What is CSS?

* CSS stands for Cascading Style Sheets
* CSS describes how HTML elements are to be displayed on screen, paper, or in other media
* CSS saves a lot of work. It can control the layout of multiple web pages all at once
* External stylesheets are stored in CSS files

# History of CSS

CSS was created HÃ¥kon Wium Lie to allow web designers to change the layout, colors, and fonts of their websites. Originally, websites were meant to be used by researchers only, so the decoration did not matter. However, when websites became widespread, the need to make them look nice grew. To learn more about HÃ¥kon Wium Lie, visit:

[HÃ¥kon Wium Lie](https://www.bu.edu/lernet/artemis/years/2020/projects/FinalPresentations/HTML/H%C3%83%C2%A5konWiumLie.html)



### The Timline of HTML

* **1994-** HAYkon Wium Lie proposed the idea of CSS.
* **1996-** The first version of CSS was invented.
* **1998-** CSS 2 was released and work on CSS 3 began. CSS 3 was very different from the other versions.
* **2011-** CSS 3 was released, which fixed the errors found in CSS 2

A CSS rule consists of a selector and a declaration block.

CSS Syntax



The selector points to the HTML element you want to style.

The declaration block contains one or more declarations separated by semicolons.

Each declaration includes a CSS property name and a value, separated by a colon.

Multiple CSS declarations are separated with semicolons, and declaration blocks are surrounded by curly braces.

**IMPLEMENTATION OF CSS**

Three Ways to Insert CSS

There are three ways of inserting a style sheet:

* INLINE CSS
* INTERNAL CSS
* EXTERNAL CSS

## Inline CSS

An inline style may be used to apply a unique style for a single element.

To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.

EX- <h1 style="color:blue;text-align:center;">This is a heading</h1>

## Internal CSS

An internal style sheet may be used if one single HTML page has a unique style.

The internal style is defined inside the <style> element, inside the head section.

<!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>

## External CSS

With an external style sheet, you can change the look of an entire website by changing just one file!

Each HTML page must include a reference to the external style sheet file inside the <link> element, inside the head section.

<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" href="mystyle.css">

</head>

<body>

<h1>This is a heading</h1>

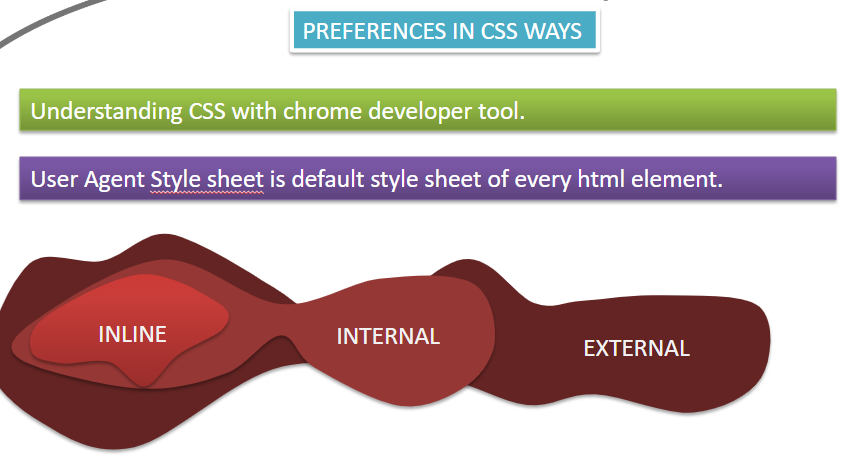
<p>This is a paragraph.</p>

</body>

</html>

NOTE: THE FIRST PRIORITY ALWAYS GOES TO INLINE CSS.

BUT WE CAN SET THE PRIORITY WITH HELP OF !important KEYWORD.



CSS Selectors

CSS selectors are used to "find" (or select) the HTML elements you want to style.

We can divide CSS selectors into five categories:

* Simple selectors (select elements based on name, id, class)
* [Combinator selectors](https://www.w3schools.com/css/css_combinators.asp) (select elements based on a specific relationship between them)
* [Attribute selectors](https://www.w3schools.com/css/css_attribute_selectors.asp) (select elements based on an attribute or attribute value)

## The CSS id Selector

The id selector uses the id attribute of an HTML element to select a specific element.

The id of an element is unique within a page, so the id selector is used to select one unique element!

To select an element with a specific id, write a hash (#) character, followed by the id of the element.

SYNTAX

#id\_Name {  
  key:”value”;  
}

## The CSS class Selector

The class selector selects HTML elements with a specific class attribute.

To select elements with a specific class, write a period (.) character, followed by the class name.

SYNTAX

.class\_Name {  
  key:”value”;  
}

## The CSS element Selector

The element selector selects HTML elements based on the element name.

SYNTAX

elementNAme {  
  key:”value”;  
}

## The CSS Universal Selector

The universal selector (\*) selects all HTML elements on the page.

SYNTAX

\*{  
  key:”value”;  
}

NOTE : THE FIRST PRIORITY ALWAYS GOES TO ID SELECTOR.

PRIORITY BASIS

* ID SELECTOR
* CLASS SLEECTOR
* ELEMENT SELECTOR
* UNIVERSAL SELECTOR

**BUT WE CAN SET THE PRIORITY WITH HELP OF !important KEYWORD.**

CSS Combinators

A CSS selector can contain more than one simple selector. Between the simple selectors, we can include a combinator.

There are four different combinators in CSS:

* descendant selector (space)
* child selector (>)
* adjacent sibling selector (+)
* general sibling selector (~)

## Descendant Selector

The descendant selector matches all elements that are descendants of a specified element.

The following example selects all <p> elements inside <div> elements:

EX-

<!DOCTYPE html>

<html>

<head>

<style>

div p {

background-color: yellow;

}

</style>

</head>

<body>

<h2>Descendant Selector</h2>

<p>The descendant selector matches all elements that are descendants of a specified element.</p>

<div>

<p>Paragraph 1 in the div.</p>

<p>Paragraph 2 in the div.</p>

<section><p>Paragraph 3 in the div.</p></section>

</div>

<p>Paragraph 4. Not in a div.</p>

<p>Paragraph 5. Not in a div.</p>

</body>

</html>

## Child Selector (>)

The child selector selects all elements that are the children of a specified element.

The following example selects all <p> elements that are children of a <div> element:

EX-

<head>

<style>

div > p {

background-color: yellow;

}

</style>

</head>

<body>

<h2>Child Selector</h2>

<p>The child selector (>) selects all elements that are the children of a specified element.</p>

<div>

<p>Paragraph 1 in the div.</p>

<p>Paragraph 2 in the div.</p>

<section>

<!-- not Child but Descendant -->

<p>Paragraph 3 in the div (inside a section element).</p>

</section>

<p>Paragraph 4 in the div.</p>

</div>

<p>Paragraph 5. Not in a div.</p>

<p>Paragraph 6. Not in a div.</p>

</body>

</html>

## Adjacent Sibling Selector (+)

The adjacent sibling selector is used to select an element that is directly after another specific element.

Sibling elements must have the same parent element, and "adjacent" means "immediately following".

The following example selects the first <p> element that are placed immediately after <div> elements:

EX-

<!DOCTYPE html>

<html>

<head>

<style>

div + p {

background-color: yellow;

}

</style>

</head>

<body>

<h2>Adjacent Sibling Selector</h2>

<p>The + selector is used to select an element that is directly after another specific element.</p>

<p>The following example selects the first p element that are placed immediately after div elements:</p>

<div>

<p>Paragraph 1 in the div.</p>

<p>Paragraph 2 in the div.</p>

</div>

<p>Paragraph 3. After a div.</p>

<p>Paragraph 4. After a div.</p>

<div>

<p>Paragraph 5 in the div.</p>

<p>Paragraph 6 in the div.</p>

</div>

<p>Paragraph 7. After a div.</p>

<p>Paragraph 8. After a div.</p>

</body>

</html>

## General Sibling Selector (~)

The general sibling selector selects all elements that are next siblings of a specified element.

The following example selects all <p> elements that are next siblings of <div> elements:

EX-

<!DOCTYPE html>

<html>

<head>

<style>

div ~ p {

background-color: yellow;

}

</style>

</head>

<body>

<h2>General Sibling Selector</h2>

<p>The general sibling selector (~) selects all elements that are next siblings of a specified element.</p>

<p>Paragraph 1.</p>

<div>

<p>Paragraph 2.</p>

</div>

<p>Paragraph 3.</p>

<section>code </section>

<p>Paragraph 4.</p>

</body>

</html>

## The CSS Grouping Selector

The grouping selector selects all the HTML elements with the same style definitions.

Look at the following CSS code (the h1, h2, and p elements have the same style definitions):

It will be better to group the selectors, to minimize the code.

To group selectors, separate each selector with a comma.

EX-

<!DOCTYPE html>

<html>

<head>

<style>

h1, h2, p {

text-align: center;

color: red;

}

</style>

</head>

<body>

<h1>Hello World!</h1>

<h2>Smaller heading!</h2>

<p>This is a paragraph.</p>

</body>

</html>

CSS Backgrounds

The CSS background properties are used to add background effects for elements.

* background-color
* background-image
* background-repeat
* background-attachment
* background-position

## CSS background-color

The background-color property specifies the background color of an element.

### **Example**

The background color of a page is set like this:

body {  
  background-color: lightblue;  
}

## CSS background-image

The background-image property specifies an image to use as the background of an element.

By default, the image is repeated so it covers the entire element.

### **Example**

Set the background image for a page:

body {  
  background-image: url("paper.gif");  
}

## CSS background-repeat: no-repeat

Showing the background image only once is also specified by the background-repeat property:

### **Example**

Show the background image only once:

body {  
  background-image: url("img\_tree.png");  
  background-repeat: no-repeat;  
}

## CSS background-position

The background-position property is used to specify the position of the background image.

### **Example**

Position the background image in the top-right corner:

body {  
  background-image: url("img\_tree.png");  
  background-repeat: no-repeat;  
  background-position: right top;  
}

**positions**

top left(default)

top center

top right

left

center

right

bottom left

bottom center

bottom right

## CSS background-attachment

The background-attachment property specifies whether the background image should scroll or be fixed (will not scroll with the rest of the page):

### **Example**

Specify that the background image should be fixed:

body {  
  background-image: url("img\_tree.png");  
  background-repeat: no-repeat;  
  background-position: right top;  
  background-attachment: fixed;  
}

### **Example**

Specify that the background image should scroll with the rest of the page:

body {  
  background-image: url("img\_tree.png");  
  background-repeat: no-repeat;  
  background-position: right top;  
  background-attachment: scroll;  
}

# CSS **background-size** Property

The background-size property specifies the size of the background images.

#example1 {

background: url(mountain.jpg);

background-repeat: no-repeat;

background-size: cover;

}

CSS Border

The CSS border is a shorthand property used to set the border on an element.

The [CSS](https://www.javatpoint.com/css-tutorial) border properties are use to specify the style, color and size of the border of an element. The CSS border properties are given below

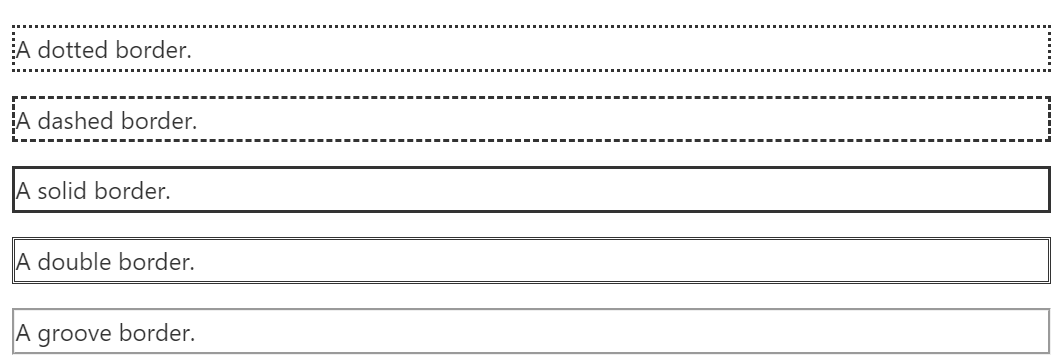
* border-style
* border-color
* border-width
* border-radius

1) CSS border-style

The Border style property is used to specify the border type which you want to display on the web page.

There are some border style values which are used with border-style property to define a border.

