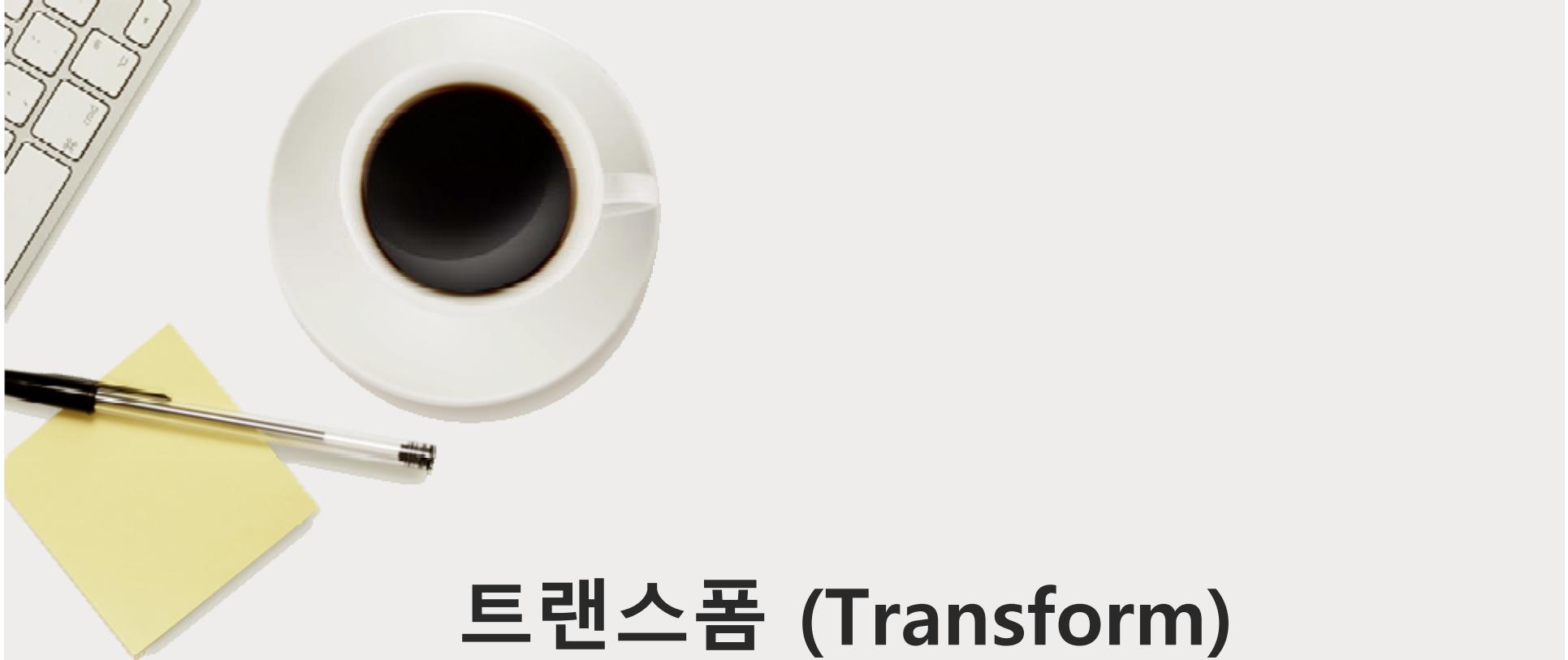


## Chapter 7

# 트랜스폼, 트랜지션, 애니메이션



# 트랜스폼 (Transform)



## ▣ 프랜스폼(Transform, 변형)

- **요소를 변형시키기 위한 속성**
  - 요소의 이동, 크기 조정, 회전, 기울기를 부여하여 변형시킴
    - : 주로 박스를 변형시키기 위해 사용
  - translate, scale, rotate, skew 라는 메서드를 값으로 지정
- **주요 함수**

함수	의미
translate	요소를 지정한 거리만큼 이동시키기 위한 메서드
scale	요소를 확대하거나 축소하기 위한 메서드
rotate	요소를 지정한 각도만큼 회전시키기 위한 메서드
skew	요소를 지정한 각도만큼 왜곡시키기 위한 메서드





## ■ translate() 함수

- **요소를 지정한 값만큼 이동시키는 메서드**
  - 이동시키는 방향은 x축이나 y축으로 지정 이동되는 거리는 픽셀로 지정
  - position 속성은 absolute나 relative 모두 가능

transform : translate(**값**)

- 값은 음수와 양수 모두 사용 가능

함수	의미
translate(dx)	요소를 지정한 dx 픽셀만큼 x축 방향으로 이동
translate(dx, dy)	요소를 지정한 dx 픽셀만큼 x축 방향으로, dy 만큼 y축 방향으로 이동
translate3d(dx, dy, dz)	요소를 지정한 dx 픽셀만큼 x축 방향으로, dy 만큼 y축 방향으로, dz 만큼 z 축 방향으로 이동 (2017년 현재 브라우저에서 수행되지 않음)
translateX(dx)	요소를 지정한 dx 픽셀만큼 x축 방향으로 이동 ( = translate(dx) )
translateY(dy)	요소를 지정한 dy 픽셀만큼 y축 방향으로 이동
translateZ(dz)	요소를 지정한 dz 픽셀만큼 z축 방향으로 이동 (2017년 현재 브라우저에서 수행되지 않음)

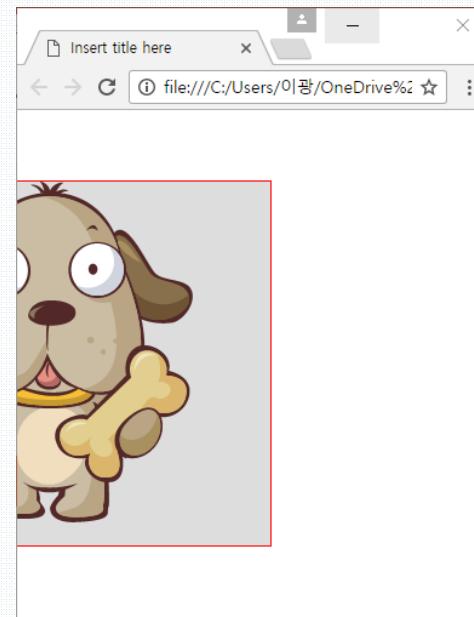


# 트랜스폼 (Transform)



## ▣ translate() 함수의 예

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Insert title here</title>
    <style>
        #mainbox {
            width:300px; height:300px;
            border: 1px solid red;
            background: #DDDDDD;
            transform: translate(-100px, 50px);
        }
    </style>
</head>
<body>
    <div id="mainbox">
        
