CSS3

CSS is the language we use to style a Web page.

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**

CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.

CSS Syntax:

```
h1{
        color:red;
    }
```

h1 => element selectors

color => property

red => value;

```
p {
  color: red;
  text-align: center;
}
```

How To Add/ Way of CSS:

- Inline CSS
- Internal CSS
- External CSS

Inline CSS

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

Add the style attribute to the relevant element.

Ex: welcome

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;

}
</style>
```

```
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

External CSS

The external CSS is the CSS linked to an HTML file using the <link> tag.

index.html

```
<html>
<head>
<link rel="stylesheet" href="style.css">
</head>
<body>
<p>Hello GFG</p>
</body>
</html>
```

Style.css

```
p {
  color: green;
}
```

How to Command the css

/* style */

CSS selectors: A CSS selector selects the HTML element(s) you want to style.

Categories:

- **Simple or Basic selectors**

- **Attribute selectors**
- **Combinator selectors** (select elements based on a specific relationship between them)
- **Pseudo-class selectors**
- **Pseudo-elements selectors**

Simple Selectors: - used to target specific HTML elements for style

These include selecting by:

1. element name (e.g., h1)
2. class (.class-Name)
3. ID (#id-Name)
4. universal (* for all elements).
5. Grouping selectors (h1,p,ul)

Element Selectors (h1): The element selector **selects HTML elements based on the element nam**

e.

Ex: h1, p, ul, li …. Etc.

```
p {
    text-align: center;
    color: red;
}
```

Class Selectors(.): The class **selector selects HTML elements with a specific class attribute**.

write a period (.) character, followed by the class name.

Ex:

<h1 class="center large">Red and center-aligned heading</h1>

```
.center {
        text-align: center;
        color: red;
}

.large{

    Font-size:55px;

}
```

ID Selectors (#): - The ID **selector selects HTML elements with a specific id attribute**.

write a period (#) character, followed by the id name.

Ex:

<h1 id="bg-color ">Red and center-aligned heading</h1>

```
#bg-color {
    background-color:red;

}
```

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

Class vs ID selectors

- **Class Selector (.):** Targets elements with a specified class attribute, **allowing multiple elements to share the same styling.**
- **ID Selector (#):** Targets a single element with **a unique ID attribute**, ensuring that **styles are applied to one specific element on the page.**

Universal Selector (*): Selects **all elements on the page and applies the same style universally.**

Ex:

```
* {
  text-align: center;
  color: blue;
}
```

Group selectors (,): The grouping selector **selects all the HTML elements with the same style definitions.**

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

h2 {
  text-align: center;
  color: red;
}

p {
  text-align: center;
  color: red;
}
```

```
h1, h2, p {
  text-align: center;
  color: red;
}
```

CSS Attribute Selector

Attribute Selector allows you to **select elements based on the presence, value, or specific characteristics of their attribute**.

1. [attribute] Selector: with the **specified attribute**

```
a[href] { color: 2px solid red; }
```

2. [attribute=”value”]: **exactly equal to the** specified value.

```
input[type="submit"] { background-color: green; color: white; }
```

3. [attribute~=”value”]: one of the words **matches the specified value**.

```
div[class~="green"] { border: 3px solid red; }
```

```
<div class="box large green"></div>
<div class="box small red"></div>
```

4.[attribute^=”value”] **starts** with the specified value.

```
a[href^="https"] { color: darkgreen; }
```

```
<a href="https://example.com">Visit Example</a>
```

5.[attribute$=”value”] **ends** with the specified value

```
div[class$="box"] {
border: 2px solid blue;
background-color: lightgray;
}
```

