# 11 트랜지션과 애니메이션

# 11–1 변형 알아보기

# 트랜지션과 애니메이션의 개요

## 변형 ( transform )

- **요소를 변형시키기 위한 속성**
  - 요소의 이동, 크기 조정,  회전, 기울기를 부여하여 변형시킴

## 트랜지션( transition )

- **특정 시간 동안 한 스타일에서 다른 스타일로 바뀌는 것을 의미**
  - [예] 특정 콘텐츠에 마우스를 올리면 요소의 스타일에 변화가 발생하도록 하는 것

## 애니메이션( animation )

- **트랜지션보다 정교한 변화를 지정할 수 있음**
  - 프랜스폼이나 트랜지션이 수행할 수 없는 효과를 지정하기 위해 사용

# transform과 변형 함수

## transform 속성

**transform** : 변형 함수

- **transform 속성은 변형 함수를 지정해 변형을 수행**
  - 요소의 이동, 크기 조정, 회전, 기울기를 부여하여 변형시킴
  - 주로 박스를 변형시키기 위해 사용

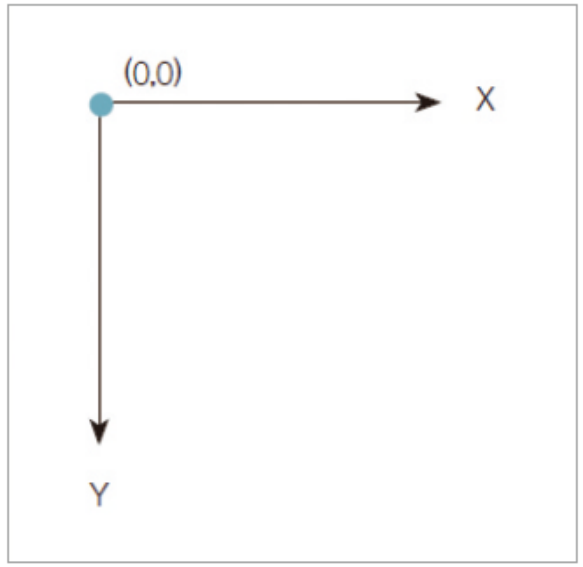
- **transform 속성의 변형 함수**

| 변형 함수 | 설명 |
|---|---|
| translate | 요소를 지정한 거리만큼 이동시키기 위한 변형 함수 |
| scale | 요소를 확대하거나 축소하기 위한 변형 함수 |
| rotate | 요소를 지정한 각도만큼 회전시키기 위한 변형 함수 |
| skew | 요소를 지정한 각도만큼 왜곡시키기 위한 변형 함수 |

# transform과 변형 함수

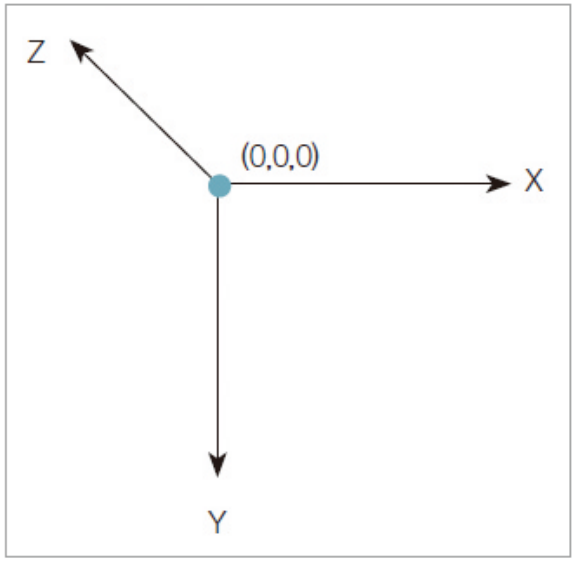
## 2차원 변형과 3차원 변형

- **2차원 변형**
  - 수평이나 수직으로 웹 요소 변형
  - 크기나 각도만 지정하면 됨
  - 2차원 좌표 사용




2차원 좌표계

- **3차원 변형**
  - x축과 y축에 원근감 추가
  - z축은 앞뒤로 이동
  - 앞쪽으로 다가올 수록 값이 더 커짐




3차원 좌표계

# transform : translate() 변형 함수

## translate() 변형 함수

transform:**translate(인자)**

- **요소를 지정한 거리만큼 이동시키기 위한 변형 함수**
  - 변형되는 과정을 보이는 것이 아니라 변형된 후의 상태를 보여줌

| 변형 함수 | 설명 |
|---|---|
| translate(dx) | 요소를 지정한 x축 방향으로 dx 만큼 이동 |
| translate(dx, dy) | 요소를 지정한 x축 방향으로 dx 만큼, 방향으로 dy 만큼 y축 이동 |
| translateX(dx) | 요소를 지정한 x축 방향으로 dx 만큼 이동 ( = translate(dx) ) |
| translateY(dy) | 요소를 지정한 y축 방향으로 dy 만큼 이동 |
| translateZ(dz) | 요소를 지정한 z축 방향으로 dz 만큼 이동 |
| translate3d(dx, dy, dz) | 요소를 지정한 x축 방향으로 dx만큼 , y축 방향으로 dy만큼 , z축 방향으로 dz 만큼 이동 |

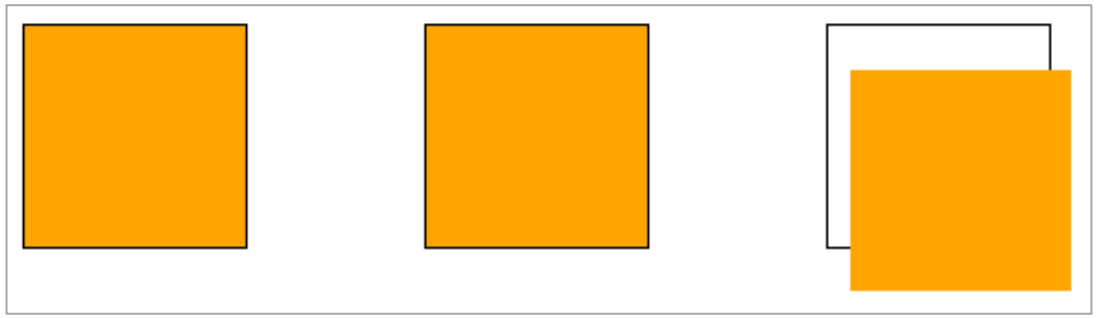
# transform : translate() 변형 함수



chapter11 > translate.html > ...

```
<!DOCTYPE html>
<html lang="ko">
    <head>
        <meta charset="UTF-8">
        <title>Transform:translate</title>
        <style>
            #container {
                width:800px; height:200px;
                margin:20px auto;
            }
            .origin {
                width: 100px; height: 100px;
                border: 1px solid black;
                float: left;
                margin: 40px;
            }
            .origin > div {
                width:100px ;height:100px;
                background-color: orange;
            }
            #movex:hover { transform: translateX(50px); }
            #movey:hover { transform: translateY(20px);  }
            #movexy:hover { transform: translate(10px, 20px);  }
        </style>
    </head>
```

```
    <body>
        <div id="container">
            <div class="origin">
                <div  id="movex"></div>
            </div>
            <div class="origin">
                <div id="movey"></div>
            </div>
            <div class="origin">
                <div id="movexy"></div>
            </div>
        </div>
        </div>
    </body>
</html>
```