    </div>
</body>
</html>
```





## ■ scale() 함수

- **요소를 지정한 값만큼 확대하거나 축소하기 위한 함수**
  - 확대나 축소시키는 방향은 x축이나 y축으로 지정
  - 확대/축소되는 정도는 상대적인 값인 정수나 실수를 사용 (기준 값은 1임)

transform : scale(값)

- 0보다 크고 1보다 작을 경우 : 요소 축소
- 1보다 클 경우 : 요소 확대
- 음수일 경우 : 대칭 변형

함수	의미
scale(sx)	요소를 가로와 세로로 sx만큼 확대
scale(sx, sy)	요소를 지정한 x축 방향으로 sx만큼, y축 방향으로 sy만큼 확대/축소
scaleX(sx)	요소를 지정한 x축 방향으로 sx만큼 확대
scaleY(sy)	요소를 지정한 y축 방향으로 sy만큼 확대
scaleZ(sz)	요소를 지정한 z축 방향으로 sz만큼 확대(2017년 현재 브라우저에서 수행되지 않음)



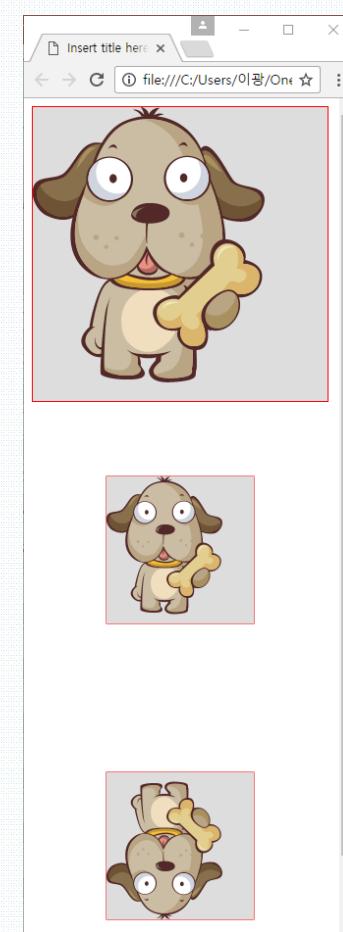
# 트랜스폼 (Transform)



## ■ scale() 함수의 예

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Insert title here</title>

    <style>
        div {
            width:300px; height:300px;
            border: 1px solid red;
            background: #DDDDDD;
        }
        #t1 { transform:scale(0.5, 0.5) }
        #t2 { transform:scale(0.5, -0.5) }
    </style>
</head>
<body>
    <div></div>
    <div id="t1"></div>
    <div id="t2"></div>
</body>
</html>
```





## ■ rotate() 함수

- 요소를 시계 방향으로 회전시킴
  - 값은 degree를 사용
    - : 음수 값을 지정할 경우 반시계 방향으로 회전
  - 요소의 중심좌표를 기준으로 회전

transform : rotate(각도)

함수	의미
rotate(angle)	요소를 지정한 각도만큼 회전
rotateX(angle)	요소를 지정한 각도만큼 수평방향으로 회전
rotateY(angle)	요소를 지정한 각도만큼 수직방향으로 회전
rotateZ(angle)	요소를 지정한 각도만큼 앞뒤로 회전 (2017년 현재 브라우저에서 수행되지 않음)



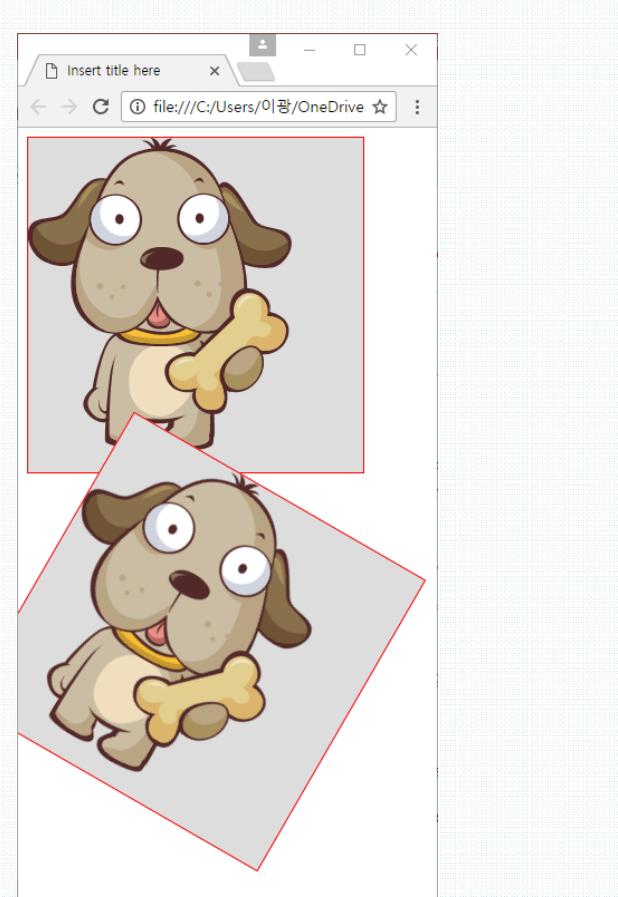
# 트랜스폼 (Transform)



## ■ rotate() 함수의 예

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Insert title here</title>
    <style>
        div {
            width:300px; height:300px;
            border: 1px solid red;
            background: #DDDDDD;
        }

        #t1 { transform:rotate(30deg) }
    </style>
</head>
<body>
    <div></div>
    <div id="t1">
        