1. {border-style: none;}
2. {border-style: dotted;}
3. {border-style: dashed;}
4. {border-style: solid;}
5. {border-style: double;}
6. {border-style: groove;}



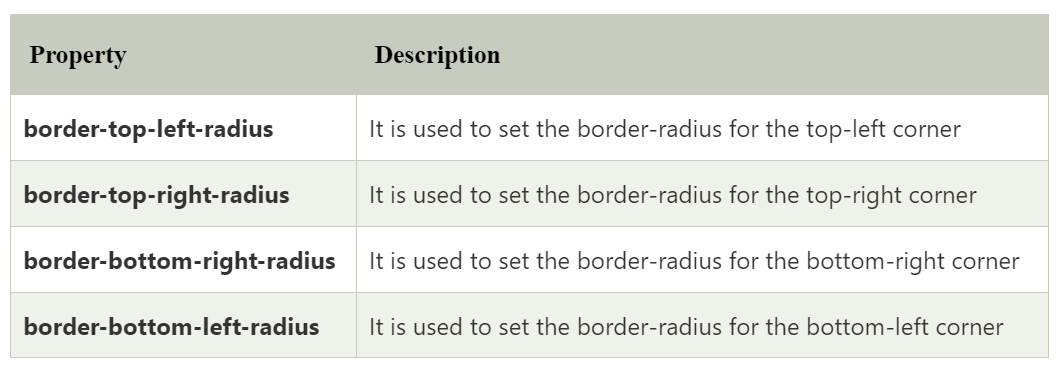
## 2) CSS border-width

The border-width property is used to set the border's width. It is set in pixels. You can also use the one of the three pre-defined values, thin, medium or thick to set the width of the border.

# CSS border-radius property

This CSS property sets the rounded borders and provides the rounded corners around an element, tags, or div. It defines the radius of the corners of an element.

It is shorthand for **border top-left-radius, border-top-right-radius, border-bottom-right-radius** and **border-bottom-left-radius**. It gives the rounded shape to the corners of the border of an element. We can specify the border for all four corners of the box in a single declaration using the border-radius. The values of this property can be defined in percentage or length units.



Let's see what happens when we provide a single value, two values, three values, and four values to this property.

* If we provide a single value **(**such as **border-radius: 30px;)** to this property, it will set all corners to the same value.
* When we specify two values **(**such as **border-radius: 20% 10% ;)**, then the first value will be used for the top-left and bottom-right corners, and the second value will be used for the top-right and bottom-left corners.
* When we use three values **(**such as **border-radius: 10% 30% 20%;)** then the first value will be used for the top-left corner, the second value will be applied on top-right, and bottom-left corners and the third value will be applied to the bottom-right corner.
* Similarly, when this property has four values **(border-radius: 10% 30% 20% 40%;)** then the first value will be the radius of top-left, the second value will be used for the top-right, the third value will be applied on bottom-right, and the fourth value is used for bottom-left.

If the border-style property has four values:

* **border-style: dotted solid double dashed;**
  + top border is dotted
  + right border is solid
  + bottom border is double
  + left border is dashed

If the border-style property has three values:

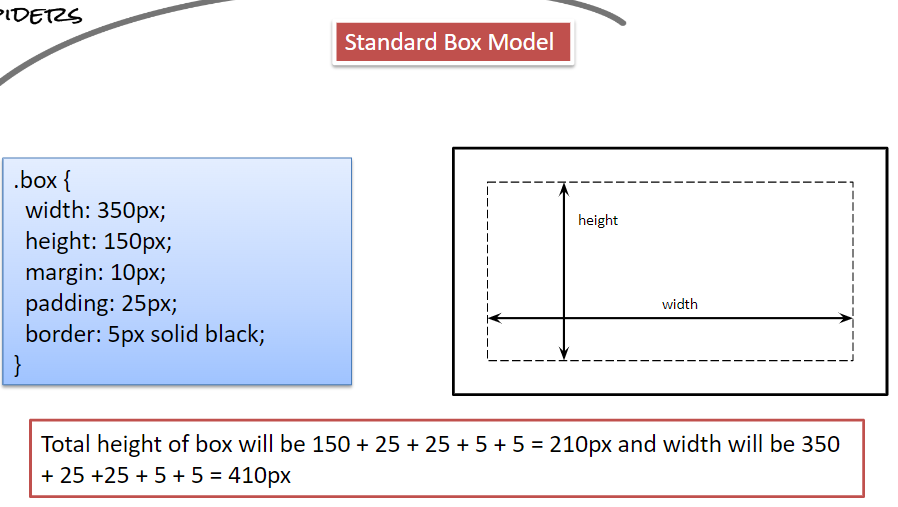
* **border-style: dotted solid double;**
  + top border is dotted
  + right and left borders are solid
  + bottom border is double

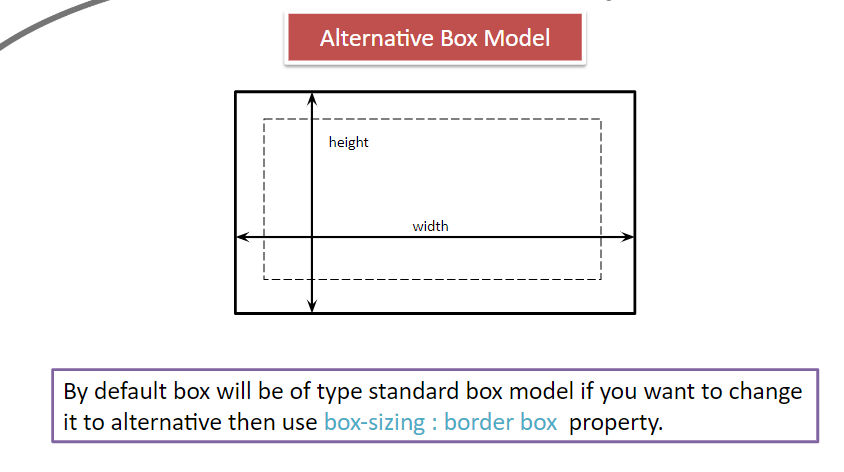
If the border-style property has two values:

* **border-style: dotted solid;**
  + top and bottom borders are dotted
  + right and left borders are solid

If the border-style property has one value:

* **border-style: dotted;**
  + all four borders are dotted





Colors are specified using predefined color names, or RGB, HEX, HSL, RGBA, HSLA values.

CSS Color Names

In CSS, a color can be specified by using a predefined color name

EX-RED,BLUE,GREEN,BLACK,WHITE ETC

CSS Text Color

You can set the color of text

CSS Background Color

You can set the background color for HTML elements:

RGB Value

In CSS, a color can be specified as an RGB value, using this formula:

rgb(red, green, blue)

Each parameter (red, green, and blue) defines the intensity of the color between 0 and 255.

For example, rgb(255, 0, 0) is displayed as red, because red is set to its highest value (255) and the others are set to 0.

To display black, set all color parameters to 0, like this: rgb(0, 0, 0).

To display white, set all color parameters to 255, like this: rgb(255, 255, 255).

HEX Value

In CSS, a color can be specified using a hexadecimal value in the form:

#rrggbb

Where rr (red), gg (green) and bb (blue) are hexadecimal values between 00 and ff (same as decimal 0-255).

For example, #ff0000 is displayed as red, because red is set to its highest value (ff) and the others are set to the lowest value (00).

To display black, set all values to 00, like this: #000000.

To display white, set all values to ff, like this: #ffffff.

HSL Value

In CSS, a color can be specified using hue, saturation, and lightness (HSL) in the form:

hsl(hue, saturation, lightness)

Hue is a degree on the color wheel from 0 to 360. 0 is red, 120 is green, and 240 is blue.

Saturation is a percentage value. 0% means a shade of gray, and 100% is the full color.

Lightness is also a percentage. 0% is black, 50% is neither light or dark, 100% is white

HSLA Value

In CSS, a color can be specified using hue, saturation, and lightness (HSL) in the form:

hsl(hue, saturation, lightness,alfa)

Hue is a degree on the color wheel from 0 to 360. 0 is red, 120 is green, and 240 is blue.

Saturation is a percentage value. 0% means a shade of gray, and 100% is the full color.

Lightness is also a percentage. 0% is black, 50% is neither light or dark, 100% is white

Alfa value is nothing but the define the opacity to increase the transperancy of an object.

range(0.1 to 1)

CSS GRADIENTS

CSS gradients let you display smooth transitions between two or more specified colors.

CSS defines three types of gradients:

Linear Gradients (goes down/up/left/right/diagonally)

Radial Gradients (defined by their center)

Conic Gradients (rotated around a center point)

1.CSS Linear Gradients

To create a linear gradient you must define at least two color stops. Color stops are the colors you want to

render smooth transitions among. You can also set a starting point and a direction (or an angle) along with the gradient effect.

Syntax

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

background-image: repeating-linear-gradient(direction, color-stop1, color-stop2, ...);

Direction - Top to Bottom (this is default)

Direction - Left to Right

The following example shows a linear gradient that starts from the left. It starts red, transitioning to yellow:

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

Direction - Diagonal

You can make a gradient diagonally by specifying both the horizontal and vertical starting positions.

The following example shows a linear gradient that starts at top left (and goes to bottom right).

It starts red, transitioning to yellow:

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

2.CSS Radial Gradients

A radial gradient is defined by its center.

To create a radial gradient you must also define at least two color stops.

Syntax

background-image: radial-gradient(shape size at position, start-color, ..., last-color);

background-image: repeating-radial-gradient(shape size at position, start-color, ..., last-color);

By default, shape is ellipse, size is farthest-corner, and position is center.

Radial Gradient - Evenly Spaced Color Stops (this is default)

The following example shows a radial gradient with evenly spaced color stops:

background-image: radial-gradient(red, yellow, green);

Radial Gradient - Differently Spaced Color Stops

The following example shows a radial gradient with differently spaced color stops:

background-image: radial-gradient(red 5%, yellow 15%, green 60%);

Set Shape

The shape parameter defines the shape. It can take the value circle or ellipse. The default value is ellipse.

The following example shows a radial gradient with the shape of a circle:

3.CSS Conic Gradients

A conic gradient is a gradient with color transitions rotated around a center point.

To create a conic gradient you must define at least two colors.

Syntax

background-image: conic-gradient([from angle] [at position,] color [degree], color [degree], ...);

background-image: repeating-conic-gradient([from angle] [at position,] color [degree], color [degree], ...);

By default, angle is 0deg and position is center.

If no degree is specified, the colors will be spread equally around the center point.

Conic Gradient: Three Colors

The following example shows a conic gradient with three colors:

background-image: conic-gradient(red, yellow, green);

Conic Gradient: Three Colors and Degrees

The following example shows a conic gradient with three colors and a degree for each color:

Example

A conic gradient with three colors and a degree for each color:

background-image: conic-gradient(red 45deg, yellow 90deg, green 210deg);

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)"

Text Alignment and Text Direction

In this chapter you will learn about the following properties:

* text-align
* text-align-last
* direction

## 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 following example shows center aligned, and left and right aligned text (left alignment is default if text direction is left-to-right, and right alignment is default if text direction is right-to-left):

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 straight (like in magazines and newspapers):

div {  
  text-align: justify;  
}

## Text Align Last

The text-align-last property specifies how to align the last line of a text.

#box1{  
  text-align-last: right;  
}  
  
#box2{  
  text-align-last: center;  
}  
  
#box3{  
  text-align-last: justify;  
}

## Text Direction

The direction property can be used to change the text direction of an element:

Note:by default direction is left right(ltr)

We can change right to left(rtl).

p {  
  direction: rtl;  
    
}

Text Decoration

* text-decoration-line
* text-decoration-color
* text-decoration-style
* text-decoration-thickness
* text-decoration

**TEXT DECORATION LINE**

The text-decoration-line property is used to add a decoration line to text.

