13. CSS3와 애니메이션

13-1 변형

13-2 변형과 관련된 속성들

13-1 트랜지션

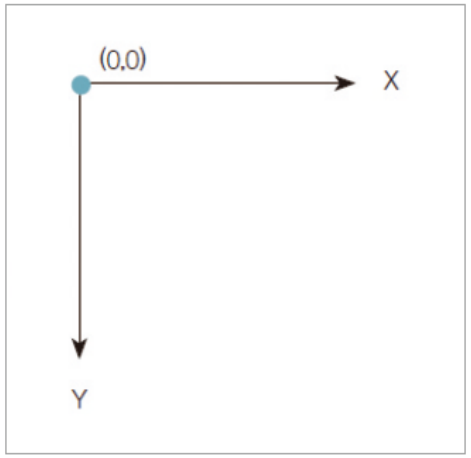
13-1 애니메이션

13-1 변형

변형(transform, 트랜스폼) : 특정 요소의 크기나 형태 등 스타일이 바뀌는 것

2차원 변형

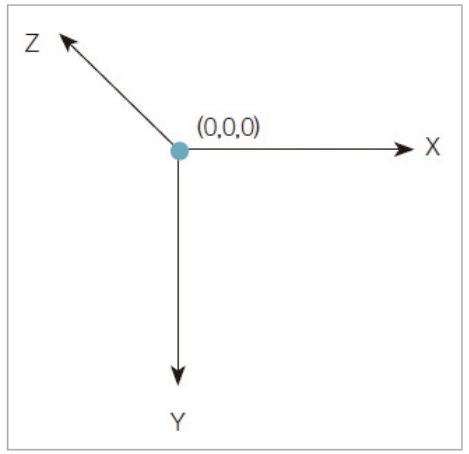
- 수평이나 수직으로 웹 요소 변형
- 크기나 각도만 지정하면 됨
- 2차원 좌표 사용



2차원 좌표계

3차원 변형

- x축과 y축에 원근감 추가
- z축은 앞뒤로 이동. 보는 사람 쪽으로 다가올 수록 값이 더 커짐



3차원 좌표계

13-1 변형

transform과 변형 함수

- 웹 요소를 변형하려면 transform: 다음에 변형 함수를 함께 입력함
- 구식 모던 브라우저까지 고려한다면 브라우저 접두사를 붙여야 함

2차원 변형 함수

변형 함수	설명
translate(tx, ty)	지정한 크기만큼 x축과 y축으로 이동합니다.
translateX(tx)	지정한 크기만큼 x축으로 이동합니다.
translateY(ty)	지정한 크기만큼 y축으로 이동합니다.
scale(sx, sy)	지정한 크기만큼 x축과 y축으로 확대/축소합니다.
scaleX(sx)	지정한 크기만큼 x축으로 확대/축소합니다.
scaleY(sy)	지정한 크기만큼 y축으로 확대/축소합니다.
rotate(각도)	지정한 각도만큼 회전합니다.
skew(ax, ay)	지정한 각도만큼 x축과 y축으로 왜곡합니다.
skewX(ax)	지정한 각도만큼 x축으로 왜곡합니다.
skewY(ay)	지정한 각도만큼 y축으로 왜곡합니다.

3차원 변형 함수

변형 함수	설명
matrix3d(n [, n])	4*4 행렬을 이용해 이동과 확대/축소, 회전 등의 변환을 지정합니다.
translate3d(tx, ty, tz)	지정한 크기만큼 x축과 y축, z축으로 이동합니다.
translateZ(tz)	지정한 크기만큼 z축으로 이동합니다.
scale3d(sx, sy, sz)	지정한 크기만큼 x축과 y축, z축으로 확대/축소합니다.
scaleZ(sz)	지정한 크기만큼 z축으로 확대/축소합니다.
rotate3d(rx, ry, rz, 각도)	지정한 각도만큼 회전합니다.
rotateX(각도)	지정한 각도만큼 x축으로 회전합니다.
rotateY(각도)	지정한 각도만큼 y축으로 회전합니다.
rotateZ(각도)	지정한 각도만큼 z축으로 회전합니다.
perspective(길이)	입체적으로 보일 수 있는 깊이 값을 지정합니다.

13-1 변형

translate 함수

지정한 방향으로 이동할 거리를 지정하면 해당 요소를 이동시킴

기본형

```
transform:translate(tx, ty)
transform:translate3d(tx, ty, tz)
transform:translateX(tx)
transform:translateY(ty)
transform:translateZ(tz)
```