    </div>
</body>
</html>
```





## ▣ skew() 함수

- 요소에 기울기를 지정
  - 값은 degree를 사용

transform : skew(각도)

함수	의미
skew(dx)	요소를 x축 방향으로 dx도 만큼 기울임
skew(dx, dy)	요소를 x축 방향으로 dx도, y축 방향으로 dy도 만큼 기울임
skewX(sx)	요소를 x축 방향으로 dx도 만큼 기울임 ( = skew(dx) )
skewY(sy)	요소를 y축 방향으로 dy도 만큼 기울임

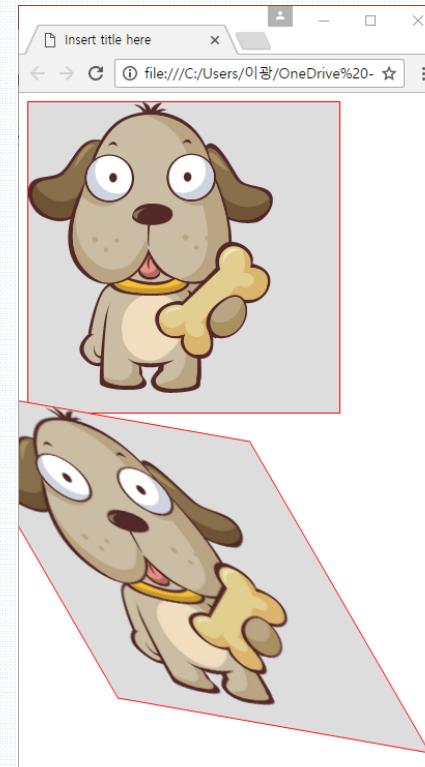


# 트랜스폼 (Transform)



## ▣ skew() 함수의 예

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Insert title here</title>
    <style>
        div {
            width:300px; height:300px;
            border: 1px solid red;
            background: #DDDDDD;
        }
        #t1 { transform:skew(30deg,10deg) }
    </style>
</head>
<body>
    <div></div>
    <div id="t1"></div>
</body>
</html>
```





## ■ matrix() 함수

- translate, scale, skew를 동시에 지정하기 위한 함수

```
transform : matrix( scaleX, skewX, skewY, scaleY, translateX, translateY)
```

- 각각의 값에는 단위를 생략하고 숫자만 지정해야 함



# 트랜스폼 (Transform)

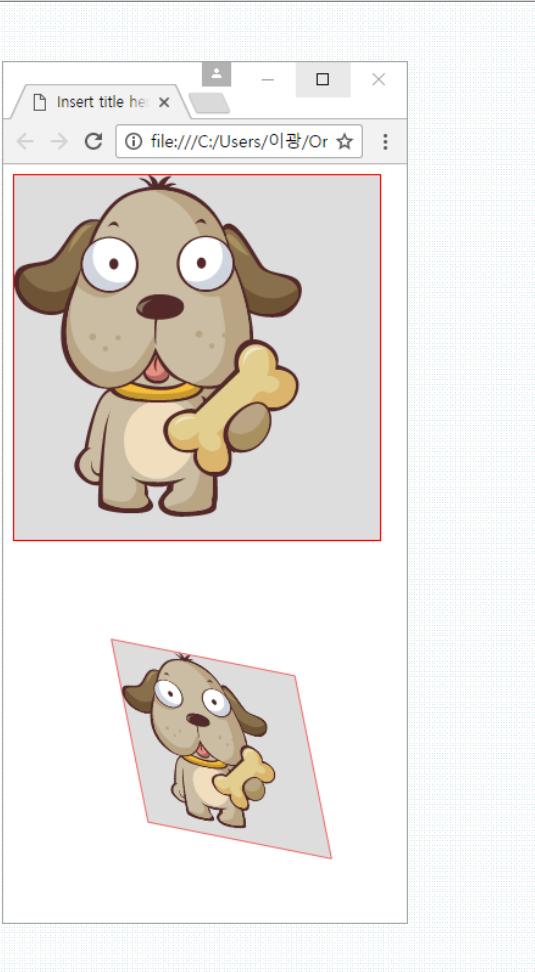


## ■ matrix() 함수의 예

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Insert title here</title>
    <style>
        div {
            width:300px; height:300px;
            border: 1px solid red;
            background: #DDDDDD;
        }

        #t1 { transform:matrix(0.5, 0.1, 0.1, 0.5, 20, 20) }

    </style>
</head>
<body>
    <div></div>
    <div id="t1"></div>
</body>
</html>
```





## ■ transform() 함수

- transform 속성에 효과를 나타내는 함수들을 일괄 지정

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

- 함수들의 순서는 중요하지 않음

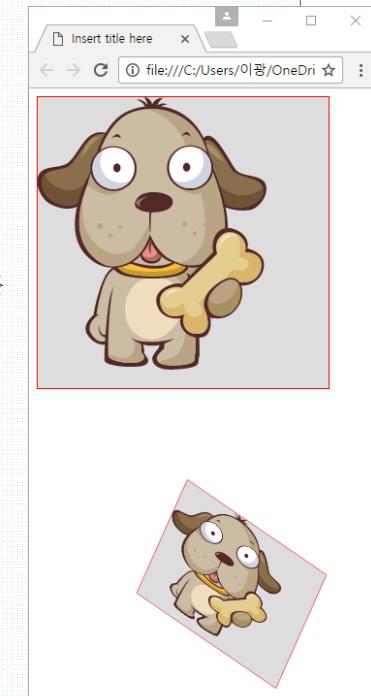


# 트랜스폼 (Transform)



## ■ transform() 함수의 예

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Insert title here</title>
    <style>
        div {
            width:300px; height:300px;
            border: 1px solid red;
            background: #DDDDDD;
        }
        #t1 { transform : translate(50px,50px) scale(0.5) skew(10deg,10deg) rotate(30deg) }
    </style>
</head>
<body>
    <div></div>
    <div id="t1"></div>
</body>
</html>
```





## ■ transform-origin() 함수

- 트랜스폼의 각종 효과를 지정하기 위한 기준점을 지정

**transform-origin(x, y)**

- X 좌표와 Y 좌표를 나타내는 두 개의 값을 가짐

- : 기본 값은 50% 50% ( 0%, 0% : 좌측 상단, 100%, 100% : 우측 하단 )
- : 좌표 대신 매크로 값을 사용 가능

매크로 값	의미
left	0 50% 과 같음
right	100% 50% 과 같음
top	50% 0 과 같음
bottom	50% 100% 과 같음
center	50% 50% 과 같음