# transform : scale() 변형 함수

## scale() 변형 함수

transform:**scale(인자)**

- **요소를 지정한 크기만큼 확대하거나 축소하기 위한 변형 함수**
  - 인자 값이 1보다 클 경우 확대, 1보다 작을 경우 축소를 의미

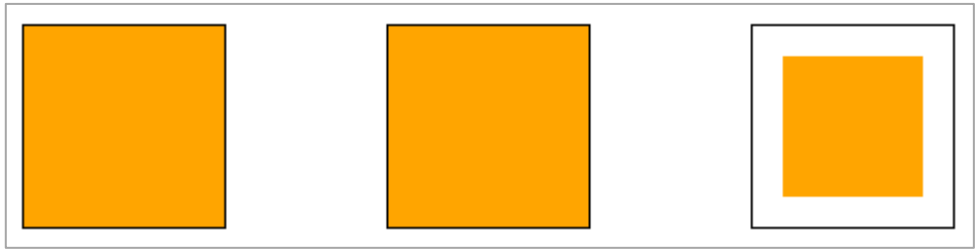
| 변형 함수 | 설명 |
|---|---|
| scale(sx) | 요소를 x축과 y축 방향으로 sx만큼 확대 또는 축소 |
| scale(sx, sy) | 요소를 지정한 x축 방향으로 sx만큼, y축 방향으로 sy만큼 확대 또는 축소 |
| scaleX(sx) | 요소를 지정한 x축 방향으로 sx만큼 확대 또는 축소 |
| scaleY(sy) | 요소를 지정한 y축 방향으로 sy만큼 확대 또는 축소 |
| scaleZ(sz) | 요소를 지정한 z축 방향으로 sz만큼 확대 또는 축소 |
| scale3d(dx, dy, dz) | 요소를 지정한 x축 방향으로 dx만큼 , y축 방향으로 dy 만큼 , z축 방향으로 dz 만큼 확대 또는 축소 |

# transform : scale() 변형 함수



chapter11 > scale.html > ...

```
<!DOCTYPE html>
<html lang="ko">
    <head>
        <meta charset="UTF-8">
        <style>
            #container{ width:600px; margin:20px auto; }
            .origin {
                width: 100px; height: 100px;
                border: 1px solid black;
                float: left;
                margin: 40px;
            }
            .origin > div {
                width:100px; height:100px;
                background-color: orange;
            }
            #scalex:hover{
                transform: scaleX(2);
            }
            #scaley:hover{
                transform: scaleY(1.5);
            }
            #scale:hover{
                transform: scale(0.7);
             }
        </style>
    </head>
    <body>
        <div id="container">
            <div class="origin">
                <div id="scalex"></div>
            </div>
            <div class="origin">
                <div id="scaley"></div>
            </div>
            <div class="origin">
                <div id="scale"></div>
            </div>
        </div>
    </body>
</html>
```



# transform : rotate() 변형 함수

## rotate() 변형 함수

transform:**rotate(인자)**

- **요소를 시계 방향으로 회전시킴**
  - 일반적인 각도(degree)나 래디안(radian)을 사용
  - 양수일 경우 시계 방향, 음수일 경우 반시계 방향으로 회전
- **요소의 중심 좌표를 기준으로 회전**
  - transform-origin 속성으로 변경 가능

| 변형 함수 | 설명 |
|---|---|
| rotate(각도) | 요소를 지정한 각도만큼 회전 |
| rotateX(각도) | 요소를 지정한 각도만큼 수평방향으로(x축 기준) 회전 |
| rotateY(각도) | 요소를 지정한 각도만큼 수직방향으로(y축 기준) 회전 |
| rotateZ(각도) | 요소를 지정한 각도만큼 앞뒤로(z축 기준) 회전 |
| rotate(dx, dy, dz) | 요소를 지정한 x축 방향으로 dx만큼 , y축 방향으로 dy 만큼 , z축 방향으로 dz 만큼 회전 |

# transform : rotate() 변형 함수



chapter11 > rotate.html > ...

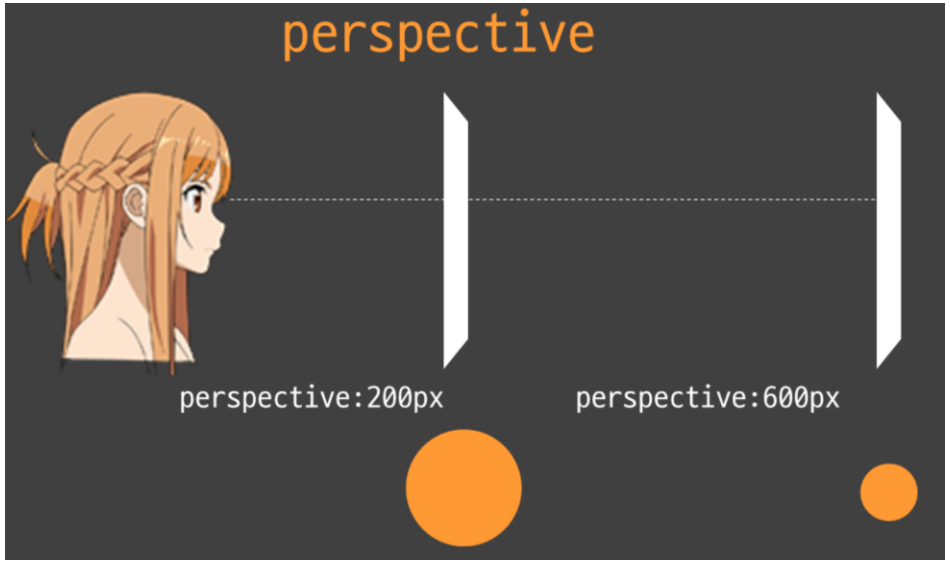
```
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <style>
        #container{
            width:800px;margin:20px auto;
        }
        .origin {
            width: 100px; height: 100px; margin: 40px;
            border:1px solid black;
            float: left;
        }
        #rotate1:hover { transform: rotate(40deg); }
        #rotate2:hover { transform: rotate(-40deg); }
    </style>
</head>
<body>
    <div id="container">
        <div class="origin" id="rotate1">
            <img src="images/tree.jpg">
        </div>
        <div class="origin" id="rotate2">
            <img src="images/tree.jpg">
        </div>
</body>
</html>
```