- **transform:translate(tx, ty)** - x축 방향으로 tx만큼, y축 방향으로 ty만큼 이동합니다. tx와 ty 두 가지 값을 사용하지만 ty 값이 주어지지 않으면 0으로 간주합니다.
- **transform:translate3d(tx, ty, tz)** - x축 방향으로 tx만큼, y축 방향으로 ty만큼, 그리고 z축 방향(앞뒤)으로 tz만큼 이동합니다.
- **transform:translateX(tx)** - x축 방향으로 tx만큼 이동합니다.
- **transform:translateY(ty)** - y축 방향으로 ty만큼 이동합니다.
- **transform:translateZ(tz)** - z축 방향으로 tz만큼 이동합니다.

```
<style>
  .movex:hover { transform: translateX(50px); }
  .movey:hover { transform: translateY(20px); }
  .movexy:hover { transform : translate(10px, 20px); }
</style>

<div class="origin">
  <div class="movex"><img src="images/bus.jpg"></div>
</div>
<div class="origin">
  <div class="movey"><img src="images/bus.jpg"></div>
</div>
<div class="origin">
  <div class="movexy"><img src="images/bus.jpg"></div>
</div>
```

13-1 변형

scale 함수

지정한 크기만큼 요소를 확대/축소

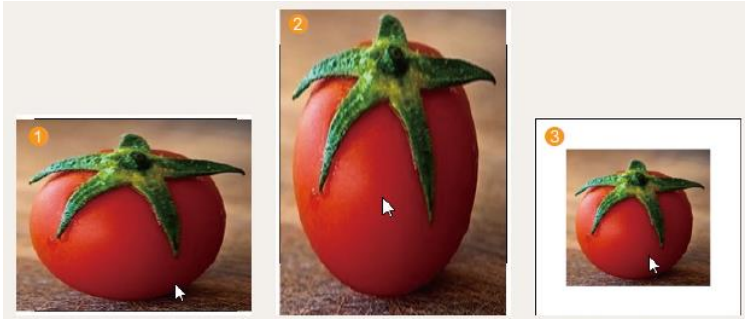
기본형

```
transform:scale(sx, sy) -1
transform:scale3d(sx, sy, sz) -2
transform:scaleX(sx) -3
transform:scaleY(sy) -4
transform:scaleZ(sz) -5
```

- **transform:scale(sx, sy)** - x축 방향으로 sx만큼, y축 방향으로 sy만큼 확대합니다. sy 값이 주어지지 않는다면 sx 값과 같다고 간주합니다. 예를 들어 scale(2.0)는 scale(2,2)와 같은 함수이며 요소를 두 배로 확대합니다.
- **transform:scale3d(sx, sy, sz)** - x축 방향으로 sx만큼, y축 방향으로 sy만큼, 그리고 z축 방향으로 sz만큼 확대합니다.
- **transform:scaleX(sx)** - x축 방향으로 sx만큼 확대합니다.
- **transform:scaleY(sy)** - y축 방향으로 sy만큼 확대합니다.
- **transform:scaleZ(sz)** -z축 방향으로 sz만큼 확대합니다.

```
<style>
  .scalex:hover { transform: scaleX(1.2); }
  .scaley:hover { transform: scaleY(1.5); }
  .scale:hover { transform: scale(0.7); }
</style>

<div class="origin">
  <div class="scalex"><img src="images/fruit.jpg"></div>
</div>
<div class="origin">
  <div class="scaley"><img src="images/fruit.jpg"></div>
</div>
<div class="origin">
  <div class="scale"><img src="images/fruit.jpg">  </div>
</div>
```



13-1 변형

rotate 함수

- 각도만큼 웹 요소를 시계 방향이나 시계 반대 방향으로 회전
- 일반 각도(degree)나 래디안(radian) 값 사용(1래디안=1/180°)

2차원 함수 기본형 `transform:rotate(각도)`

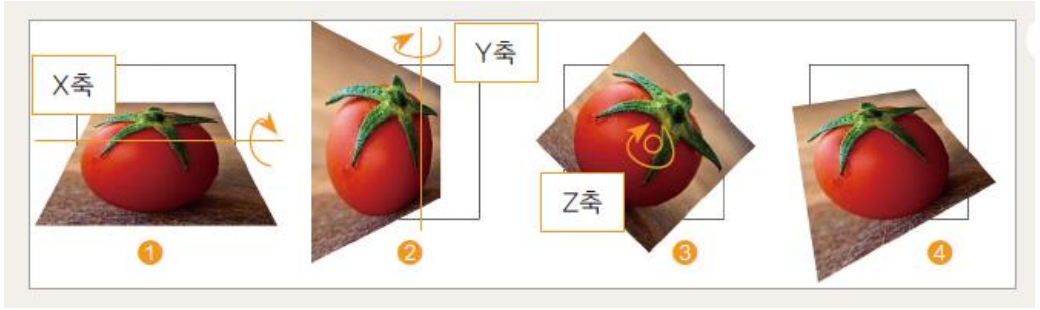
```
<style>
  .rotate1:hover { transform: rotate(20deg); }
  .rotate2:hover { transform: rotate(-40deg); }
</style>

<div class="origin">
  <div class="rotate1"> <img src="images/fruit.jpg"></div>
</div>
<div class="origin">
  <div class="rotate2"> <img src="images/fruit.jpg"></div>
</div>
```