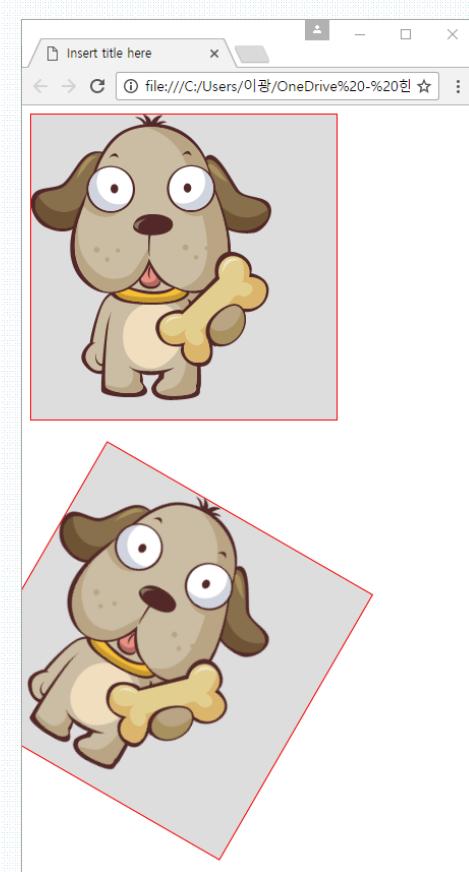
# 트랜스폼 (Transform)



## ■ transform-origin() 함수의 예

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Insert title here</title>
    <style>
        div {
            width:300px; height:300px;
            border: 1px solid red;
            background: #DDDDDD;
        }

        #t1 {
            transform-origin : 0 50%;
            transform:rotate(30deg)
        }
    </style>
</head>
<body>
    <div></div>
    <div id="t1"></div>
</body>
</html>
```





## ▣ perspective 속성

- **하위요소에 대해 원근감을 지정하는 속성**
  - rotate()와 함께 사용

**perspective : length | none**

- : none : 원근감을 지정하지 않음
- : length : 원근감의 정도를 픽셀로 표현 ( 클수록 none에 가까움 )

## ▣ perspective-origin 속성

- **원금감의 기준을 지정하는 속성**
  - perspective 속성과 함께 사용

**perspective-origin : 가로위치 세로위치**

- 위치는 퍼센트(%)나 매크로 값 사용 가능
  - : 50% 50% = center center
  - : 기본 값은 50% 50% 임

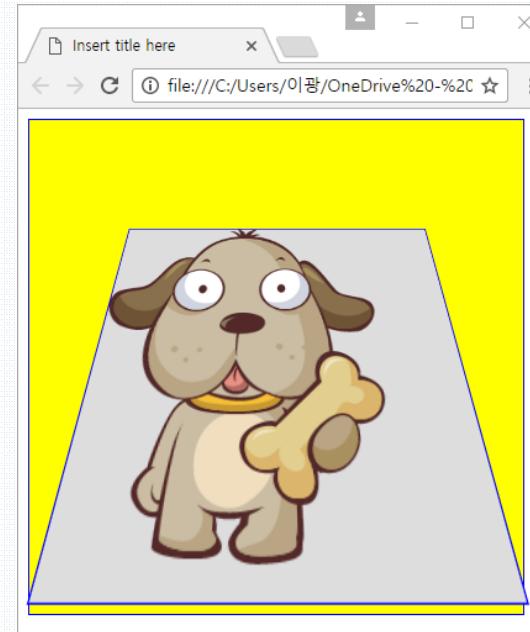


# 트랜스폼 (Transform)



## □ perspective 속성의 예

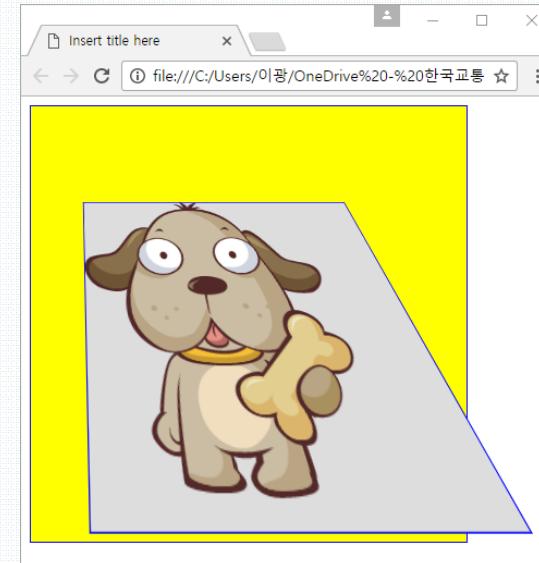
```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
<style>
div {
    width:400px; height:400px; background: yellow;
    border: 1px solid blue;
    perspective: 200px;
}
div#t1 {
    position:absolute;
    left:50px; top:50px;
    width:300px; height:300px; background: #DDDDDD;
    border: 1px solid blue;
    transform:rotateX(20deg);
}
</style>
</head>
<body>
<div>
    <div id="t1"></div>
</div>
</body>
</html>
```





## ▣ perspective-origin 속성의 예

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Insert title here</title>
    <style>
        div {
            width:400px; height:400px; background: yellow;
            border: 1px solid blue;
            perspective: 200px;
            perspective-origin: 10% 50%;
        }
        div#t1 {
            position:absolute;
            left:50px; top:50px;
            width:300px; height:300px; background: #DDDDDD;
            border: 1px solid blue;
            transform:rotateX(20deg);
        }
    </style>
</head>
<body>
    <div>
        <div id="t1"></div>
    </div>
</body>
</html>
```



