<!-- ! transition -->

The transition property in CSS is used to create smooth animations when changing the properties of an element.
 It allows you to control the speed of these property changes over a specified duration, rather than having them occur instantaneously.

 Syntax

transition: property duration timing-function delay;

### Components of a Transition

1. **property**

   - **Description**: Specifies the CSS property that the transition effect is applied to (e.g., width, height, background-color, etc.).
   - **Special Value**: all - Applies the transition to all changeable properties.

   - **Example**:
     transition: background-color 0.5s ease;

2. **duration**

   - **Description**: Specifies the length of time the transition takes to complete. The duration is defined in seconds (s) or milliseconds (ms).

   - **Example**:
     transition: background-color 1s;

3. **timing-function**

   - **Description**: Defines the speed curve of the transition. It controls how the intermediate states of the transition are calculated.

   - **Common Values**:
     - ease: Starts slow, then fast, then ends slow (default value).
     - linear: Constant speed from start to end.
     - ease-in: Starts slow, then fast.
     - ease-out: Starts fast, then slow.
     - ease-in-out: Starts slow, speeds up, then slows down.

   - **Example**:
     transition: width 2s ease-in;

4. **delay**

   - **Description**: Specifies a delay before the transition starts. This can be in seconds (s) or milliseconds (ms).
   - **Example**:

```
transition: height 0.5s ease 0.3s;
```

### Shorthand Property
The transition property is often written in shorthand to include all the above components. You can omit any component, and it will use the default value.

Example:

```
transition: all 0.3s ease-in-out;
```

<!-- !   transform -->

Definition:

- The transform property in CSS allows you to apply various transformations to an element, such as moving, rotating, scaling, or skewing it.

Transform Functions:

1. **translate()**:

   - Moves the element from its current position.
   - translate(x, y) moves the element horizontally by x and vertically by y.

   - Example: transform: translate(50px, 100px); (moves the element 50px to the right and 100px down).

2. **rotate()**:

   - Rotates the element around a fixed point (the center by default).
   - rotate(angle) rotates the element by the specified angle in degrees.

   - Example: transform: rotate(45deg); (rotates the element 45 degrees clockwise).

3. **scale()**:

- Resizes the element.
- `scale(x, y)` scales the element by `x` horizontally and `y` vertically.

- Example: `transform: scale(2, 1.5);` (doubles the width and increases the height by 50%).

4. **skew()**:

- Skews the element along the X and Y axes.
- `skew(x-angle, y-angle)` skews the element by the specified angles.
- Example: `transform: skew(30deg, 10deg);` (skews the element 30 degrees along the X-axis and 10 degrees along the Y-axis).

Transform Origin:

- **transform-origin**:
  - Defines the point around which the transformation occurs.
  - Can be set using values like `center`, `top`, `bottom`, `left`, `right`, or specific coordinates.
  - Example: `transform-origin: top;`