CSS / CSS 3

I. Introduction to cascading style sheet:

What is CSS?

Cascading Style Sheets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable.

- CSS handles **the look and feel part** of a web page.
- Using CSS, you can control the **color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used,** as well as a variety of other effects.
- CSS is easy to learn and understand but it provides a powerful control over the **presentation** of an HTML document.
- Most commonly, CSS is combined with the markup languages HTML or XHTML

Advantages of CSS

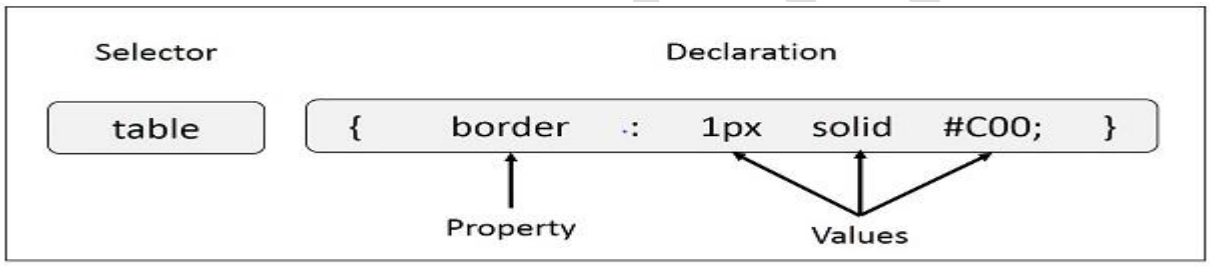
- **CSS saves time** - You can **write CSS once and then reuse the same sheet in multiple** HTML pages. You can define a style for each HTML element and apply it to as many web pages as you want.
- **Pages load faster** - If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So, less code means faster download times.
- **Easy maintenance** - To make a global change, simply change the style, and all the elements in all the web pages will be updated automatically.

CSS SYNTAX:

A CSS comprises of style rules that are interpreted by the browser and then applied to the corresponding elements in your document. A style rule is made of three parts:

- **Selector:** A selector is an HTML tag at which a style will be applied. This could be any tag like <h1> or <table> etc.
- **Property:** A property is a **type of attribute of HTML tag**. Put simply, all the HTML attributes are converted into CSS properties. They could be color, border, etc.
- **Value**: Values are assigned to properties. For example, color property can have the value either red or #F1F1F1 etc.

CSS Style Rule Syntax as follows: **selector { property: value }**



TYPES OF CSS

There are 3 types of CSS. They are:

1. **External style sheet**
2. **Internal style sheet**
3. **Inline style sheet**

1.External (in a separate file)

- External style sheets are **separate files full of CSS instructions** (with the file extension .css).
- An external CSS stylesheet can be applied to **any number of HTML documents** by placing a **<link> element in each HTML document**.
- The **attribute rel of the <link> tag** has to be set to **"stylesheet",** and the **href attribute to the relative or absolute path to the stylesheet.**

- While using **relative URL paths is generally considered good practice**, absolute paths can be used, too. In HTML5 the type attribute can be omitted.
- It is recommended that the **<link> tag be placed in the HTML file's <head> tag so that the styles are loaded before the elements** they style. Otherwise, users will see a flash of unstyled content.

Example :

1. hello-world.html

```
<!DOCTYPE html>
 <html>
<head>
<meta charset="utf-8" />
 <link rel="stylesheet"  type="text/css"  href="style.css">   </head>
   <body>
<h1>Hello world!</h1>
<p>I ♥ CSS</p>
</body>
 </html>
```

2.style.css

```
h1 {
 color: green;
text-decoration: underline;
}

p {
 font-size: 25px;
 font-family: 'Trebuchet MS', sans-serif;
}
```

2.Internal (at the top of a web page document)

- Internal styles are **placed at the top of each web page document**, before any of the content is listed.
- This is the next best thing to external, because they're easy to find, yet allow you to **'override' an external style sheet** -- for that special page that wants to be a nonconformist!
- CSS enclosed in **<style></style> tags within an HTML document functions like an external stylesheet**, except that it lives in the HTML document it styles instead of in a separate file,
- and therefore can only be applied to the document in which it lives. Note that this element must be **inside the <head> element for HTML validation** (though it will work in all current browsers if placed in body).

