CSS

🡪CSS stands for cascade style sheets.

🡪useful to create the responsive webpages.

🡪useful to style the websheets.

🡪provides equality for the both web browsers and devices.

🡪we can add colors, animations, fonts,etc.. to the webpages freely.

**Syntax**

Selector

{

Property1:value1;

Property2:value2;

And so on...

}

🡪selector will points to the element of HTML.

🡪Multiple declarations are separated by ‘;’ and those are enclosed in {…..}.

**Example**

P

{

Text-align:left;

color:red;

Font-size:100px;

}

**CSS SELECTORS**

🡪 selectors are based on the elements of HTML.

**Element Selectors**

🡪 Element selectors will be applied on elements like <p>, <body>.

**Example**

P

{

Text-align:center;

color:red;

}

🡪 all elements of <p> inside <p>…..</p> text is aligned to center and color is red.

**Id selector**

🡪 id will be used to point the unique element in HTML.

🡪 id will be unique on a webpage. we can select the element with ‘#’ along with id.

🡪 **Syntax**:#id1

**Example**

#id1

{

Text-align:center;

color:red;

}

**Class Selector**

🡪 class selector will be used to point the class element in HTML.

🡪 we can select the element with ‘.’ along with id.

🡪 **Syntax**:#id1

**Example**

.perk

{

Text-align:center;

color:red;

}

🡪here all the classes with perk are affected with this.

🡪we can make only specific element affected by class.

**Example**

p.class

{

Text-align:center;

color:red;

}

🡪 in the above example only element p contains class will be affected.

🡪HTML Elements are also affected by multiple classes.

**Example:** <p class=”cla,clb”>This is a paragraph</p>

🡪in the above example both the classes are affected for <p>

Universal Selector

🡪universal selector applies to all elements of HTML.

🡪 represented by ‘\*’.

Example:

\*

{

Text-align:center;

color:red;

}

**Group Selectors**

🡪 group selectors are used when we same set of styling for for more than one element.

🡪 To group selectors, all the elements are separated by ‘,’.

**Example**

h1,h2,p

{

Text-align:center;

color:red;

}

**How To Add CSS**

**🡪**there are three ways to add css I.e internal, external, inline.

**External**

**🡪** writing css code in a separate file and include in HTML using <link> Tag.

🡪 File must be saved in .css format and should not contain any HTML Tags

**Example**

**<link rel=”stylesheet” href=”m1.css”>**

**In m1.css**

p

{

Text-align:center;

color:red;

}

**Internal**

**🡪** if one webpage has one unique styling then we use internal i.e within HTML page.

🡪 css block will be enclosed in <style>…</style>

**Example**

<!DOCTYPE html>  
<html>  
<head>  
<style>  
body {  
  background-color: linen;  
}  
  
h1 {  
  color: maroon;  
  margin-left: 40px;  
}  
</style>  
</head>  
<body>  
<h1>This is a heading</h1>  
<p>This is a paragraph.</p>  
</body>  
</html>

**Inline style Sheet**

🡪in this case, styling is applied within element.

**Example**:<p style=”align:center;”>This is a paragraph</p>

The border-style property specifies what kind of border to display.

|  |  |
| --- | --- |
| **Style** | **Description** |
| Dotted | Defines a dotted border |
| Dashed | Defines a dashed border |
| solid | Defines a solid border |
| double | Defines a double border |
| groove | Defines a 3D grooved border. The effect depends on the border-color value |
| ridge | Defines a 3D ridged border. The effect depends on the border-color value |
| inset | Defines a 3D inset border. The effect depends on the border-color value |
| outset | Defines a 3D outset border. The effect depends on the border-color value |
| hidden | Defines a hidden border |
| none | Defines no border |

**Example**

p.dotted {border-style: dotted;}  
p.dashed {border-style: dashed;}  
p.solid {border-style: solid;}  
p.double {border-style: double;}  
p.groove {border-style: groove;}  
p.ridge {border-style: ridge;}  
p.inset {border-style: inset;}  
p.outset {border-style: outset;}  
p.none {border-style: none;}  
p.hidden {border-style: hidden;}  
p.mix {border-style: dotted dashed solid double;}

**CSS border width**

🡪 To specify the width of the border can be done by border-width.

