Css :-

* Css stands for :- Cascading Style Shape.
* It is used to apply style to web pages to make web pages has 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 lea and he Leaded the
* Css invented by Company w3c (world wide
* web Consorhum
* Css was Established in 1996 Dec 17
* Current version of Css is 2.1
* Syntax :-

```
<style>
Selector {
        Css property : value;
        property : value;
}
</style>
```

Declaration.

* Selector points to html Element (Tag) which we want to style. decleration block contains one or more decleration Seperated by Semicolun,
* Each decleration contains Css property and value Seperated by Colann
* Decleration blocks are surrounded by Curleybraces

</head>
Ex <Style>
h1 { color: red; font-size: 50px;
}
</Style>
</head>
<body>
<h1> Good morning </h1>
</body>

→ Types of Css :-

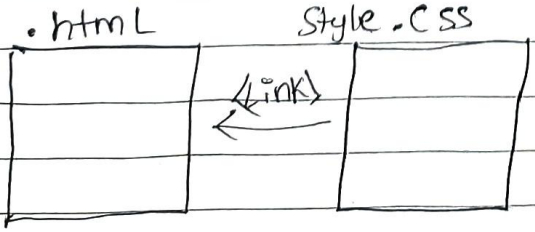
From a) Inline Css
b) Internal Css
c) External Css

a) Inline Css :-
It Contains Css property inside the body sesection attached with tag is known as inline Css

Ex <h1 Style = " color: red; font size: 100 Px;> HII </h1>

b) Internal Css :-
The Css rule said should be with in the HTML file in the head Section that is (ie) the Css is Embeded with in the html

c) External Css :-
External Contains a Seperate Css file that Contains Only styling properties with the help of link tag we can link Css file to html.



Syntax:-

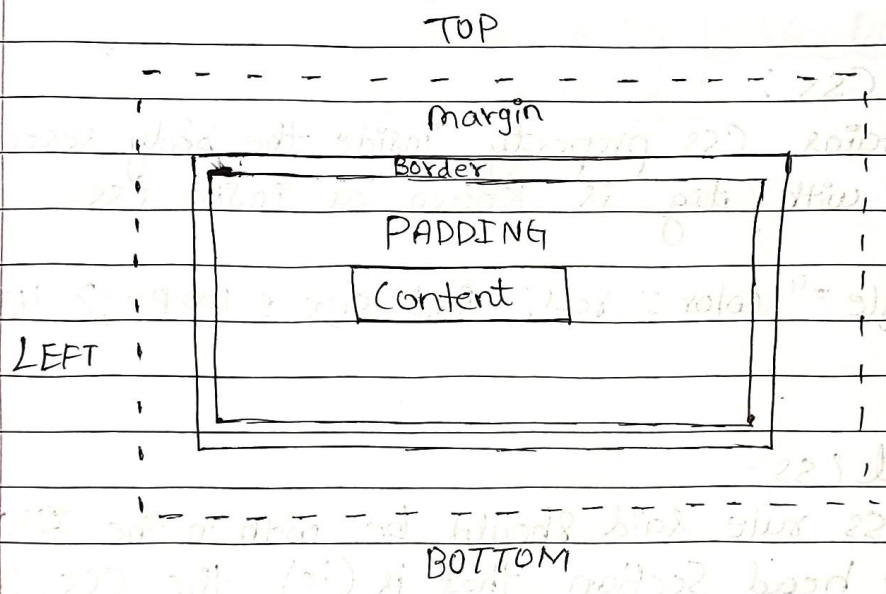
```
<link rel="stylesheet" href="style.css">
```

Css Box model:-

* Css box model is a box that wraps

* It consist of margin, padding, borders and actual Content

- box model illustration:-



Content:-
The Content of the box where text, images vedios appear

Padding:-
It clears the area around the Content and it is transperent

classmate
Date
Page

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 transperent.

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 colorname;
```

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 of the box)

```
Box shadow: value1 value2 value3 Colorname inset;
```

b) Class selector:-

- This selector selects the html element with specific class attribute
- It uses "(.) period" to select on class attribute followed by classname
- Class name can be duplicated.

Syntax:

```
.classname {

}
```

Ex

```
<style>
    .title {
        color: colorname;
    }
</style>
<body>
    <h1 class="title"> This is an Example (classname) 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> Hellow </h1>
    <h2> Good </h2>
    <h3> morning </h3>
</body>
```

Opacity:

- It define opacity or transperency of an selected element
- It takes value from 0.0 to 1.0

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
- Types of Combinators :-

(i) Descendent Selector(_ Space)
(ii) Child Selector (> gt)
(iii) Adjacent Sibiling Selector (+)
(iv) General Sibiling Selector. ('~' 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 Sibiling Selector:-

- it is used to select an Element that is directly after another specified Element
- Adjacent means immediately after following
- it used "+" 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>
 This is paragraph
<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 ruleset

4) General sibiling Selector

- This Element selects all the Elements that are next sibiling of Selected Element
- It uses "Squiggly" (~) 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> paragraph 2
    <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 :-

a) Style an Element when a user mouse hover it
b) Style links, visited links, unvisited links
c) Style an element when it get focused
d) Types of pseudo class :-

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

```
<body>
    <a href = "https// www.google.com"> google</a>
</body>
```

```
<style>
    a:link {
        color: colour nam;
    }
</style>
```

ii) Active :- → It is used to select and style active link
→ A link becomes active when user clicks on it

```
<style>
    a:active {
        color: colornam;
    }
</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 ~~the~~ 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:      ;
    }
```

```
<a href " https: // "www.twitter.com" > twitter </a>
```

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

This selector is allowed on the Elements that accepts Keyboard Events or user input

~~User~~

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

```
Username : <input type = "text" }<br> <br>
password : <input type = "password >
```

(vi) First-child:- This selector is used to select the specified selectors 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>
<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 parenthies & by using number, Even, odd

```
P: nth-child (3) {
        color:
        }
```

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

Even: Even number childs 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 {
        // css 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 :- Pyspiders

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

* Example:-

```
P:: first leter {
        color: colorname;
        }
```

<P> This Element aplies styles to the first line</P>

O/P:- THIS Element applies Styles 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:- Pyspiders 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

```
#para::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 whole width of the window
- 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: in line;
    height: 50px;
    width: 100px;
}
```
No Effect.

<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 ~~formatted~~ formated 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 Evenely;
: Center;

* align-items : baseline;
: Center;
: flex start;
: flex End;

* flex-direction: Column;
Column-reverse;
row;
row-reverse;

* flex-wrap : wrap;
nowrap;
wrap-reverse;

Position :-

- This property define 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: 200 px;
    }
```

(iv) Sticky:- Position of Element with respect to its parent Element in Sticky condition

```
#Selector {
        position : sticky ;
        top : 100px;
        left : 1200px;
```

(v) fixed: Position of Element can be placed ~~any~~ 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
- Types 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:- Drags the Element to given value.

```
transform: translate(-500px);
transform: translateY(500px);
transform: translateZ(500px);
transform: translate(150px, 150px);
```

(4) 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 keywords:-

- Animation is used to animate CSS 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 as "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, dely,
           iteration-count, direction;
```

Example :-

```
animation: anmi 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 :-

```
@keyframes animation-name {
    0% {
        // CSS Property
    }
    50% {
        // CSS property
    }
    100% {
        // CSS property
    }
}
```