* **Tip:** You can combine more than one value, like overline and underline to display lines both over and under a text.

h1 {

text-decoration: overline;

}

h2 {

text-decoration: line-through;

}

h3 {

text-decoration: underline;

}

p.ex {

text-decoration: overline underline;

}

## Specify a Color for the Decoration Line

The text-decoration-color property is used to set the color of the decoration line.

h1 {  
  text-decoration-line: overline;  
  text-decoration-color: red;  
}

## Specify a Style for the Decoration Line

The text-decoration-style property is used to set the style of the decoration line.

h1 {  
  text-decoration-line: underline;  
  text-decoration-style: solid;  
}  
  
h2 {  
  text-decoration-line: underline;  
  text-decoration-style: double;  
}  
  
h3 {  
  text-decoration-line: underline;  
  text-decoration-style: dotted;  
}  
  
p.ex1 {  
  text-decoration-line: underline;  
  text-decoration-style: dashed;  
}  
  
p.ex2 {  
  text-decoration-line: underline;  
  text-decoration-style: wavy;  
}  
  
p.ex3 {  
  text-decoration-line: underline;  
  text-decoration-color: red;  
  text-decoration-style: wavy;  
}

## Specify the Thickness for the Decoration Line

The text-decoration-thickness property is used to set the thickness of the decoration line.

h1 {  
  text-decoration-line: underline;  
  text-decoration-thickness: auto;  
}  
  
h2 {  
  text-decoration-line: underline;  
  text-decoration-thickness: 5px;  
}  
  
h3 {  
  text-decoration-line: underline;  
  text-decoration-thickness: 25%;  
}  
  
p {  
  text-decoration-line: underline;  
  text-decoration-color: red;  
  text-decoration-style: double;  
  text-decoration-thickness: 5px;  
}

The Shorthand Property

The text-decoration property is a shorthand property for:

* text-decoration-line (required)
* text-decoration-color (optional)
* text-decoration-style (optional)
* text-decoration-thickness (optional)

h1 {  
  text-decoration: underline;  
}  
  
h2 {  
  text-decoration: underline red;  
}  
  
h3 {  
  text-decoration: underline red double;  
}  
  
p {  
  text-decoration: underline red double 5px;  
}

## A Small Tip

All links in HTML are underlined by default. Sometimes you see that links are styled with no underline. The text-decoration: none; is used to remove the underline from links, like this:

a {  
  text-decoration: none;  
}

## 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:

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

Text Spacing

* text-indent
* letter-spacing
* line-height
* word-spacing

## Text Indentation

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

p {  
  text-indent: 50px;  
}

## 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:

h1 {  
  letter-spacing: 5px;  
}

## Line Height

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

p.small {  
  line-height: 0.8;  
}  
  
p.big {  
  line-height: 1.8;  
}

## 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:

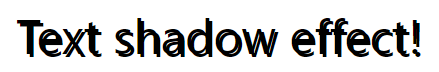
p.one {  
  word-spacing: 10px;  
}  
  
p.two {  
  word-spacing: -2px;  
}

## Text Shadow

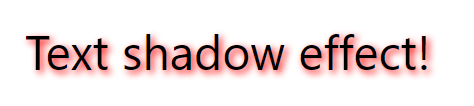
The text-shadow property adds shadow to text.

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

h1 {  
  text-shadow: 2px 2px;  
}



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



The position property specifies the type of positioning method used for an element

The position Property

The position property specifies the type of positioning method used for an element (static, relative, fixed, absolute or sticky).

There are five different position values:

* static
* relative
* fixed
* absolute
* sticky

## position: static;

HTML elements are positioned static by default.

Static positioned elements are not affected by the top, bottom, left, and right properties.

An element with position: static; is not positioned in any special way; it is always positioned according to the normal flow of the page:

## position: relative;

An element with position: relative; is positioned relative to its normal position.

Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position. Other content will not be adjusted to fit into any gap left by the element.

## position: fixed;

An element with position: fixed; is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled. The top, right, bottom, and left properties are used to position the element.

A fixed element does not leave a gap in the page where it would normally have been located.

## position: absolute;

An element with position: absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed).

However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling.

**Note:** Absolute positioned elements are removed from the normal flow, and can overlap elements.

## position: sticky;

An element with position: sticky; is positioned based on the user's scroll position.

A sticky element toggles between relative and fixed, depending on the scroll position. It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like position:fixed).

CSS Overflow

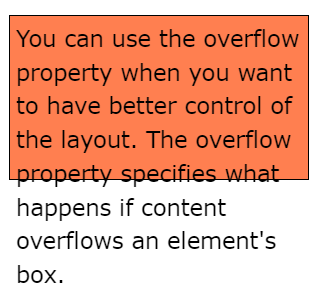
The overflow property specifies whether to clip the content or to add scrollbars when the content of an element is too big to fit in the specified area.

The overflow property has the following values:

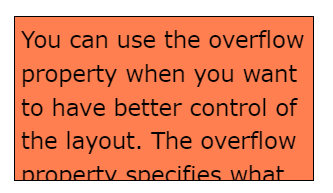
* visible - Default. The overflow is not clipped. The content renders outside the element's box
* hidden - The overflow is clipped, and the rest of the content will be invisible
* scroll - The overflow is clipped, and a scrollbar is added to see the rest of the content
* auto - Similar to scroll, but it adds scrollbars only when necessary

overflow: visible

By default, the overflow is visible, meaning that it is not clipped and it renders outside the element's box:

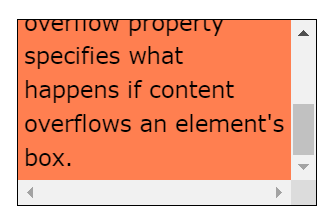


## overflow: hidden

With the hidden value, the overflow is clipped, and the rest of the content is hidden:

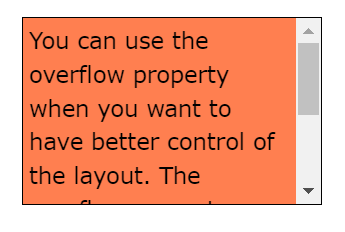
## overflow: scroll

Setting the value to scroll, the overflow is clipped and a scrollbar is added to scroll inside the box. Note that this will add a scrollbar both horizontally and vertically (even if you do not need it):



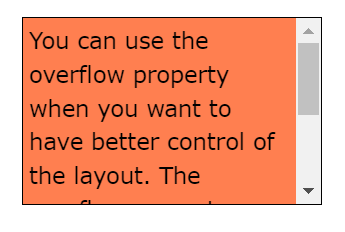
## overflow: auto

The auto value is similar to scroll, but it adds scrollbars only when necessary:



## overflow-x and overflow-y

The overflow-x and overflow-y properties specifies whether to change the overflow of content just horizontally or vertically (or both):overflow-x specifies what to do with the left/right edges of the content.  
overflow-y specifies what to do with the top/bottom edges of the content.



DETAILS TAG

The <details> tag specifies additional details that the user can

open and close on demand.

The <details> tag is often used to create an interactive widget

that the user can open and close. By default, the widget is

closed. When open, it expands, and displays the content within.

Any sort of content can be put inside the <details> tag.

**Tip:** The <summary> tag is used in conjunction

with <details> to specify a visible heading for the details.

<details>

<summary>Epcot Center</summary>

<p>Epcot is a theme park at Walt Disney

World Resort featuring exciting attractions,

international pavilions, award-winning

fireworks and seasonal special events.</p>

</details>

FIELDSET TAG

The <fieldset> tag is used to group related elements in a form.

The <fieldset> tag draws a box around the related elements.

Tips and Notes

**Tip:** The <legend> tag is used to define a caption for

the <fieldset> element.

<form action="/action\_page.php">

<fieldset>

<legend>Personalia:</legend>

<label for="fname">First name:</label>

<input type="text" id="fname" name="fname"><br><br>

<label for="lname">Last name:</label>

<input type="text" id="lname" name="lname"><br><br>

<label for="email">Email:</label>

<input type="email" id="email" name="email"><br><br>

<label for="birthday">Birthday:</label>

<input type="date" id="birthday" name="birthday"><br><

br>

<input type="submit" value="Submit">

</fieldset>

</form>

IFRAME TAG

The <iframe> tag specifies an inline frame.

An inline frame is used to embed another document within the

current HTML document.

**Tip:** Use CSS to style the <iframe> (see example below).

**Tip:** It is a good practice to always include a title attribute for

the <iframe>. This is used by screen readers to read out what

the content of the <iframe>

<iframe src="https://www.google.com" title="google

webpage"></iframe>

AUDIO TAG

The <audio> tag is used to embed sound content in a document,

such as music or other audio streams.

The <audio> tag contains one or more <source> tags with

different audio sources. The browser will choose the first source

it supports.

The text between the <audio> and </audio> tags will only be

displayed in browsers that do not support the <audio> element.

There are three supported audio formats in HTML: MP3, WAV,

and OGG.

<audio controls src="horse.ogg" type="audio/ogg">

</audio>

VIDEO TAG

The <video> tag is used to embed video content in a document,

such as a movie clip or other video streams.

The <video> tag contains one or more <source> tags with

different video sources. The browser will choose the first source

it supports.

The text between the <video> and </video> tags will only be

displayed in browsers that do not support the <video>

element.

There are three supported video formats in HTML: MP4, WebM,

and OGG.

<video width="320" height="240" controls

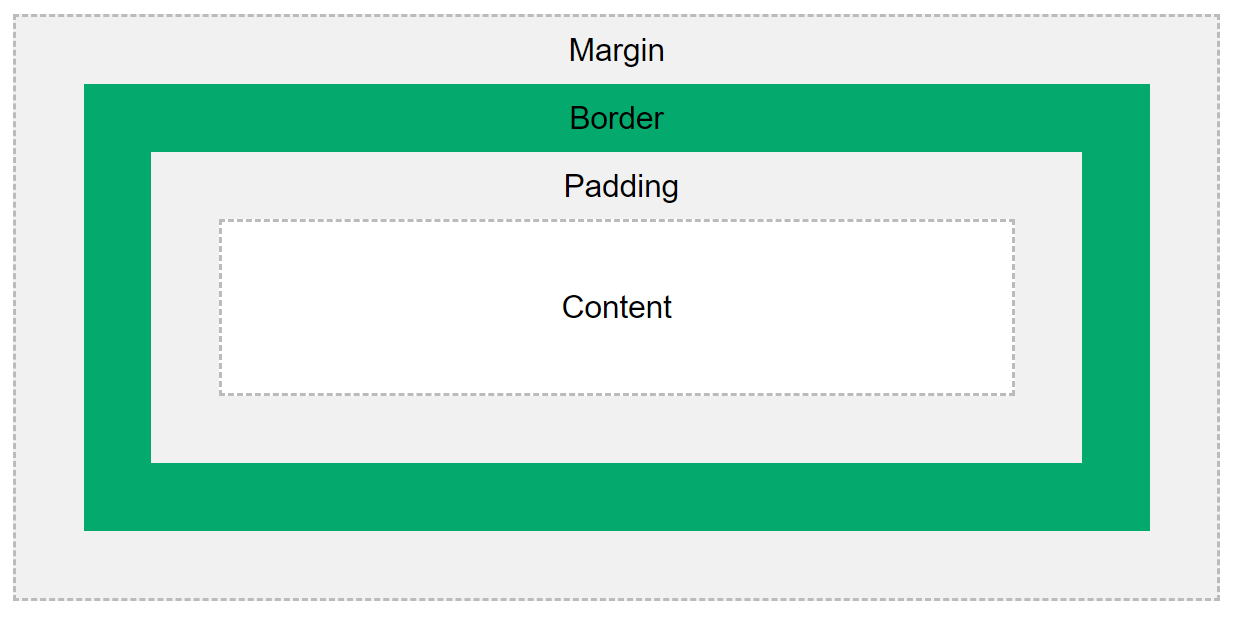
src="movie.mp4" type="video/mp4">

</video>

## The CSS Box Model

In CSS, the term "box model" is used when talking about design and layout.

The CSS box model is essentially a box that wraps around every HTML element. It consists of: margins, borders, padding, and the actual content. The image below illustrates the box model:



* **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

CSS Margins

The CSS margin properties are used to create space around elements, outside of any defined borders.

With CSS, you have full control over the margins. There are properties for setting the margin for each side of an element (top, right, bottom, and left).

Margin - Individual Sides

CSS has properties for specifying the margin for each side of an element:

* margin-top
* margin-right
* margin-bottom
* margin-left

Margin - Shorthand Property

To shorten the code, it is possible to specify all the margin properties in one property.

The margin property is a shorthand property for the following individual margin properties:

* margin-top
* margin-right
* margin-bottom
* margin-left

So, here is how it works:

If the margin property has four values:

* **margin: 25px 50px 75px 100px;(TRBL)**
  + top margin is 25px
  + right margin is 50px
  + bottom margin is 75px
  + left margin is 100px

If the margin property has three values:

* **margin: 25px 50px 75px;(TRB)**
  + top margin is 25px
  + right and left margins are 50px
  + bottom margin is 75px

If the margin property has two values:

* **margin: 25px 50px;**
  + top and bottom margins are 25px
  + right and left margins are 50px

If the margin property has one value:

* **margin: 25px;**
  + all four margins are 25px

## The auto Value

You can set the margin property to auto to horizontally center the element within its container.

The element will then take up the specified width, and the remaining space will be split equally between the left and right margins.

CSS Padding

The CSS padding properties are used to generate space around an element's content, inside of any defined borders.

With CSS, you have full control over the padding. There are properties for setting the padding for each side of an element (top, right, bottom, and left).

Padding - Individual Sides

CSS has properties for specifying the padding for each side of an element:

* padding-top
* padding-right
* padding-bottom
* padding-left

Padding - Shorthand Property

To shorten the code, it is possible to specify all the padding properties in one property.