# 변형 관련 속성 : perspective

## perspective 속성

**perspective** : none | 크기

- **원근감을 지정하기 위한 속성**
- **perspective는 눈에서부터 모니터까지의 가상의 거리를 의미**
  - perspective값이 작으면 물체를 가까이 보는 것과 같고, 값이 크면 물체를 멀리서 보는 것과 같음
  - perspective 값이 클 수록 원근에 대한 왜곡도 커지게 됨
  - 크기는 0보다는 커야 하며, 픽셀로 표현함



# 변형 관련 속성 : perspective



chapter11 > perspective.html > ...

```html
<!DOCTYPE html>
<html lang="ko">
    <head>
        <meta charset="UTF-8">
        <style>
        .origin{
            width:152px; height:180px;
            border:1px solid black;
            margin:30px;
            float:left;
        }
        .origin > div {
            width:152px; height:180px;
        }
        .rotatex:hover {
            transform: rotateX(50deg);
        }
        #pers {
            perspective:300px;
        }
        </style>
    </head>
    <body>
        <div class="origin">
            <img src="images/sunset.jpg" alt=""> <!--원본이미지-->
        </div>
        <div class="origin">
            <div class="rotatex">
                <img src="images/sunset.jpg" alt="">
            </div>
        </div>
        <div class="origin" id="pers">
            <div class="rotatex">
                <img src="images/sunset.jpg" alt="">
            </div>
        </div>
    </body>
</html>
```

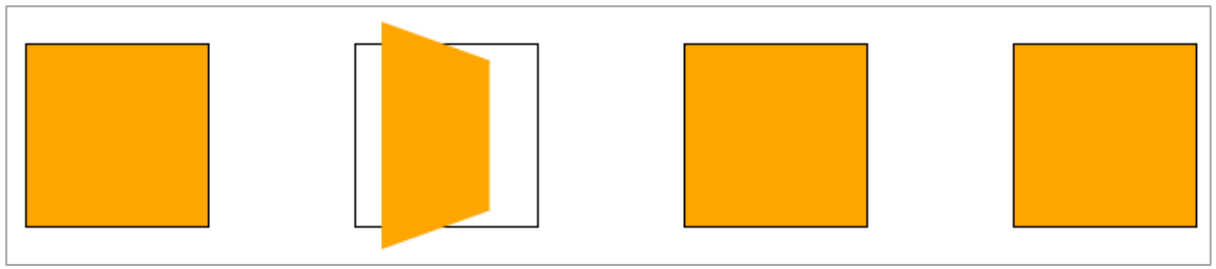
# 변형 관련 속성 : perspective



chapter11 > rotate3d.html > ...

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="UTF-8">
  <style>
    #container{
      width:800px; margin:20px auto;
    }
    .origin {
        width: 100px;
        height: 100px;
        margin: 40px;
        float: left;
        border: 1px solid black;
        perspective: 200px;
    }
    .origin > div {
        width:100px; height:100px;
        background-color: orange;
    }
    #rotatex:hover { transform: rotateX(55deg); }
    #rotatey:hover { transform: rotateY(55deg); }
    #rotatez:hover { transform: rotateZ(55deg); }
    #rotatexyz:hover { transform: rotate3d(2.5, 1.2, -1.5, 55deg); }
  </style>
</head>
```