Example:

```
<head>
    <style>
        h1 {
            color: green;
            text-decoration: underline;
        }
        p {
            font-size: 25px;
            font-family: 'Trebuchet MS', sans-serif;
        }
    </style>
</head>
<body>
    <h1>Hello world!</h1>
    <p>I ♥ CSS</p>
</body>
```

3.Inline (right next to the text it decorates)

- Inline styles are **placed right where you need them**, next to the text or graphic you wish to decorate.
- You can insert **inline styles anywhere in the middle of your HTML code**, giving you real freedom to specify each web page element.
- Use inline styles to apply styling to a specific element. Placing style rules in **a <style> tag or external CSS file** is encouraged in order to maintain a distinction between **content and presentation**.
- Inline styles override any CSS in a <style> tag or external style sheet. this fact more often than not reduces a project's maintainability.
- The styles in the following example apply directly to the elements to which they are attached.
- Inline styles are generally the safest way to ensure rendering compatibility across various email clients, programs and devices, but can be time-consuming to write and a bit challenging to manage.

  Example:

  <h1 style="color: green; text-decoration: underline;">Hello world!</h1>

  ** I ♥ CSS **

II. Working with text and Fonts :

CSS TEXT:

1.Text Color

The color property is used **to set the color of the text**. The color is specified by:

- **a color name - like "red"**
- **a HEX value - like "#ff0000"**
- **an RGB value - like "rgb(255,0,0)"**

The default text color for a page is defined in the body selector.

Example:

```
body {
  color: blue;
}

h1 {
  color: green;
}
```

2.Text Alignment

The text-align property is used **to set the horizontal alignment of a text**.

A text can be **left or right aligned, centered, or justified**.

- The left alignment is default if text direction is left-to-right, and right alignment is default if text direction is right-to-left.

Example:

```
h1 {
   text-align: center;
  }

h2 {
   text-align: left;
   }

h3 {
   text-align: right;
  }
```

- When the text-align property is set to "**justify**", each line is stretched so that every line **has equal width**, and the left and right margins are **straigh**t (like in magazines and newspapers):

Example:

```
div {
  text-align: justify;
}
```

3.Text Decoration

- The text-decoration property is used **to set or remove decorations from** text.
- The value text-decoration: none; is often used to remove underlines from links:

Example: for anchor tag

```
a {
text-decoration: none;
 }
```

The other text-decoration values are used to decorate text:

```
  h1 {
text-decoration: overline;
  }

 h2 {
text-decoration: line-through;
  }

 h3 {
text-decoration: underline;
  }
```

4.Text Transformation

- The text-transform property is used to **specify uppercase and lowercase** letters in a text.
- It can be used to turn everything into uppercase or lowercase letters, or capitalize the first letter of each word.

Example:

```
p.uppercase {
  text-transform: uppercase;
}

p.lowercase {
  text-transform: lowercase;}
p.capitalize {
  text-transform: capitalize;
}
```

Here the p is the paragraph tag. Among multiple paragraphs we use the class attribute to specify .it is written within the body tag.

```
<p class="uppercase">This is some text.</p>

<p class="lowercase">This is some text.</p>

<p class="capitalize">This is some text.</p>
```

5.Text Indentation

The text-indent property is used to **specify the indentation of the first line** of a text:

Example:

```
p {
    text-indent: 50px;
  }
```

6.Letter Spacing

The letter-spacing property is used to **specify the space between the characters in a text.**

The following example demonstrates how to increase or decrease the space between characters.

Example:

```
h1 {
  letter-spacing: 3px;
  }

h2 {
  letter-spacing: -3px;
  }
```

7.Line Height

The line-height property is used to **specify the space between lines.**

Example:

```
 p.small {
       line-height: 0.8;
      }

 p.big {
```

```
  line-height: 1.8;
}
```

8.Text Direction

The direction property is used to **change the text direction of an element**.

Example:

```
p {
  direction: rtl;
 }
```

9.Word Spacing

The word-spacing property is used to specify **the space between the words in a text.**

The following example demonstrates how to increase or decrease the space between words:

Example:

```
h1 {
 word-spacing: 10px;
 }

h2 {
  word-spacing: -5px;
  }
```

10.Text Shadow

The text-shadow property adds **shadow to text**.

