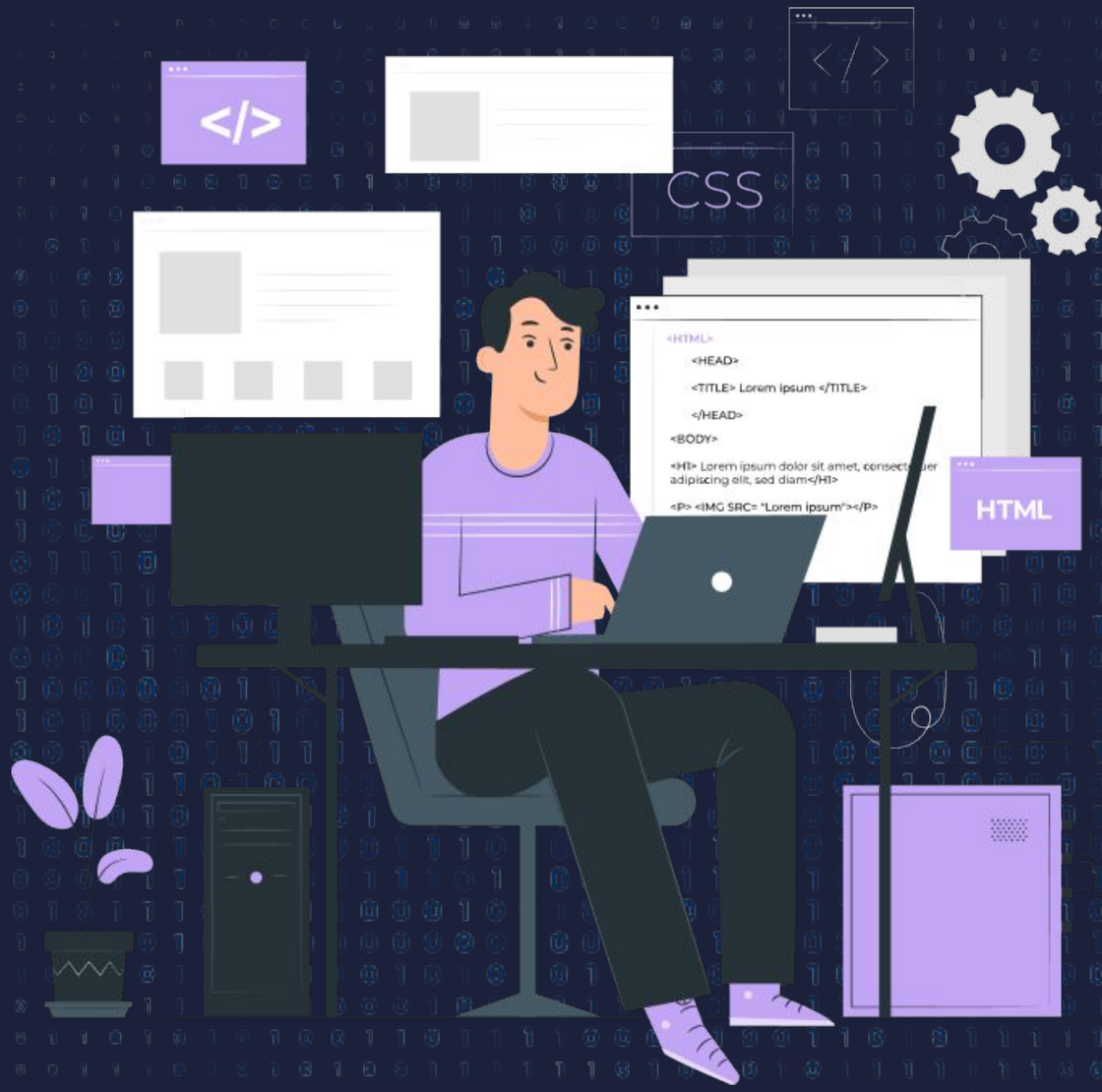
PW SKILLS

Transform



Topics

- CSS transform
- Translate
- Scale
- Rotate
- Skew
- matrix

CSS Transform

- This transform property allows you to translate, rotate, scale, and skew elements.
- Transformation is an effect that is used to change shape, size, and position.