```
<body>
  <div id="container">
    <div class="origin">
      <div id="rotatex"></div>
    </div>
    <div class="origin">
      <div id="rotatey"></div>
    </div>
    <div class="origin">
      <div id="rotatez"></div>
    </div>
    <div class="origin">
      <div id="rotatexyz"></div>
    </div>
  </div>
</body>
</html>
```

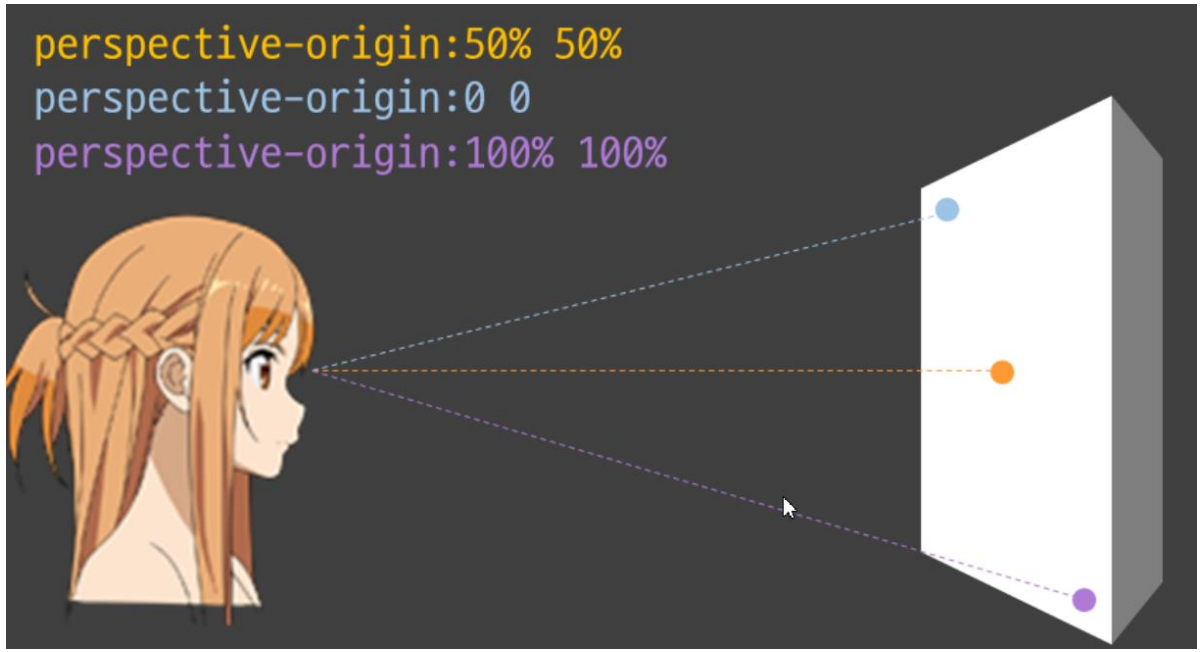


# 변형 관련 속성 : perspective-origin

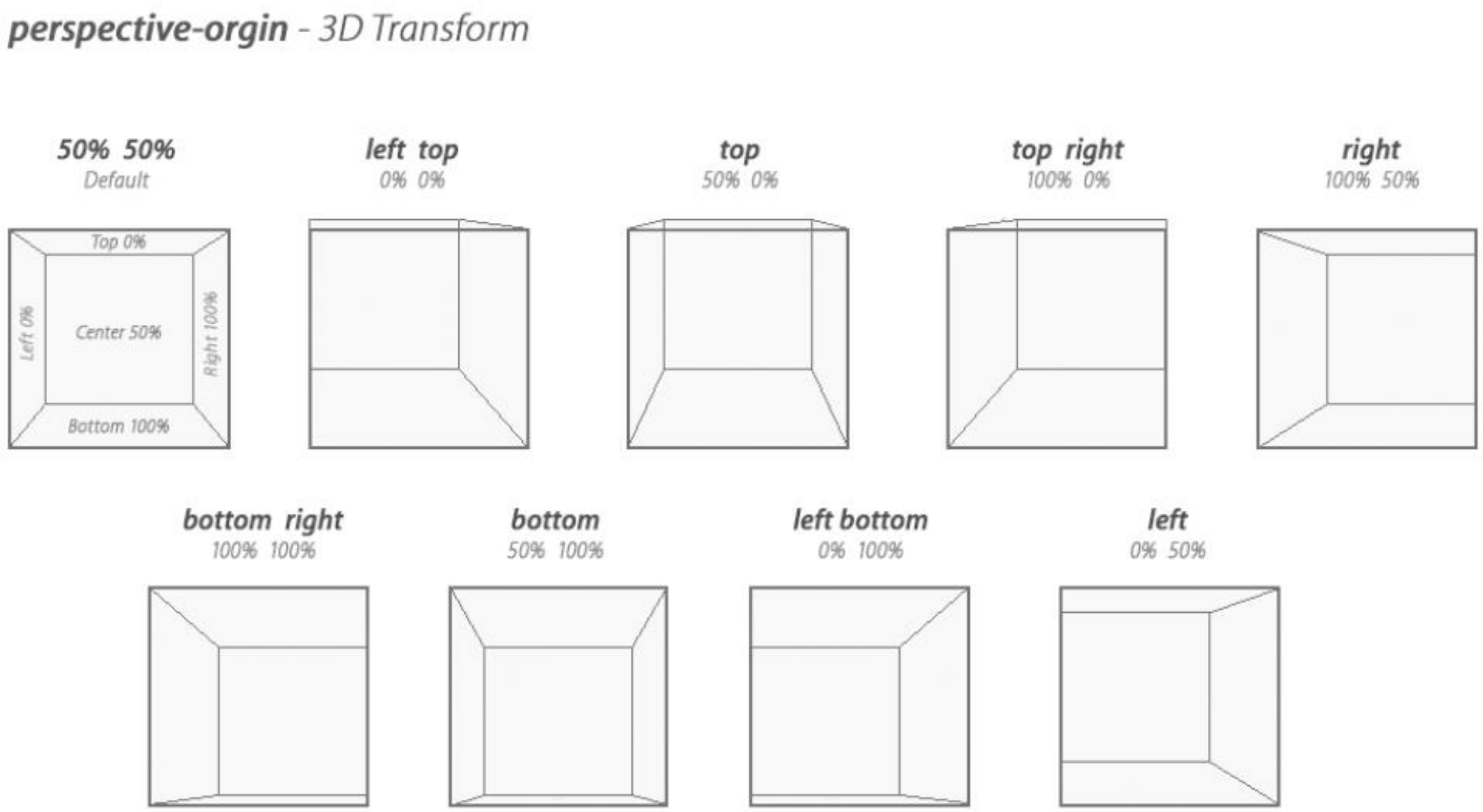
## perspective-origin 속성

**perspective-origin** : <가로 위치> <세로 위치>

- **원근감의 기준을 지정하기 위한 속성으로 사용자가 어디를 쳐다 보는가를 지정하는 값을 의미**
  - perspective 속성과 함께 사용
  - 위치는 퍼센트(%)나 매크로 값 사용 가능
  - 50% 50% = center center
  - 기본 값은 50% 50% 임



# 변형 관련 속성 : perspective-origin



# 변형 관련 속성 : perspective-origin



chapter11 > perspective-origin.html > ...

```html
<!DOCTYPE html>
<html lang="ko">
    <head>
        <meta charset="UTF-8">
        <style>
        .origin{
            width:152px; height:180px;
            border:1px solid black;
            margin:30px;
            float:left;
        }
        .rotatex:hover { transform: rotateX(50deg); }
        #pers {
            perspective:300px;
            perspective-origin: 10% 100%;
        }
        </style>
    </head>
    <body>
        <div class="origin" id="pers">
            <div class="rotatex">
                <img src="images/sunset.jpg" alt="">
            </div>
        </div>
    </body>
</html>
```

# transform : skew() 변형 함수

## skew() 변형 함수

```
transform:skew(인자)
```

- **요소를 지정한 각도만큼 비틀어 왜곡 시키는 변형 함수**
  - 2차원 변형만 가능한 함수로

| 변형 함수 | 설명 |
|---|---|
| skew(dx) | 요소를 x축 방향으로 dx도 만큼 왜곡시킴 |
| skew(dx, dy) | 요소를 x축 방향으로 dx도, y축 방향으로 dy도 만큼 왜곡시킴 |
| skewX(sx) | 요소를 x축 방향으로 dx도 만큼 왜곡시킴 ( = skew(dx) ) |
| skewY(sy) | 요소를 y축 방향으로 dy도 만큼 왜곡시킴 |

# transform : skew() 변형 함수



chapter11 > skew.html > ...

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <style>
        .origin {
            width: 160px; height: 240px;
            border: 1px solid black;
            float: left; margin: 100px;
        }
        .skewx:hover { transform: skewX(30deg); }
        .skewy:hover { transform: skewY(15deg); }
        .skewxy:hover { transform: skew(-25deg, -15deg); }
    </style>
</head>
<body>
    <div class="origin">
        <div class="skewx"><img src="images/rose.jpg"></div>
    </div>
    <div class="origin">
        <div class="skewy"><img src="images/rose.jpg"></div>
    </div>
    <div class="origin">
        <div class="skewxy"><img src="images/rose.jpg"></div>
    </div>
</body>
</html>
```

# 변형 함수 동시 지정

## 변형 함수의 동시 지정

**transform**: translate(), scale(), rotate(), skew()

- transform 속성에 효과를 나타내는 변형 함수들을 동시에 지정 가능
  - 인자들의 순서는 중요하지 않음

chapter11 > transform1.html > ...