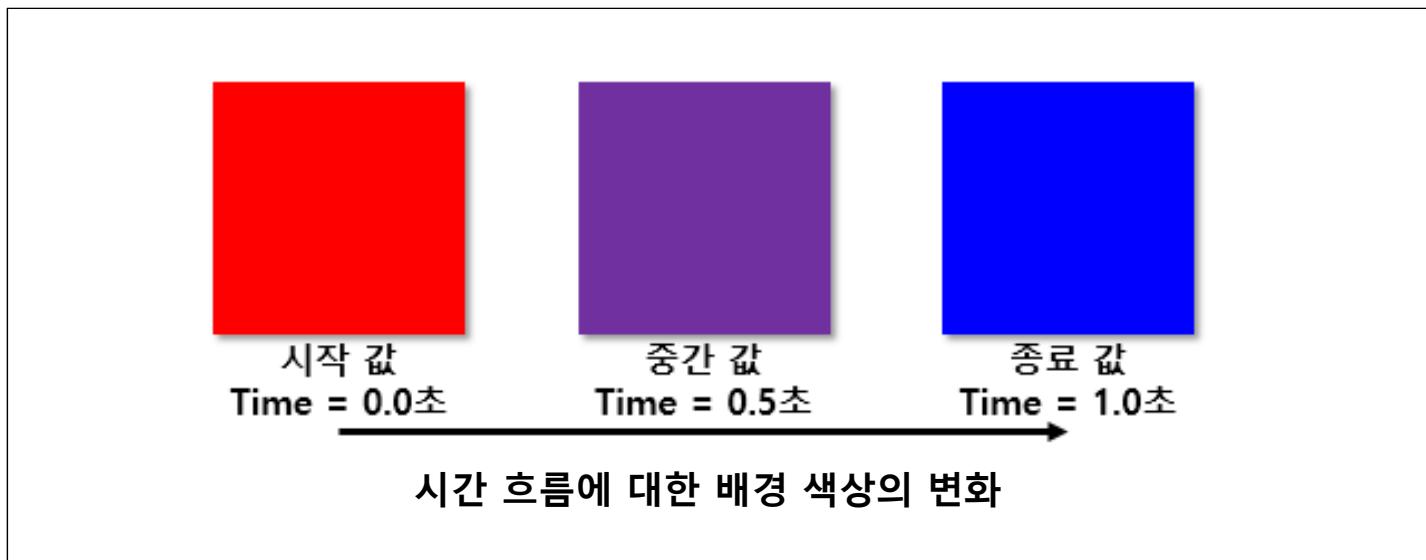


# 트랜지션 (Transition)



## ▣ 트랜지션(transition)

- 특정 스타일에서 다른 스타일로 바뀌는 것을 의미
  - [예] 특정 콘텐츠에 마우스를 올리면 요소의 스타일에 변화가 발생하도록 하는 것
- 변화가 발생할 수 있는 CSS 속성들만 트랜지션에서 사용할 수 있음
  - [예] 색상, 길이나 크기 등





## ▣ 트랜지션의 지정 과정과 주요 속성

- **지정 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 딜레이 시간을 지정

- **주요 속성**

주요 속성	의미
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-property 속성

- **트랜지션을 지정할 대상을 지정하기 위한 속성**
  - 트랜지션을 적용할 수 있는 대상은 CSS의 속성으로 제한됨

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

- **none** : 어떤 트랜지션도 발생하지 않도록 지정
- **all** : 요소에 지정된 모든 속성에 대해 트랜지션이 발생되도록 함
- **css속성** : 지정된 속성에만 트랜지션이 발생되도록 함

[예] transition-property : width;

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





## ■ transition-duration 속성

- 트랜지션이 지속되는 시간을 초단위로 지정하기 위한 속성

**transition-duration : 지속시간**

[예] transition-duration : 5s; ( 5초 동안 지속 )

## ■ transition-delay 속성

- 트랜지션이 발생되기 까지의 지연시간을 초단위로 지정하기 위한 속성

**transition-delay : 지연시간**

[예] transition-delay : 3s; ( 3초 후 트랜지션 발생 )



# 트랜지션 (Transition)

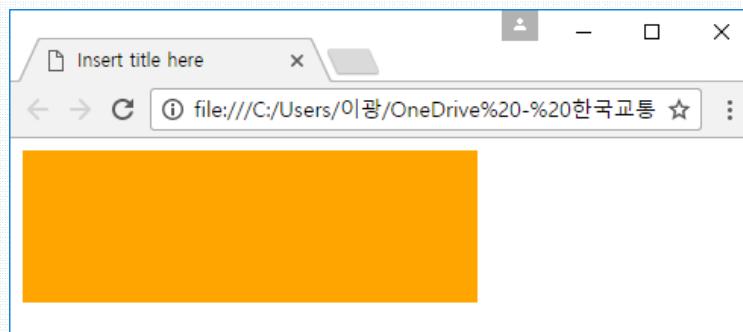
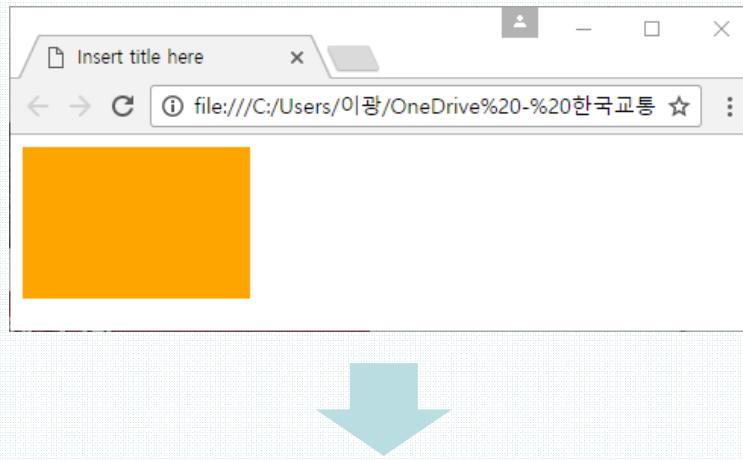


## ■ transition의 예 - 1

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Insert title here</title>
    <style type="text/css">

        div {
            width:150px; height:100px;
            background:orange;
            transition-property:width;
            transition-duration:2s;
            transition-delay:1s;
        }
        div:hover { width:300px; }

    </style>
</head>
<body>
    <div></div>
</body>
</html>
```



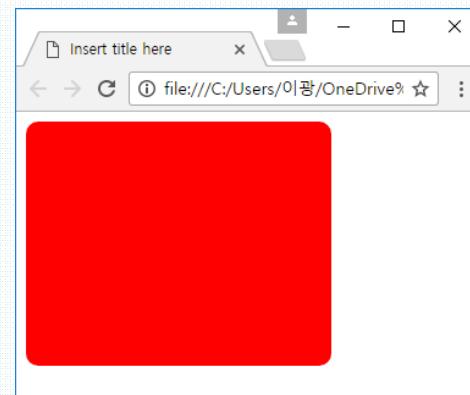
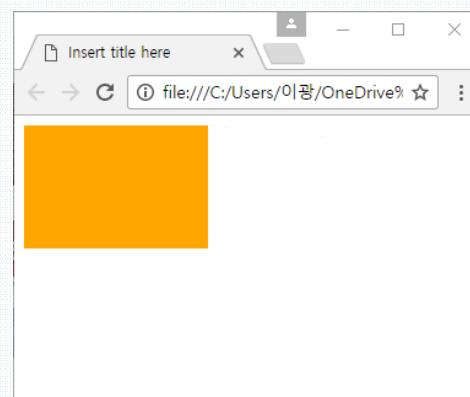
# 트랜지션 (Transition)



## transition의 예 - 2

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Insert title here</title>
    <style type="text/css">

        div {
            width:150px; height:100px;
            background:orange;
            transition-property:all;
            transition-duration:2s;
            transition-delay:1s;
        }
        div:hover {
            width:250px; height:200px;
            background:red;
            border-radius:10px
        }
    </style>
</head>
<body>
    <div></div>
</body>
</html>
```