The padding property is a shorthand property for the following individual padding properties:

* padding-top
* padding-right
* padding-bottom
* padding-left

So, here is how it works:

If the padding property has four values:

* **padding: 25px 50px 75px 100px;(TRBL)**
  + top padding is 25px
  + right padding is 50px
  + bottom padding is 75px
  + left padding is 100px

If the padding property has three values:

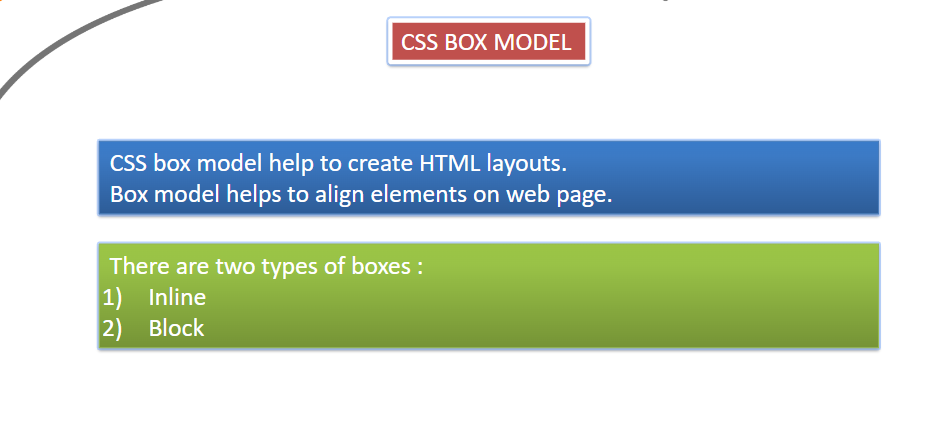
* **padding: 25px 50px 75px;(TRB)**
  + top padding is 25px
  + right and left paddings are 50px
  + bottom padding is 75px

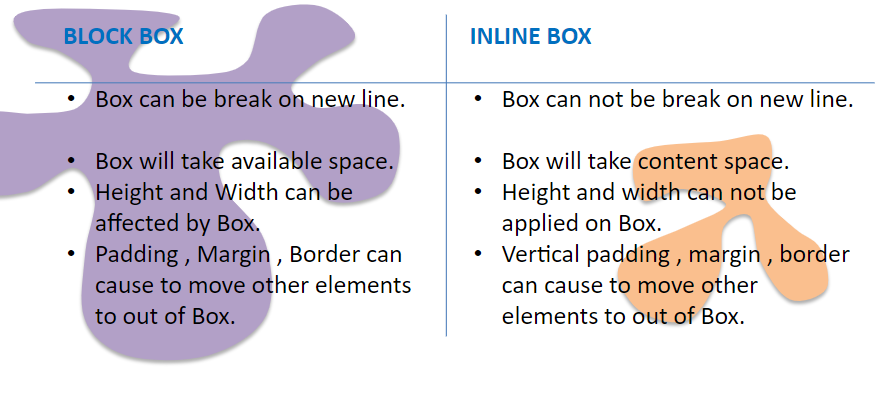
If the padding property has two values:

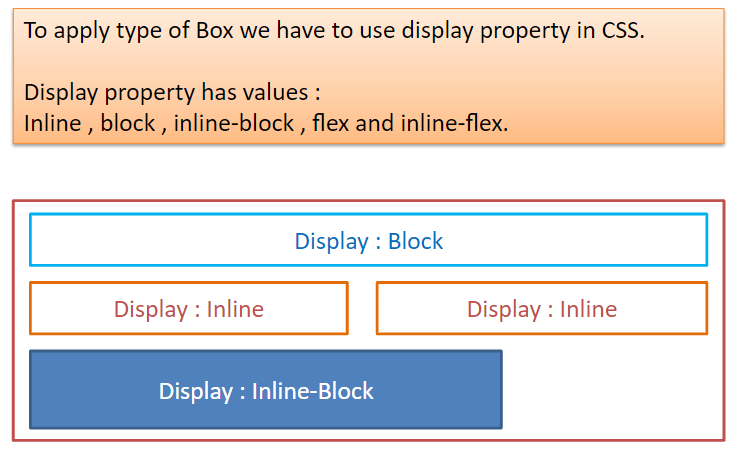
* **padding: 25px 50px;(TR)**
  + top and bottom paddings are 25px
  + right and left paddings are 50px

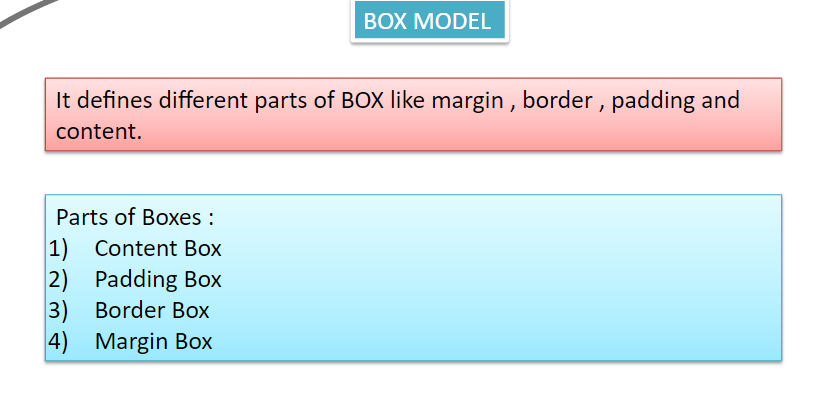
If the padding property has one value:

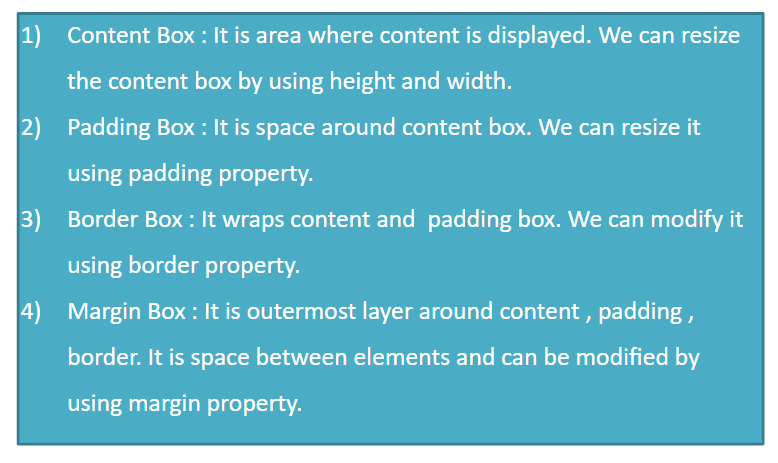
* **padding: 25px;**
  + all four paddings are 25px