Translate

The translate is used to move the element alone on the x-axis and y-axis.

Syntax:

```
translate(x, y)
```

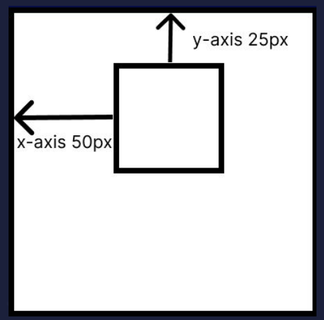
The **x** and **y** parameters specify the distance that the element should be moved along the X and Y axis, respectively. They can be specified in various units, such as “px”, “em” or “%”.

Translate example

Move the element 50 pixels to the right and 25 pixels down

CSS

```
.box {
  border: solid;
  height: 50px;
  width: 50px;
  transform: translate(50px, 25px);
}
```

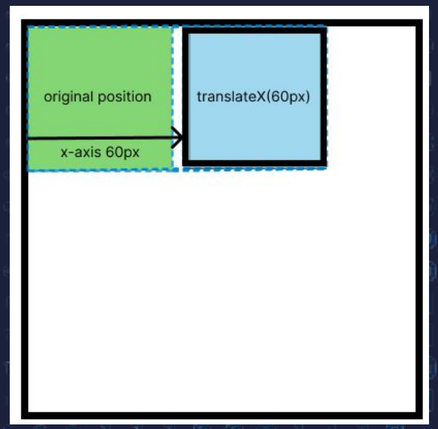


Translate example

Move the element 60 pixels to the right using translateX()

CSS

```
.box {
  border: solid;
  height: 50px;
  width: 50px;
  transform: translateX(60px);
}
```

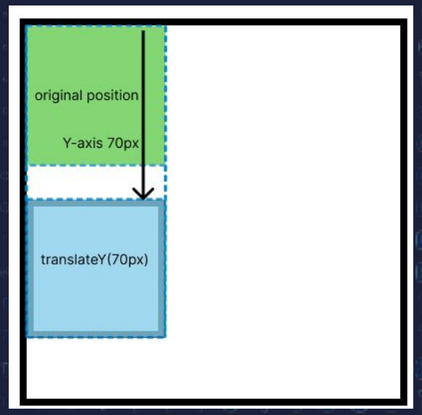


Translate example

Move the element 70 pixels to the bottom using translateY():

CSS

```
.box {
  border: solid;
  height: 50px;
  width: 50px;
  transform: translateY(70px);
}
```



Scale

It is used to change the width and height of an element.

Syntax:

```
transform: scale(x, y);
```

where x and y are the scaling factors. A value of 1 represents the original size of the element. Values greater than 1 will scale the element up, while values between 0 and 1 will scale the element down.

Scale example

Scale an element to 120% of its original size in the horizontal direction and 80% of its original size in the vertical direction.

CSS:

```
.image{
transform: scale(1.2,
0.8);
 }
```

Output:

before

after

Scale example

Scale an element to 150% of its original size in both the horizontal and vertical directions:

CSS:

```
.image{
    transform:
scale(1.5);
}
```

Output:

before

after

Scale example

Increase the height and decrease the width of elements using scaleY() and scaleX():

CSS:

```
.image{
    transform:
scaleX(1.2);
    transform:
scaleX(0.5);
}
```

Output:

before

after

Rotate

It is used to rotate the element on the basis of an angle.

Syntax:

```
transform: rotate(angle);
```

where angle is the amount of rotation in degrees. Positive values rotate the element clockwise, while negative values rotate it counterclockwise.