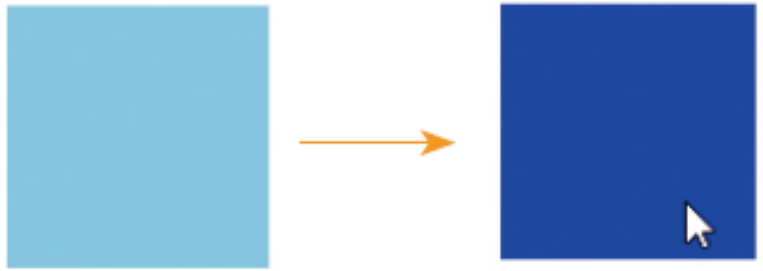
```
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4       <meta charset="UTF-8">
5       <style>
6           .origin {
7               width: 160px; height: 240px;
8               border: 1px solid black;
9               margin:50px;
10          }
11          .all{ transform: translate(20px, 30px) scale(0.9) rotate(20deg) }
12      </style>
13  </head>
14  <body>
15      <div class="origin">
16          <div class="all"><img src="images/rose.jpg"></div>
17      </div>
18  </body>
19  </html>
```

# 11–2 트랜지션 알아보기

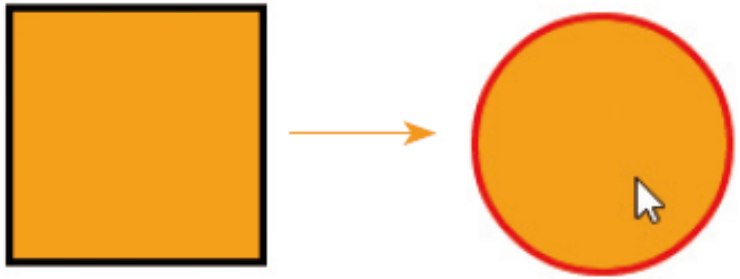
# 트랜지션의 개요

## 트랜지션(transition)

- **이미 지정되어 있는 스타일에서 주어진 시간에 따라 다른 스타일로 바뀌는 것을 의미**
  - 특정 이벤트가 발생했을 때 지정되어 있던 CSS 속성의 값이 점차적으로 변하며 다른 속성으로 지정되는 것
  - [예] 특정 요소에 마우스를 올리면 요소의 모양이 변하거나 색상이 변하는 효과를 발생



사각형의 배경 색이 바뀌는 트랜지션



사각형의 모양과 테두리 색이 바뀌는 트랜지션

- **트랜지션을 위해 CSS 속성을 사용하지만 모든 CSS의 속성이 사용될 수 있는 것은 아님**
  - 변화가 발생할 수 있는 CSS 속성들만 트랜지션에서 사용할 수 있음
  - [예] 색상, 길이나 크기 등

# 트랜지션과 속성

## 트랜지션 주요 속성과 수행 과정

- **주요 속성**

| 주요 속성 | 설명 |
|---|---|
| transition-property | 트랜지션을 수행할 CSS 속성을 지정 |
| transition-delay | 트랜지션이 시작되기까지의 지연시간을 지정 ( 단위는 초를 사용 ) |
| transition-duration | 트랜지션의 지속시간을 지정 ( 단위는 초를 사용 ) |
| transition-timing-function | 트랜지션 지속 동안 속도의 변화를 지정 |
| transition | 트랜지션 관련 주요 주요 속성을 동시에 지정 |

- **수행 과정**
  - 1. 어떠한 이벤트에 스타일의 변화를 줄 것인지 지정
    [예] 마우스를 올리거나 클릭하는 등의 행위
  - 2. 변화를 주고자 하는 속성의 처음 상태와 최종 상태를 지정
    [예] 너비에 변화를 줄 경우 처음 너비와 최종 너비를 지정
  - 3. transition 속성들을 사용하여 움직임의 속도나 지연(delay) 시간을 지정

# 트랜지션 속성 : transition-property

## transition-property 속성

**transition-property** : all | none | <CSS속성>

- **트랜지션을 지정할 대상을 지정하기 위한 속성**

| 속성값 | 설명 | 비고 |
|---|---|---|
| all | 요소에 지정된 모든 속성에 대해 트랜지션이 발생되도록 함<br>기본 값으로 transition-property 속성을 지정하지 않으면 all로 지정됨 | |
| none | 어떤 트랜지션도 발생하지 않도록 지정 | |
| CSS속성 | 지정된 속성에만 트랜지션이 발생되도록 함<br>속성이 하나 이상일 경우 쉼표(,)로 구분해 지정 | transition-property : background-color<br>transition-property : width, height |

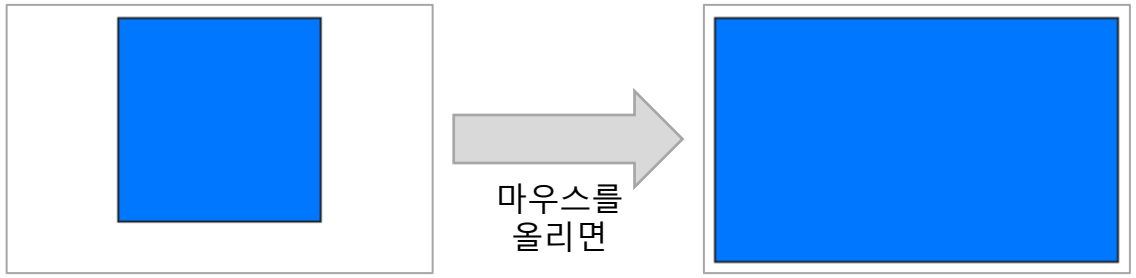
- **주로 이벤트와 함께 트랜지션이 발생하도록 지정함**
  - 예를 들어, 마우스의 동작과 같은 이벤트를 사용해 트랜지션이 발생하도록 지정함

# 트랜지션 속성 : transition-duration

## transition-duration 속성

**transition-duration** : 지속시간

- **트랜지션이 지속되는 시간을 지정하기 위한 속성**
  - 초(second)나 밀리 초(millisecond) 지정 가능
  - 초에는 s문자를 함께 표시해야 함
  - [예] transition-duration : 5s;
    5초 동안 지속




chapter11 > tr-1.html > ...

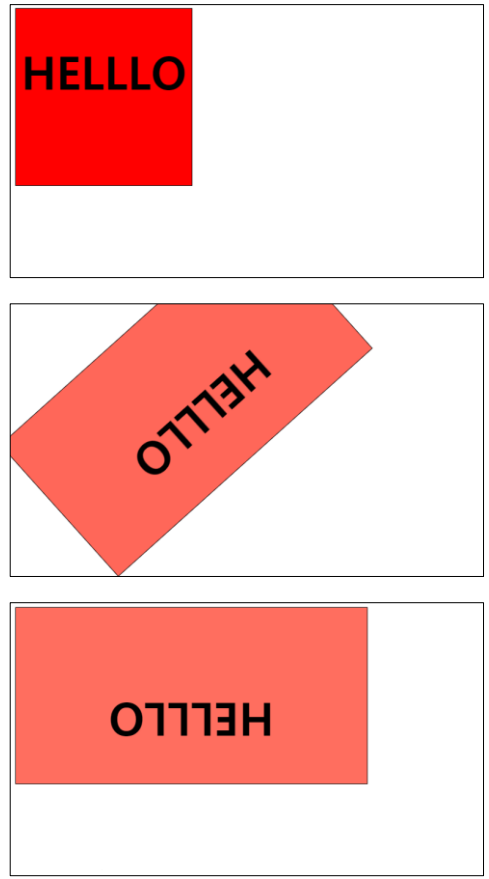
```
1   <!DOCTYPE html>
2   <html lang="ko">
3   <head>
4     <meta charset="UTF-8">
5     <style>
6       .box {
7           margin:20px auto;
8           width: 100px;
9           height: 100px;
10          background-color: #07f;
11          border: 1px solid #222;
12          transition-property: width, height;
13          transition-duration: 2s, 1s;
14      }
15      .box:hover {
16          width:200px;
17          height:120px;
18      }
19    </style>
20  </head>
21  <body>
22    <div class="box"></div>
23  </body>
24  </html>
```