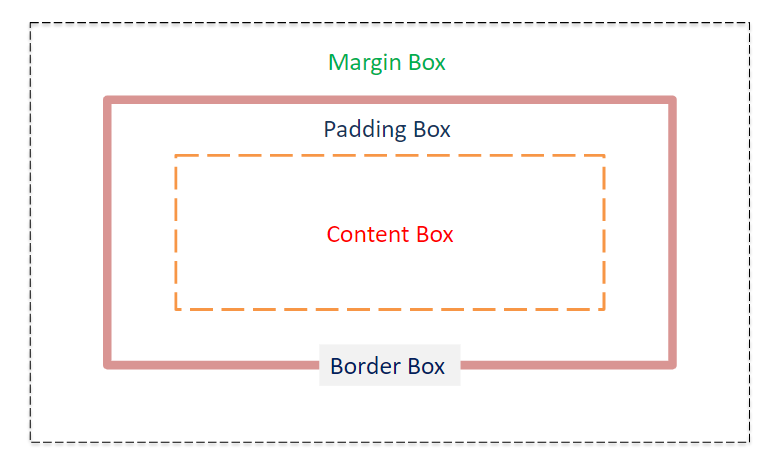


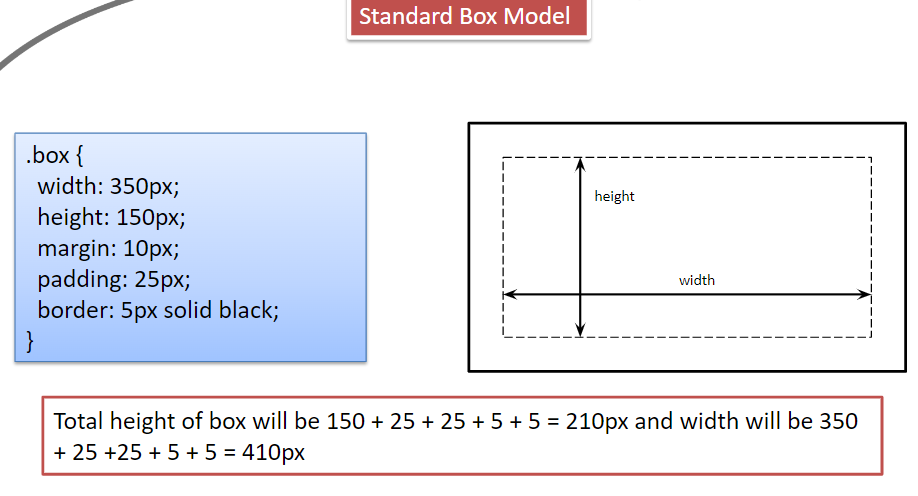


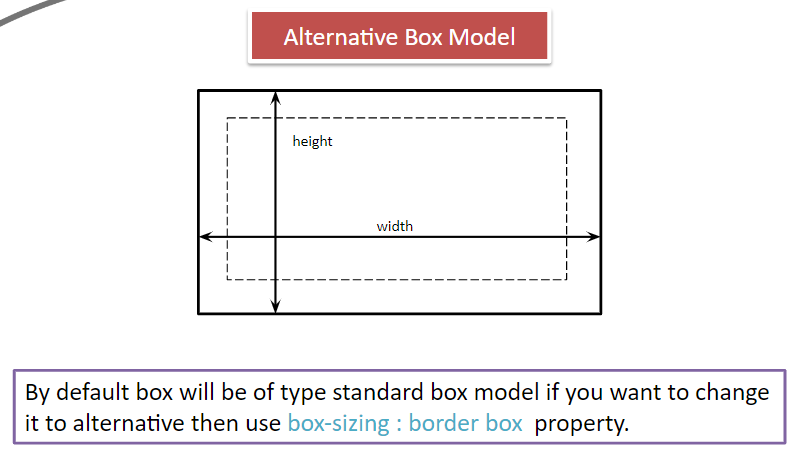


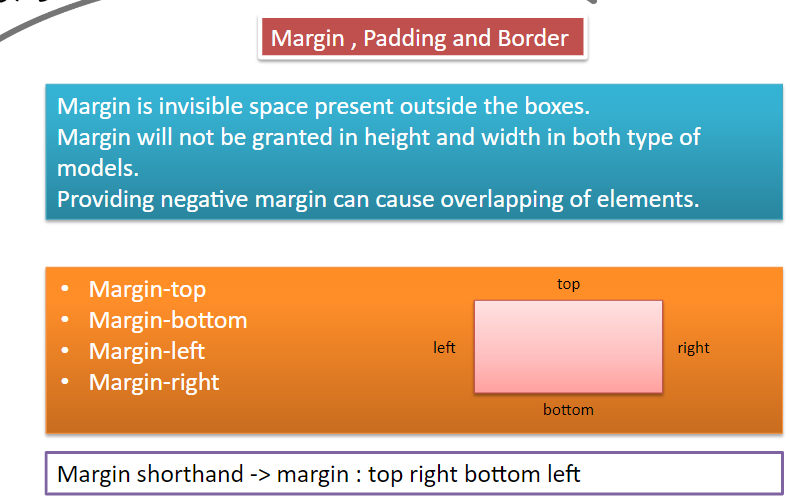


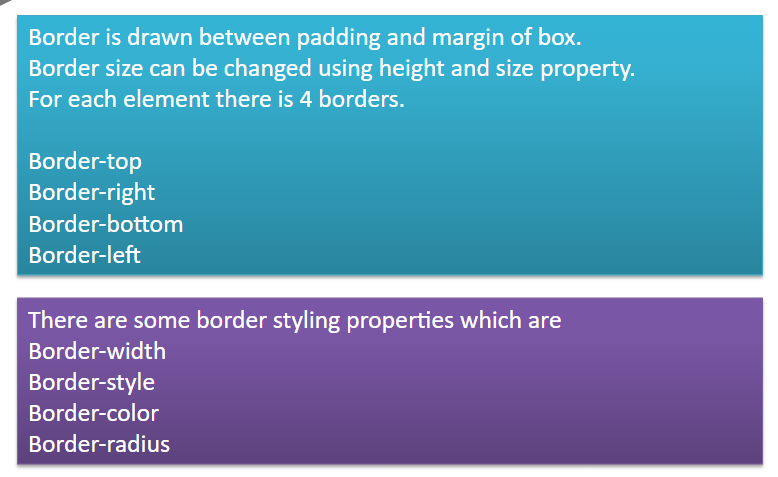


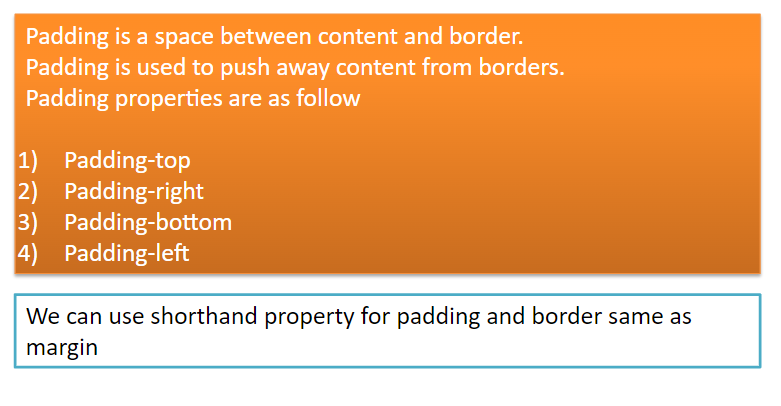


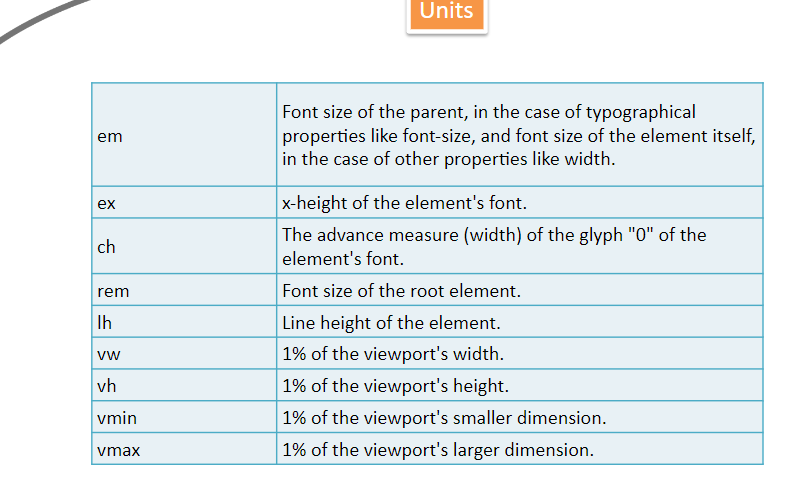




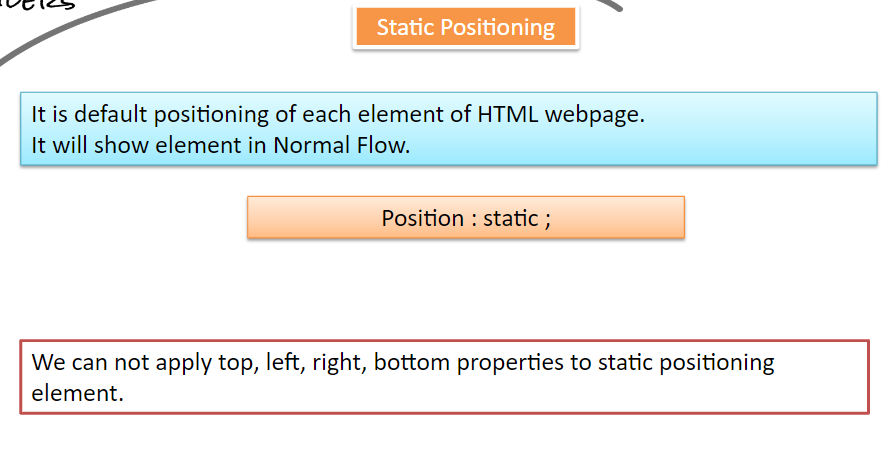


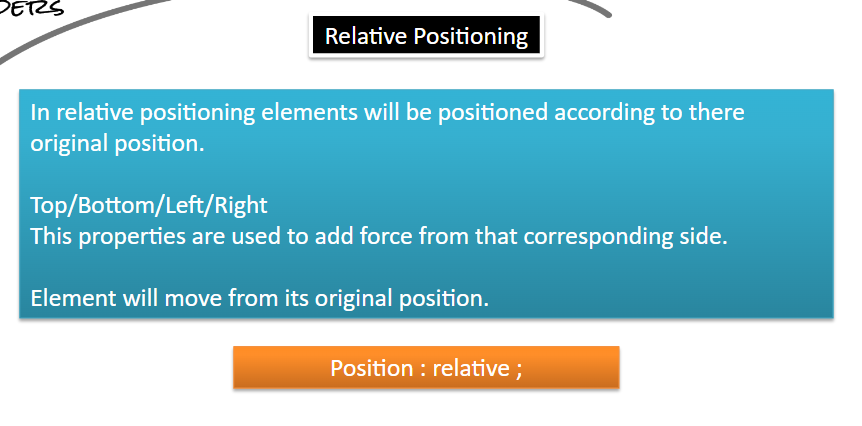


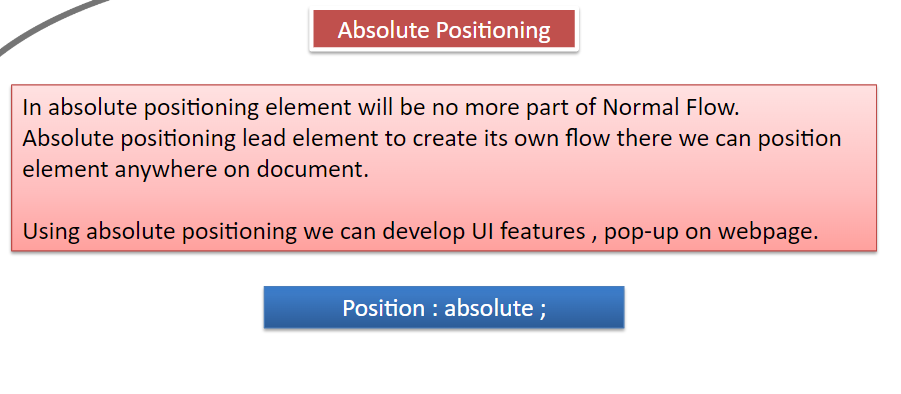


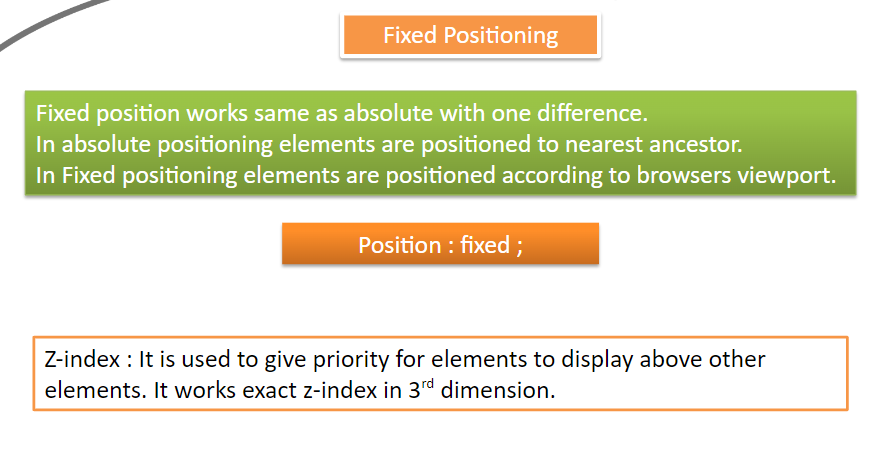










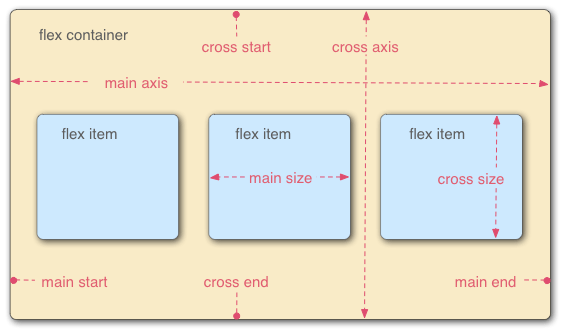


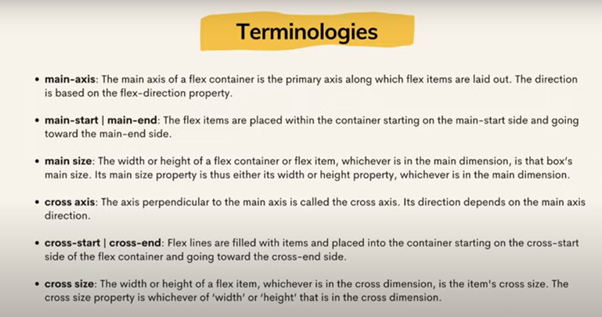


-------------------------------------------------------------------------------------------------------------------------**CSS FLEX BOX** :

Flex box is a one-dimensional layout method for laying out items in rows or columns.

**THE FLEX MODEL :**

****

****

**Display : Flex**

It is used to apply flex on child elements. This property must be used in parent element.

FLEX-DIRECTION :

Ø The flex-direction property specifies the direction of the flex- items.

Ø We have to write in parent container.

**flex-direction**: row|row-reverse|column|column-reverse

1) row-> Default value. The flexible items are displayed horizontally, as a row

2) row-reverse: -> Same as row, but in reverse order

3) column : ->The flexible items are displayed vertically, as a column

4) column-reverse:-> Same as column, but in reverse order

**Justify content :**

The CSS justify-content property defines how the browser distributes space between and around content items along the main-axis of a flex container,

**Justify-content: center | space-between | space-around |space-evenly | flex-start | flex-end**

**flex-start** (default): items are packed toward the start line

**flex-end**: items are packed toward to end line

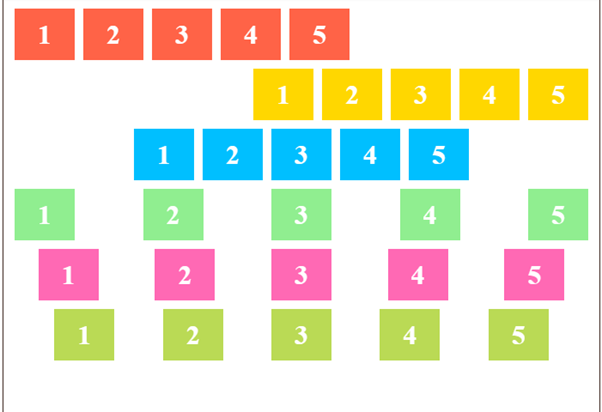
**center**: items are centered along the line

**space-between**: items are evenly distributed in the line; first item is on the start line, last item on the end line

**space-around:** items are evenly distributed in the line with equal space around them

**space-evenly**: items are distributed so that the spacing between any two adjacent alignment subjects, before the first alignment subject, and after the last alignment subject is the same

EX:



* The red list is set to flex-start
* The yellow is set to flex-end
* The blue is set to center.
* The green is set to space-between
* The pink is set to space-around
* The light green is set to space-evenly.

**ALIGN-ITEMS :**

The align-items property defines the default behavior for how items are laid out along the cross axis (perpendicular to the main axis).

The align-items property accepts following values:

**flex-start**: cross-start margin edge of the items is placed on the cross-start line

**flex-end**: cross-end margin edge of the items is placed on the cross-end line

**center**: items are centered in the cross-axis

**stretch (default**): stretch to fill the container (still respect min-width/max-width)

**short hand property :**

Ø **flex-flow : <flex-direction> <flex-wrap>**

This is a shorthand for the flex-direction and flex-wrap properties, which together define the flex container’s main and cross axes. The default value is row nowrap.

***FLEX WRAP :***

**By default, flex items will all try to fit onto one line. You can change that and allow the items to wrap as needed with this property.**

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

**nowrap (default): all flex items will be on one line**

**wrap: flex items will wrap onto multiple lines, from top to bottom.**

**wrap-reverse: flex items will wrap onto multiple lines from bottom to top.**

Order Property :

**>** **The order CSS property sets the order to lay out an item in a flex.**

**>we have to give order property inside the flex-item to order them.**

**Syntax : order: 5;**

ALIGN-SELF :

It makes possible to override the align-items value for specific flex items(child).

NOTE: The align-self property accepts the same values as the align-items

**FLEX-GROW :**

This property specifies how much of the remaining space in the flex container should be assigned to the item (the flex grow factor).

Ø It accepts a unitless value.

**FLEX-SHRINK:**

It specifies the “flex shrink factor”, which determines how much the flex item will shrink relative to the rest of the flex items in the flex container when there isn’t enough space on the row.

