

# NOTES

DATE-10/1/2023

CSS

Note

-subham mohanty

# CSS

## Anatomy of CSS Rule.

<Style>

selector → tag to which style is to apply, here it is paragraph, heading

p {

color : blue;  
font-size : 200px;

}

Properties

Values

colon :

} Declarations which property and value separated by semicolon. ; is not complete

h1 {

color : green;

text-align : center;

}

</style>

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 blue
```

2) Class selector - We define selector as a class with **.classname** selector and this class is assigned to required tags, many classes can be assigned one element as **class="Blue Red"**.

```
.Blue {
```

```
}
```

```
color: blue;
```

```
<p class="Blue">  
<p class="Blue"> --- <p>  
<p> --- <p>  
      ↑  
unaffected.  
      Blue text
```

3) id selector

```
#name {
```

```
    color: blue;  
}
```

```
<p> --- <p>
```

```
<div id="name"> --- </div>
```

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>

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 whether it is direct element or not gets that style.

## Various selector combinations

• Colored p {

color: blue;

}

every element inside elements with class colored gets style

article > .colored {

color: blue;

}

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

## Adjacent sibling Selector

div+p {

background-color: yellow;

}

adjacent sibling is element next to given element at same

<div>

<p> </p>

<div>

<p> </p>

<p>

this element gets it

## General sibling Selector

div~p {

background-color: yellow;

}

All the siblings given element.

<div>

<p> </p>

<div>

<p> — </p>

<p> this gets <

<p> this also gets

## Pseudo-Class Selectors

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

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 released

5) nth-child(x) - Particular 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.

P :: first-letter {  
}

3) before - it is used to insert something before like emoji with Content property

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

4) after - used to 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 [attribute = "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 [attribute ^="Value"]

[ title ^= "flower"]

when title start with flower word flower may not be separated like flowerofflower

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

[ title \*= "flower" ]

when flower word occurred somewhere like in [ ] but may not be separated by space or (-) like [myfloweris] is title.

10

\* 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 = "stylesheet" 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 useful when there are lots of pages to style as particular 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}

#### 3) EXADecimal-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,s,l);
  - 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; right  
left top; left bottom; left  
right center; left center;  
top center; bottom center;

## III) Background Attachment

- i) background-attachment : scroll; background image scrolls  
fixed; background image stays in place.

## IV) Background Shorthand - This property used to specify a background property in single 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 ( $\alpha$ )

## IV)

### Background Shadow

i) text-shadow: 2px 2px 2px rgba(R,G,B, $\alpha$ );

↑              ↑              ↑              ↑  
Horizontal    Vertical    Blurr      Color

Horizontal Vertical

can be negative for upward or left side

ii) box-shadow: 2px 2px 2px rgba(R,G,B, $\alpha$ )

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 (ef).

### Border width

- i) border-width : medium;  
thick ;  
2px ;
- ii) border-width : xpx ypx zpx medium / thick ;  
top right ~~left~~ bottom left.
- iii) border-width : ~~xpx~~ ypx ;  
top bottom

### Border color

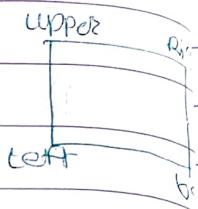
- i) border-color ; color ; hsl(h,s,l,|);  
rgb(R,G,B); rgba(R,G,B,A);  
# FFFFFF;

### Border sides

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

## Border Radius

- i) border-radius : 15px upper right bottom left ;
- ii) border-radius : 15px upper right-left bottom ;
- iii) border-radius : 15px upper-bottom right-left ;
- iv) border-radius : 15px all ;
- v) border-top-left-radius : 2em ;
- vi) border-top-right-radius : 1 ;
- vii) border-bottom-left-radius : 50% ;
- viii) border-bottom-right-radius : 5px ;



## Border Shorthand Property

border : 6px solid red ;

### Order of values

- i) width
- ii) style (compulsory)
- iii) color

## CSS outlines

outline is used to making element stand out by creating outside borders with some spacing.

It has all the properties same as borders.

only one different property as follows :

outline-offset : 2px ;

# CSS Margins Padding & Height, Widths

## Margins

margin : auto;  
 margin : 2px;  
 margin : 2em;  
 margin : 2%;

// browser calculates  
 // length  
 // length  
 // % width

margin - top : auto;  
 margin - bottom : 2px;  
 margin - right : 2em;  
 margin - left : 2%;

margin : 2px, 4px, 2px, 2px;  
 margin : 2px, 4px, 2px, 2px;  
 margin : 2px 4px;  
 margin : 2px  
 all.

## Padding

Same as that of margin.

## Height & width

- i) height : auto; 2px; 2em; 2%; inherit; initial;
- ii) width : auto; 2px; 2em; 2%; inherit; initial;

auto - default value, browser calculate height width

2px - length

2em - length

2% - % of box containing it.

inherit - sets value to default.

initial - Parent's value.

i) min-width: 2px; 2em; 2%; auto; inherit; initial;  
ii) max-width: 2px; 2em; 2%; auto; inherit; initial;  
iii) min-height: 2px; 2em; 2%; auto; inherit; initial;  
iv) max-height: 2px; 2em; 2%; auto; inherit; initial;

max-min property used when resizing is concerned  
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(R,G,B,α); #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 is used for multiple fonts when first font is not supported then next font is 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, serif;
}
```