🡪 Default sizes are thick, thin,medium.

**Example**

border-width: medium;

border-width: thick;

border-width: thin;

**CSS border color**

**🡪** we can specify the colors of the border based on RGB, RGBA, HEX, HSL.

**Example:**

**P**

**{**

**Border-color:red;**

**}**

**🡪** wecan also have an option for specific side color(top, right,bottom,left).

**Example:**

p.one

{  
  border-style: solid;  
  border-color: red green blue yellow; /\* red top, green right, blue bottom and yellow left \*/  
}

**CSS Border**

**🡪** we can apply specific direction to the specific direction.

🡪 left, bottom, right, left.

**Example**

p

{  
  border-top-style: dotted;  
  border-right-style: solid;  
  border-bottom-style: dotted;  
  border-left-style: solid;  
}

P

{

Border-style: dotted solid dotted solid;

}

**CSS Margins**

**🡪** Margins are used to create the space around elements outside borders.

**Example**

P

{

margin-left:100px;

margin-right:50px;

margin-bottom:60px;

margin-left:23px;

}

**Shorthand**

P

{

margin: 100 50 60 23;

}

Margin Collapse

🡪 sometimes, margins can be collapsed into a single margin.

🡪 Top and bottom margins of elements are sometimes collapsed into a single margin that is equal to the largest of the two margins.

🡪 This does not happen on left and right margins! Only top and bottom margins.

**Example**

h1 {  
  margin: 0 0 50px 0;  
}  
  
h2 {  
  margin: 20px 0 0 0;  
}

**PADDING**

🡪 padding is used to generate around element.

🡪 padding can be done in 4 directions(top, right, bottom, left).

🡪 values are represented in the form of px, pd, %, inherit.

**Example**

P

{

padding-left:10px;

padding-right:10px;

padding-top:10px;

padding-bottom:10px;

}

In one shorthand

P

{

padding: 10px 10px 10px 10px;

}

**CSS Height And Width**

**🡪** height and width are used to set height and width of an element.

🡪 it doesn’t includes borders, margins, padding. It sets height and width inside borders, margins, padding.

**Example**

P

{

width:200px;

height:100px;

}

🡪 we can use max-width to set maximum width of an element.

🡪 for example, if we use both width and max-width within same element, max-with will be used as max-width dominates the width. finally, width won’t apply to the element.

**Example**

P

{

width:200px;

height:100px;

max-width:150px;

}

**CSS BOX Model**

🡪 box model wraps around HTML pages. It includes content, padding, border, margins.

🡪 **Content** - The content of the box, where text and images appear

**Padding** - Clears an area around the content. The padding is transparent

**Border** - A border that goes around the padding and content

**Margin** - Clears an area outside the border. The margin is transparent

**Example**

div {  
  width: 300px;  
  border: 15px solid green;  
  padding: 50px;  
  margin: 20px;  
}

**CSS OUTLINE**

🡪 Outline is the line that is drawn outside the borders to make the element stand out.

🡪 style are as same as border styles.

Example

p.dotted {outline-style: dotted;}  
p.dashed {outline-style: dashed;}  
p.solid {outline-style: solid;}  
p.double {outline-style: double;}  
p.groove {outline-style: groove;}  
p.ridge {outline-style: ridge;}  
p.inset {outline-style: inset;}  
p.outset {outline-style: outset;}

**OUTLINE WIDTH**

**🡪** outline-width is used to set the width of outline.

🡪 usually represented in fixed like thin(1px), medium(3px), large(5px) and specific value and units will be ‘%’, ‘px’, ‘cm’, ‘em’.

**OUTLINE COLOR**

**🡪** outline-color is used to set the color of outline.

🡪 colors are based on RGB, HEX, HSX, HSXA, invert.

**Example**

p.ex1 {outline: dashed;}  
p.ex2 {outline: dotted red;}  
p.ex3 {outline: 5px solid yellow;}  
p.ex4 {outline: thick ridge pink;}

Shorthand

p.ex1 {outline: dashed;}  
p.ex2 { outline: dotted red;}  
p.ex3 {outline: 5px solid yellow;}  
p.ex4 {outline: thick ridge pink;}

**Outline-offset property**

🡪 The outline-offset property adds space between an outline and the edge/border of an element. The space between an element and its outline is transparent.