CSS ANIMATIONS :

**>CSS animations** make it possible to animate transitions from one CSS style configuration to another.

WHAT IS EXACTLY ANIMATIONS :

**Animation** is a method in which [figures](https://en.wikipedia.org/wiki/Image) are manipulated to appear as [moving images](https://en.wikipedia.org/wiki/Motion_picture).

ADVANTAGE OF CSS ANIMATIONS

1. EASY TO USE SIMPLE ANIMATION
2. IT PERFORM SMOOTH ON UNDER MODERATE SYSTEM LOAD

[**animation-name**](https://developer.mozilla.org/en-US/docs/Web/CSS/animation-name)

Specifies the name of the [@keyframes](https://developer.mozilla.org/en-US/docs/Web/CSS/@keyframes) at-rule describing the animation’s keyframes.

[**animation-duration**](https://developer.mozilla.org/en-US/docs/Web/CSS/animation-duration)

Configures the length of time that an animation should take to complete one cycle.

[**animation-timing-function**](https://developer.mozilla.org/en-US/docs/Web/CSS/animation-timing-function)

Configures the timing of the animation; that is, how the animation transitions through keyframes, by establishing acceleration curves.

[**animation-delay**](https://developer.mozilla.org/en-US/docs/Web/CSS/animation-delay)

Configures the delay between the time the element is loaded and the beginning of the animation sequence.

[**animation-iteration-count**](https://developer.mozilla.org/en-US/docs/Web/CSS/animation-iteration-count)

Configures the number of times the animation should repeat; you can specify infinite to repeat the animation indefinitely.

[**animation-direction**](https://developer.mozilla.org/en-US/docs/Web/CSS/animation-direction)

Configures whether or not the animation should alternate direction on each run through the sequence or reset to the start point and repeat itself.

[**animation-fill-mode**](https://developer.mozilla.org/en-US/docs/Web/CSS/animation-fill-mode)

Configures what values are applied by the animation before and after it is executing.

-----------------------------------------------------------------------------------------------------------------------**CSS FLEX BOX** :

Flex box is a one-dimensional layout method for laying out items in rows or columns.

**THE FLEX MODEL :**

**Display : Flex**

It is used to apply flex on child elements. This property must be used in parent element.

FLEX-DIRECTION :

Ø The flex-direction property specifies the direction of the flex- items.

Ø We have to write in parent container.

**flex-direction**: row|row-reverse|column|column-reverse

1) row -> Default value. The flexible items are displayed horizontally, as a row

2) row-reverse: -> Same as row, but in reverse order

3) column : ->The flexible items are displayed vertically, as a column

4) column-reverse:-> Same as column, but in reverse order

**Justify content :**

The CSS justify-content property defines how the browser distributes space between and around content items along the main-axis of a flex container,

**Justify-content: center | space-between | space-around |space-evenly | flex-start | flex-end**

**flex-start** (default): items are packed toward the start line

**flex-end**: items are packed toward to end line

**center**: items are centered along the line

**space-between**: items are evenly distributed in the line; first item is on the start line, last item on the end line

**space-around:** items are evenly distributed in the line with equal space around them

**space-evenly**: items are distributed so that the spacing between any two adjacent alignment subjects, before the first alignment subject, and after the last alignment subject is the same

EX:

* The red list is set to flex-start
* The yellow is set to flex-end
* The blue is set to center.
* The green is set to space-between
* The pink is set to space-around
* The light green is set to space-evenly.

**ALIGN-ITEMS :**

The align-items property defines the default behavior for how items are laid out along the cross axis (perpendicular to the main axis).

The align-items property accepts following values:

**flex-start**: cross-start margin edge of the items is placed on the cross-start line

**flex-end**: cross-end margin edge of the items is placed on the cross-end line

**center**: items are centered in the cross-axis

**stretch (default**): stretch to fill the container (still respect min-width/max-width)

**short hand property :**

Ø **flex-flow : <flex-direction> <flex-wrap>**

This is a shorthand for the flex-direction and flex-wrap properties, which together define the flex container’s main and cross axes. The default value is row nowrap.

***FLEX WRAP :***

**By default, flex items will all try to fit onto one line. You can change that and allow the items to wrap as needed with this property.**

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

**nowrap (default): all flex items will be on one line**

**wrap: flex items will wrap onto multiple lines, from top to bottom.**

**wrap-reverse: flex items will wrap onto multiple lines from bottom to top.**

Order Property :

**>** **The order CSS property sets the order to lay out an item in a flex.**

**>we have to give order property inside the flex-item to order them.**

**Syntax : order: 5;**

ALIGN-SELF :

It makes possible to override the align-items value for specific flex items(child).

NOTE: The align-self property accepts the same values as the align-items

**FLEX-GROW :**

This property specifies how much of the remaining space in the flex container should be assigned to the item (the flex grow factor).

Ø It accepts a unitless value.

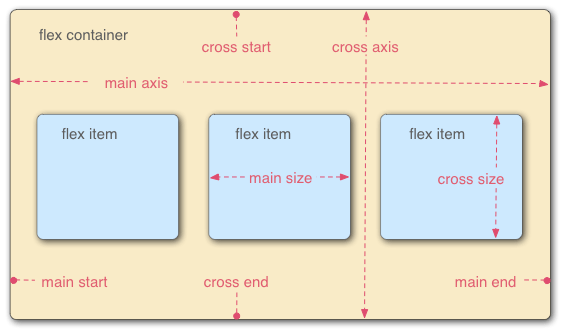
**FLEX-SHRINK:**

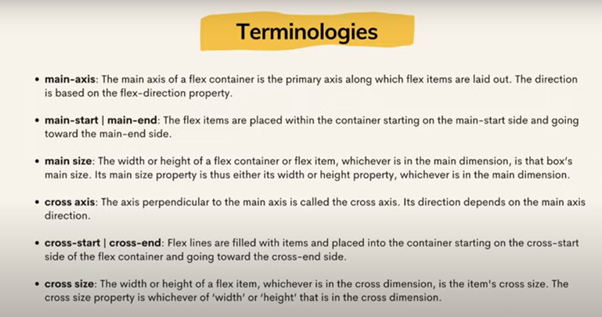
It specifies the “flex shrink factor”, which determines how much the flex item will shrink relative to the rest of the flex items in the flex container when there isn’t enough space on the row.

**CSS FLEX BOX** :

Flex box is a one-dimensional layout method for laying out items in rows or columns.

**THE FLEX MODEL :**

****

****

**Display : Flex**

It is used to apply flex on child elements. This property must be used in parent element.

FLEX-DIRECTION :

Ø The flex-direction property specifies the direction of the flex- items.

Ø We have to write in parent container.

**flex-direction**: row|row-reverse|column|column-reverse

1) row-> Default value. The flexible items are displayed horizontally, as a row

2) row-reverse: -> Same as row, but in reverse order

3) column : ->The flexible items are displayed vertically, as a column

4) column-reverse:-> Same as column, but in reverse order

**Justify content :**

The CSS justify-content property defines how the browser distributes space between and around content items along the main-axis of a flex container,

**Justify-content: center | space-between | space-around |space-evenly | flex-start | flex-end**

**flex-start** (default): items are packed toward the start line

**flex-end**: items are packed toward to end line

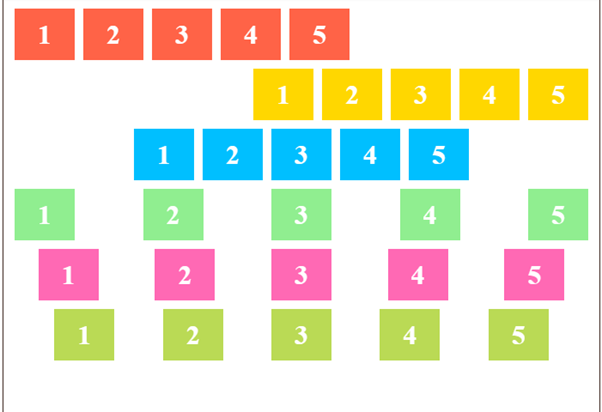
**center**: items are centered along the line

**space-between**: items are evenly distributed in the line; first item is on the start line, last item on the end line

**space-around:** items are evenly distributed in the line with equal space around them

**space-evenly**: items are distributed so that the spacing between any two adjacent alignment subjects, before the first alignment subject, and after the last alignment subject is the same

EX:



* The red list is set to flex-start
* The yellow is set to flex-end
* The blue is set to center.
* The green is set to space-between
* The pink is set to space-around
* The light green is set to space-evenly.

**ALIGN-ITEMS :**

The align-items property defines the default behavior for how items are laid out along the cross axis (perpendicular to the main axis).

The align-items property accepts following values:

**flex-start**: cross-start margin edge of the items is placed on the cross-start line

**flex-end**: cross-end margin edge of the items is placed on the cross-end line

**center**: items are centered in the cross-axis

**stretch (default**): stretch to fill the container (still respect min-width/max-width)

**short hand property :**

Ø **flex-flow : <flex-direction> <flex-wrap>**

This is a shorthand for the flex-direction and flex-wrap properties, which together define the flex container’s main and cross axes. The default value is row nowrap.

***FLEX WRAP :***

**By default, flex items will all try to fit onto one line. You can change that and allow the items to wrap as needed with this property.**

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

**nowrap (default): all flex items will be on one line**

**wrap: flex items will wrap onto multiple lines, from top to bottom.**

**wrap-reverse: flex items will wrap onto multiple lines from bottom to top.**

Order Property :

**>** **The order CSS property sets the order to lay out an item in a flex.**

**>we have to give order property inside the flex-item to order them.**

**Syntax : order: 5;**

ALIGN-SELF :

It makes possible to override the align-items value for specific flex items(child).

NOTE: The align-self property accepts the same values as the align-items

**FLEX-GROW :**

This property specifies how much of the remaining space in the flex container should be assigned to the item (the flex grow factor).

Ø It accepts a unitless value.

**FLEX-SHRINK:**

It specifies the “flex shrink factor”, which determines how much the flex item will shrink relative to the rest of the flex items in the flex container when there isn’t enough space on the row.

---------------------------------------------------------------------------------------------

MediaQueries :

Media queries are useful when you want to modify your site or app depending on a device's general type (such as print vs. screen) or specific characteristics and parameters (such as screen resolution or browser viewport width).

CSS ANIMATIONS :

**>CSS animations** make it possible to animate transitions from one CSS style configuration to another.

WHAT IS EXACTLY ANIMATIONS :

**Animation** is a method in which [figures](https://en.wikipedia.org/wiki/Image) are manipulated to appear as [moving images](https://en.wikipedia.org/wiki/Motion_picture).

ADVANTAGE OF CSS ANIMATIONS

1. EASY TO USE SIMPLE ANIMATION
2. IT PERFORM SMOOTH ON UNDER MODERATE SYSTEM LOAD

[**animation-name**](https://developer.mozilla.org/en-US/docs/Web/CSS/animation-name)

Specifies the name of the [@keyframes](https://developer.mozilla.org/en-US/docs/Web/CSS/@keyframes) at-rule describing the animation’s keyframes.

[**animation-duration**](https://developer.mozilla.org/en-US/docs/Web/CSS/animation-duration)

Configures the length of time that an animation should take to complete one cycle.

[**animation-timing-function**](https://developer.mozilla.org/en-US/docs/Web/CSS/animation-timing-function)

Configures the timing of the animation; that is, how the animation transitions through keyframes, by establishing acceleration curves.

[**animation-delay**](https://developer.mozilla.org/en-US/docs/Web/CSS/animation-delay)

Configures the delay between the time the element is loaded and the beginning of the animation sequence.

[**animation-iteration-count**](https://developer.mozilla.org/en-US/docs/Web/CSS/animation-iteration-count)

Configures the number of times the animation should repeat; you can specify infinite to repeat the animation indefinitely.

[**animation-direction**](https://developer.mozilla.org/en-US/docs/Web/CSS/animation-direction)

Configures whether or not the animation should alternate direction on each run through the sequence or reset to the start point and repeat itself.

[**animation-fill-mode**](https://developer.mozilla.org/en-US/docs/Web/CSS/animation-fill-mode)

Configures what values are applied by the animation before and after it is executing.

Pseudo class is a selectors [in pseudo class there only one :] is a keyword added to a selector

that specifies a special state of the selected elements.

Syntax: selectors:pseudo class

{ Property:value;}

 Anchor pseudo class

Represents the state of links as unvisited,visited or currently selected.

Anchoe also enables you to activate the html elements or apply a speacified style to an

element when the mouse pointer is kept over.

The anchor pseudo class include the following :

 :link[applies styles to non visitied links]

 :visited[applies styles to visited links]

 :hover[ to an element over which the mouse-pointer moves]

 :active[applies styles to an active element]

{visited=purple, unvisited=blue, active=red}

TRANSFORMATION

* AN OBJECT CAN CHANGE THE ORIGINAL FORM TO DESIRE FORM THIS IS CALLED AS TRANSFORMATION.

