CSS Topics

---

1. Font-weight

- Controls how **bold** or **light** your text appears.
- Common values:
  - `normal` → default weight
  - `bold` → thicker text
  - `100–900` → numeric weight (100 = thin, 900 = very bold)

Example:

```
p {
  font-weight: bold;
}

h1 {
  font-weight: 700;
}
```

---

2. text-align

- Controls **horizontal alignment** of text inside an element.
- Values:
  - `left`
  - `right`

- `center`
- `justify` → spaces text so edges are aligned

Example:

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

---

3. Overflow

- Controls what happens when content is **too big** for its container.
- Values:
  - `visible` → default, content spills out
  - `hidden` → extra content is cut off
  - `scroll` → scrollbars appear
  - `auto` → scrollbars appear only if needed

Example:

```
div {
  width:100px;
  height:100px;
  overflow: scroll;
}
```

---

4. List-style: none

- Removes bullets or numbers from lists (`ul`, `ol`).

```
ul {
  list-style: none;
}
```

---

5. text-decoration: none

- Removes underlines from links or text.

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

---

6. Position

Determines how an element is positioned on the page.

relative

- Positioned **relative to its original position**.

absolute

- Positioned **relative to its nearest positioned parent** (NOT the page unless no parent has position).

fixed

- Stays in place even when you scroll.
- Positioned relative to the **viewport**.

sticky

- Behaves like `relative`, but becomes `fixed` when scrolling reaches a certain point.

Example:

```
div {
  position: absolute;
  top: 20px;
  left: 30px;
}
```

---

7. Display

flex

- 1-dimensional layout (row or column).

grid

- 2-dimensional layout (rows + columns).

---

8. Flexbox Basics

Main Axis & Cross Axis

- Depends on `flex-direction`:
    - `row` → main axis is horizontal
    - `column` → main axis is vertical

justify-content (JC)

- Aligns items **along the main axis**.

Common values:

- `flex-start`
- `center`
- `flex-end`
- `space-between`
- `space-around`

align-items (AI)

- Aligns items **along the cross axis**.

Values:

- `flex-start`
- `center`
- `flex-end`
- `stretch`

flex-grow

- Controls how much an item **expands** to fill space.

flex-shrink

- Controls how items **shrink** when space is limited.

Example:

```
.container {
  display: flex;
  justify-content: center;
  align-items: center;
}
```

---

9. CSS Grid Basics

grid-template-rows / grid-template-columns

Defines number and size of rows/columns.

```
grid-template-columns: 200px 1fr 200px;
```

repeat() function

```
grid-template-columns: repeat(3, 1fr);
```

gap

Space between grid items.

1fr

Fractional unit → takes up remaining space.

grid-row / grid-column

Places an item in specific grid tracks.

```
grid-column: 1 / 3;  /* from line 1 to line 3 */
```

span

Spans across multiple rows/columns.

```
grid-column: span 2;
```

Grid Alignment

- **justify-content (JC)** → aligns grid as a whole horizontally
- **justify-items (JI)** → aligns items inside their grid cells horizontally
- **align-content (AC)** → aligns entire grid vertically
- **align-items (AI)** → aligns items vertically inside cells

---

10. Media Queries

Used for **responsive design**.

max-width

Apply styles when screen is **less than** a certain width.

```
@media (max-width: 600px) {
  body {
    background: lightblue;
  }
}
```

only screen

Used to target screen devices.

```
@media only screen and (max-width: 768px) { ... }
```

orientation

- `portrait` → height > width
- `landscape` → width > height

```
  @media (orientation: landscape) {
```

```
  body { background: yellow; }
}
```

---

11. Transition, Transform, Animation

transition

Smoothly changes styles.

```
div {
  transition: all 0.3s;
}
```

1. `transition` (shorthand)

Defines all transition properties in one line.

2. `transition-property`

Which CSS property will animate (e.g., `width`, `opacity`).

3. `transition-duration`

How long the animation takes (e.g., `0.5s`).

4. `transition-timing-function`

Speed curve of the transition:

- `linear`
- `Ease (default)`
- `ease-in`
- `ease-out`

- `ease-in-out`

5. `transition-delay`

How long to wait **before** starting the transition.

transform

Rotate, scale, translate (move), skew.

```
transform: scale(1.2);
```

animation

Uses keyframes for custom animations.

```
@keyframes move {
  from { left: 0; }
  to { left: 100px; }
}
```

1. `@keyframes`

Defines animation frames (from/to or 0%–100%).

2. `animation` (shorthand)

Set all animation properties in one line.

3. `animation-name`

Name of the `@keyframes` to run.

4. `animation-duration`

Length of the animation (e.g., `2s`).

5. `animation-timing-function`

Speed curve of the animation:

- `ease`
- `linear`
- `ease-in`
- `ease-out`
- `ease-in-out`
- `steps()`
- `cubic-bezier(...)`

6. `animation-delay`

How long to wait before animation starts.

7. `animation-iteration-count`

How many times the animation repeats:

- number (e.g., `3`)
- `infinite`

8. `animation-direction`

Direction of animation play:

- `normal`

- reverse
- alternate
- alternate-reverse

9. animation-fill-mode

How the animation applies styles before/after play:

- none
- forwards
- backwards
- both

---

12. Border

Border sides:

- border-top
- border-left
- border-right
- border-bottom

Border colors:

- white

- `transparent` → hides border but keeps space.

Border-radius

Rounds corners.

```
border-radius: 10px;
```

JavaScript Topics

---

1. var, let, const

var

- Re-declare, Re-intialize

let

- No Re-declare, Re-intialize

const

- No Re-declare, No Re-intialize

---

2. Function Syntax

```
function greet() {
  console.log("Hello");
}
```

---

3. if, else if, else

```
if (x > 10) {
  ...
} else if (x === 10) {
  ...
} else {
  ...
}
```

---

4. for loop

```
for (let i=0; i<5; i++) {
  console.log(i);
}
```

---

5. == vs ===

- `==` → compares value (loose comparison)
- `===` → compares value + type (strict comparison)

---

6. DOM Selectors

To select HTML elements:

- `document.getElementById()`
- `document.getElementsByClassName()`
- `document.querySelector()`
- `document.querySelectorAll()`

---

7. alert()

Shows a popup message.

```
alert("Hello!");
```

---

8. element.style.color

Changes CSS style using JavaScript.

```
document.getElementById("text").style.color = "red";
```

---

9. innerText / textContent / innerHTML

- `innerText` → visible text
- `textContent` → all text (including hidden)
- `innerHTML` → can include HTML tags

---

10. Input field value

```
let name = document.getElementById("inputName").value;
```

---

11. onclick

Inline event:

```
<button onclick="sayHello()">Click</button>
```

---

12. addEventListener

Better way to attach events.

```
button.addEventListener("click", sayHello);
```

---

13. createElement, add content, attach

```
let p = document.createElement("p");
p.textContent = "Hello";
document.body.appendChild(p);
```

appendChild()
Adds a node.

append()
Adds nodes .

---

14. removeChild / child.remove()

```
parent.removeChild(child);
// OR
child.remove();
```

---

15. trim, Number, JSON.stringify, JSON.parse

trim()

Removes spaces around text.

```
name.trim();
```

Number()

Converts value to number.

```
let age = Number("20");
```

JSON.stringify()

Converts object → string.

```
let str = JSON.stringify(obj);
```

JSON.parse()

Converts string → object.

```
let obj = JSON.parse(str);
```