🡪 The following example specifies an outline 15px outside the border edge:

**Example**

p {  
  margin: 30px;  
  background: yellow;  
  border: 1px solid black;  
  outline: 1px solid red;  
  outline-offset: 15px;  
}

**Text Color**

**🡪** used to color the text for an element.

**🡪** Background color is used to set the background of the color.

**Example**

body

{  
  background-color: lightgrey;  
  color: blue;  
}

**Text Alignment**

**🡪 Text Alignment is used to algn the text. Attributes are left, right, center, justified.**

**Example:**

h1 {  
  text-align: center/left/right/justify;  
}

**Text Direction**   
 **🡪** useful to set the direction of the text.

🡪 text-direction, Unicode-bidi used to set the direction of text.

**Example**

p

{  
  direction: rtl;  
  unicode-bidi: bidi-override;  
}

**Vertical Aligment**

**🡪**Usedto align vertically for an element.

**Example**

img.top {  
  vertical-align: top;  
}

**Text Decoration**

**🡪 used to set or remove decorations from text.**

**Example**

a {  
  text-decoration: none/ line/overline/linethrough;  
}

🡪 none—No Effect

underline—Below Text

overline—above Text

Linethrough—Strike off  
**Text Transformation**

**🡪 Text** transformation is used to specify uppercase/lowercase/capitalize

**Example**

p.uppercase {  
  text-transform: uppercase/lowercase/capitalize;  
}

**Text Indendation**

🡪 text-indent property is used to specify the indent of first page.

**Example**

P

{

text-indentaion:50px;

}

**Letter Spacing**

**🡪** Letter spacing is used to represent the space between the characters.

🡪attribute value will be positive or negative. positive increases the space and negative decreases the space.

**Example**

h1 {  
  letter-spacing: 9px;  
}  
  
h2 {  
  letter-spacing: -2px;  
}

**Line Spacing**

**🡪** Linespacing is used to specify the space between thelines**.**

**Example**

p.small {  
  line-spacing: 0.8;  
}

**Word Spacing**

**🡪** word spacing is used to specify the word between thelines**.**

**🡪** attribute value will be positive or negative. positive increases the space and negative decreases the space

**Example**

p.small {  
  word-spacing: 0.8;  
}

**White spacing**

**🡪** White spacing property is used to manage the white spaces.

🡪 below example shows how to disable text wrapping.

**Example**

P

 {  
  white-space: nowrap;  
}

## Text Shadow

🡪 The text-shadow property adds shadow to text.

🡪 In its simplest use, you only specify the horizontal shadow (2px) and the vertical shadow (2px), blur

**Example**

h1 {  
  text-shadow: 2px 2px 5px red;  
}

**Font Size**

**🡪** Choosing the right font catches the viewer attraction.

🡪 There are several font families like **Serif**, **Sans-serif**, **Monospace**, **Cursive**, **Fantasy**

**Example**

.p1 {  
  font-family: "Times New Roman", Times, serif;  
}

**Font Style**

**🡪** used to specify the style of the font.

🡪 attributes are none(normal), italics(italic), oblique(leaning).

Example:

p.normal {  
  font-style: normal/italics/oblique;  
}

**Font weight**

**🡪** used to specify the font weight.

🡪 attributes are normal/bold

**Example**

p.normal {  
  font-weight: normal/bold;  
}

## Font Variant

🡪 The font-variant property specifies whether or not a text should be displayed in a small-caps font.

🡪 In a small-caps font, all lowercase letters are converted to uppercase letters. However, the converted uppercase letters appears in a smaller font size than the original uppercase letters in the text.