The transform property applies a transformation to an element. This property allows you to rotate, scale, skew, etc., elements.

BASICALLY THERE ARE THREE TYPE OF TRANSFORMATION IS THERE.

1.rotate – x-axis,y-axis and z-axis syntax: transform:rotate(value)

2.skew - x-axis,y-axis and z-axis syntax: transform:skew(value)

3.scale - x-axis,y-axis syntax: transform:scale(value)

<!DOCTYPE html>

<html>

<head>

<style>

div.a {

width: 150px;

height: 80px;

background-color: yellow;

transform: rotate(20deg);

}

div.b {

width: 150px;

height: 80px;

background-color: yellow;

transform: skewY(20deg);

}

div.c {

width: 150px;

height: 80px;

background-color: yellow;

transform: scaleY(1.5);

}

</style>

</head>

<body>

<h1>The transform Property</h1>

<h2>transform: rotate(20deg):</h2>

<div class="a">Hello World!</div>

<br>

<h2>transform: skewY(20deg):</h2>

<div class="b">Hello World!</div>

<br>

<h2>transform: scaleY(1.5):</h2>

<div class="c">Hello World!</div>

</body>

</html>

TRANSFORM-ORIGIN

The transform-origin property allows you to change the position of transformed elements.

2D transformations can change the x- and y-axis of an element. 3D transformations can also change the z-axis of an element.

**Note:** This property must be used together with the [transform](https://www.w3schools.com/cssref/css3_pr_transform.php) property.

Syntax: transform-origin:value;

1.center

2.bottom right

3.bottom left

4.top left

5.top right

TRANSITION

A transition occurs when a CSS property changes from one value to another value over a period of time.

|  |  |
| --- | --- |
| [*transition-property*](https://www.w3schools.com/cssref/css3_pr_transition-property.php) | Specifies the name of the CSS property the transition effect is for |
| [*transition-duration*](https://www.w3schools.com/cssref/css3_pr_transition-duration.php) | Specifies how many seconds or milliseconds the transition effect takes to complete |
| [*transition-timing-function*](https://www.w3schools.com/cssref/css3_pr_transition-timing-function.php) | Specifies the speed curve of the transition effect |
| [*transition-delay*](https://www.w3schools.com/cssref/css3_pr_transition-delay.php) | Defines when the transition effect will start |

TRANSLATE

The translate property allows you to change the position of elements.

The translate property defines x- and y-coordinates of an element in 2D. You can also define the z-coordinate to change position in 3D.

Coordinates can be given as only x-coordinates, x- and y-coordinates, or x-, y- and z-coordinates.

**Tip:** You need to define a value for CSS perspective property for the z-property to take effect.

**Note:** An alternative technique to translate an element is to use CSS transform property with CSS translate() function. The CSS translate property, as explained on this webpage, is arguably a simpler and more direct way to translate an element.

## ­­­­­­­­­­Grid Layout

The CSS Grid Layout Module offers a grid-based layout system, with rows and columns, making it easier to design web pages without having to use floats and positioning.

## Display Property

An HTML element becomes a grid container when its display property is set to grid.

<!DOCTYPE html>

<html>

<head>

<style>

.grid-container {

display: grid;

grid-template-columns: auto auto auto;

background-color: #2196F3;

padding: 10px;

}

.grid-item {

background-color: rgba(255, 255, 255, 0.8);

border: 1px solid rgba(0, 0, 0, 0.8);

padding: 20px;

font-size: 30px;

text-align: center;

}

</style>

</head>

<body>

<h1>display: grid</h1>

<p>Use display: grid; to make a block-level grid container:</p>

<div class="grid-container">

<div class="grid-item">1</div>

<div class="grid-item">2</div>

<div class="grid-item">3</div>

<div class="grid-item">4</div>

<div class="grid-item">5</div>

<div class="grid-item">6</div>

<div class="grid-item">7</div>

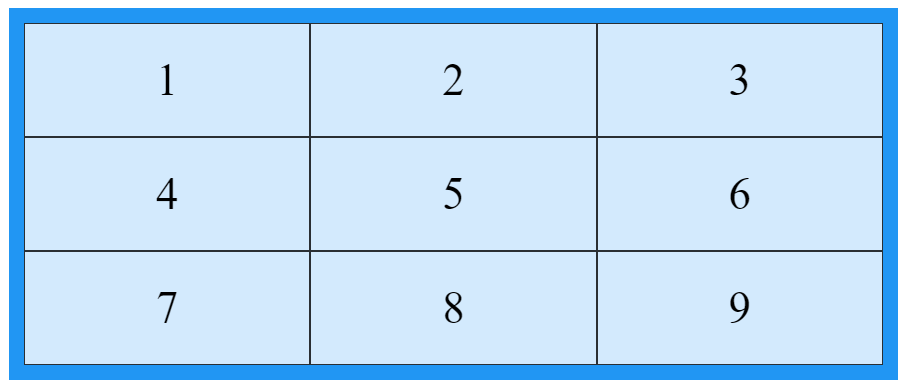
<div class="grid-item">8</div>

<div class="grid-item">9</div>

</div>

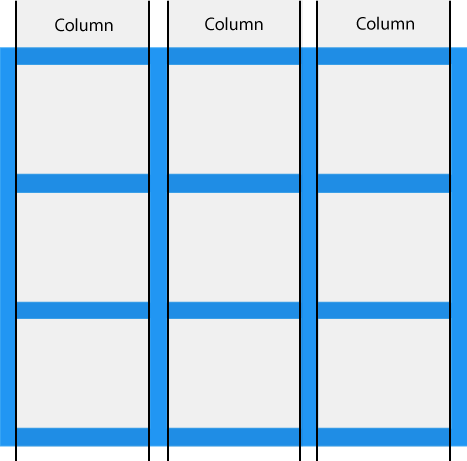
</body>

</html>



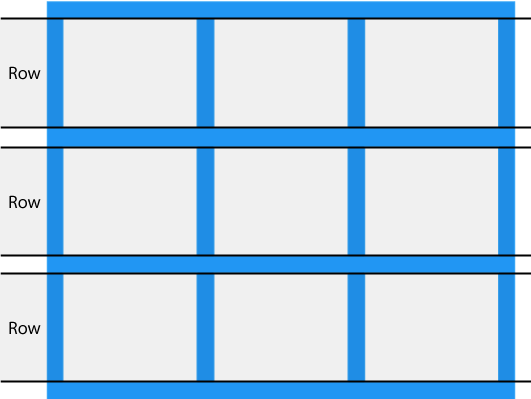
## Grid Columns

The vertical lines of grid items are called columns.



## Grid Rows

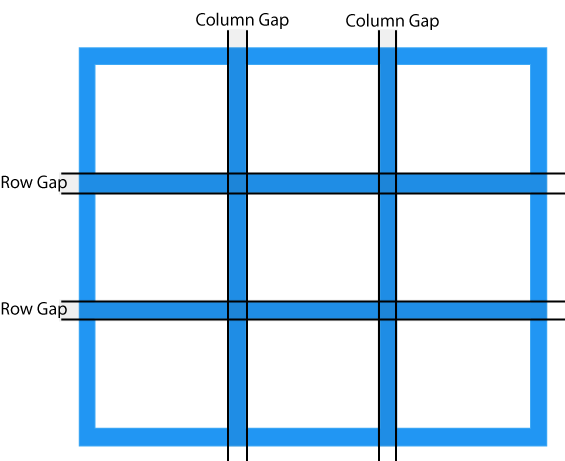
The horizontal lines of grid items are called rows.



.grid-container {  
  display: grid;  
**row-gap: 50px;**}

## Grid Gaps

The spaces between each column/row are called gaps.



.grid-container {  
  display: grid;  
**column-gap: 50px;**  
}

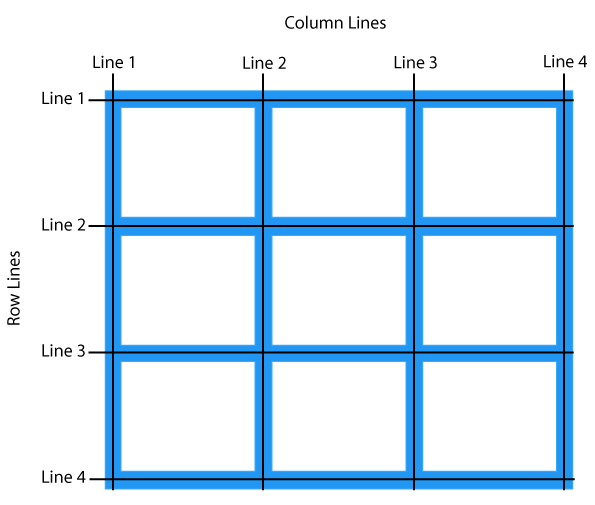
The gap property is a shorthand property for the row-gap and the column-gap properties:

.grid-container {  
  display: grid;  
**gap: 50px 100px;**}

Grid Lines

The lines between columns are called *column lines*.

The lines between rows are called *row lines*.



<!DOCTYPE html>

<html>

<head>

<style>

.grid-container {

display: grid;

grid-template-columns: auto auto auto;

gap: 10px;

background-color: #2196F3;

padding: 10px;

}

.grid-container > div {

background-color: rgba(255, 255, 255, 0.8);

text-align: center;

padding: 20px 0;

font-size: 30px;

}

.item1 {

grid-column-start: 1;

grid-column-end: 3;

}

</style>

</head>

<body>

<h1>Grid Lines</h1>

<div class="grid-container">

<div class="item1">1</div>

<div class="item2">2</div>

<div class="item3">3</div>

<div class="item4">4</div>

<div class="item5">5</div>

<div class="item6">6</div>

<div class="item7">7</div>

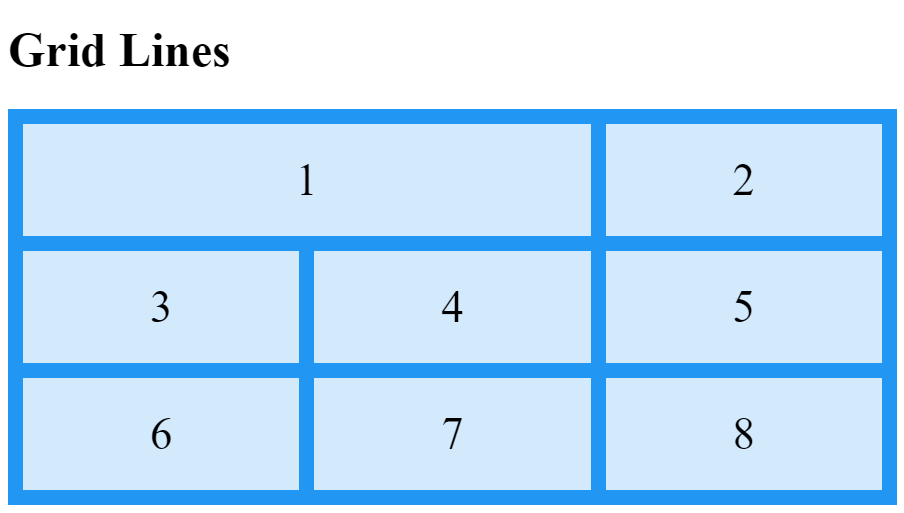
<div class="item8">8</div>

</div>

<p>You can refer to line numbers when placing grid items.</p>

</body>

</html>



<!DOCTYPE html>

<html>

<head>

<style>

.grid-container {

display: grid;

grid-template-columns: auto auto auto;

gap: 10px;

background-color: #2196F3;

padding: 10px;

}

.grid-container > div {

background-color: rgba(255, 255, 255, 0.8);

text-align: center;

padding: 20px 0;

font-size: 30px;

}

.item1 {

grid-row-start: 1;

grid-row-end: 3;

}

</style>

</head>

<body>

<h1>Grid Lines</h1>

<div class="grid-container">

<div class="item1">1</div>

<div class="item2">2</div>

<div class="item3">3</div>

<div class="item4">4</div>

<div class="item5">5</div>

<div class="item6">6</div>

<div class="item7">7</div>

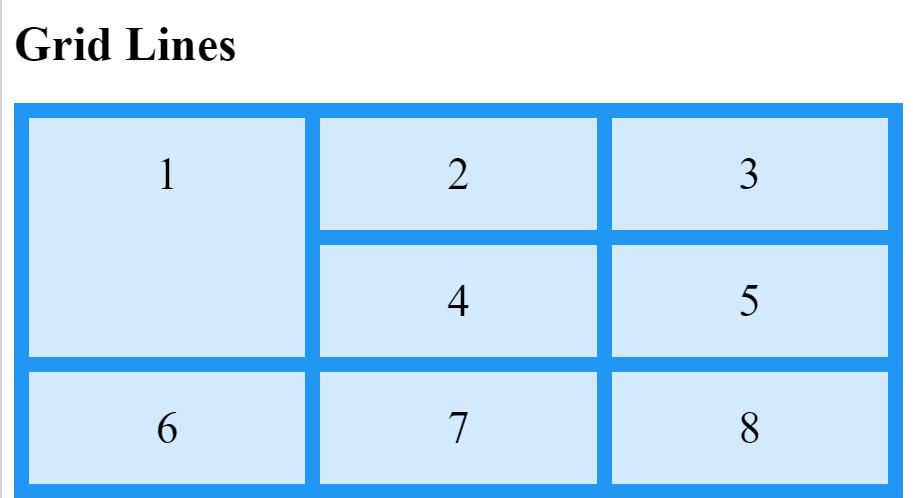
<div class="item8">8</div>

</div>

<p>You can refer to line numbers when placing grid items.</p>

</body>

</html>



**MEDIA QUERY**

The Media query in CSS is used to create a responsive web design. It means that the view of a web

page differs from system to system based on screen or media types. The breakpoint specifies for

what device-width size, the content is just starting to break or deform.

