

CSS - Cascading Style Sheet

- * CSS stands for:- Cascading Style Sheet.
- * It is used to apply style to web pages to make web pages more presentable.
- * CSS increases standards overall look for web page that makes easier for the user to interact with web page.
- * CSS invented by Tim Berners Lee and he leaded the W3C (World Wide Web Consortium).
- * CSS was established in 1996 Dec 17.
- * Current version of CSS is 2.1.

* Syntax:-

`<style>`

Selector {

 CSS property : Value ;

 property : Value ;

}

 Declaration.

`</style>`

\$

- * Selector points to HTML Element (Tag) which we want to style. Declaration block contains one or more declaration separated by semicolon.
- * Each declaration contains CSS property and value separated by colon.
- * Declaration blocks are surrounded by curly braces.

<head>

<Style>

```
hi{ color: red; font-size: 50px; }
```

</Style>

</head>

<body>

```
<h1> Good morning </h1>
```

</body>

→ Type of CSS :-

a) Inline CSS

b) Internal CSS

c) External CSS

a) Inline CSS :-

It contains CSS property inside the body section attached with tag it is known as inline CSS

Ex <h1 style="color: red; font-size: 100px;"> HII </h1>

b) Internal CSS:-

The CSS rule laid should be within the HTML file in the head section that is (ie) the CSS is embedded within the html

c) External CSS:-

External contains a separate CSS file that contains only styling properties with the help of link tag we can link CSS file to html.

.html

Style.css

<link>

Syntax:-

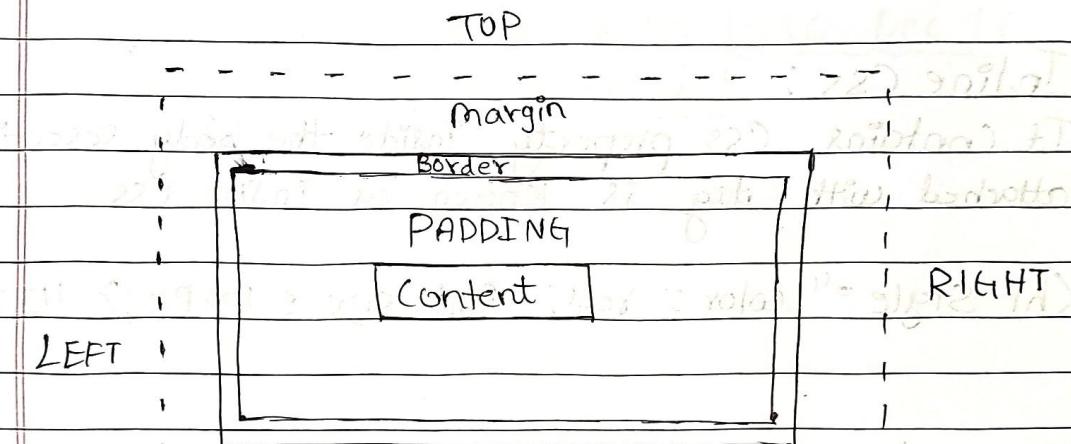
```
<link rel="stylesheet" href="style.css">
```

CSS Box model:-

- * ~~Css box model is a box that wraps~~

- * It consists of margin, padding, borders and actual Content

- box model illustration:-



Content:-

The Content of the box where text, images etc. appear

Padding:-

It clears the area around the Content and it is transparent

Border:-

Border goes around padding and content and it is visible.

Syntax:-

border : valuepx color Solid;
Ex border: 3px red Solid;

Margin:-

It clears the area outside the border and it is transparent.

Padding and margin takes the values in top, right, bottom and left (clockwise direction)

Shadow to the Box:-

Syntax:- (outside) of the box)

Box shadow : value1 value2 value3 color name;

Value 1:- Applying shadow to the right of the box

Value 2:- Applying shadow to the bottom of the box

Value 3:- Applying shadow & blur effect to the shadow

Syntax:- (inside) the box)

Box shadow: value1 value2 value3 color name inset;

b) Class Selector:-

- This Selector Selects the html Element with Specific class attribute
- It uses "(.) period" to Select on class attribute followed by class name
- Class name can be duplicated.