**Example**

p.normal {  
  font-variant: normal;  
}  
  
p.small {  
  font-variant: small-caps;  
}

shorthand

<head>  
**<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Audiowide|Sofia|Trirong">**  
<style>  
h1.a {font-family: "Audiowide", sans-serif;}  
h1.b {font-family: "Sofia", sans-serif;}  
h1.c {font-family: "Trirong", serif;}  
</style>  
</head>

**Using Google Fonts**

<head>  
**<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Audiowide|Sofia|Trirong">**  
<style>  
h1.a {font-family: "Audiowide", sans-serif;}  
h1.b {font-family: "Sofia", sans-serif;}  
h1.c {font-family: "Trirong", serif;}  
</style>  
</head>

**CSS Icons**

**🡪**we can add icons in HTML by icon library like font awesome.

🡪 Font awesome icons: <script src=<https://kit.fontawesome.com/yourcode.js> crossorigin="anonymous"></script>

**🡪BootStrap icons:** <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css"

🡪 **Google icons:** <link rel="stylesheet" href="https://fonts.googleapis.com/icon?family=Material+Icons">

🡪 Add the name of icon using <i> or <span> .**<i class="fas fa-cloud"></i>**

**CSS LINKS**

**🡪 we can decorate the links using CSS**

**Example**

/\* unvisited link \*/  
a:link {  
  color: red;  
}  
  
/\* visited link \*/  
a:visited {  
  color: green;  
}  
  
/\* mouse over link \*/  
a:hover {  
  color: hotpink;  
}  
  
/\* selected link \*/  
a:active {  
  color: blue;  
}

**Link Buttons**

a:link, a:visited {  
  background-color: #f44336;  
  color: white;  
  padding: 14px 25px;  
  text-align: center;  
  text-decoration: none;  
  display: inline-block;  
}  
  
a:hover, a:active {  
  background-color: red;  
}

**Image as List Item Marker**

**🡪**we can also add an image for a list item marker.

**Example**

ul

{  
  list-style-image: url('sqpurple.gif');  
}

**List Style position**

**🡪** list-style-position is useful to specify the position of list elements.

🡪 attributes are inside, means bullets inside the list item and outside means bullets outside the list item.

**Example**

ul.a

{  
  list-style-position: outside;  
}  
  
ul.b

{  
  list-style-position: inside;  
}

**CSS Display**

🡪 display property is the most important property in controlling the layout.

🡪 attributes are inline, border, none.

**none**

**🡪** nothing will be displayed for the element applied.

**Example**

p

{

display:none;

}

**Inline  
 🡪** all the elements are displayed in a Single line**.**

**Example**

p

{

display:inline;

}

**block**

**🡪** all the elements are displayed in the block level.

**Example**

p

{

display:block;

}

**Hide an Element**

**🡪** we can hide the element using “visibility” attribute. and set visibility to “hidden”

**Example**

P

{

visibility:hidden;

}

🡪 The main difference between visibility and “display:none” is space. Visibility consumes the space whereas “display:none” does not consumes any space.

**CSS Overflow**

🡪 The CSS overflow property controls what happens to content that is too big to fit into an area.

**🡪** Attributes are visible, hidden, auto, scroll.

**🡪 visible** indicates content is not clipped and displayed outside the element(default).

🡪 **hidden** indicates the content which is inside the margin will be displayed and the rest will be hidden. There is a loss in content

🡪 **scroll** indicates all the content is clipped and the scrollbar is attached where we can view all the content of particular element.

🡪 Auto is similar to scroll and the main difference is Auto adds scroll bar only when necessary.

**Example**

P

{

overflow:scroll/hidden/auto;

}

**Overflow-x and Overflow –y**

🡪 to deal with horizontal or vertical overflow we will use overflow-x an overflow-y.

**Example**

P

{

overflow-x:hidden;

overflow-y:scroll;

}

**CSS wordwrap**

**🡪** CSS wordwrap allows to break the word if there is any overflow occurs**.**

**Example**

p

{

word-wrap:word-break;

}

**CSS Word Break**

**🡪 word-break** Specifies the word breaking rules

**Example**

**🡪 “keep-all”** will ends with hyphen(‘-‘)where as **“break-all”** do the same without hyphens(‘-‘).

p.test1 {

width: 140px;

border: 1px solid #000000;

word-break: keep-all;

}

p.test2 {

width: 140px;

border: 1px solid #000000;

word-break: break-all;

}

**CSS Writing Mode**

**🡪 writing-mode** specifies whether text laid horizontally or vertically.

**Example**

P

{

Writing-mode:vertical-rl/horizontal-tb;

}

## The float Property

🡪 The float property is used for positioning and formatting content e.g. let an image float left to the text in a container.