## Font Style

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

## Font Weight

```
font-weight: normal;
bold;
900;
```

Boldness level [100-900] in multiple of hundred

## 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)

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

## 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

}

a:hover { /\* hover over

}

a:visited { /\* Visited link

text-decoration: none;

}

a:active { /\* clicked li

}

## CSS Lists

### Unordered list

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

list-style-image: url('imageurl');

### 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 etc. 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 : 2px;
- v) height : 2px;
- vi) text-align : left; Right;
- vii) padding : 2px;
- 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 completely  
auto - scrollbar only when necessary.

- ii) overflow-x
- iii) overflow-y

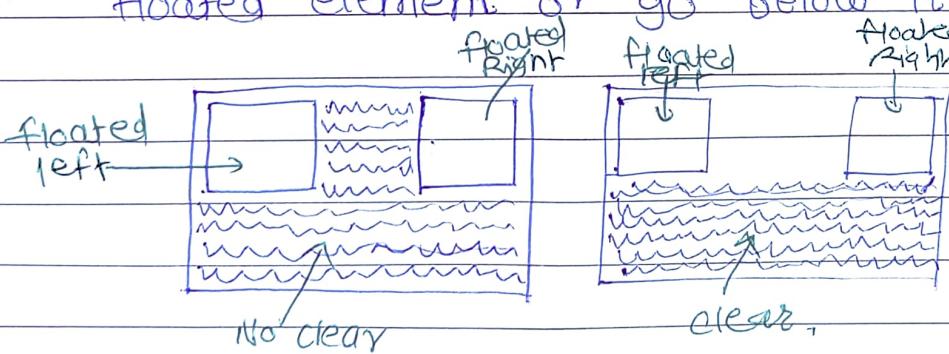
// To handle overflow in x or y directions only.

QUESTION 235

## 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 whether 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 position method used for an element (static, relative, fixed, absolute or sticky)

### Static

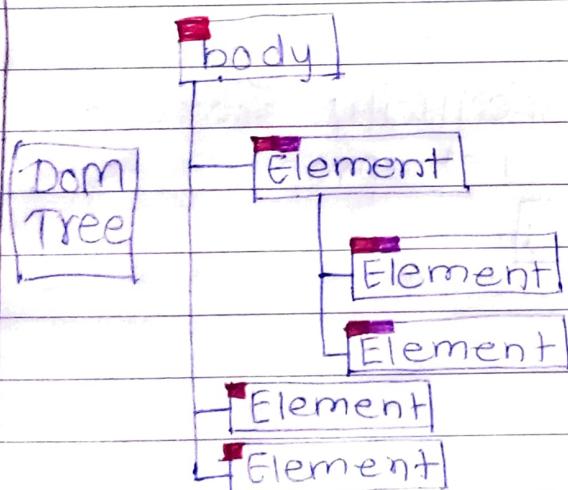
## Conflict Resolution

- 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"