Media queries can be used to check many things:

width and height of the viewport

width and height of the device

Orientation

Resolution

A media query consist of a media type that can contain one or more expression which can be either

true or false. The result of the query is true if the specified media matches the type of device the

document is displayed on. If the media query is true then a style sheet is applied.

Syntax:

@media (min-width:value1) and (max-width:value2){

}

CSS ANIMATIONS :

**>CSS animations** make it possible to animate transitions from one CSS style configuration to

another.

WHAT IS EXACTLY ANIMATIONS :

**Animation** is a method in which figures are manipulated to appear as moving images.

ADVANTAGE OF CSS ANIMATIONS

1. EASY TO USE SIMPLE ANIMATION

2. IT PERFORM SMOOTH ON UNDER MODERATE SYSTEM LOAD

**animation-name**

Specifies the name of the @keyframes at-rule describing the animation’s

keyframes.

**animation-duration**

Configures the length of time that an animation should take to complete one cycle.

**animation-timing-function**

Configures the timing of the animation; that is, how the animation transitions

through keyframes, by establishing acceleration curves.

**animation-delay**

Configures the delay between the time the element is loaded and the beginning of

the animation sequence.

**animation-iteration-count**

Configures the number of times the animation should repeat; you can specify

infinite to repeat the animation indefinitely.

**animation-direction**

Configures whether or not the animation should alternate direction on each run

through the sequence or reset to the start point and repeat itself.

**animation-fill-mode**

Configures what values are applied by the animation before and after it is executing.

Pseudo class is a selectors [in pseudo class there only one :] is a

keyword added to a selector

that specifies a special state of the selected elements.

Syntax: selectors:pseudo class

{ Property:value;}

Anchor pseudo class

Represents the state of links as unvisited,visited or currently

selected.

Anchoe also enables you to activate the html elements or apply a

speacified style to an

element when the mouse pointer is kept over.

The anchor pseudo class include the following :

:link[applies styles to non visitied links]

:visited[applies styles to visited links]

:hover[ to an element over which the mouse-pointer moves]

:active[applies styles to an active element]

{visited=purple, unvisited=blue, active=red}

TRANSFORMATION

- AN OBJECT CAN CHANGE THE ORIGINAL FORM TO DESIRE FORM THIS IS CALLED

AS TRANSFORMATION.

The transform property applies a transformation to an element. This property

allows you to rotate, scale, skew, etc., elements.

BASICALLY THERE ARE THREE TYPE OF TRANSFORMATION IS THERE.

1.rotate – x-axis,y-axis and z-axis syntax: transform:rotate(value)

2.skew - x-axis,y-axis and z-axis syntax: transform:skew(value)

3.scale - x-axis,y-axis syntax: transform:scale(value)

<!DOCTYPE html>

<html>

<head>

<style>

div.a {

width: 150px;

height: 80px;

background-color: yellow;

transform: rotate(20deg);

}

div.b {

width: 150px;

height: 80px;

background-color: yellow;

transform: skewY(20deg);

}

div.c {

width: 150px;

height: 80px;

background-color: yellow;

transform: scaleY(1.5);

}

</style>

</head>

<body>

<h1>The transform Property</h1>

<h2>transform: rotate(20deg):</h2>

<div class="a">Hello World!</div>

<br>

<h2>transform: skewY(20deg):</h2>

<div class="b">Hello World!</div>

<br>

<h2>transform: scaleY(1.5):</h2>

<div class="c">Hello World!</div>

</body>

</html>

TRANSFORM-ORIGIN

The transform-origin property allows you to change the position of

transformed elements.

2D transformations can change the x- and y-axis of an element. 3D

transformations can also change the z-axis of an element.

**Note:** This property must be used together with the transform property.

Syntax: transform-origin:value;

1.center

2.bottom right

3.bottom left

4.top left

5.top right

TRANSITION

A transition occurs when a CSS property changes from one value to another value

over a period of time.

*transition-property* Specifies the name of the CSS property the transition effect is for

*transition-duration* Specifies how many seconds or milliseconds the transition effect takes

to complete

*transition-timingfunction*

Specifies the speed curve of the transition effect

*transition-delay* Defines when the transition effect will start

TRANSLATE

The translate property allows you to change the position of elements.

The translate property defines x- and y-coordinates of an element in 2D. You

can also define the z-coordinate to change position in 3D.

Coordinates can be given as only x-coordinates, x- and y-coordinates, or x-,

y- and z-coordinates.

**Tip:** You need to define a value for CSS perspective property for the zproperty

to take effect.

**Note:** An alternative technique to translate an element is to use

CSS transform property with CSS translate() function. The

CSS translate property, as explained on this webpage, is arguably a simpler

and more direct way to translate an element

**CSS Interview Questions**

1. **What is CSS?**

Answer: CSS (Cascading Style Sheets) is a stylesheet language used to describe the presentation of a document written in HTML or XML. It controls the layout, colors, fonts, and overall appearance of web pages.

1. **What is the difference between class selectors and ID selectors in CSS?**

Answer: Class selectors are used to select multiple elements and are denoted by a period (.), e.g., .classname. ID selectors are used to select a single unique element and are denoted by a hash (#), e.g., #idname.

1. **How can you include CSS in a web page?**

Answer: CSS can be included in three ways:

Inline CSS: Using the style attribute within HTML elements.

Internal CSS: Using a <style> tag within the <head> section of an HTML document.

External CSS: Using a <link> tag to link an external CSS file within the <head> section.

1. **What is the box model in CSS?**

Answer: The box model consists of four areas: content, padding, border, and margin. It defines how elements are structured and spaced on a web page.

1. **What are pseudo-classes in CSS?**

Answer: Pseudo-classes are keywords added to selectors that specify a special state of the selected elements. For example, :hover applies a style when the user hovers over an element.

1. **How does the CSS float property work?**

Answer: The float property is used to position an element to the left or right within its container, allowing text and inline elements to wrap around it. The possible values are left, right, none, and inherit.

1. **What is the difference between absolute, relative, fixed, and sticky positioning in CSS?**

Answer:

Absolute: The element is positioned relative to the nearest positioned ancestor or the initial containing block.

Relative: The element is positioned relative to its normal position in the document flow.

Fixed: The element is positioned relative to the browser window and does not move when the page is scrolled.

Sticky: The element toggles between relative and fixed positioning depending on the scroll position.

1. **What is Flexbox in CSS?**

Answer: Flexbox (Flexible Box Layout) is a layout model that allows you to design complex layouts more efficiently. It is a one-dimensional layout method for arranging items in rows or columns, providing control over the alignment, direction, order, and size of the items.

1. **What is CSS Grid Layout?**

Answer: CSS Grid Layout is a two-dimensional layout system for the web. It allows developers to create complex and responsive layouts using rows and columns.

1. **How do you create a responsive design in CSS?**

Answer: Responsive design can be achieved using media queries, flexible grid layouts, flexible images, and ensuring the viewport is properly configured. Media queries allow styles to be applied based on the screen size, orientation, and other properties of the viewing device.

1. **What are CSS preprocessors and name a few?**

Answer: CSS preprocessors are scripting languages that extend CSS by allowing variables, nested rules, and functions. They need to be compiled into regular CSS before being used in a web project. Examples include SASS (Syntactically Awesome Stylesheets), LESS, and Stylus.

1. **Explain the concept of CSS specificity.**

Answer: CSS specificity determines which styles are applied to an element when there are conflicting rules. It is calculated based on the number of type selectors, class selectors, and ID selectors in a rule. Inline styles have the highest specificity, followed by IDs, classes/attributes, and element selectors.

1. **What is the z-index property and how does it work?**

Answer: The z-index property determines the stack order of positioned elements (elements with position values of absolute, relative, fixed, or sticky). Elements with a higher z-index value will appear in front of those with lower values.

1. **How do you use CSS transitions and animations?**

Answer: CSS transitions allow you to change property values smoothly (over a given duration) from one state to another. Animations are more complex and allow you to create keyframes that define styles at different points during the animation sequence.

/\* Transition Example \*/

.example {

transition: background-color 0.5s ease-in-out;

}

/\* Animation Example \*/

@keyframes example {

from { background-color: red; }

to { background-color: yellow; }

}

.animate {

animation: example 3s infinite;

}

1. **What are CSS variables and how do you use them?**

Answer: CSS variables, also known as custom properties, allow you to store values that you can reuse throughout your stylesheet. They are defined with a -- prefix and accessed using the var() function.

:root {

--main-color: #3498db;

}

.example {

color: var(--main-color);

}

Q.What is css?

Q.What does cascading mean in css?

Q.What are the different ways to add the css in our document?

Q.Explain the advantages of css?

Q.List out the components of css style?

Q.Explain the types of selector in css ?

Q.What is the universal selector in css?

Q.What is the difference between id and class selector in css?

Q.Is css a case sensitive language?

Q. Which property is used to change the font face in css?

Q. How to use grouping in css?

Q. Explain the child selector in css?

Q. What is the use of float property in css?

Q. Which property is used to control the position in the background for an image?

Q. Mention the property name which is used for making the font oblique in css?

Q. List out all the media types in css?

Q. List all the font attributes in css?

Q. How can we eliminate the border around the linked images in the web page in css?

Q. List out the elements of the css box model?

Q. What is the use of z-index in css?

Q. How to lighten the font weight in css?

Q. Which css property is used for setting the type of cursor in css?

Q. List out any 5 properties of the cursor in css?

Q. List some properties that are added in css3?

Q. What is the difference between the display none and the visibility hidden in css?

Q. Which property is used for formatting the text in css?

Q. List out the possible value for attribute “position” in css?

Q. Which property is used to remove the underline below the link?

Q. Can I give more than one class to an html element?

Q. How to add the comment in css?

Q. Which property is used for aligning the text in a document ?

Q. What is meant by pseudo classes in css?

Q. How to give rounded corners in css3?

Q. List out the properties of rounded corners in css3?

Q. Which new backgrounds are added in css3?

Q. What do you mean by word wrapping in css?

Q. List out the properties of transition in css3?

Q. What is opacity in css3?

Q. List some css framework?

Q. How can css be integrated into a html page?

Q. What is meant by rgb stream?

Q. What is the difference between class and id?

Q. Define z-index?

Q. Name the media types allowed by css?

Q. How can you avoid the image repetition in css?

Q. How to fix the background in css or which property is used to avoid the background scroll?

Q. What is shorthand property in css?

Q. What is a box model in css?

Q. What is the difference between margin and padding in css?

Q. How can you center an element horizontally and vertically in css?

Q. How do media queries work in responsive web design with css?

Q. How can you add external fonts in a web page using css?

Q. Tell us about the use of a rule set?

Q. What is css specificity?

Q. What is the difference between rem, em, and px?

Q. What are the float properties in css?

Q. How is the border-box different from the content box?

Q. What is the grid system?

Q. What are the different ways to hide the element using css?

Q. What is the difference between css grid vs flexbox?

Q. Differentiate between absolute and relative position in css?

Q. What is the difference between display:flex and display:inline-flex?

Q. Does margin-top and margin-bottom have an effect on inline elements?

Q. What are the advantages and disadvantages of an external style sheet?

Q. What is the embedded stylesheet and how is it used?

Q. What does ! important mean in css?

Q. What is the difference between internal , external , and inline css?

Q. How do you change the text color of an element in css?

Q. What is the css:hover pseudo class used for?

Q. How can you create a fixed header or footer in css?

Q. Describe the purpose of the @important rule in css?

Q. How can you create a dropdown menu using only html and css?

Q. What are the combinators?

Q. What is property in css?