🡪 The float property can have one of the following values:

* left - The element floats to the left of its container
* right - The element floats to the right of its container
* none - The element does not float (will be displayed just where it occurs in the text). This is default
* inherit - The element inherits the float value of its parent

**Example**

img {  
  float: right;  
}

## Float Next To Each Other

Normally div elements will be displayed on top of each other. However, if we use float: left we can let elements float next to each other:

**Example**

div {  
  float: left;  
  padding: 15px;  
}  
.div1 {  
  background: red;  
}  
.div2 {  
  background: yellow;  
}  
.div3 {  
  background: green;  
}

**CSS:display:inline-block**

🡪 same as display:inline but the major diference Is padding on top and bottom and it doesn’t add ant line breakup between the elements.

**Example**

span.a {  
  display: inline; /\* the default for span \*/  
  width: 100px;  
  height: 100px;  
  padding: 5px;  
  border: 1px solid blue;  
  background-color: yellow;  
}  
  
span.b {  
  display: inline-block;  
  width: 100px;  
  height: 100px;  
  padding: 5px;  
  border: 1px solid blue;  
  background-color: yellow;  
}

**Center an image  
 🡪** center an image can be done by setting left and the right margin to auto.

**Example**

img {  
  display: block;  
  margin-left: auto;  
  margin-right: auto;  
  width: 40%;  
}

🡪 we can also perform center verticallty using line-height which is same as height.

**Example**

.center {  
  line-height: 200px;  
  height: 200px;  
  border: 3px solid green;  
  text-align: center;  
}

🡪 we can also perform center verticallty transform.

**Example**

.center {  
  height: 200px;  
  position: relative;  
  border: 3px solid green;  
}  
  
.center p {  
  margin: 0;  
  position: absolute;  
  top: 50%;  
  left: 50%;  
  transform: translate(-50%, -50%);  
}

CSS Psuedo-class

🡪 pseudo class is used for special state

🡪 style an element for both the visited and un-visited links.

**Syntax**

Selector:pseudo-class

{

attrtibute:value;

}

🡪 pseudo classes can also combined with CSS selectors and also can be applied on tooltip, <div>

a.highlight:hover

{

Background-color:pink;

}

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

🡪Style the first letter, or line, of an element. Insert content before, or after, the content of an element

**Opacity**

🡪 opacity refers to the transparency of an element.

🡪 Ranges in between 0 to 1.1 refers to default

**Example**

.img

{

opacity:0.4;

}

🡪 hover can also applied along with transparency when mouse pulls on it. When mouse on the image transparency will get avoided and when the mouse transparency stays applied.

**Example**

.img:hover

{

opacity:0.4;

}

**Image Sprites**

**🡪**An image sprite is a collection of images put into a single image.

**Example**

#home {  
  width: 46px;  
  height: 44px;  
  background: url(img\_navsprites.gif) 0 0;  
}

🡪 Image sprites supports hover feature.