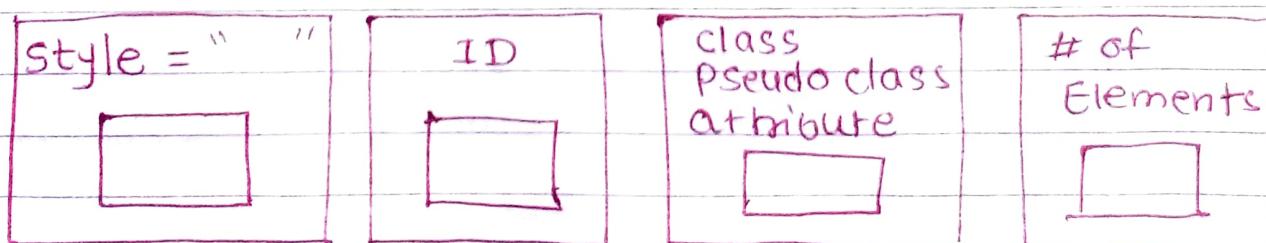
## Inheritance



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 score



div #my para {

color : blue;

}

div.big p {

color : green;

}

style = " "	ID	class	#Element
0	1	0	1

Score = 101

style =	ID	class	#Element
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 e.g. above example will show green color irrespective of their specificity score if we add important tag after property as -

color : green !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**: Arial, 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)** for transparency.

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 **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.

e.g. **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 character

`word-wrap:` `break-word` ⇒ If break word is not fitted in box

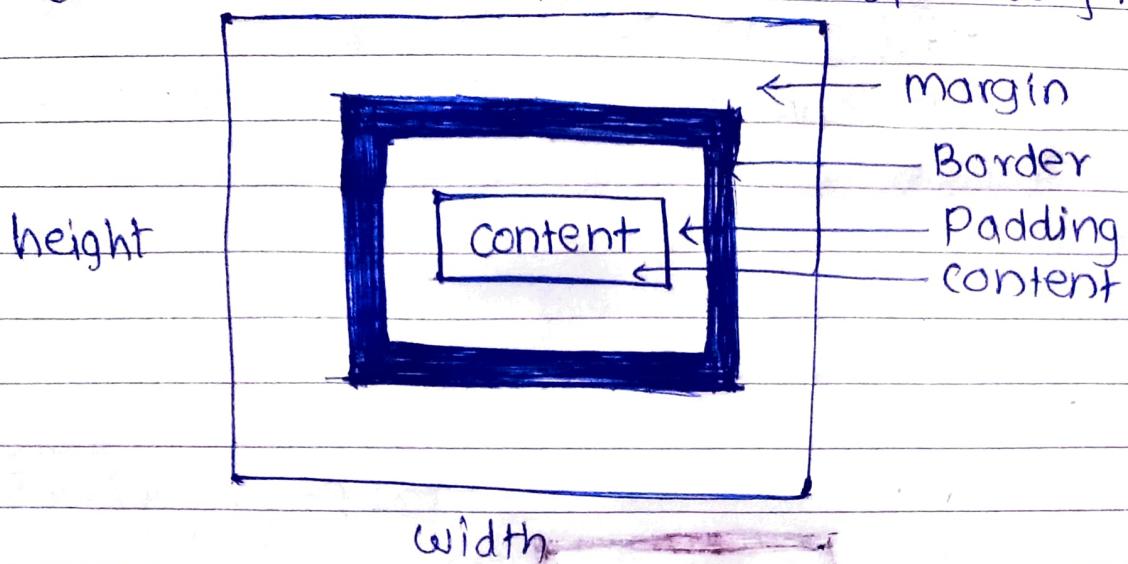
`writing-mode: horizontal-tb;` ⇒ write horizontally  
`vertical-rl;` ⇒ write upside down

```
}
```

## Box Model

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

The Box is the wrapper that wraps every HTML element



$$\text{width} = \text{width} + 2 \times \text{Border} + 2 \times \text{padding} + 2 \times \text{margin}$$

$$\text{height} = \text{height} + 2 \times \text{Border} + 2 \times \text{padding} + 2 \times \text{margin}.$$

<style>

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

#box {

```
background-color : green;  
padding : 10px, 10px, 10px, 10px / (T,R,B,L);  
border : 5px solid black;  
margin : 40px;  
width : 30px;  
height : 10px;
```