```
<div class="small-box">This is a small box</div>
<div class="large-box">This is a large box</div>
<div class="circle">This is not a box</div>
```

CSS Combinators

It defines the relationship between two selectors.

Paragraph 1 in the div.

Paragraph 2 in the div.

Paragraph 3 in the div.

Paragraph 4. Not in a div.

Paragraph 5. Not in a div.

Types of CSS Combinators

- Descendant combinator (space)
- Child combinator (>)
- Next sibling combinator (+)
- Subsequent-sibling combinator (~)

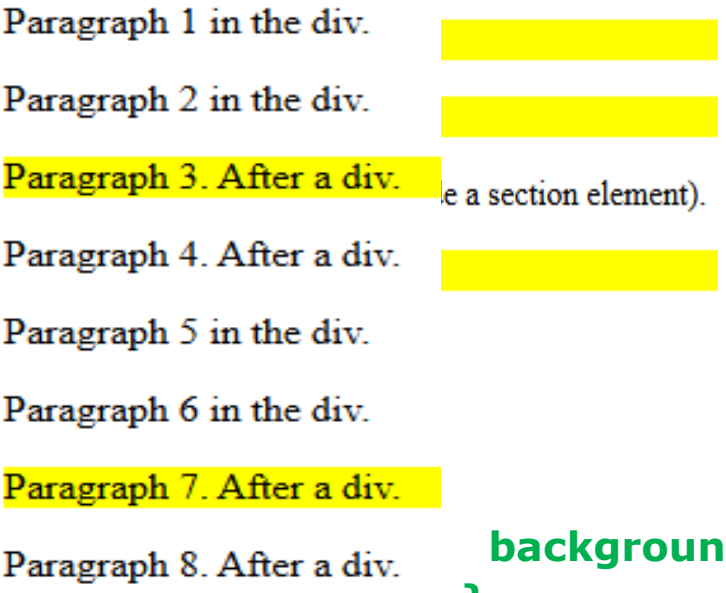
Descendant Selectors

(space): selects all elements that are **descendants of a specified element**. (**parent and all childs concepts**)

Ex:

```
<style>
div p {
  background-color: yellow;
}
</style>

<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>
```



Child Selectors (>): selects all elements that are the children of a specified element. (**Parent and only childs concepts**)

Ex:

```
<style>
div > p {
  background-color: yellow;
}
</style>

<div>
  <p>Paragraph 1 in the div.</p>
  <p>Paragraph 2 in the div.</p>
  <section>
    <p>Paragraph 3 in the div (inside a section element).</p>
  </section>
  <p>Paragraph 4 in the div.</p>
</div>
```

Next Sibling Selectors (+) select an element that is **directly after another specific element. (sibling concepts)**

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

Ex:

```
<style>
div + p {
    background-color: yellow;
        }
```

Paragraph 1.

Paragraph 2.

Paragraph 3.

Some code.

Paragraph 4.

```
    </style>
<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>
```

Subsequent-sibling Combinator (~) selects **all elements that are next siblings** of a specified element.

Ex:

```
<style>
div ~ p {
  background-color: yellow;
}
</style>


<p>Paragraph 1.</p>
<div>
  <p>Paragraph 2.</p>
</div>
<p>Paragraph 3.</p>
<code>Some code.</code>
<p>Paragraph 4.</p>
```

-
==========
==========
==========

CSS Comments

CSS comments are not displayed in the browser

Comments are ignored by browsers. A CSS comment is placed inside the <style> element, and starts with /* and ends with */

CSS Colors

Colors are specified using predefined color names, or RGB, HEXA, HSL, RGBA, HSLA values. (Text colors)

Color Names

Ex: red, green, yellow, blue, white, black

RGB Value

rgb (***red, green***, ***blue***)

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

To display black, rgb (0, 0, 0).

To display white, rgb(255, 255, 255).

RGBA Value

RGBA color values are an extension of RGB color values with an **alpha** channel - **which specifies the opacity/transprancy for a color.**

rgba(***red, green***, ***blue, alpha***)

The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (not transparent at all)

0, 0.1, 0.2, 0.3, 0.4,0.5,0.6,0.7,0.8,0.9,1

Ex:

rgba (255,99,71,0.5);

CSS HEX Colors

A hexadecimal color is specified with: #RRGGBB, where the RR (red), GG (green) and BB (blue) hexadecimal integers specify the components of the color.

*#**rrggbb*** **hexadecimal values between 00 and ff (same as decimal 0-255).**

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

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

Sometimes you will see a 3-digit hex code in the CSS source. #fc9; /* same as #ffcc99 */ #f0f; /* same as #ff00ff */ #b58; /* same as #bb5588 */

HEXA

CSS HSL Colors HSL stands for hue, saturation, and lightness.

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**

Ex: hsl(0, 100%, 50%)

hsla(*hue, saturation, lightness, alpha*)

The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (not transparent at all)

==========

Backgrounds

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

- background-image
- background-repeat
- background-attachment
- background-position
- background

```
body {
  background-color: lightblue;
}
```

Opacity / Transparency

The opacity property specifies the opacity/transparency of an element

```
div {
        background-color: green;
        opacity: 0.3;
}
```

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.

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

background-repeat

By default, the background-image property **repeats an image both horizontally and vertically.**

background-repeat: repeat-x;

background-repeat: repeat-y;

background-repeat: no-repeat;

background-position

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

left top
left center
left bottom
right top
right center
right bottom
center top
center center
center bottom

x% y% - 10% 40% .

xpos ypos – 50px 100px

background-attachment

The **background-attachment** property **specifies whether the background image should scroll or be fixed**

background-attachment: scroll; image will scroll by default

background-attachment: fixed; image will fixed

CSS background - Shorthand