**Example**

#home a:hover

{  
  background: url('img\_navsprites\_hover.gif') 0 -45px;  
}

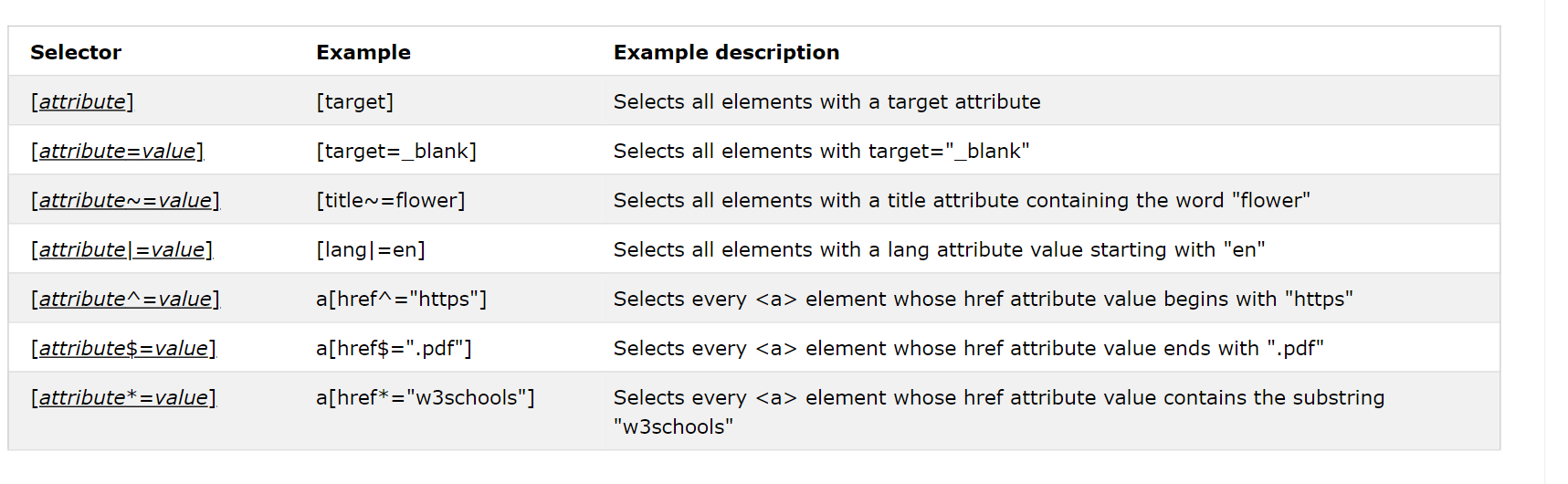
**CSS Attribute selector**

**🡪** in css attribute selector, we can select the particular attribute to style.

**Example**

a[target]

{  
 background-color: yellow;  
}

****

**CSS Forms**

🡪 **width is used to set the width of an element**

**Example**

Input

{

width:100%;

}

🡪 Use the padding property to add space inside the text field.

**Tip:** When you have many inputs after each other, you might also want to add some margin, to add more space outside of them

**Example**

Input[type=”text”]

{

width:100%;

padding: 12px 20px;

margin: 8px 0;

box-sizing:border-box;

}

🡪 Animated search inputs

Input[type=”text”]:focus

{

Transition: width:0.4s ease-in-out;

}

Input[type=”text”]:focus

{

width:100%;

}

**CSS Counters**

**🡪** Counters are like variables used to count the elements and also incrementable.

🡪 attributes are:

counter-reset: reset the container

counter- increment: increases the counter value.

content: inserts the generated content.

**Example**

body

{

counter-reset: section;

}

h1

{

counter-reset: subsection;

}

h1::before

{

counter-increment: section;

content: "Section " counter(section) ". ";

}

h2::before

{

counter-increment: subsection;

content: counter(section) "." counter(subsection) " ";

}

**Website Layout**

****

**Header**

**🡪**Header lays on the top of the website layout**.**

**Example**

**.**header

{

background-color: #f1f1f1;

padding: 20px;

text-align: center;

}

**Navigation Bar**

**🡪** Navigationbar is used to navigate through websites**.**

**Example**

.topnav a

{

float: left;

display: block;

color: #f2f2f2;

text-align: center;

padding: 14px 16px;

text-decoration: none;

}

**CONTENT**

**🡪** Content is the place where the actual content present.

🡪**1-column** (often used for mobile browsers)

**2-column** (often used for tablets and laptops)

**3-column layout** (only used for desktops)

**Example**

.column.middle

 {  
  width: 50%;  
}

**Footer**

**🡪** Footer stays at the bottom of the webpage. It contains data like copyright, author information, company name etc.

**Example**

.footer

{

background-color:gray;

text-align:center;}

## !important

🡪 The !important rule in CSS is used to add more importance to a property/value than normal.

🡪 In fact, if you use the !important rule, it will override ALL previous styling rules for that specific property on that element!

