

CSS Chapter 3

Transforms, Transitions, Animations & Media Query



CSS3 2D Transforms

With CSS3 transform, we can move, scale, turn, spin, and stretch elements.

How Does it Work?

A transform is an effect that lets an element change shape, size and position.

2D Transform methods

Translate()

Rotate()

Scale()

Skew()

Matrix()

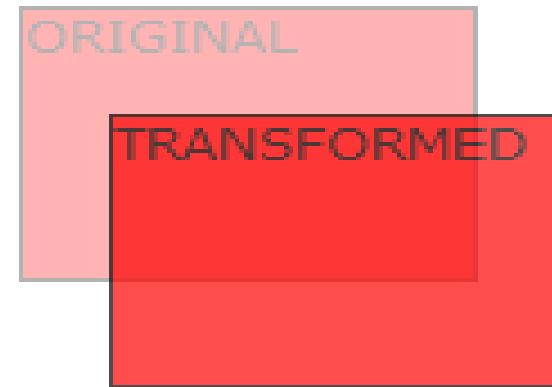


Transform Translate

With the `translate()` method, the element moves from its current position, depending on the parameters given for the left (X-axis) and the top (Y-axis) position.

Example

```
transform: translate(50px,100px);
```

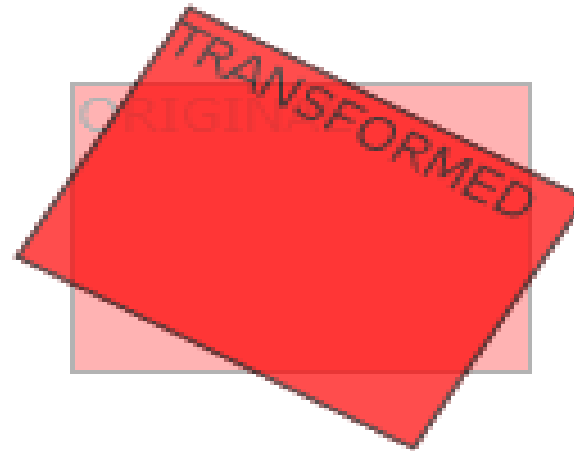


Transform Rotate

With the `rotate()` method, the element rotates clockwise at a given degree. Negative values are allowed and rotates the element counter-clockwise.

Example

```
transform: rotate(30deg);
```



Transform Scale

With the `scale()` method, the element increases or decreases the size, depending on the parameters given for the width (X-axis) and the height (Y-axis).

Example

```
transform: scale(2,4);
```

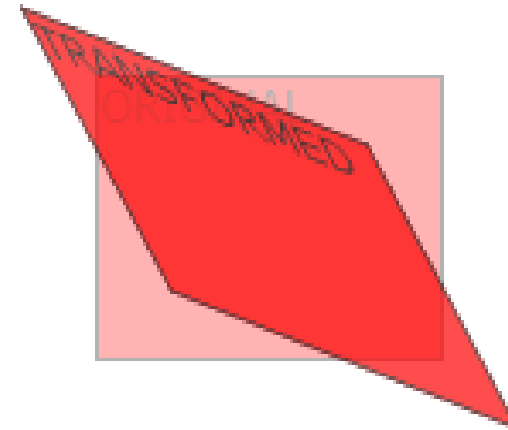


Transform Skew

With the `skew()` method, the element turns at a given angle, depending on the parameters given for the horizontal (X-axis) and the vertical (Y-axis) lines.