```
body {
  background: #ffffff url("img_tree.png") no-repeat right top;
}
```

1 – color , 2 – image , 3 – repeat , 4 - positions

==========
==========

background-origin

property specifies the origin position (the background positioning area) of a background image.

padding-box – By Default.

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

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

background-origin: content-box, border-box;

background: url(img_tree.gif), url(paper.gif);

==========

background-size

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

the keyword syntax ("auto", "cover" and "contain")

```
background-size: 100% 100%;
background-size: contain;
background-size: cover;
```

Gradient-colors

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)**

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

Direction - Top to Bottom (this is default)

```
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.**

EG;

background-image: linear-gradient(*angle*, *color-stop1*, *color-stop2*);

```
background-image: linear-gradient(red 10%, yellow, green);
```

```
 background-image: linear-gradient(to right, red,orange,yellow,green,blue,indigo,violet);
```

```
background-image: repeating-linear-gradient(red, yellow 10%, green 20%);
```

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.

```
/* background-image: radial-gradient(red,green,orange);
*/
```

```
/* background-image: radial-gradient(red 0%,blue 50%);
*/

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

Syntax

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

background-image: radial-gradient(closest-side at 60% 55%, red, yellow, black);

background-image: radial-gradient(farthest-side at 60% 55%, red, yellow, black);

background-image: radial-gradient(closest-corner at 60% 55%, red, yellow, black);

background-image: radial-gradient(farthest-corner at 60% 55%, red, yellow, black);

background-image: repeating-radial-gradient(red, yellow 10%, green 15%);

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*], ...);

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

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

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

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

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

```
background-image: conic-gradient(red, yellow, green, blue, black);
  border-radius: 50%;
```

```
 background-image: conic-gradient(red 0deg, red 90deg, yellow 90deg, yellow 180deg, green 180deg, green 270deg, blue 270deg);
  border-radius: 50%;
```

```
background-image: conic-gradient(from 90deg, red, yellow, green);
```

```
background-image: conic-gradient(at 60% 45%, red, yellow, green);
```

```
background-image: repeating-conic-gradient(red 10%, yellow 20%);
  border-radius: 50%;
```

The **background-blend-mode Property** defines how the element's background image should blend with each other and with the element's background-color.

```
background-blend-mode:
normal|multiply|screen|darken|lighten|
color-dodge|saturation|difference|luminosity|overlay;
```

CSS Units (Sizing concepts)
CSS has several different units for expressing a length.

CSS properties take "length" values, such as width, margin, padding, font-size, etc.

There are two types of length units: **absolute** and **relative**.

Absolute Units

The absolute length units are fixed

1. Cm – centimetres,
2. mm - millimetres,
3. in - inches (1in = 96px = 2.54cm)
4. px * pixels (1px = 1/96th of 1in)
5. pt points (1pt = 1/72 of 1in) .
6. pc picas (1pc = 12 pt)

Relative Units

7. Em - Relative to the Parent Element
8. Rem - Relative to the root element
9. Vw - Relative to the width of the viewport* (viewport width)
10. Vh - Relative to 1% of the height of the viewport* (viewport height)

11. Vmin - Relative to 1% of viewport's* smaller dimension
12. Vmax - Relative to 1% of viewport's* larger dimension
13. % - Relative to the parent element (screen size)

CSS Margins (Spacing concepts)

Margins are used to **create space around elements, outside** of any **defined borders.**

each side of an element (**top, right, bottom, and left**).

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

Values

auto - the **browser calculates the margin**

length - specifies a margin **in px, pt, cm, etc.**

% - specifies a margin **in % of the width** of the containing element

inherit specifies that the margin should be inherited from **the parent element**

```
p {
  margin-top: 100px;
  margin-bottom: 100px;
  margin-right: 150px;
  margin-left: 80px;
}
```

Margin - Shorthand

- **margin: 25px 50px 75px 100px;**

    - top margin is 25px
    - right margin is 50px
    - bottom margin is 75px
    - left margin is 100px
- **margin: 25px 50px 75px;**
    - top margin is 25px
    - right and left margins are 50px
    - bottom margin is 75px
- **margin: 25px 50px;**
    - top and bottom margins are 25px
    - right and left margins are 50px
- **margin: 25px;**
    - all four margins are 25px

box center alignment :

```
div{
    width: 300px;
    margin:0 auto;
}
```

CSS Padding

Padding is used to **create space inside an element's content, inside** of any defined borders.

each side of an element (top, right, bottom, and left).

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

Values

auto - the browser calculates the margin

length - specifies a margin **in px, pt, cm, etc.**

% - specifies a margin **in % of the width** of the containing element

inherit specifies that the margin should be inherited from **the parent element**