**Example**

#myid {  
  background-color: blue;  
}  
  
.myclass {  
  background-color: gray;  
}  
  
p {  
  background-color: red **!important**;  
}

**Note:** The only way to override an !important rule is to include another !important rule on a declaration with the same (or higher) specificity in the source code - and here the problem starts! This makes the CSS code confusing and the debugging will be hard, especially if you have a large style sheet!

**CSS Rounded corners**

**🡪** with **border-radius** we can make rounded corners towards borders.

🡪 we can specify four, three ,two, one attribute.

**Example**

#rcorners3 {

border-radius: 15px 50px;

background: #73AD21;

padding: 20px;

width: 200px;

height: 150px;

}

🡪 we can also make elliptical borders.

**CSS Border Images**

**🡪** with **border-image** property we can addimages to the border

🡪 we can also add borders, slice an image, decide whether middle section you specify.

**Example**

#borderimg

{

border: 10px solid transparent;

padding: 15px;

border-image: url(border.png) 30 round;

}

**Different slice values**

**🡪** There will be a chance of having multiple slices will change the looking of the border.

**Example**

#borderimg1

{

border: 10px solid transparent;

padding: 15px;

border-image: url(border.png) 50 round;

}

**CSS Background Images**

**🡪** we can also add multiple images to the background.

**Example:**

#example1

{

background-image: url(img\_flwr.gif), url(paper.gif);

background-position: right bottom, left top;

background-repeat: no-repeat, repeat;

padding: 15px;

}

🡪 we can also define sizes to the background image.

**Example**

#example1

{

background: url(img\_tree.gif) left top no-repeat, url(img\_flwr.gif) right bottom no-repeat, url(paper.gif) left top repeat;

padding: 15px;

background-size: 50px, 130px, auto;

}

**HTML Full Size Background Image**

**🡪** it’s also possible to set background images.

**Example**

html

{

background: url(img\_man.jpg) no-repeat center fixed;

background-size: cover;

}

**CSS Background-origin property**

🡪 The CSS background-origin property specifies where the background image is positioned.

🡪 The property takes three different values:

🡪 border-box - the background image starts from the upper left corner of the border

🡪 padding-box - (default) the background image starts from the upper left corner of the padding edge

🡪 content-box - the background image starts from the upper left corner of the content

**Example**

#example2

{

border: 10px solid black;

padding: 35px;

background: url(img\_flwr.gif);

background-repeat: no-repeat;

background-origin: border-box;

}

**Transparent**

**🡪** transparent is used to make the color transparent.

**Example**

Div

{

Background-color-transparent;

}

**Currentcolor Property**

**🡪** CSS holds the current color property of an element

**Example**

Div

{

color: blue;

border: 10px solid currentcolor;

padding: 15px;

}

**CSS Gradients**

**🡪 CSS Gradients** are used to display smooth transitions for more than one color.

🡪There are two types of gradients.

linear gradients:(goes up/down/left/right/diagonally).

radial gradients:(defined by the center).

🡪 **Syntax**: background-image: linear-gradient(direction, color-stop1, color-stop2, ...);

**Example**

#grad1

{

height: 200px;

background-color: red; /\* For browsers that do not support gradients \*/

background-image: linear-gradient(to right, red , yellow);

}

**Example**

#grad2

{

height: 100px;

background-color: red; /\* For browsers that do not support gradients \*/

background-image: linear-gradient(90deg, red, yellow);

}

**CSS Tooltip**

🡪 tooltip is used to display the additional information to the user when user hovers the mouse on it.

**Example**

.tooltip .tooltiptext

{

visibility: hidden;

width: 120px;

background-color: black;

color: #fff;

text-align: center;

border-radius: 6px;

padding: 5px 0;

/\* Position the tooltip \*/

position: absolute;

z-index: 1;}

**CSS Reflection**

**🡪** used to reflect the image

**Example**

img

{

-webkit-box-reflect: below/right;

}

**CSS Object position property**

**🡪 used to specify the object to be positioned inside the element.**

**🡪 object-fit** is used to fit an image in webpage**.**

**🡪 object-position** is used to place an image in an appropriate person/

**Example**

img

{

width: 200px;

height: 300px;

object-fit: cover;

object-postion 80% 100%;

}