Syntax:

• Classname {

}

Ex

<Style>

• title {

color: colorname;

}

</Style>

<body>

<h1> class="title" </h1> This is an Example
(class name) for class <h1>

</body>

c) Id Selector:-

- Id Selector uses the id attribute of an html Element to Select a specific Element
- It uses "Hash(#)" to select an id attribute followed by idname
- Id should be Unique in web page

Syntax:

#idname {

}

Ex `<Style>`

```
#head {  
    color: colorname;  
}
```

`</Style>`

`<body>`

```
<h1 id="head"> Good morning <h1>  
</body>
```

d) Universal Selector:-

- This selector selects all the html Elements on the page
- To select all Elements we write `* (Star)` character

Syntax:-

```
* {  
    CSS:Property;  
}
```

Ex `<Style>`

`*`

`Color: colorname;`

`background color: colorname;`

`}`

`</Style>`

`<body>`

`<h1> Hello </h1>`

`<h2> Good </h2>`

`<h3> morning </h3>`

`</body>`

Opacity:

- It defines opacity or transparency of an selected element
- It takes values from ~~0.0~~ to 1.0

~~Ex~~ Syntax
div {

 opacity: 0.0 to 1.0

}

Ex

<Style>
div {

 height:

 width:

 background-color:

 opacity:

}

</Style>

<body>

 <div> </div>

</body>

CSS Combinators:-

- Combinators are explaining the relationship b/w Selector.
- CSS Selector are pattern to select an HTML Element for styling
- We connect more than one Selector using combinator.
- Type of Combinators :-
 - (i) Descendent Selector (Space)
 - (ii) Child Selector (> gt)
 - (iii) Adjacent sibling Selector (+)
 - (iv) General sibling Selector ('n' squiggly)

(i) Descendent Selector:-

- Descendent Selector matches all the elements that are descendent of specified Element. It uses Space of after selecting first Selector.

Ex <style>

div p {

 color: color name;

}

</style>

</head>

<body>

 <h1> CSS Combinators </h1>

 <div>

 <h1>

 <p> This is inner paragraph </p>

 <h1>

<h2> This is heading h2 </h2>
</div>

<p> This is outer paragraph </p>
</body>

(ii) Child Selector

- It Select all the Elements that are children of Specified Element.
- It uses greaterthan character to combine the selectors.

Ex <style>

div>p{

color: color name;

background: color name

}

</style>

<body>

<h1> This is boy </h1>

<h1> This is boy </h1>

<p> This is girl </p>

<h1> This is boy </h1>

<p> This is girl </p>

<p> This is girl </p>

</body>

3) Adjacent Sibling Selector:

- It is used to select an element that is directly after another specified element.
- Adjacent means immediately after following.
- It uses "+" character to combine the selector.

h1 + h2 {

color: colorname;

background-color: colorname;

}

<h1> This is heading one </h1>

<h2> This is heading two </h2>

<h3> This is heading three </h3>

<p> This is paragraph </p>

<h3> This is heading two </h3>

<h2> This is heading three </h2>

- The whole example h3 tag is placed immediately after h1 tag so this satisfied our CSS rule.

4) General Sibling Selector

- This Element selects all the Elements that are next sibling of Selected Element.
- It uses "Squiggle" (~) to combine the selector.

<Ex>

<Style>

h1 ~ h3 {

color : colorname;

}

</Style>

<body>

<h1> heading 1

<p> paragraph 1

<h3> heading 3

<h2> heading 2

<p> paragraph2

<h3> heading 3

<h2> heading 2

<body>

CSS Pseudo-class :- (:)

A Pseudo class is used to define special state of an Element.

Uses of Pseudo class :-

- Style an Element when a user mouse hover it
- Style links , visited links , unvisited links
- Style an Element when it get focused
- Type of pseudo class :-

(i) ~~link~~ link :- It is used to style the link on the web page.