# 트랜지션 속성 : transition-duration



chapter11 > tr-2.html > ...

```html
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <style>
        h1 { text-align:center; font-size:52px; }
        .tr1 {
            width: 200px; height: 200px;
            text-align:center;
            background-color: red;
            border: 1px solid black;
            transition-property: background-color, transform, width, height;
            transition-duration: 2s, 3s;
          }
        .tr1:hover {
            width: 400px; height: 200px;
            background-color: #ff6e5f;
            transform: rotate(180deg);
        }
    </style>
</head>
<body>
    <div class="tr1"><h1>HELLLO</h1> </div>
</body>
</html>
```



# 트랜지션 속성 : transition-timing-function

## transition-timing-function 속성

**transition-timing-function** : ease | linear | ease-in | ease-out | ease-in-out

- **지속시간 동안 속도 변화의 종류를 지정하기 위한 속성**
  - 시작 시간과 종료 시간 사이에 다양한 형태의 속도 변화 지정 가능

| 속성값 | 설명 |
|---|---|
| ease | 처음에는 천천히 시작해서 점점 빠르게 진행되다가 후반부에는 다시 느려지는 형태 (기본 값) |
| linear | 처음부터 끝까지 동일한 속도로 트랜지션을 진행 |
| ease-in | 트랜지션의 시작을 느리게 해서 후반부로 갈수록 진행 속도가 빨라짐 |
| ease-out | 트랜지션의 시작을 빠르게 해서 후반부로 갈수록 진행 속도가 느려짐 |
| ease-in-out | 트랜지션의 시작을 느리게 시작해서 느리게 끝남 ( 전반부 : ease-in , 후반부 : ease-out ) |

# 트랜지션 속성 : transition-delay, transition

## transition-delay 속성

**transition-delay** : 지연 시간

- **트랜지션이 발생되기까지의 지연시간을 지정하기 위한 속성**
  - 지연시간에는 주로 초 단위를 사용
  - 초는 s문자를 함께 표시해야 함
  - [예] transition-delay:3s; – 3초 후에 트랜지션 발생

## transition 속성

**transition** : transition-property transition-duration transition-timing-function transition-delay

- **property, duration, delay, timing-function을 일괄 지정하기 위한 속성**
  - [예] transition : width 5s ease-in 3s;
- **일부 속성은 생략 가능**
  - 시간을 하나만 입력할 경우, 지속시간으로 인정됨

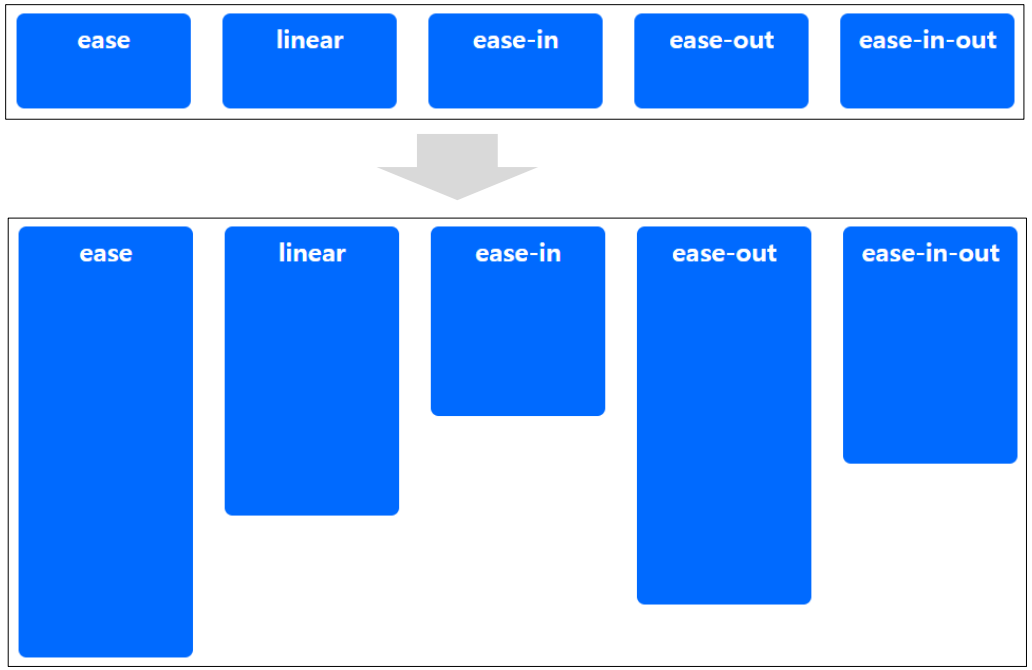
# 트랜지션 속성 : transition-timing-function



chapter11 > tr-function.html > ...

```html
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="UTF-8">
  <style>
    #ex div{
        float:left;
        width:100px;
        height:50px;
        margin:5px 10px;
        padding:5px;
        color: white;
        background-color: #006aff;
        border-radius:5px;
        text-align:center;
        font-weight:bold;
    }
    #ex:hover div{ height:400px; }
    #ex .ease { transition: 3s ease 1s;}
    #ex .linear{ transition:3s linear 1s; }
    #ex .ease-in{ transition:3s ease-in 1s; }
    #ex .ease-out{ transition:3s ease-out 1s  }
    #ex .ease-in-out{ transition:3s ease-in-out 1s; }
  </style>
</head>
```

```html
<body>
  <div id="ex">
    <div class="ease"> ease </div>
    <div class="linear"> linear </div>
    <div class="ease-in"> ease-in </div>
    <div class="ease-out"> ease-out </div>
    <div class="ease-in-out"> ease-in-out </div>
  </div>
</body>
</html>
```



# 11-3 애니메이션 알아보기

# 애니메이션의 개요

## 애니메이션( animation )

- **트랜지션보다 역동적인 동작 지정 가능**
  - transition이나 transform 속성으로는 상세한 움직임을 표현하는데 한계가 있음