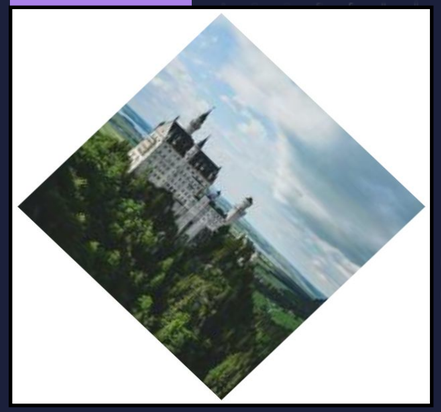
Rotate example

Rotate an element 45 degrees clockwise:

CSS:

```
.image {
transform: rotate(45deg);
 }
```

Output:



Rotate example

Rotate an element 45 degrees anticlockwise

CSS:

```
.image {
transform:
rotate(-45deg);
 }
```

Output:

Skew

It specifies the skew transformation along the X and Y axis corresponding to the skew angles.

Syntax:

```
transform: skew(x-angle,
y-angle);
```

Where x-angle and y-angle are the angles of skew in degrees. Positive values skew the element in the direction of the positive axis, while negative values skew it in the opposite direction.

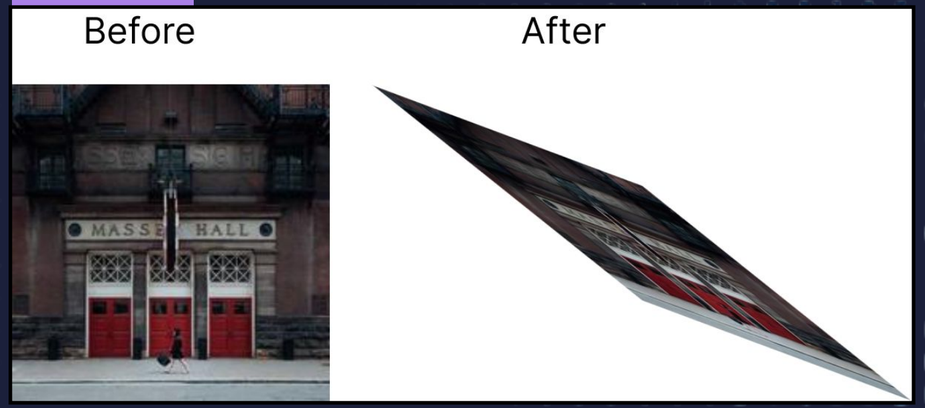
Skew example

Skew an element 45 degrees in the X direction and 25 degrees in the Y direction.

CSS:

```
.image {
transform :
skew(45deg,25deg)
}
```

Output:



Skew example

Skew an element 30 degrees in the X direction using skewX()

CSS:

```
.image {
transform: skewX(30deg)
 }
```

Output:

Skew example

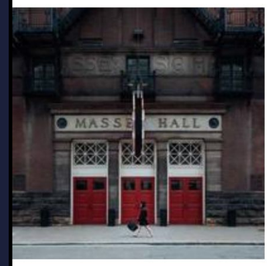
Skew an element 45 degrees in the Y direction using skewY()

CSS:

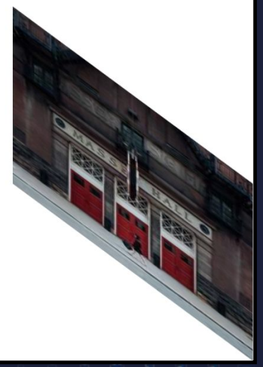
```
.image {
transform: skewY(45deg)
 }
```

Output:

Before



After



matrix

It takes six parameters, containing mathematical functions that allow you to rotate, scale, move (translate), and skew elements.

Syntax :

```
transform: matrix(a, b, c, d, tx, ty);
// matrix(scaleX(), skewY(), skewX(), scaleY(), translateX(), translateY())
```

Matrix example

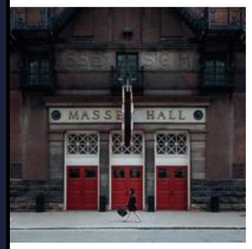
Scale the element in horizontal and vertical directions:

CSS:

```
.image{
    transform: matrix(2, 0, 0, 2, 0, 0);
}
```

Output:

Before



After


SKILLS
THANK YOU