The following example specifies the position of the horizontal shadow (3px), the position of the vertical shadow (2px) and the color of the shadow (red):

Example:

```
h1 {
text-shadow: 3px 2px red;
}
```

CSS FONTS

CSS Font Families

In CSS, there **are two types of font family names**:

- **generic family** - a group of font families with a similar look (like "Serif" or "Monospace")
- **font family** - a specific font family (like "Times New Roman" or "Arial")

Font Family

- The font family of a text is set with the `font-family` property.
- The `font-family` property should hold several font names as a "fallback" system
- **Note**: If the name of a font family is **more than one word, it must be in quotation marks,** like: "Times New Roman".
- More than one font family is specified in a comma-separated list.

Example:

```
p {
font-family: "Times New Roman", Times, serif;
}
```

Font Style

The font-style property is **mostly used to specify italic text**.

This property has three values:

- **normal** - The text is shown normally
- **italic** - The text is shown in italics
- **oblique** - The text is "leaning" (oblique is very similar to italic, but less supported)

example:

```
p.normal {
  font-style: normal;
}

p.italic {
  font-style: italic;
}

p.oblique {
  font-style: oblique;
}
```

Font Size

- The font-size property **sets the size of the text**.
- Always use the proper HTML tags, like <h1> - <h6> for headings and <p> for paragraphs.
- The font-size value can be an **absolute, or relative size**.

Absolute size:

- Sets the text to a specified size

- Does not allow a user to change the text size in all browsers (bad for accessibility reasons)
- Absolute size is useful when the physical size of the output is known

Relative size:

- Sets the size relative to surrounding elements
- Allows a user to change the text size in browsers

Note: If you do not specify a font size, **the default size for normal text, like paragraphs, is 16px**.

Set Font Size With Pixels

Example

```
h1 {
    font-size: 40px;
  }

h2 {
    font-size: 30px;
  }


 p {
   font-size: 14px;
   }
```

Set Font Size With Em

To allow users to resize the text (in the browser menu), many developers use em instead of pixels.

The em size unit is recommended by the W3C.

1em is equal to the current font size. The default text size in browsers is 16px. So, the default size of 1em is 16px.

The size can be calculated from pixels to em using this formula: *pixels*/16=*em*

Example

```
h1 {
  font-size: 2.5em; /* 40px/16=2.5em */
}

h2 {
  font-size: 1.875em; /* 30px/16=1.875em */
}

p {
  font-size: 0.875em; /* 14px/16=0.875em */
}
```

Use a Combination of Percent and em

The solution that works in all browsers, is to set a default font-size in percent for the <body> element:

Example

```
body {
  font-size: 100%;
}

h1 {
  font-size: 2.5em;
}

h2 {
```

```
  font-size: 1.875em;
}

p {
  font-size: 0.875em;
}
```

Float Property CSS – Left and Right

- CSS float is a property that forces any element to float (right, left, none, inherit) inside its parent body with the rest of the element to wrap around it.
- This property can be used to place an image or an element inside its container and other inline elements will wrap around it.

For eg. : Images in newspaper and articles are placed in certain position with rest of the content wrapped around it.

Syntax :

```
float: left | right | none | inherit ;
```

Values :

```
float : right;    /* Floats the element to right of it's container */

float : left;     /*  Floats the element to left of it's container */

float : none;     /*  It will restrict the element to float */

float : initial;  /*  The element remains to it's default position */

float : inherit;  /*  Enables the element to inherit the property from its parent element */
```

CSS SELECTORS

Introduction to selectors:

- CSS selectors **identify specific HTML elements as targets for CSS styles.**
- Selectors use a wide range of over 50 selection methods offered by the **CSS language, including elements, classes, IDs, pseudo-elements and pseudo-classes, and patterns.**
- CSS selectors form **the basis of Cascading Style Sheets, allowing us to target specific elements within a HTML document and apply style.**

List of CSS Selectors:

1. **universal selector (*)**
2. **Type selector / element selector**
3. **Class Selector (.)**
4. **ID selector (#)**

1.universal selector: (*)

- **CSS universal selectors** select any type of elements in an HTML page. It matches a single element.
- An **asterisk (i.e. "*")** is used to denote a **CSS universal selector**. An asterisk can also be followed by a selector.
- This is useful when you want to set a style for of all the elements of an HTML page or for all of the elements within an element of an HTML page.

Syntax:

```
synatx:

* { property:value }
```

Example:

```
*{
  color: blue;
```

```
        background: silver;
        }
```

2.Type selectors:

Type Selector is also called as Element selector.