```
div {
  padding-top: 50px;
  padding-right: 30px;
  padding-bottom: 50px;
  padding-left: 80px;
}
```

Padding - Shorthand

- **padding: 25px 50px 75px 100px;**
  - top padding is 25px
  - right padding is 50px
  - bottom padding is 75px
  - left padding is 100px
- **padding: 25px 50px 75px;**
  - top padding is 25px
  - right and left paddings are 50px
  - bottom padding is 75px
- **padding: 25px 50px;**
  - top and bottom paddings are 25px
  - right and left paddings are 50px
- **padding: 25px;**
  - all four paddings are 25px

Padding and Element Width **box-sizing: border-box; (width + padding size will increase so add border-box)**

CSS Borders

The CSS border properties allow you to specify the style, width, and color of an element's border.

CSS Border Style

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

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

***border-style property can have from one to four values (for the top border, right border, bottom border, and the left border).**

The border-width property specifies the width of the four borders. set as a specific size (in px, pt, cm, em, etc) three pre-defined values: thin, medium, or thick:

```
border-width: 25px 10px 4px 35px;
```

The border-color property is used to set the color of the four borders.

```
 border-color: red green blue yellow;
```

```
p {
  border-top-style: dotted;
```

```
  border-right-style: solid;
  border-bottom-style: dotted;
  border-left-style: solid;
}

p {
  border-style: dotted solid double dashed;
}

/* Three values */
p {
  border-style: dotted solid double;
}

/* Two values */
p {
  border-style: dotted solid;
}

/* One value */
p {
  border-style: dotted;
}
```

```
p{
   /* border */

   /* border-style: dashed; */
   /* border-style: dotted; */
   /* border-style: double; */
   /* border-style:groove; */
   /* border-style:hidden; */
   /* border-style:inset; */
   /* border-style: outset; */
   /* border-style:none; */
   /* border-style:ridge; */
   /* border-style:solid; */
   /* border-width: 55px; */
   /* border-color: blueviolet; */
```

```
  /* border-radius: 50px; */
 /* border-top-style: dashed;
 border-top-width: 30px;
 border-top-color: blue;

 border-right-style: groove;
 border-right-width: 30px;
 border-right-color: rebeccapurple;

 border-bottom-style: double;
 border-bottom-width: 30px;
 border-bottom-color: palegreen;

 border-left-style: outset;
 border-left-width: 30px;
 border-left-color: peru; */

 /* shorthand */
 /* border-top: 10px dotted darkblue;
 border-right: 30px double darkcyan;
 border-bottom: 30px ridge darkmagenta;
 border-left: 30px groove darkred; */

 /* shorthand */
 border: 5px solid darkblue;

 /* border-top-left-radius: 10px;
 border-top-right-radius: 20px;
 border-bottom-right-radius: 40px;
 border-bottom-left-radius: 30px; */
 border-radius: 50%;

  width: 200px;
}
```

CSS Border – Shorthand

```
p {
  border: 5px solid red;
}

p {
  border-left: 6px solid red;
}
```

The border-radius property is used to add rounded borders to an element:

```
p {
  border: 2px solid red;
  border-radius: 5px;
}
```

CSS Outline

An outline is a line drawn outside the element's border.

- outline-style – dotted, dashed. ref in borders
- outline-color – red, green, yellow
- outline-width – thin, thick, medium
- outline-offset - adds space between an outline and the edge/border of an element
- outline - outline: 5px solid yellow;

```
    /* outline */

     /* outline-style: dashed;
     outline-style: dotted;
     outline-style: double;
     outline-style: groove;
     outline-style: hidden;
     outline-style: inset;
     outline-style: none;
     outline-style: outset;
     outline-style: ridge;
     outline-style: solid; */
     /* outline-style: unset; */

```

```
    /* outline-width: 30px;

    outline-color: #2be25f7e; */

    outline: 5px double darkgreen;