- **트랜지션과 애니메이션의 차이점**
  - 트랜지션

    한 스타일에서 다른 스타일로 변경될 때 진행 시간을 지정해 부드럽게 변경되도록 함
  - 애니메이션

    키프레임(keyframe) 사용하여 특정 시점에 애니메이션을 지정할 수 있음

    재생횟수, 진행방향, 일시 정지 등을 지정할 수 있어 트랜지션보다 동적인 효과를 지정 가능

- **애니메이션을 동작을 위한 여러 속성들을 제공하고 있음**

## 키프레임 (keyframe)

- **애니메이션의 시작과 종료 사이에서 세밀한 움직임을 지정**
  - @keyframe 내에 타임 라인(time line)을 지정
    타임라인의 시작과 끝은 각각 from과 to를 사용
    타임라인은 퍼센트를 사용해 지정할 수도 있음 ( 0% ~ 100 % )
    타임라인의 수가 많을 수록 세밀한 애니메이션 지정 가능
  - 애니메이션 이름
    @keyframe에서 지정한 애니메이션의 이름을 의미
    animation-name 속성의 값으로 지정해 애니메이션을 실행

```
@keyframes 애니메이션 이름 {
        타임라인1 { 해당 타임에서의 상태 }
        타임라인2 { 해당 타임에서의 상태 }
                ……
        타임라인n { 해당 타임에서의 상태 }
}
```

```
@keygrames myAnimation {
        0%{ width:100px; background:red; }
       25%{ width:200px; background:green; }
       50%{ width:300px; background:red; }
       75%{ width:400px; background:green; }
      100%{ width:500px; background:blue; }
}
```

```
@keygrames myAnimation {
        from { width:100px; background:red; }
       25%{ width:200px; background:green; }
       50%{ width:300px; background:red; }
       75%{ width:400px; background:green; }
      to{ width:500px; background:blue; }
}
```

# 애니메이션의 속성

## animation-name 속성

**animation-name** : 애니메이션 이름

- **실행할 애니메이션을 지정**
  - @keyframes에서 지정한 애니메이션 이름을 값으로 지정해 애니메이션을 실행

## animation-duration 속성

**animation-duration** : 지속 시간

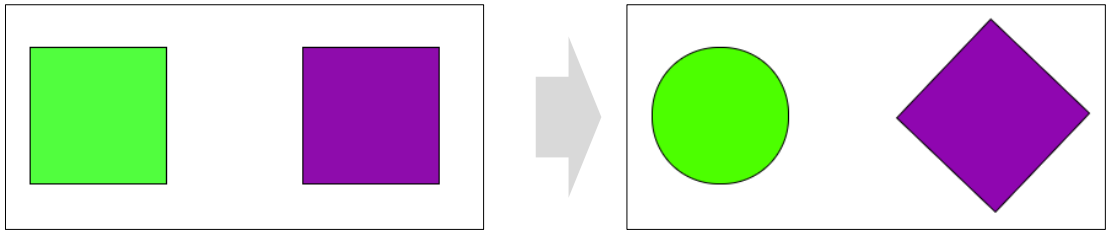
- **애니메이션이 수행되는 지속시간을 초단위로 지정하기 위한 하는 속성**
  - [예] 애니메이션의 이름이 myAnimation이고 지속시간이 5초인 경우

```
animation-name : myAnimation
animation-duration : 5s;
```

# 애니메이션의 속성

chapter11 > ani-1.html > ...

```html
<!DOCTYPE html>
<html lang="ko">
<head>
    <meta charset="UTF-8">
    <style>
        #container{
            width:500px;
            margin:20px auto;
        }
        .box{
            width: 100px; height: 100px;
            float:left;
            margin:50px;
        }
        #box1 {
            background-color: #4cff00;
            border: 1px solid transparent;
            animation-name: shape;
            animation-duration: 3s;
        }
        #box2 {
            background-color: #8f06b0;
            border: 1px solid transparent;
            animation-name: rotate;
            animation-duration: 3s;
        }
        @keyframes shape {
            from { border: 1px solid transparent; }
            to {
                border: 1px solid #000;
                border-radius: 50%;
            }
        }
        @keyframes rotate {
            from { transform:rotate(0deg) }
            to { transform: rotate(45deg); }
        }
    </style>
</head>
<body>
    <div id="container">
        <div class="box" id="box1"></div>
        <div class="box" id="box2"></div>
    </div>
</body>
</html>
```



# 애니메이션의 속성

## animation-iteration-count 속성

**animation-iteration-count** : 정수 | infinite

- **애니메이션의 반복횟수를 지정**
  - 반복할 횟수를 정수로 지정
  - infinite를 지정할 경우 무한 반복 수행

## animation-delay 속성

**animation-delay** : 지연 시간

- **애니메이션의 시작하기까지의 지연시간을 지정**

## animation-timing-function 속성

**animation-timing-function** : 타이밍

- **애니메이션의 진행과정 중 속도의 변화를 지정**
  - 트랜지션에서 transition-timing-function 속성과 같은 값을 가짐

# 애니메이션의 속성

## animation-direction 속성

**animation-direction** : normal | alternate | reverse | alternate-reverse

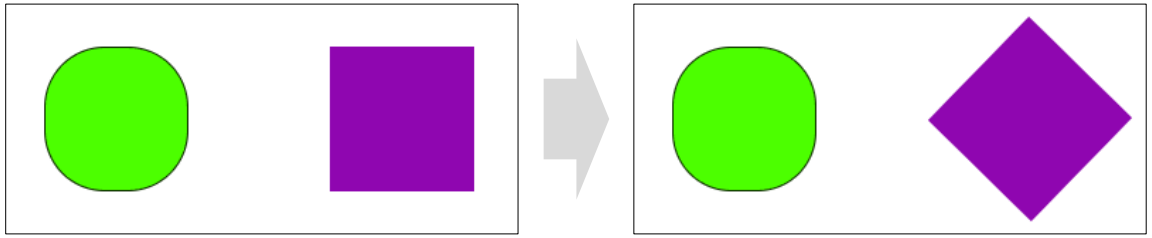
- **애니메이션의 진행방향을 지정하기 위한 속성**
  - 순방향 또는 역방향으로 지정 가능

| 속성값 | 설명 |
|---|---|
| normal | – 애니메이션 진행을 순방향으로 지정<br>– 한쪽 방향으로만 진행 ( 반복 횟수가 2일 경우 : 단방향으로만 2회 발생 ) |
| reverse | – 애니메이션 진행을 역방향으로 (to에서 from으로) 지정<br>– 방향만 다를 뿐 normal과 동일 |
| alternate | – 순방향과 역방향으로 애니메이션이 연속해서 발생<br>– 반복 횟수가 2일 경우 순방향과 역방향 왕복으로 발생 |
| alternate-reverse | – 역방향과 순방향으로 애니메이션이 연속해서 발생<br>– 순서만 다를 뿐 alternate와 동일 |

# 애니메이션의 속성

chapter11 > ani-2.html > ...