## ▣ transition-timing-function 속성

- **지속시간 동안 속도 변화의 종류를 지정하기 위한 속성**
  - 시작시간과 종류시간 사이에 다양한 형태의 속도 변화 지정 가능

주요 속성	의미
ease	진행 속도가 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 전반부의 형태는 'ease-in'과 같고 후반부는 "ease-out"과 같음





## ■ transition-timing-function 속성의 예

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Insert title here</title>
    <style>
        div { width:150px; height:50px; border: 1px solid black;
               background-color:yellow;
               text-align:center; line-height:50px;
               transition-property:width; transition-duration: 3s;
        }
        #Box1 { transition-timing-function: linear }
        #Box2 { transition-timing-function: ease }
        #Box3 { transition-timing-function: ease-in }
        #Box4 { transition-timing-function: ease-out }
        #Box5 { transition-timing-function: ease-in-out }
        div:hover { width: 400px; }
    </style>
</head>
<body>
    <div id="Box1">linear</div>
    <div id="Box2">ease</div>
    <div id="Box3">ease-in</div>
    <div id="Box4">ease-out</div>
    <div id="Box5">ease-in-out</div>
</body>
</html>
```





## ▣ transition 속성

- **property, duration, delay, timing-function**을 일괄 지정하기 위한 속성
  - 각 속성의 순서는 중요하지 않음

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

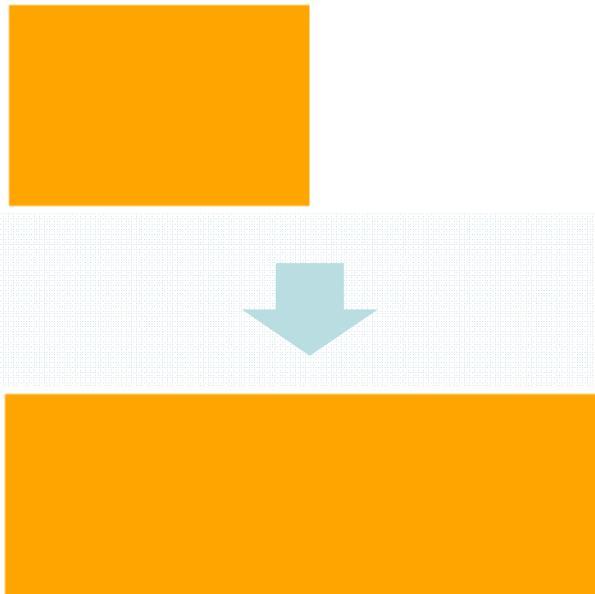
[예] transition : width 5s ease-in 3s;





## ■ transition의 예 - 3

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Insert title here</title>
    <style type="text/css">
        div {
            width: 150px; height: 100px;
            background: orange;
            transition:width 2s ease-in 1s;
        }
        div:hover { width: 300px ;}
    </style>
</head>
<body>
    <div></div>
</body>
</html>
```