<body>

 Google

<body>

```
<style>
```

```
a:link {
```

```
color: colorname;
```

```
}
```

```
</style>
```

Nar
Name
Ay

ii) Active :- It is used to select and style active link

→ A link becomes active when user clicks on it

```
<style>
```

```
a:active {
```

```
color: colorname;
```

```
}
```

```
</style>
```

```
<body>
```

```
<a href="https://facebook.com">Facebook</a>
```

```
</body>
```

iii) Visited :- (i) A link changes the style when the user visited the link

(ii) When user clicks on link it becomes visited

```
<style>
```

```
a:visited {
```

```
color: colorname;
```

```
}
```

```
</style>
```

```
<body>
```

```
<a href="https://Instagram.com">Instagram</a>
```

```
</body>
```

iv) Hover:- It is used to style an element when user takes mouse over it.

Hover can be used on all html Elements.

```
a:hover {  
    color: colorname;  
    font-family: ;  
}
```

`twitter`

v) Focus:- It is used to select the element that has focus.

This Selector is allowed on the Elements that accept Keyboard Events or user input.

User

```
input:focus {  
    color:  
    background-color:  
}
```

Username : `<input type="text">

`

password : `<input type="password">`

(vi) First-child:- This selector is used to select the specified Selector only if it is the first child of its parent.

```
p:first-child {  
    color:  
}
```

`<p> Paragraph </p>`

`<p> paragraph </p>`

`<p> paragraph </p>`

vi) Last-child :- This selector is used to select the specified selector only if it is the last child of its parent

p: last-child {
color:
}

<P> Paragraph </P>
<P> Paragraph </P>
?a. <P> Paragraph </P>

vii) nth-child :- This selector is used to select the specified selector by calling the which child-selector we want in the parentheses {} by using number, even, odd

p: nth-child (3) {
color:
}

<P> Hello </P>
<P> Hello </P>
<P> Hello </P>
~~~~~

Even: Even number children are Selected

p: nth-child (even) {  
color:  
}

## Pseudo Elements :- ( :: )

- \* Pseudo Elements are used to style specified part of an Element
- \* for Example styling the first letter or first line of an Element.
- \* Insert content before or after an Element

### Syntax :-

```
Selector :: pseudo Element {  
    / / or property ;  
}
```

### \* Types of pseudo Elements :-

- (i) first letter :-
- (ii) first line
- (iii) Before
- (iv) After
- (v) Selection

(i) first letter :- It is used to add special stylings to the first letter of text

Example:-

```
h1:: first-letter {  
    color : colorname  
}
```

<h1> PySpiders </h1>

Output:- **P**ySpiders

2) ::First-line :- This Element applies style to the first line of block level Element.

\* Example:-

```
P:: first-line {  
    color: colorname;  
}
```

<P> This Element applies style to the first line(<P>)

O/P:- THIS Element applies style to the first line.

3) ::Before :- This Element is used to insert Content just before the Selected Element

Ex. h2:: before {  
 content: "PySpider";  
 color: colorname;  
}

<h2> Element </h2>

O/P:- PySpider Element.

4) :: After :- This Element is used to insert Content just after the Selected Element

```
h3:: after {  
    content: " Elements";  
    color: Black;  
}
```

<h3> PySpider </h3>

O/P :- PySpider Elements

5) :: Selection:- This element matches the portion of an Element that is selected by user.

```
#Parea :: Selection {  
    background: color;  
    color: colorname;  
}
```

`<h3>` This element matches the portion of an Element `</h3>`

Display property :-

- a) Block :- Displays an Element as Block content
- Always it starts on new line
- It takes ~~to~~ whole width of the window
- ~~We can apply~~ we can apply height and width.

Syntax :-

Selector {

```
    display: block;  
}
```

b) Inline :-

- It displays Elements as in line Element.
- Height and width properties will have no Effect

Syntax :-

Selector {