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="UTF-8">
  <style>
    #container{
      width:500px;  margin:20px auto;
    }
    .box{
      width: 100px; height: 100px; float:left;
      margin:50px;
    }
    #box1 {
      background-color: #4cff00;
      border: 1px solid transparent;
      animation-name: shape;
      animation-duration: 3s;
      animation-iteration-count: infinite;
    }
    #box2 {
      background-color: #8f06b0;
      border: 1px solid transparent;
      animation-name: rotate;
      animation-duration: 3s;
      animation-iteration-count: infinite;
    }
    @keyframes shape {
      from {  border: 1px solid transparent; }
      to {
        border: 1px solid #000;
        border-radius: 50%;
      }
    }
    @keyframes rotate {
      from { transform:rotate(0deg)  }
      to { transform: rotate(45deg);  }
    }
  </style>
</head>
<body>
  <div id="container">
    <div class="box" id="box1"></div>
    <div class="box" id="box2"></div>
  </div>
</body>
</html>
```



# 애니메이션의 속성

## animation-play-state 속성

**animation-play-state** : running | pasud

- **애니메이션의 진행이나 또는 정지를 지정하기 위한 속성**

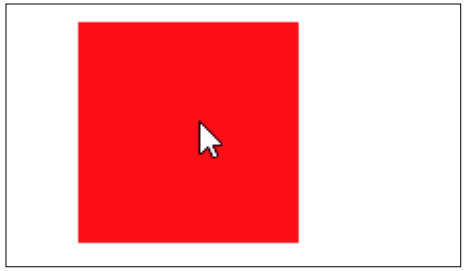
| 속성값 | 설명 |
|---|---|
| running | – 애니메이션을 진행상태로 지정<br>– running 상태로 지정되어 있는 동안은 애니메이션이 정상적으로 진행됨 |
| paused | – 애니메이션을 일시 정지 상태로 지정<br>– 애니메이션의 동작 뿐만 아니라 animation-duration 에서 지정한 시간도 일시정지됨 |

# 애니메이션의 속성



chapter11 > ani-play-state.html > ...

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <style>
        div {
            width: 100px; height: 100px; background: red;
            position: relative;
            animation-name: mymove;
            animation-iteration-count:2;
            animation-duration: 5s;
            animation-direction:alternate;
        }
        div:hover {
            animation-play-state: paused;
        }
        @keyframes mymove {
            from {left: 0px;}
            to {left: 200px;}
        }
    </style>
</head>
<body>
    <div></div>
</body>
</html>
```



마우스를 올리면 애니메이션이 정지됨

# 애니메이션의 속성

## animation 속성

**animation** : 이름 지속시간 [타이밍 함수] [지연시간] [반복 횟수] [방향]

- **애니메이션의 속성들을 일괄 지정하기 위한 속성**
  - 속성의 순서는 중요하지 않음
- **일부 속성은 생략할 수 있음**
  - animation-name과 animation-duration은 생략할 수 없음
    시간(초)를 하나만 지정할 경우 지속 시간으로 인식함
  - 시간(초)을 두 개 모두 지정한 경우
    앞은 지속 시간으로 인식하고 뒤는 지연시간으로 인식함
- **animation-play-state 속성은 일괄 지정에 포함시킬 수 없음**

# 애니메이션의 속성



chapter11 > ani-3.html > ...

```html
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="UTF-8">
  <style>
    .box {
      width: 100px; height: 100px;
      margin: 60px auto;
      animation: rotate 1.5s infinite, background 1.5s infinite alternate;
    }
    @keyframes rotate {  /* 0도 -> x축 -180도 회전 -> y축 -180도 회전 */
      from { transform: perspective(120px) rotateX(0deg) rotateY(0deg); }
      50% { transform: perspective(120px) rotateX(-180deg) rotateY(0deg); }
      to { transform: perspective(120px) rotateX(-180deg) rotateY(-180deg); }
    }
    @keyframes background {  /* 배경색 빨강 -> 초록 -> 파랑 */
      from { background-color: red; }
      50% { background-color: green }
      to { background-color: blue; }
    }
  </style>
</head>
<body>
  <div class="box"> </div>
</body>
</html>
```

# 애니메이션의 속성

## animation-fill-mode 속성

**animation-fill-mode** : none | forwards | backwards | both

- **애니메이션의 시작하기 전이나 종료 후의 상태를  지정하기 위한 속성**
  - 애니메이션의 시작

    지연시간을 지정할 경우 지연시간 동안 애니메이션 요소가  브라우저에 나타나지 않은 문제점 발생

    키 프레임의 첫번째 요소가 지연시간 동안 출력되도록 해 문제점을 해결할 수 있음
  - 애니메이션 종료 시점

    애니메이션을 마치면 첫 번째 키 프레임 상태로 전환된

    경우에 따라 애니메이션 종료 후에도 마지막 키 프레임이 유지되도록 할 수 있음

| 속성값 | 설명 |
|---|---|
| none | 애니메이션 효과가 없도록 지정 |
| forwards | 애니메이션 종료 후 키프레임의 마지막 타임라인의 값이 지정되도록 설정 |
| backwards | 애니메이션 시작 전 키프레임의  첫번째 타임라인의 값이 지정되도록 설정 |
| both | forwards와 backwards를 동시에 지정 |

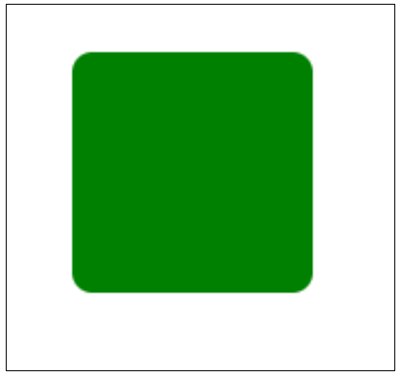
# 애니메이션의 속성



chapter11 > ani-fill-mode.html > ...

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <style>
        div {
            width: 100px; height: 100px;
            margin:30px; padding:10px;
            font-weight:bold; font-size:16px;
            border-radius:10px;
        }
        @keyframes myDIV1 {
            from { background-color: blue; }
            33%  { background-color: red; }
            66%  { transform:scale(1.5, 1.5); }
            to {background-color: green;}
        }
        #myDIV1 {
            animation-name: myDIV1;
            animation-duration: 5s;
            animation-delay:2s;
            animation-iteration-count:1;
            animation-fill-mode:both;
        }
    </style>
</head>
```

```
<body>
    <div id="myDIV1"></div>
</body>
</html>
```



# 수고하셨습니다