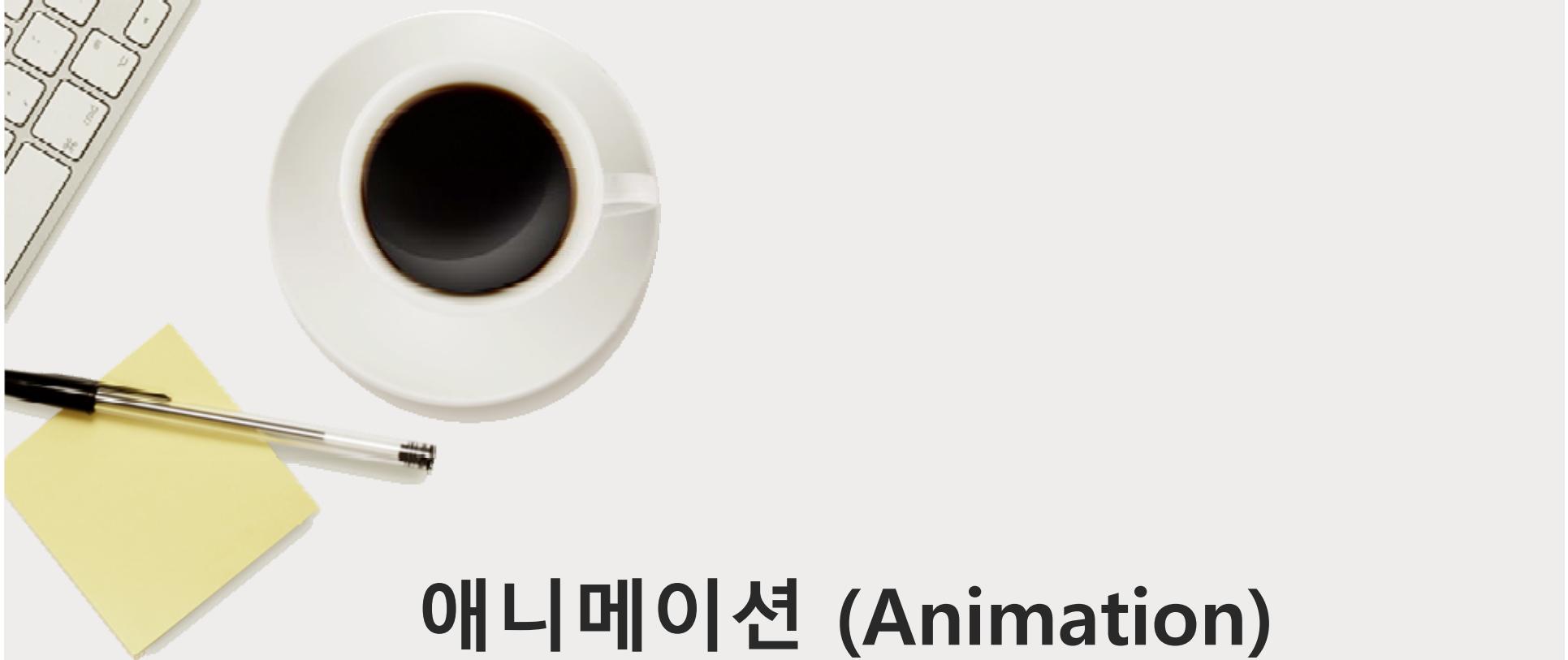




## ■ transition의 예 - 4

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Insert title here</title>
    <style type="text/css">
        div {
            width: 150px; height: 100px;
            position: relative; left: 10px;
            background: orange;
            transition-property: width, left, color;
            transition-duration: 2s, 2s, 2s;
            transition-timing-function: ease-in, linear, ease-in-out;
            transition-delay: 0s, 1s, 1s
        }
        div:hover {
            width: 300px; left: 200px; background: red;
        }
    </style>
</head>
<body>
    <div></div>
</body>
</html>
```





# 애니메이션 (Animation)



## ■ 애니메이션 (animation)

- 요소를 동적으로 표현하는 또 다른 방법으로 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) 규칙을 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 위한 방법 제공

- @keyframe 내에 타임 라인(time line)을 지정
  - : 타임라인은 퍼센트를 사용해 지정 (0% ~ 100 %)
  - : 타임라인의 수가 많을 수록 세밀한 애니메이션 지정 가능

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

```
@keyframes 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; }  
}
```





## ■ animation-name 속성

- 애니메이션의 이름을 지정하여 해당 애니메이션이 실행되도록 하는 속성
  - @keyframes에 지정한 애니메이션 이름을 사용

```
animation-name : 애니메이션 이름
```

```
animation-name : myAnimation
```

## ■ animation-duration 속성

- 애니메이션의 지속시간을 초단위로 지정하기 위한 속성

```
animation-duration : 지속시간
```

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





## ■ animation-timing-function 속성

- 애니메이션의 진행과정 중 속도의 변화를 지정하기 위한 속성
  - 트랜지션에서 transition-timing-function 속성과 같은 값을 가짐

animation-timing-function : 타이밍

animation-timing-function : ease-in;

## ■ animation-iteration-count 속성

- 애니메이션의 반복횟수를 지정하기 위한 속성
  - 반복할 횟수를 정수로 지정
  - infinite를 지정할 경우 무한 반복 수행

animation-iteration-count : 정수 | infinite

animation-iteration-count : 3;





## ■ animation-direction 속성

- 애니메이션의 진행방향을 지정하기 위한 속성
  - 순방향 또는 역방향으로 지정 가능
  - 역방향으로 지정할 경우 timing-function도 역방향으로 지정됨

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

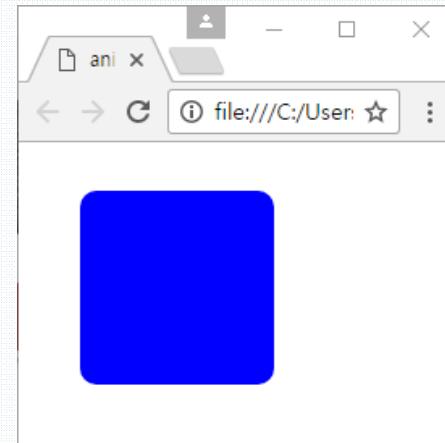
속성 값	의미
normal	<ul style="list-style-type: none"><li>- 애니메이션 진행을 순방향으로 지정</li><li>- 단방향으로만 진행 ( 반복 횟수가 2일 경우- 단방향으로만 2회 발생 )</li></ul>
alternate	<ul style="list-style-type: none"><li>- 순방향과 역방향으로 애니메이션이 연속해서 발생</li><li>- 반복 횟수가 2일 경우 순방향과 역방향 왕복으로 발생</li></ul>
reverse	<ul style="list-style-type: none"><li>- 애니메이션 진행을 순방향으로 지정</li><li>- 방향만 다를 뿐 normal과 동일</li></ul>
alternate-reverse	<ul style="list-style-type: none"><li>- 역방향과 순방향으로 애니메이션이 연속해서 발생</li><li>- 순서만 다를 뿐 alternate와 동일</li></ul>





## ■ 애니메이션 예제 - 1

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<style type="text/css">
div {
    width: 100px; height: 100px; margin:30px; padding:10px;
    font-weight:bold; font-size:16px;
    border-radius:10px; background-color: blue;
}
@keyframes myDIV1 {
    from { background-color: blue; }
    50% { background-color: red; }
    to { transform:scale(1.5, 1.5); }
}
#myDIV1 {
    animation-name: myDIV1;
    animation-duration: 5s;
    animation-iteration-count:2;
    animation-timing-function : linear;
    animation-direction:alternate;
}
</style>
</head>
<body>
    <div id="myDIV1"></div>
</body>
</html>
```





## ■ animation-delay 속성

- 애니메이션의 시작하기 전까지의 지연시간을 지정하기 위한 속성
  - transition-delay와 거의 유사

**animation-delay : 지연시간**

```
animation-delay : 3s;
```

## ■ animation-fill-mode 속성

- 애니메이션의 시작하기 전이나 종료 후의 상태를 지정하기 위한 속성

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

속성 값	의미
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
alternate-reverse	forwards와 backwards를 동시에 지정





## ■ animation-fill-mode 속성의 예

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">

<style type="text/css">
    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;}
    }

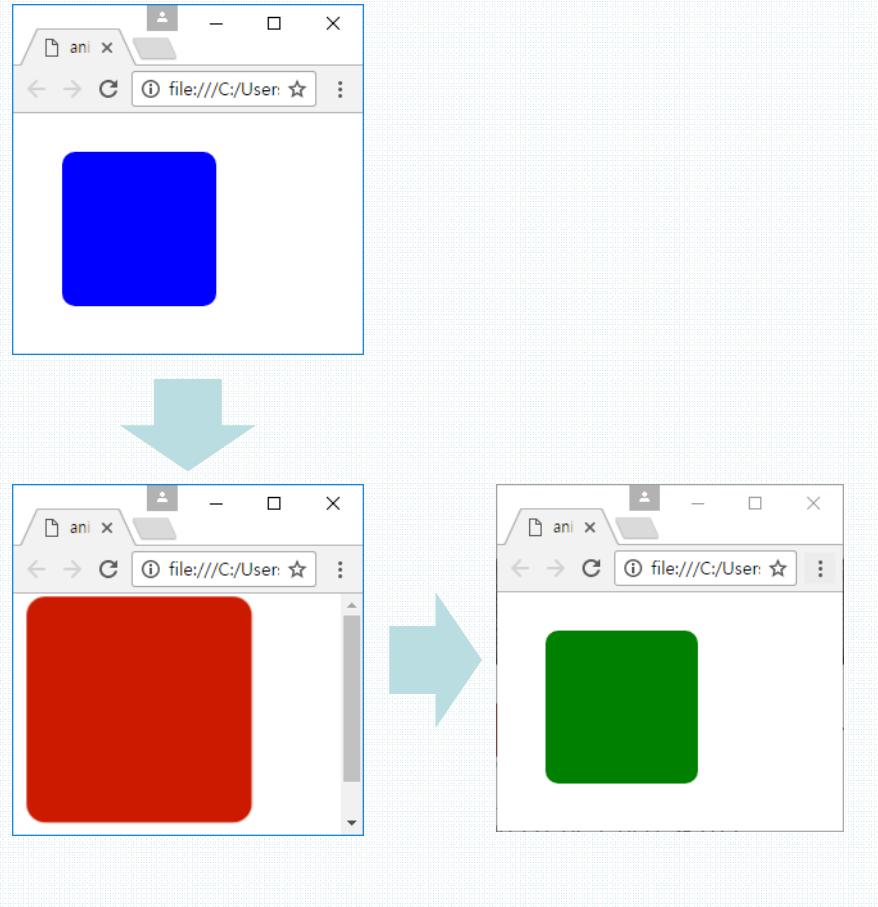
```



# 애니메이션 (Animation)