    outline-offset: 30px;
```

CSS Text

Text Color - color

Text Alignment and Text Direction

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

  left alignment is default, text-align: center, left, right, justify

- text-align-last – right, center, left, justify
- direction
- unicode-bidi

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

- vertical-align - property sets the **vertical alignment of an element. (img)**

  sub, super

Text Decoration - property is used to add a decoration line to text.

- text-decoration-line - overline, line-through, underline,

- text-decoration-line: overline underline
- text-decoration-color - red, green
- text-decoration-style - solid, double, dotted, dashed, wavy
- text-decoration-thickness - 5px
- text-decoration - text-decoration: underline red double 5px;
- text-decoration: none;

Text Transformation - property is used to specify **uppercase and lowercase letters in a text.**

text-transform: uppercase, lowercase, capitalize;

Text Spacing

- text-indent: 50px - property is used to specify the **indentation of the first line of a text**
- letter-spacing: 2px, -2px; = property is used to specify **the space between the characters in a text.**
- line-height:0.8, 1.8 => property is used to specify **the space between lines**:
- word-spacing: 10px, -2px => property is used to specify **the space between the words in a text.**
- white-space: nowrap property specifies **how white-space inside an element is handled.**

Text Shadow

- The text-shadow property adds shadow to text.
- text-shadow: 2px 2px; **horizontal** shadow (2px) and the **vertical** shadow (2px):
- text-shadow: 2px 2px red; **(red)** to the shadow:
- text-shadow: 2px 2px 5px red; a **blur effect (5px)** to the shadow:
- text-shadow: 0 0 3px #ff0000, 0 0 5px #0000ff;

- text-shadow: 5px 17px 4px #110f0fa1, 12px -16px 8px red, -20px -1px 4px #1b682ee6;

-
- box-shadow: 6px 14px 20px 20px #acacb3;
- 1 – x axis
- 2 – y axis
- 3 – blur
- 4 – spread
- 5 - color

CSS Fonts – important

1. **Serif** fonts **have a small stroke at the edges of each letter.** They create a **sense of formality and elegance.**
2. **Sans-serif** fonts have **clean lines (no small strokes attached).** They create **a modern and minimalistic look.**
3. **Monospace** fonts - here all the **letters have the same fixed width.** They **create a mechanical look.**
4. **Cursive** fonts **imitate human handwriting.**
5. **Fantasy** fonts are **decorative/playful fonts.**

Difference Between Serif and Sans-serif Fonts

Serif	Times New Roman Georgia Garamond

Sans-serif	Arial Verdana Helvetica
Monospace	Courier New Lucida Console Monaco
Cursive	Brush Script MT Lucida Handwriting
Fantasy	Copperplate Papyrus

font-family property to **specify the font of a text.**

font-style property is mostly used to **specify italic text**.
Normal , italic , oblique

font-weight property **specifies the weight of** a font:
normal, bold , lighter ,
100,200,300,400,500,600,700,800,900

font-variant property **specifies -text should be displayed in a small-caps font**.

small-caps font, **all lowercase letters are converted to uppercase letters**. Normal , small-caps;

font-size property sets **the size of the text.**

font: italic small-caps bold 12px/30px Georgia, serif;

Google Fonts - take fonts from google

The CSS height and width properties are used to set the height and width of an element.

The CSS max-width property is used to set the maximum width of an element.

CSS Box Model

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

- **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 Pseudo-classes selectors

Syntax:

```
selector:pseudo-class {
  property: value;
}
```

a – hyperlink , ancher, a, link

```
a:link {
  background-color: yellow;
}

a:visited {
```

```
  background-color: cyan;
}

a:hover {
  background-color: lightgreen;
}

a:active {
  background-color: hotpink;
}

a:link, a:visited {
  background-color: #f44336;
  color: white;
 }

a:hover, a:active {
  background-color: red;
}

a.highlight:hover {
  color: #ff0000;
}
```

:first-child pseudo-class matches a specified element that is the first child of another element.

:checked

input:checked
option:checked

```
input[type="radio"]:checked {
  box-shadow: 0 0 5px 3px blue;
}
input[type="checkbox"]:checked {
  box-shadow: 0 0 5px 3px maroon;
}
option:checked {
  color: blue;
```

```
  background-color: pink;
}

input:optional {
  background-color: lightgreen;
}

input:required {
  background-color: pink;
  border-color: red;
}


:nth-child()
li:nth-child(2) {
  background-color: lightgreen;
}
p:nth-child(odd) {
  background-color: red;
}

p:nth-child(even) {
  background: lightgreen;
}
p:nth-child(3n+1) {  n=>0,1,2,3,4,5
  background-color: red;
}
p:nth-child(3n-1) {
  background-color: red;
}

p:nth-last-child(3n+0) {

  background-color: red;

}
```

Cursor

```
<span style="cursor: auto">auto</span><br>
<span style="cursor:
crosshair">crosshair</span><br>
<span style="cursor: default">default</span><br>
<span style="cursor: e-resize">e-
resize</span><br>
<span style="cursor: help">help</span><br>
<span style="cursor: move">move</span><br>
<span style="cursor: n-resize">n-
resize</span><br>
<span style="cursor: ne-resize">ne-
resize</span><br>
<span style="cursor: nw-resize">nw-
resize</span><br>
<span style="cursor: pointer">pointer</span><br>
<span style="cursor:
progress">progress</span><br>
<span style="cursor: s-resize">s-
resize</span><br>
<span style="cursor: se-resize">se-
resize</span><br>
<span style="cursor: sw-resize">sw-
resize</span><br>
<span style="cursor: text">text</span><br>
<span style="cursor: w-resize">w-
resize</span><br>
<span style="cursor: wait">wait</span>
```

