation is not playing

```
animation-fill-mode: none // no style before after
                   : forwards // keeps last keyframe
                   : backwards // keeps first keyframe
                   : both. // extend property in forward & backward
```

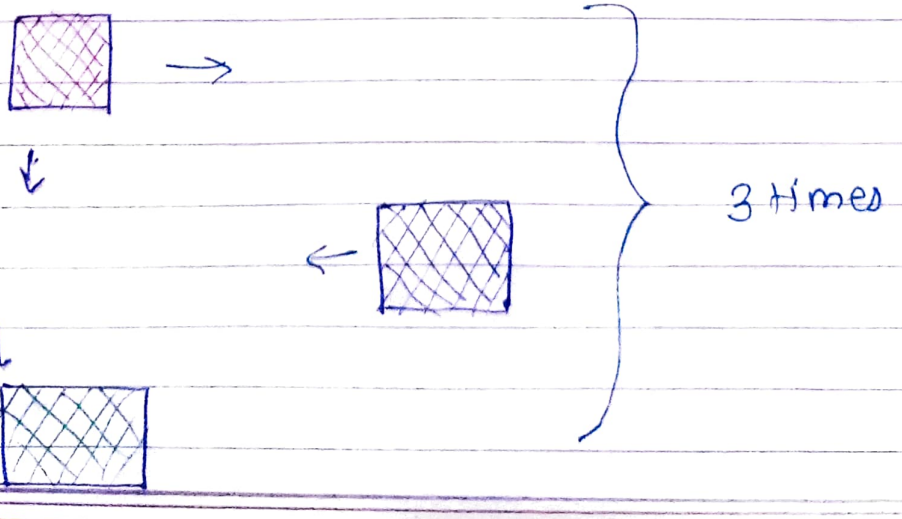
```
div {
    width : 100px;
    height : 100px;
    background-color : red;
    position : relative;
    animation-name : example;
    animation-duration : 4s;
    animation-iteration-count : 3;
    animation-direction : ~~alternate~~;
    animation-delay : 2s;
    animation-timing-function : ease
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

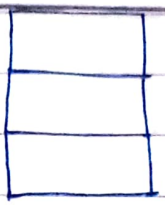
3) 100%



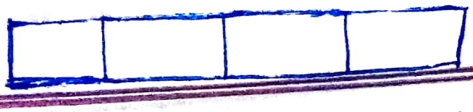
CSS Navigation bar

Navigation bar is a kind of list of URLs.
It has base of html list elements and there are two kinds of Navigation bars.

1) Vertical Navigation 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 whether 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: #4CAE50;
             color : white;
}
```



Horizontal Nav Bar

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 : 14PX 16PX;
    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

M T W T F S S
Page No.
Date:
YOUVA

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.

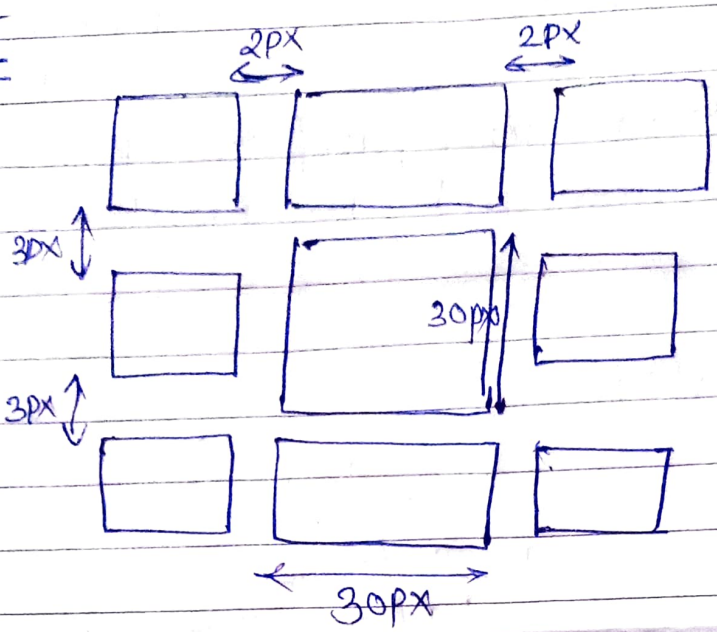
```
.grid-container{
        display : grid;
        grid-column-gap : 2PX ;
        grid-row-gap : 3 PX ;
        //grid-gap : 2PX 3PX ;
        grid-template-columnn : auto 30px auto ;
        grid-template-rows : atuto 30px auto ;
```

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>
```

Output



CSS transitions

M T W T F S S

Page No. | YOUVA

Date

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 ; linear;
ease-in; cubic-bezier(n,n,n,n);
ease-out;
ease-in-out;

CSS Units

M T W T F S S
Page No
Date:
YOUVA

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
% - retivale to parent element