Example

```
transform: skew(30deg,20deg);
```



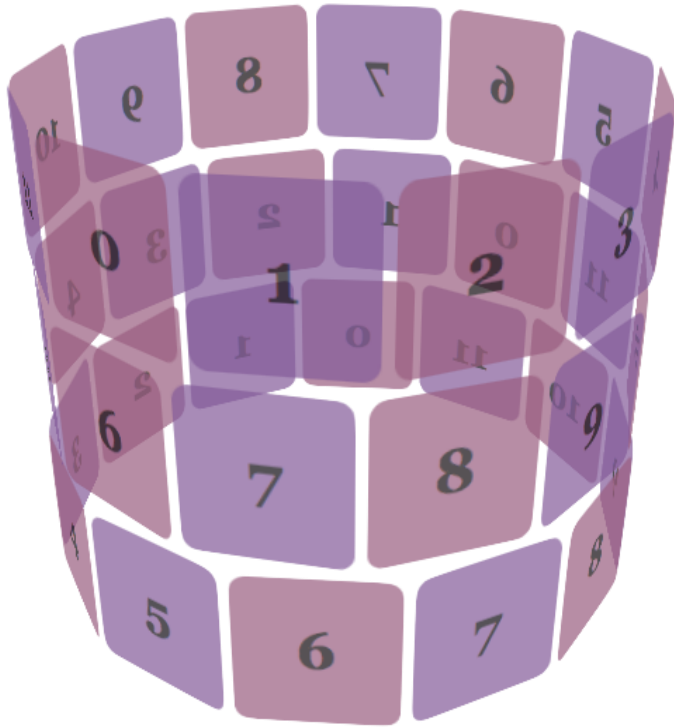
Transform Matrix

- The `matrix()` method combines all of the 2D transform methods into one.
- The `matrix` method takes six parameters, containing mathematic functions, which allow you to: rotate, scale, move (translate), and skew elements.

Example

```
transform: matrix(0.866,0.5,-0.5,0.866,0,0);
```

CSS 3D Transforms



3D transform methods:

- `rotateX()`
- `rotateY()`
- `rotateZ()`

3D Transform Rotate X

- With the **rotateX()** method, the element rotates around its X-axis at a given degree.

Example

```
transform: rotateX(120deg);
```



3D Transform Rotate Y

- With the rotateY() method, the element rotates around its Y-axis at a given degree.

Example

`transform: rotateY(130deg);`



3D Transform Rotate Z

- With the rotateZ() method, the element rotates around its Z-axis at a given degree.

Example

transform: rotateZ(130deg);



Transitions in CSS-3

- CSS3 transitions allow you to add smooth and gradual changes to the CSS properties of an element, such as its colour, size, or position.
- In this chapter you will also learn about the following properties:
 - transition-delay
 - transition-duration
 - transition-property
 - transition-timing-function



Transition-Delay

- The transition-delay property specifies when the transition effect will start.
- The transition-delay value is defined in seconds (s) or milliseconds (ms).

Transition-Duration

The transition-duration property specifies how many seconds (s) or milliseconds (ms) a transition effect takes to complete.

Transition-Timing-Function

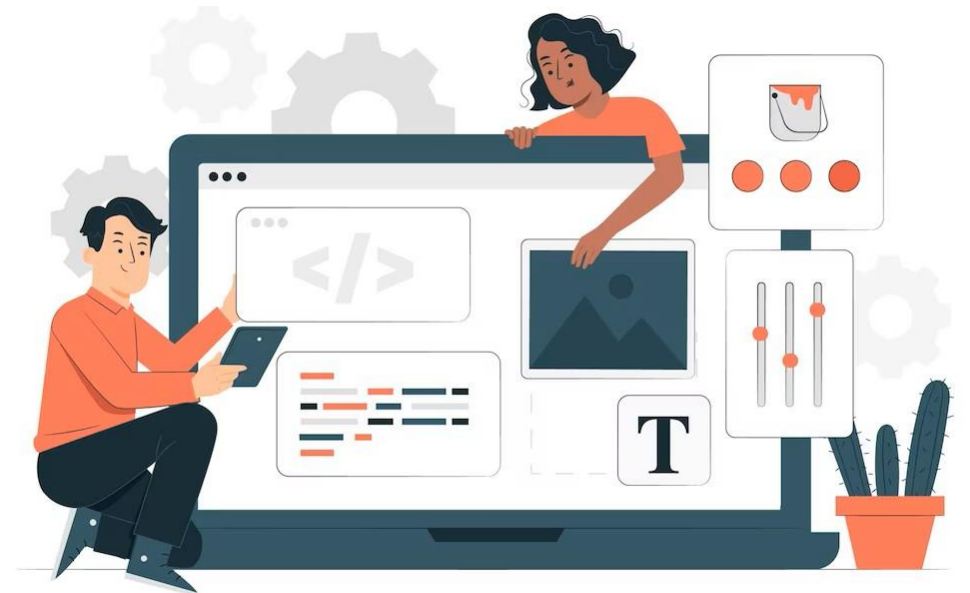
The transition-timing function property specifies the speed curve of the transition effect.

The transition-timing-function property can have the following values:

- ease - specifies a transition effect with a slow start, then fast, then end slowly.
- linear - specifies a transition effect with the same speed from start to end
- ease-in - specifies a transition effect with a slow start
- ease-out - specifies a transition effect with a slow end
- ease-in-out - specifies a transition effect with a slow start and end
- cubic-bezier(n,n,n,n) - lets you define your own values in a cubic-bezier function

Transition-Property

- The transition-property specifies the name of the CSS property the transition effect is for (the transition effect will start when the specified CSS property changes).