```
cursor: url('icons8-mouse-scrolling-50.png') , auto;
```

CSS Lists

```
list-style-type: circle , square , upper-roman , lower-
alpha

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

list-style-position: outside , inside;
```

list-style-type: none;

list-style: square inside url("sqpurple.gif");
shorthand

CSS Layout - The display Property

The display property is the most important CSS property for controlling layout.

display: none;

display: inline;

display: block;

display: inline-block;

the element will still take up the same space as before. The element will be hidden, but still affect the layout:

visibility: hidden;

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

- Static - normal
- relative - is positioned relative to its normal position.
- Fixed - is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled.
- Absolute - document body,
- sticky -is positioned based on the user's scroll position.

Elements are then positioned using the top, bottom, left, and right properties.

```
top: 80px;
right: 0;
```

left, right

The z-index property specifies the stack order of an element.

CSS Pseudo-elements

```
selector::pseudo-element {
  property: value;
}
```

::first-line pseudo-element is used to add a special style to the first line of a text.

::first-letter pseudo-element is used to add a special style to the first letter of a text.

::before pseudo-element can be used to insert some content before the content of an element. Content: url()

::after pseudo-element can be used to insert some content after the content of an element.

::marker pseudo-element selects the markers of list items.

::selection pseudo-element matches the portion of an element that is selected by a user.
::selection: color, background, cursor, and outline.

==========

Display Property

It is used to create the layout the Html elements.

Property:

Display: inline; -> it is used to change block level to inline level , it doesn't affect the width & height

: block; -> it is used to change inline level to block level ,it will affect the width & height

: inline-block; -> both we can use , and also affect the width & height

:none; -> hidden the layout of the html element

:table;

:flex;

:inline-flex;

:grid;

:inline-grid;

CSS Transition

It is used to transition the element smoothly movement.

While hover the html element , it will affect the transitions.