- The CSS element Selector or the type selector matches occurrences of those tags specified in the list.
- **Type selectors target element types on the page.** They're also called **element selectors because we use the element's HTML tag as the selector.**

```
synatx:

element { property:value }
```

e.g. `p { font-size:32px; }`

3.class selectors: (.)

The class name selector select all elements with the targeted class name. For example, the **class name. warning** would select the following element:

```
<div class="warning">

<p>This would be some warning copy.</p>

</div>
```

We can **combine class names to target elements** more specifically. Let's build on the example above to showcase a more complicated class selection.

CSS

```
.important {  color: orange; }
```

```
.warning {   color: blue; }
.warning, .important {    color: red; }
```

HTML

```
<div class="warning">
 <p>This would be some warning copy</p>
</div>
<div class="important warning">
  <p class="important">This is some really important warning copy</p>
</div>
```

4.ID selector: (#)

ID selectors **select DOM elements with the targeted ID**. To select an element by a specific ID in CSS, **the # prefix** is used.

Example:

```
<div id="hello">
   <p>Example</p>
</div>
```

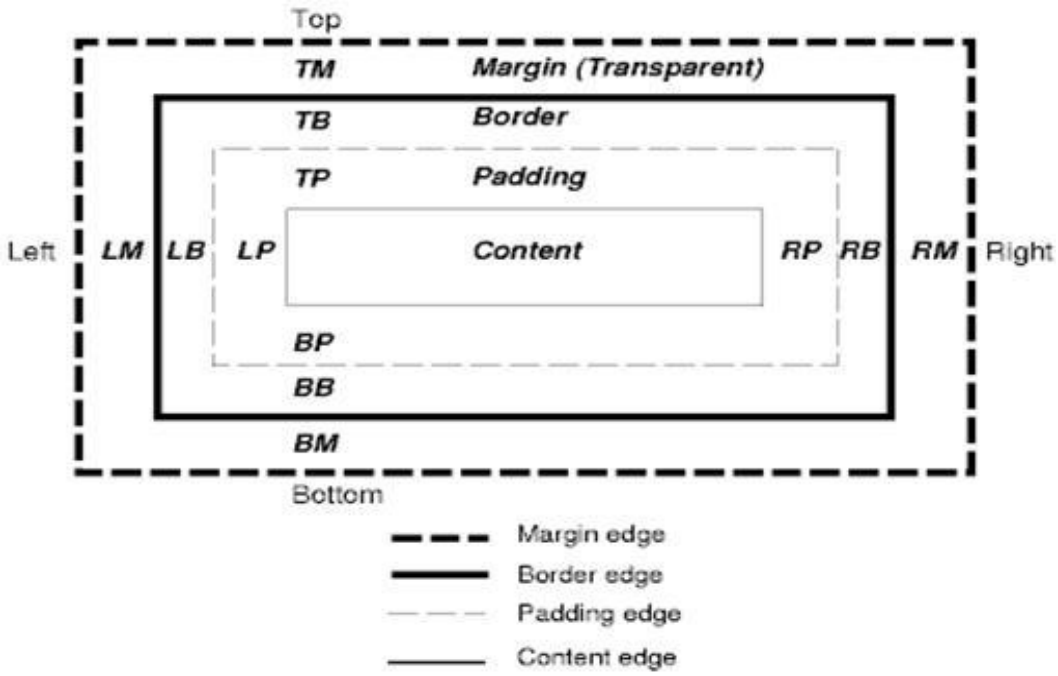
In css:

```
#hello {
 width: 20px;
}
```

Note: the id selector is a unique. And only one tag can be called at a time.

LAYOUT

BOX MODEL



- The browser creates a rectangle for each element in the HTML document.
- The Box Model describes how the **padding, border, and margin** are added to the content to create this rectangle.
- The perimeter of each of the four areas is called an **edge**. **Each edge defines a box**.
- The innermost rectangle is the **content box**. The width and height of this depends on the element's rendered content (text, images and any child elements it may have).
- Next is the **padding box**, as defined by the padding property. If there is no padding width defined, the padding edge is equal to the content edge.

- the **border box**, as defined by the border property. If there is no border width defined, the border edge is equal to the padding edge.
- The outermost rectangle is the **margin box**, as defined by the margin property. If there is no margin width defined, the margin edge is equal to the border edge.