```
#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>
```





## ■ animation-play-state 속성

- 애니메이션의 진행 또는 정지를 지정하기 위한 속성

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

animation-play-state : running;

### - running

- : 애니메이션을 진행상태로 지정
- : running 상태로 지정되어 있는 동안은 애니메이션이 정상적으로 진행됨

### - paused

- : 애니메이션을 일시 정지 상태로 지정
- : paused 상태로 지정되면 애니메이션의 동작 뿐만 아니라 animation-duration 값과 같은 애니메이션 실행 관련 시간도 일시정지됨





## ■ animation 속성

- 애니메이션의 속성을 일괄 지정하기 위한 속성
  - 속성의 순서는 중요하지 않음

```
animation : 이름 지속시간 타이밍함수 자연시간 반복횟수 방향
```

```
animation : myAnimation 5s ease-out 2s 2 alternate;
```

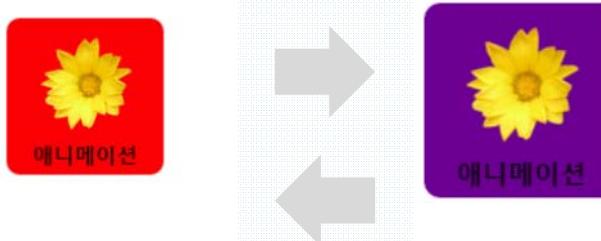
- 일부 속성은 생략할 수 있음
  - animation-name과 animation-duration은 생략할 수 없음
  - 시간(초)를 하나만 지정할 경우
    - : 지속 시간으로 인식함
  - 시간(초)을 두 개 모두 지정한 경우
    - : 앞은 지속 시간으로 인식하고 뒤는 자연시간으로 인식함
- animation-play-state 속성은 일괄지정에 포함시킬 수 없음





## ■ 애니메이션 예제 - 2

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<style type="text/css">
    img { width: 80%; height: 80% }
    div {
        width: 100px; height: 100px; margin: 20px;
        font-weight:bold; font-size:16px;
        padding:10px; border-radius:10px;
    }
    @keyframes myDIV1 {
        from { background-color: blue; }
        50% { transform: scale(1.3, 1.3); }
        to { background-color: red; }
    }
    #myDIV1 {
        animation: myDIV1 5s infinite linear;
    }
</style>
</head>
<body>
    <div id="myDIV1" align=center><br>애니메이션</div>
</body>
</html>
```





## ■ 애니메이션 예제 - 3

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<style type="text/css">

    img { width: 80%; height: 80% }

    div {
        width: 100px; height: 100px; margin: 20px;
        font-weight:bold; font-size:16px;
        padding:10px; border-radius:10px;
        position:relative;
    }

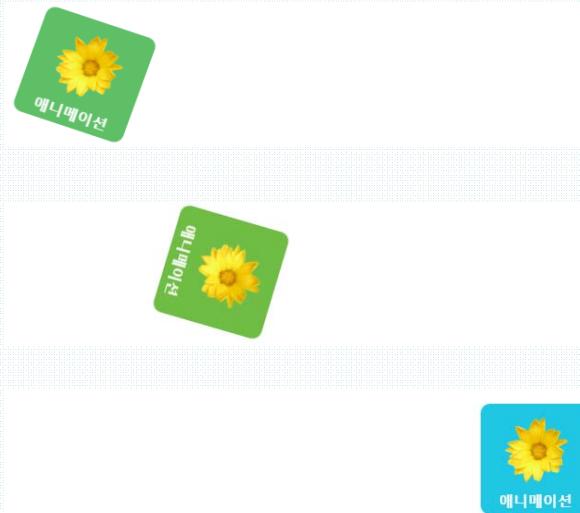
    @keyframes myDIV1 {
        0% { transform: rotate(0deg);left:0px; }
        25% { transform: rotate(20deg);left:0px; }
        50% { transform: rotate(0deg);left:500px; }
        55% { transform: rotate(0deg);left:500px; }
        70% { transform: rotate(0deg);
              left:500px;
              background:#1ec7e6;
        }
        100% { transform: rotate(-360deg);left:0px; }
    }
</style>
```



# 애니메이션 (Animation)



```
#myDIV1 {  
    background:#92B901; color:#ffffff;  
    animation:myDIV1 5s infinite;  
}  
</style>  
</head>  
<body>  
    <div id="myDIV1" align=center><br>애니메이션</div>  
</body>
```





## ■ 애니메이션 예제 - 4

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<style type="text/css">
    img { width: 80%; height: 80% }
    div {
        width: 100px; height: 100px; margin: 20px;
        font-weight:bold; font-size:16px;
        padding:10px; border-radius:10px;
    }

    @keyframes myDIV1 {
        from, to {background:red; left:0px; top:0px;}
        25% {background:yellow; left:200px; top:0px;}
        50% {background:blue; left:200px; top:200px;}
        75% {background:green; left:0px; top:200px;}
    }

    #myDIV1 {
        background: red; position: relative;
        animation:myDIV1 5s infinite;
    }
</style>
</head>
```



# 애니메이션 (Animation)



```
<body>  
  
    <div id="myDIV1" align=center>  
        <br>애니메이션  
    </div>  
  
</body>  
</html>
```