Property

1. **Transition-duration ****
2. **Transition-delay**
3. **Transition-timing-function**
4. **Transition-property**
5. **Transition // shorthand**

CSS3 Transition Properties

- The following table lists all the transition prope

Property	Description
transition	A shorthand property for setting the four into a single property
transition-delay	Specifies when the transition effect will s
transition-duration	Specifies how many seconds or milliseco takes to complete
transition-property	Specifies the name of the CSS property t
transition-timing-function	Specifies the speed curve of the transitio

6. **Transition-timing-function: ease, ease-in, ease-out, linear, ease-in-out, cubic-baziear();**

- ease - star slow, then fast , then end slow
- ease-in - start slow,then end fast
- ease-out - start fast ,then end slow
- ease-in-out - both ease-in,ease-out
- linear - same speed maintain
- cubic-bazear() -> customize work

1.

CSS OVERFLOW

It is used to handle the overflow content by overflow property

Overflow: hidden;

: visible;

Scroll;

Overflow-x : hidden;

: visible;

Scroll;

Overflow-y: hidden;

: visible;

Scroll;

Auto;

Float

It is used to floating the element and then create the layout left or right

Property

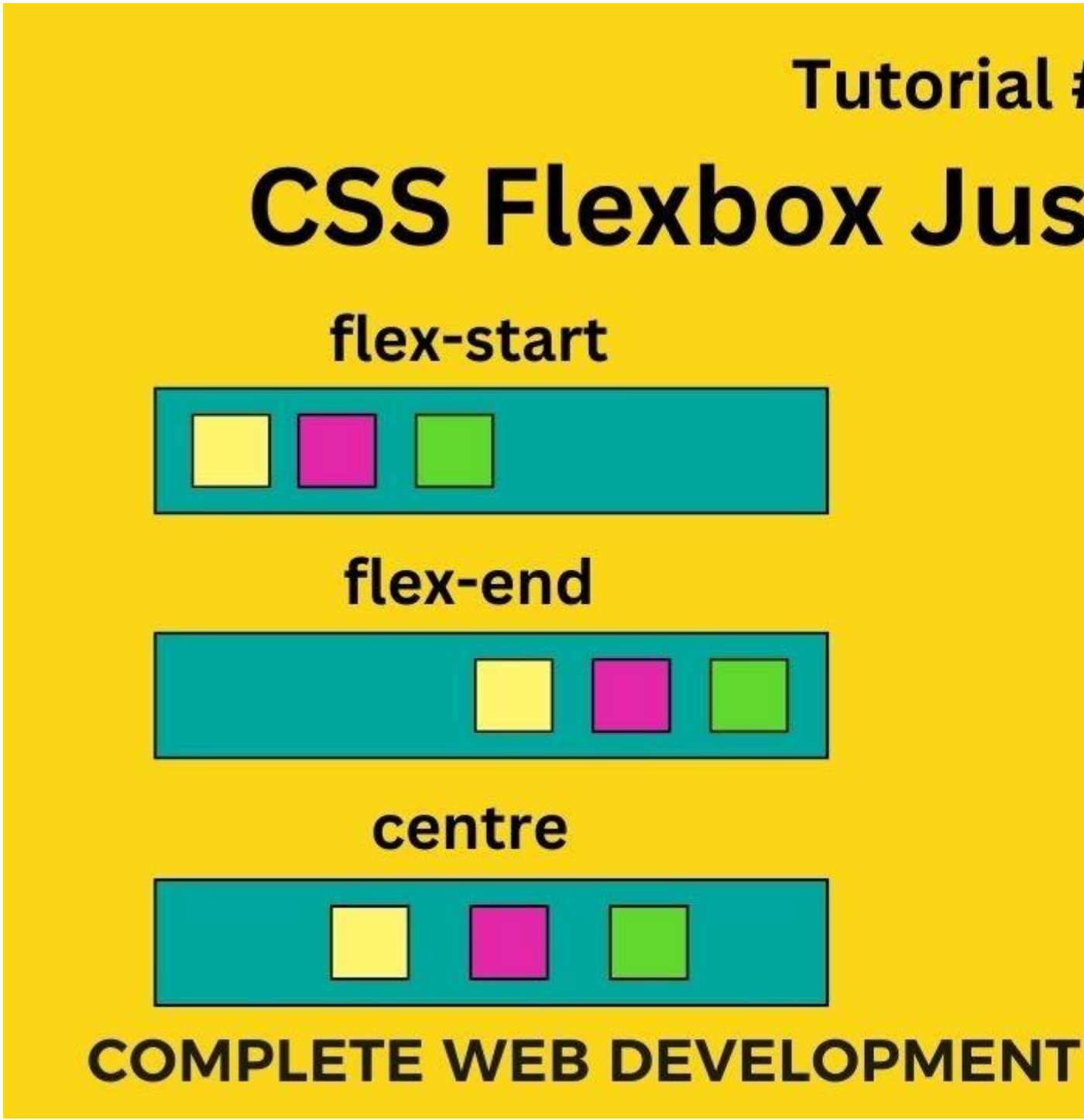
Float: left

Float: right

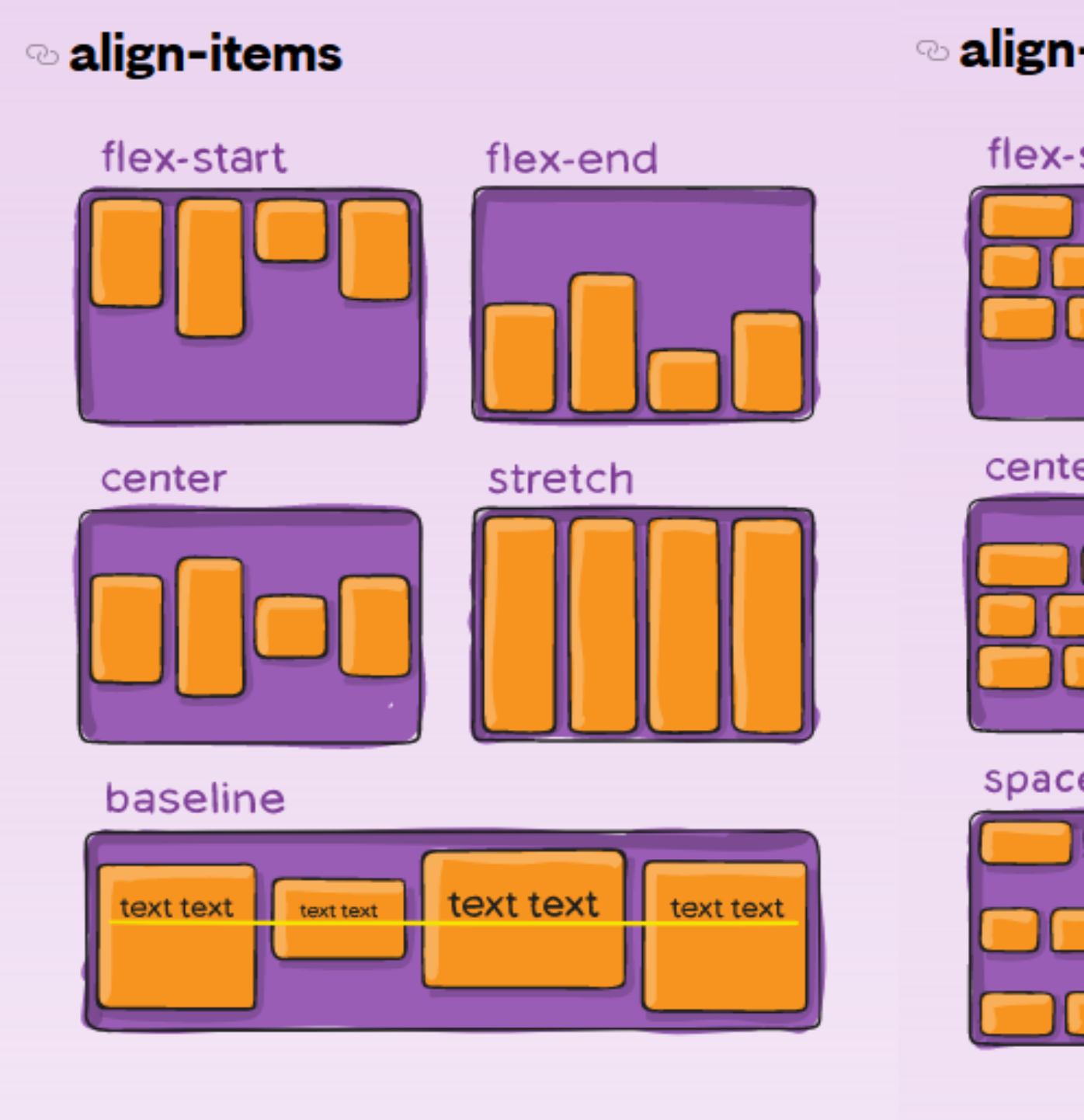
Float: none;

Flex Box Ch

Property	Value(s)
#1 display	flex
#2 flex-direction	row||column||colu
#3 justify-content	flex-start || flex-e
	space-between |
	space-evenly
#4 align-items	flex-start || flex-e
	stretch || baseline
#5 align-content	flex-start || flex-e
	stretch||space-be
#6 align-self	auto || flex-start |
	baseline || stretch
#7 Order	/* Any positive Va
#8 flex-grow	/* Any positive Va
#9 flex-shrink	/* Any positive Va
#10flex-wrap	nowrap || wrap ||



Justify-content – is used to align the flex items main-axis or horizontally or x-axis



Align-items: it is used to align the flex items cross –axis or vertically or y-axis , we need to add the height in flex container

Align-content: it is also as align-items. ** we need to add the flex-wrap, then only it will work**

The transform CSS property allows for various transformations of an element, including rotation, scaling, skewing, and translation.

The transform property uses functions like rotate(), scale(), translate(), and skew() to apply specific transformations.