```
    display: inline;  
}
```

h1, h2 {

display: inline;

height: 50px; } No Effect.

width: 100px; }

}

<h1> Good morning </h1>

<h1> Have a nice day </h1>

<h2> Happy holi </h2>

### c) In-line block:-

- It displays an Element as in-line level container, the Element itself is forwarded as an inline Element.
- We can apply height and width to the Element.

#### Syntax:-

Selector {

    display: inline-block;

### d) Flex:-

- It displays Elements as block level flex container
- To apply flex to Element we need to create parents and children

Syntax :- {Display : flex;}

Ex

.container {

    display: flex;

    height: 100px;

    width: 100%;

}

<div class="container">

    <div class="box1"></div>

    <div class="box2"></div>

    <div class="box3"></div>

</div>

## flex properties:-

- \* `justify-content` : `flex-start`;  
  : `flex-end`;  
  : `space-around`;  
  : `space-between`;  
  : `space-evenly`;  
  : `center`;
  
- \*  `: baseline;  
  : center;  
  : flex-start;  
  : flex-end;`
  
- \* `flex-direction`: `column`;  
  `column-reverse`;  
  `row`;  
  `row-reverse`;
  
- \* `flex-wrap` : `wrap`;  
  `nowrap`;  
  `wrap-reverse`;

## Position :-

- This property defines the methods of positioning to an HTML Element.
- Positions takes the value in top, right, bottom and left.
- There are five types in position
  - (i) Static
  - (ii) absolute
  - (iii) Relative
  - (iv) Sticky
  - (v) Fixed

(i) Static:- It is default position property

(ii) Absolute:- Position of an Element with respect to parent Element

# Selector {

```
Position: absolute;  
top: 0;  
left: 0;  
}
```

(iii) Relative:- Position of an Element with respect to its original position.

# Selector {

```
Position: relative;  
top: 100px;  
left: 200px;  
}
```

(iv) Sticky :- Position of Element with respect to its parent Element  
in      **# Selector {**

Sticky condition

Position : Sticky ;  
top : 100px ;  
left : 1200px ;

(v) fixed : Position of Element can be placed anywhere with respect to window.

- We can fix the Selected Element wherever we want inside the window

**# Selector {**

position : fixed ;  
top : 100px ;  
left : 1200px ;

## CSS Transform:-

- It is used to apply 2d or 3d transformation to an selected Element
- This property allows us to rotate, move, scale, skew etc
- Type of transform :-

a) Rotate

b) Skew

c) translate

d) Scale

a) Rotate:-

Example:-

Card: hover {

transform: rotate(360deg);

transform: rotateY(360deg);

rotate: 360deg;

}

Scale:-

b) ~~Skew~~:- It is used to expand selected Element

• It takes the value more than 1.0

Card: transform: scale(1, 2);

transform: scale();

d) translate:- Change the Element to given value.

transform: translate(-500px);

transform: translateY(500px);

transform: translateZ(500px);

transform: translate(150px, 150px);

c) Skew:- It makes the element look like 3d it rotates the element like top and bottom.

`transform: skew(30deg);`

### CSS Animation:-

- (i) Animation using from and to keyword:-
- Animation is used to animate all properties with more control.
- we can use "@Keyframes" rule to change the animation from given style to new style

### Syntax:-

```
@ Keyframe animation-name {
    from {
        //CSS - property
    }
    to {
        //CSS - property
    }
}
```

### Properties to add animation :-

Animation-name : Name of the animation (User define)

Animation-duration : Time taken by the animation

Animation-timing-function : It defines speed curve of animation.

- It takes the values "ease", "ease-in", "ease-out" and "ease-in-out"

Animation-delay :- Delay in starting the animation

Animation-iteration-count :- No of times an animation should run.

Animation direction : (i) It specifies direction of animation

\* Animation short hand property :-

Syntax:-

Animation: name, duration, timing-function, delay,  
iteration-count, direction;

Example:-

animation: anim 5s ease 1s infinite normal;

(ii) Animation using (%) Percentage value:-

We can use % value to indicate what should happen when a certain percent of animation is completed

Syntax:-

@Keyframe animation-name {

0% {

// CSS property

}

50% {

// CSS property

}

100% {

// CSS property

}