Property	Description
<u>transition</u>	A shorthand property for setting the four transition properties into a single property
<u>transition-delay</u>	Specifies a delay (in seconds) for the transition effect
<u>transition-duration</u>	Specifies how many seconds or milliseconds a transition effect takes to complete
<u>transition-property</u>	Specifies the name of the CSS property the transition effect is for
<u>transition-timing-function</u>	Specifies the speed curve of the transition effect

Example:

Get Hired By Compnay Around The World

Discover Global Career Opportunities And Land Your
Dream Job Anywhere In The World With Our Extensive
Job Listings Of Top Companies

Apply Now!

Learn More



Code Behind:

```
<div class="abc">









</div>
```

CSS Behind:

```
.abc:hover
{
    scale: 1.15;
    transition: scale 2s;
    transition-timing-function: linear;
}
```

Example:

HireMe Now!

The Best Place To Find Your Dream Company

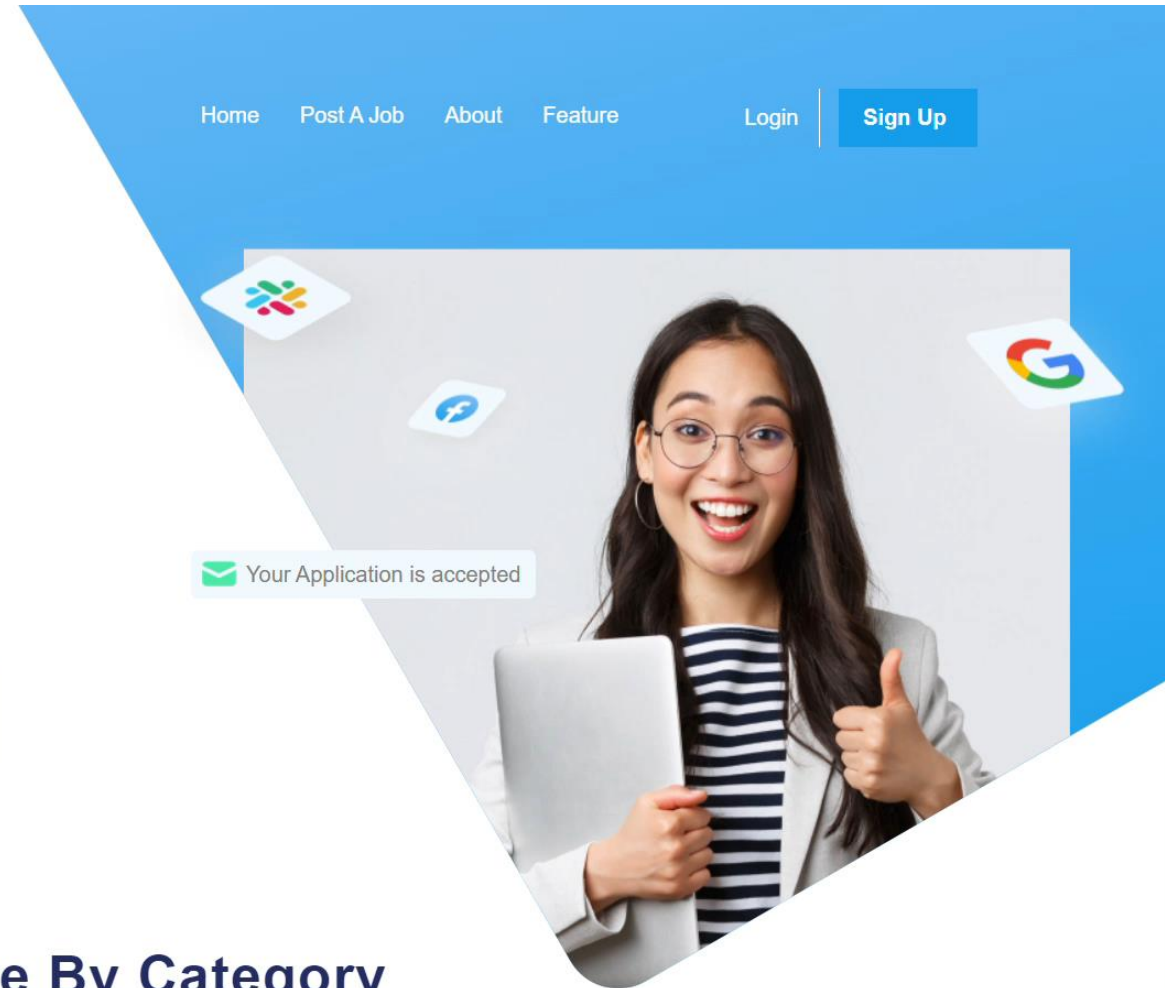
Ideal platform for job seekers passionate about startups and seeking new career opportunities.