```
transform: translate()
      ========================
      it is used to move the html elements
      translateX(200px); - move x-axis
      translateY(200px); => move y-axis
      translateZ(200px); => move y-axis

      translate(200px ,200px )=> move x & y axis

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

      transform:rotate()
      ==================
       it is used to rotate the elements

       parameter => degree => 0-360
       rotateX() => rotate x-axis
       rotateY() => rotate Y-axis
       rotateZ() => rotate Z-axis
       rotate()
       ---------------------
       transform:scale()
       ======================
       it is used to zoom in zoom out the element

       scaleX(1)
       scaleY(2)
       scaleZ(1)
       scale(0.5,1)
```

```
parent element
<div>
   img
</div>

div{
  width:200px;
  height:200px;
}

img{
  scaleX(2)
  scale(2,2)
}

1- width
2 - height

2 x 200px = 400
0.5 x 200px = 100
```

Skew()

It is used to Skew the html elements

- Use 0-360 deg

Skew(30deg,40deg)

skewX(30deg)

skewY(40deg)

Overflow

Float

Display:flex

Transition

Transform 2d ,3d

Animations

Display:grid

Media query

CSS Height, Width and Max-width

The CSS height and width properties are used to set the height and width of an element.

The CSS max-width property is used to set the maximum width of an element.

The height and width properties may have the following values:

- auto - This is default. The browser calculates the height and width
- length - Defines the height/width in px, cm, etc.
- % - Defines the height/width in percent of the containing block
- initial - Sets the height/width to its default value
- inherit - The height/width will be inherited from its parent value