Example

```
div {
border: 5px solid red;
margin: 50px;
padding: 20px;
}
```

box-sizing

The default box model (content-box) can be counter-intuitive, since the width / height for an element will not represent its actual width or height on screen as soon as you start adding padding and border styles to the

The following example demonstrates this potential issue with content-box:

Eg:

```
textarea {
    width: 100%;
    padding: 3px;
    box-sizing: content-box; /* default value */ }
```

Colors and Borders:

Backgrounds

With CSS you can set **colors, gradients, and images as the background of an element.**
It is possible to specify various combinations of **images, colors.**

Background Color :

The background-color property sets the background color of an element using a color value or through keywords, such **as transparent, inherit or initial**.

- **transparent**, specifies that the background color should be transparent. This is default.
- **inherit,** inherits this property from its parent element.
- **initial**, sets this property to its default value.

Colors in CSS can be specified by different methods.

1.Color names

CSS

```
div {
background-color: red;  /* red */
}
```

HTML

```
<div>This will have a red background</div>
```

2.Hex color codes

Hex code is used **to denote RGB components** of a color in **base-16 hexadecimal notation**.

#ff0000, for example, is bright red, where the red component of the color is 256 bits (ff) and the corresponding green and blue portions of the color is 0 (00).

Note: If both values in each of the three RGB pairings (R, G, and B) are the same, then the color code can be shortened into three characters (the first digit of each pairing).
#ff0000 can be shortened to #f00, and

#ffffff can be shortened to #fff.

Note: Hex notation is case-insensitive.

Html:

```
body {
  background-color: #de1205; /* red */
}
```

Css:

```
.main {
  background-color: #00f; /* blue */
}
```

RGB / RGBA

Another way to declare a color is to use RGB or RGBA.

RGB stands for **Red, Green and Blue,** and requires of three separate values between **0 and 255**, put between brackets, that correspond with the decimal color values for respectively red, green and blue.

RGBA allows you to add an additional alpha parameter **between 0.0 and 1.0 to define opacity.**

Html:

```
header {
 background-color: rgb(0, 0, 0); /* black */
 }
footer {
-color: rgba(0, 0, 0, 0.5); /* black with 50% opacity */
}
```

HSL / HSLA

Another way to declare a color is to use HSL or HSLA and is similar to RGB and RGBA.

HSL stands for **hue, saturation, and lightness**, and is also often called HLS:

- ✓ **Hue** is a degree on the color wheel **(from 0 to 360).**
- ✓ **Saturation** is a percentage between **0% and 100%.**
- ✓ **Lightness** is also a percentage between **0% and 100%.**

HSLA allows you to add an additional alpha parameter **between 0.0 and 1.0 to define opacity.**

```
li a {
 background-color: hsl(120, 100%, 50%); /* green */
 }
```

Css:

```
#p1 {
 background-color: hsla(120, 100%, 50%, .3); /* green with 30% opacity */
 }
```

Interaction with background-image

The following statements are all equivalent:

```
body {
background: red;
background-image: url(partiallytransparentimage.png);
}
body {
 background-color: red;
background-image: url(partiallytransparentimage.png);
}
body {
  background-image: url(partiallytransparentimage.png);
background-color: red;
}
body {
background: red url(partiallytransparentimage.png);
 }
```

They will all lead to the red color being shown underneath the image, where the parts of the **image are transparent,** or the image is not showing (perhaps as a result of background-repeat).

Note that the following is not equivalent:

```
body {
 background-image: url(partiallytransparentimage.png);
background: red;
}
```

Here, the value of **background overrides your background-image.**

Background Image

The background-image property is used to specify a background image to be applied to all matched elements. By default, this image is tiled to cover the entire element, excluding margin.

```
.myClass {
background-image: url('/path/to/image.jpg');
}
```

To use multiple images as background-image, define comma separated url()

```
.myClass {
 background-image: url('/path/to/image.jpg') ,
url('/path/to/image2.jpg');
}
```

- ✓ The images will stack according to their order with the first declared image on top of the others and so on.
- ✓ Value Result url('/path/to/image.jpg') Specify background image's path(s) or an image resource specified with data URI schema (apostrophes can be omitted), separate multiples by comma none No background image initial **Default value inherit Inherit parent's value**

Background – size:

Keywords syntax-

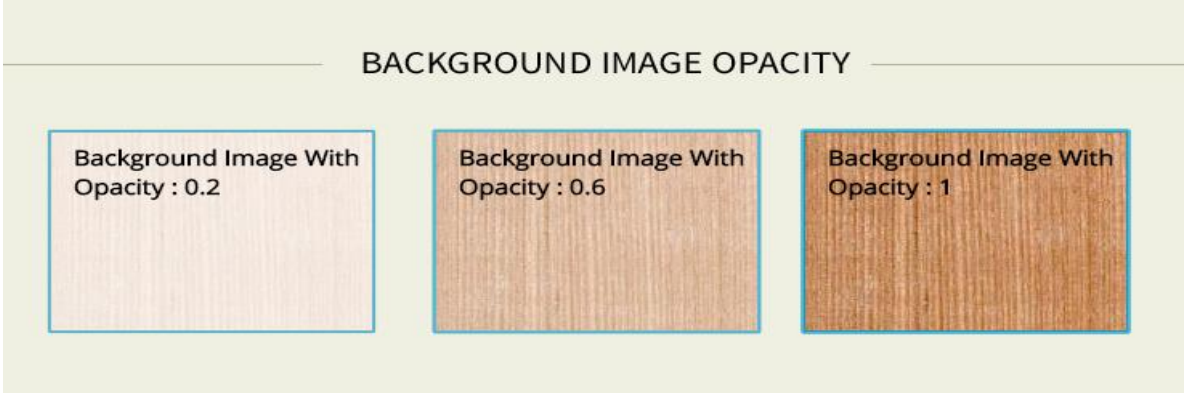
background-size: **cover ;** /*Background image will cover both the coordinate of containing box.*/

background-size: **contain ;** /*Small size images will spread till the containing box covers completely.*/

Background Image Opacity

When we opt for Background opacity property of CSS for an HTML element generally what happen is it will not only change the opacity of image in background but also reflects the opacity changes in its child elements.

The default initial value for opacity is 1(100% opaque).



Background Opacity Syntax

opacity: **1.0 ;** /* The default opacity where image looks complete solid. */

opacity: **0.6 ;** /* The image will look less solid, more opaque*/

opacity: **0.1 ;** /* The image will become nearly transparent and text gets visible more clearly */

opacity: **inherit ;** /* The default opacity of actual image/*

Border Property: Style, Width, Color

The CSS Border property allows you to customize the borders around an HTML elements.

It is a shorthand property to set individual border property values in a single place.

You can set the thickness or width, color and style of each border.

There are mainly three border properties:

- **The border-style** :- Specifies whether a border should be solid, dashed line, double line, or one of the other possible values
- **The border-color** :- Specifies the color of a border
- **The border-width** :- Specifies the width of a border

Border-style:

This border-style property defines the type of border to display. Below given is the syntax for the type of styles that can be used:

Syntax:

border-style: **none** /* Defines no border */

border-style: **solid**

border-style: **double**

border-style: **dotted**

border-style: **dashed**

border-style: **groove**

border-style: **inset**

border-style: **outset**

border-style: **ridge**

border-style: **hidden**

Border-width:-

The border-width property helps you to set the width of the border. It comes up with a predefined values **thin, medium and thick.**

You can also set the custom width of borders but in pixels.

The predefined values are :

border-width: **thin;**

border-width: **thick;**

border-width: **medium;**

Rounded Borders :

Border-radius :

The **border-radius property** defines **the radius of the element's corners**.

This property **can have from one to four values**. Here are the rules:

1. **Four values - border-radius: 15px 50px 30px 5px;**

- (first value applies to top-left corner,
- second value applies to top-right corner,
- third value applies to bottom-right corner, and
- fourth value applies to bottom-left corner)

2. **Three values - border-radius: 15px 50px 30px;**

- (first value applies to top-left corner,
- second value applies to top-right and bottom-left corners, and
- third value applies to bottom-right corner)

3. **Two values - border-radius: 15px 50px;**

- (first value applies to top-left and bottom-right corners, and
- the second value applies to top-right and bottom-left corners):

4. **One value - border-radius: 15px;**

- the value applies to all four corners, which are rounded equally:

box-shadow Property

The box-shadow property attaches one or more shadows to an element.

Examples:

```
Img {
  box-shadow: 10px 10px 8px red;

box-shadow: 10px 10px 8px 10px #eee ;

 box-shadow: 5px 5px blue, 10px 10px red, 15px 15px green;
}
```

CSS 3 NOTES