Explore

Land your Dream Job

Recruit the Perfect Fit

Explore By Category



Code Behind:

```
<div class="rect-over-images">





</div>
```

CSS Behind:

```
.slack
{
  animation: rotation 2s infinite;
}

.fb
{
  animation: rotation 2s infinite;
}

.google
{
  animation: rotation 2s infinite;
}
```

Animations

With CSS3, we can create animations that replace animated images, Flash animations, and JavaScript in many web pages.

CSS3 @keyframes Rule

The @keyframes rule is where the animation is created. Specify a CSS style inside the @keyframes rule and the animation will gradually change from the current style to the new style.

Specify when the change will happen in per cent, or the keywords "from" and "to", which is the same as 0% and 100%.



Example:

```
@keyframes myfirst{  
    0% {background: red;}  
    25% {background: yellow;}  
    50% {background: blue;}  
    100% {background: green;}  
}
```

```
@-webkit-keyframes myfirst {  
    0% {background: red;}  
    25% {background: yellow;}  
    50% {background: blue;}  
    100% {background: green;}  
}
```

Properties included in animations

- @keyframes
- animation-name
- animation-duration
- animation-delay
- animation-iteration-count
- animation-direction
- animation-timing-function
- animation-fill-mode
- animation



Media Query

- Media query is a CSS technique introduced in CSS3
- It uses the `@media` rule to include a block of CSS properties only if a certain condition is true.
- Media queries will allow us to change our layouts to suit the exact need of different devices – without changing the content
- CSS can be used to specify how a document is presented in different media.



Example:

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<style>
body {
  background-color: lightgreen;
}

@media only screen and (max-width: 600px) {
  body {
    background-color: lightblue;
  }
}
</style>
</head>
<body>

</body>
</html>
```

Media query syntax: external

```
<link href="narrow.css" rel="stylesheet"  
media="only screen and (max-width:500px)">
```

```
@import url(narrow.css) only screen and  
(max-width:500px);
```



What we have discussed so far...



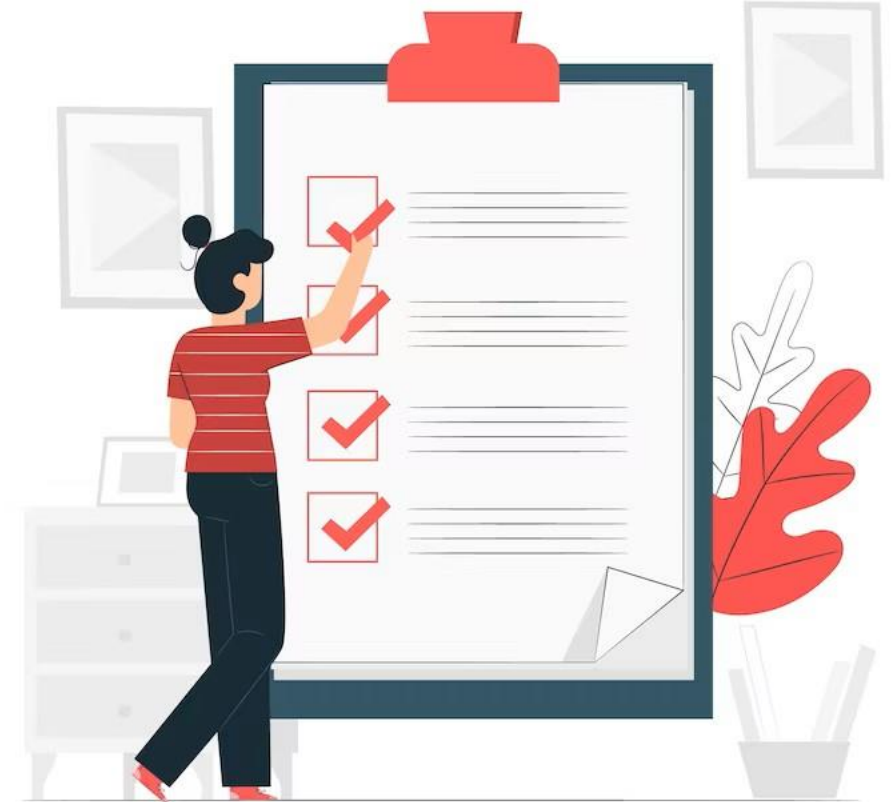
CSS-Transform



CSS-Transition



CSS-Media Query



Questions?