#content {

```
background-color : violet;
```

}

</style>

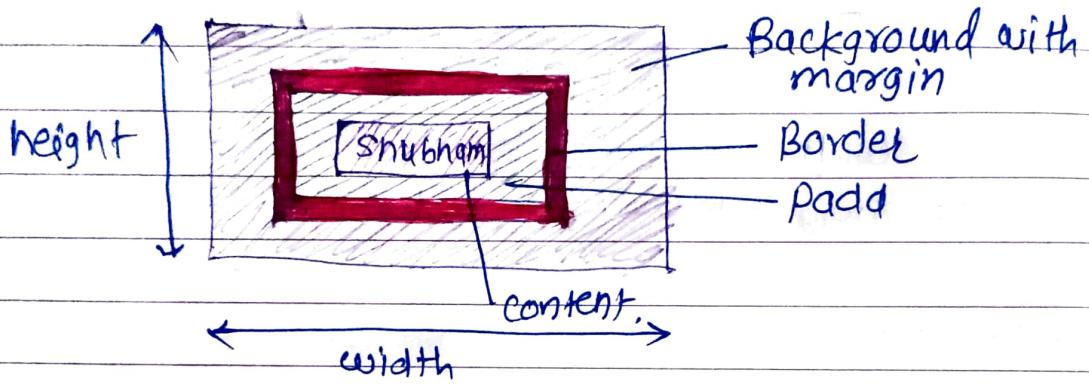
```
<div id="Box">
```

```
  <div id="content">
```

```
</div>
```

Shubham </div>

Output



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

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

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;

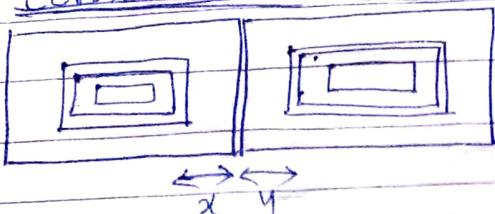
inside that element styling or to parent like body, or \* selector for all the element.

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

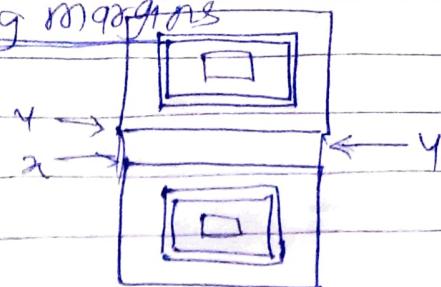
```
body {  
  box-sizing: border-box; // for child elements  
}
```

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

## Commulative and collapsing margins



margin 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 necces.  
scroll - It always provides two scrollbar even when not required.



## 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 [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: 50px;
    padding: 4px;
    background-color: white;
```

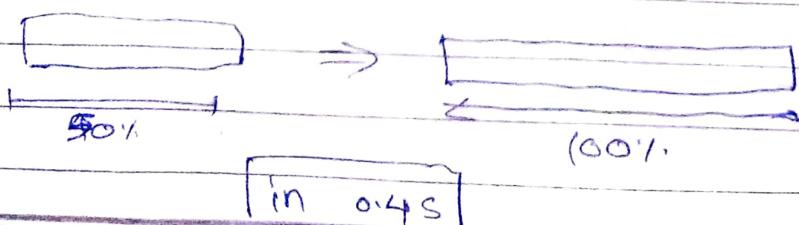
```
}
```

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

```
    width: 100%;
```

```
}
```

#### Output



6)

Textareas input - use resize property to decide whether textareas should be resizable or not.

~~text area~~  
text area {

resize: none;  
width: 100%;  
height: 50%;

{

3) Select menus

select {

width: 100%;  
height: 20px;  
border: 1px solid black;

{

# CSS Animation

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

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
10%
50%
100% : end

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

Syntax

@keyframes animationname {

keyframe {CSS-style}  
selector

}

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 times it repeats.

animation-iteration-count: 3;

we can set to infinite for forever runs of animation.

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

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 speed

: ease-in // slow start

: ease-out // slow end

: ease-in-out // slow start & end

: cubic-bezier (n,n,n,n) // define own curve

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 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;
}
  
```

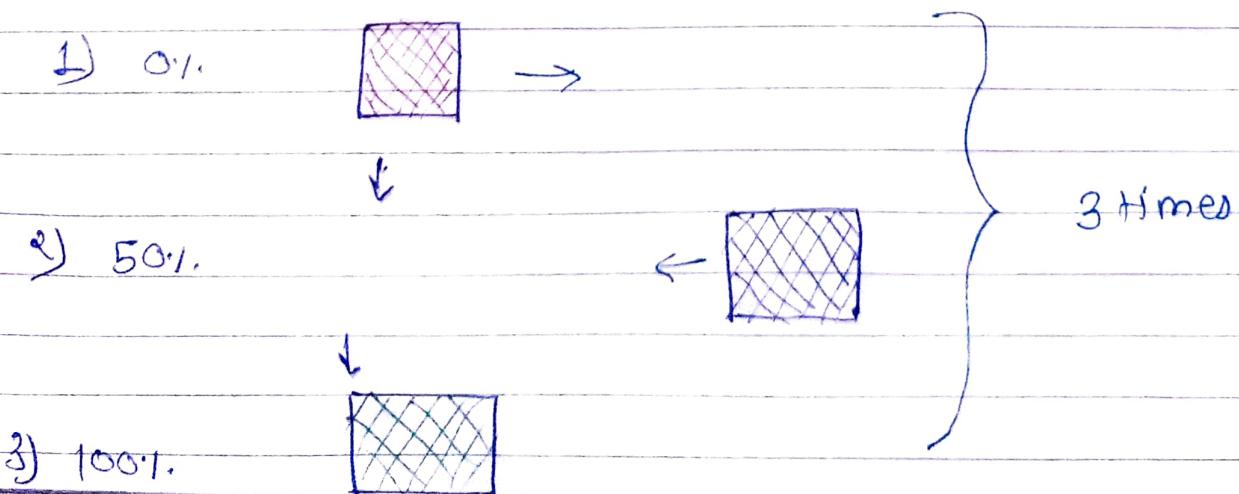
### @keyframes example {

```

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



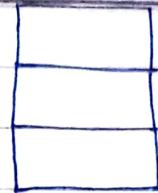
# CSS Navigation bar.

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

Navigation bar is a kind of list of URLs.

It has base of HTML list elements and there are two kinds of navigation bars.

## I) Vertical Navigation bar



Vertical  
NavBar

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 bar with respect to webpage whether it should be fixed or move with page scroll.
overflow: auto;                // overflow defines what should happen with unfitted elements of block.  
(Tabs of Nav bar)
```

}

li a {

```
display: block;                // to make block elements
color: #000000;                // color of text of tab
```

}

li { text-align: center; }

// text aligns in tab

}

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

{

// color of text when hover.

li a: active {

background-color: #CAE5E0;

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 <sup>border</sup>  
 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: 2px;
grid-row-gap: 3px;
// grid-gap: 2px 3px;
grid-template-columns: 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  $3 \times 3$  created.

**.grid-container > div {**

```
background-color: green;
text-align: center;
font-size: 20px;
```

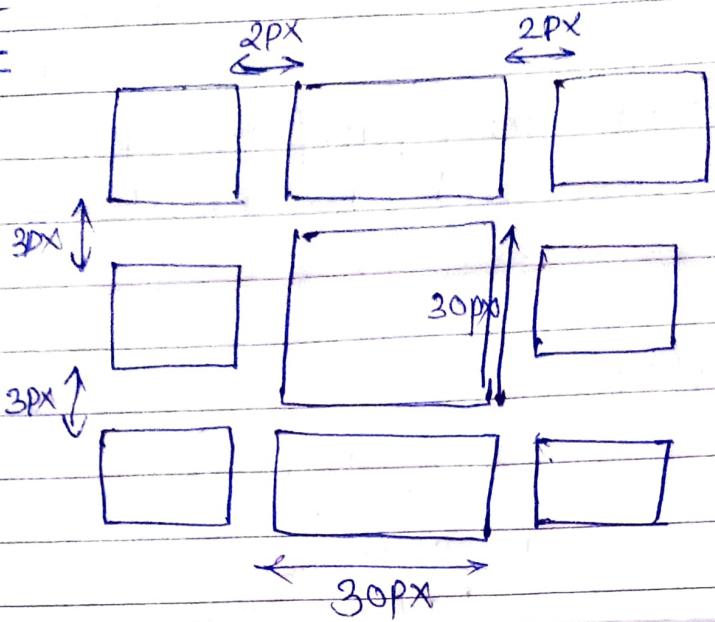
}

<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

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: 100px;
    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(nnn);  
ease-out;  
ease-in-out;

# CSS Units

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

## Absolute lengths

cm - centimeter

mm - millimeter

in - inch

PX - pixels

pt - points

pc - picas

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

pixels are relative to viewing device screen

for low-dpi-device one pixel 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 o width (zero's width)

rem - relative to font size of root element

vw - relative to  $\frac{1}{100}$  of width of viewport

vh - relative to  $\frac{1}{100}$  of height of viewport

vmin - relative to  $\frac{1}{100}$  viewport's smaller dimension

vmax - relative to  $\frac{1}{100}$  viewport's larger dimension

% - relative to parent element