3차원 함수 기본형
```
transform:rotate(rx, ry, 각도)
transform:rotate3d(rx, ry, rz, 각도)
transform:rotateX(각도)
transform:rotateY(각도)
transform:rotateZ(각도)
```

```
<style>
  .origin { perspective: 200px; }
  .rotatex { transform: rotateX(45deg); }
  .rotatey { transform: rotateY (45deg); }
  .rotatez { transform: rotateZ(45deg); }
  .rotatexyz { transform: rotate3d(2.5, 1.2, -1.5, 45deg); }
</style>

<div class="origin"><div class="rotatex"><img src="images/fruit.jpg"
alt=""></div></div>
<div class="origin"><div class="rotatey"><img src="images/fruit.jpg"
alt=""></div></div>
<div class="origin"><div class="rotatez"><img src="images/fruit.jpg"
alt=""></div></div>
<div class="origin"><div class="rotatexyz"><img src="images/fruit.jpg"
alt=""></div></div>
```



13-1 변형

skew 함수

요소를 지정한 각도만큼 비틀어 왜곡

```
기본형   transform:skew(ax, ay) ❶
         transform:skewX(ax) ❷
         transform:skewY(ax) ❸
```

- **transform:skew(ax, ay)** - 첫 번째 각도는 x축에서의 왜곡 각도이고 두 번째 각도는 y축에서의 왜곡 각도입니다. 두 번째 값이 주어지지 않으면 y축에 대한 왜곡 각도를 0으로 간주해 y축으로는 왜곡이 생기지 않습니다.
- **transform:skewX(ax)** - x축에서만 주어진 각도만큼 왜곡합니다.
- **transform:skewY(ay)** - y축에서만 주어진 각도만큼 왜곡합니다.

```
3차원 함수 기본형   transform:rotate(rx, ry, 각도)
                    transform:rotate3d(rx, ry, rz, 각도)
                    transform:rotateX(각도)
                    transform:rotateY(각도)
                    transform:rotateZ(각도)
```

```
<style>
  .skewx:hover { transform: skewX(30deg); }
  .skewy:hover { transform: skewY(15deg); }
  .skewxy:hover { transform: sk (-25deg, -15deg); }
</style>

<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>
```

13-2 변형과 관련된 속성들

transform-origin 속성

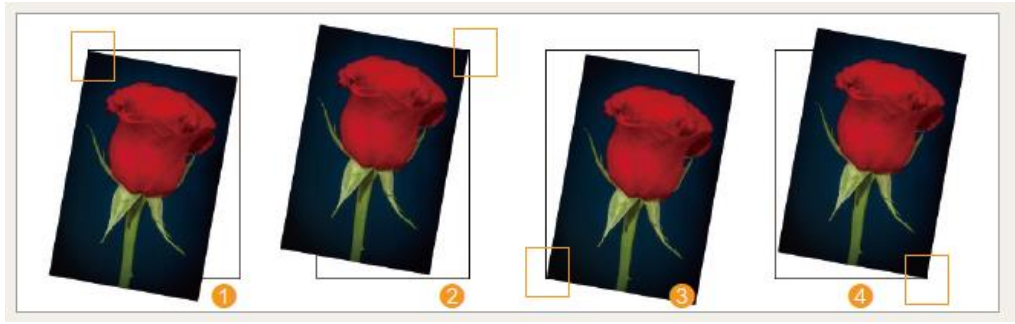
특정 지점을 변형의 기준으로 설정

기본형 `transform-origin: <x축> <y축> <z축> | initial | inherit ;`

속성 값	설명
<x축>	원점 기준의 x 좌표값으로 길이 값이나 <백분율>, left, center, right 중에서 사용할 수 있습니다.
<y축>	원점 기준의 y 좌표값으로 길이 값이나 <백분율>, top, center, bottom 중에서 선택할 수 있습니다.
<z축>	원점 기준의 z 좌표값으로 길이 값만 사용할 수 있습니다.

```
<style>
  .rose {transform: rotateZ(10deg);}
  .ltop .rose{ transform-origin: left top; }
  .rtop .rose { transform-origin: right top; }
  .lbottom .rose { transform-origin: left bottom; }
  .rbottom .rose { transform-origin : right bottom; }
</style>

<div class="origin">
  <div class="ltop"><img src="images/rose.jpg" class="rose"></div>
</div>
<div class="origin">
  <div class="rtop"><img src="images/rose.jpg" class="rose"></div>
</div>
<div class="origin">
  <div class="lbottom"><img src="images/rose.jpg" class="rose"></div>
</div>
<div class="origin">
  <div class="rbottom"><img src="images/rose.jpg" class="rose"></div>
</div>
```



13-2 변형과 관련된 속성들

perspective 속성

- 원근감을 갖게 함.
- 속성 값은 0보다 커야 하며 값이 클수록 사용자로부터 멀어짐.

기본형 `perspective: <크기> | none;`

속성 값	설명
<크기>	원래 위치에서 사용자가 있는 방향으로 얼마나 이동하는지를 픽셀 크기로 지정합니다.
none	perspective를 지정하지 않습니다. 기본 값입니다.★

perspective-origin 속성

- 입체적으로 표현할 요소의 아랫부분(bottom) 위치 지정
- 좀더 높은 곳에서 원근을 조절하는 듯한 느낌을 갖게 함.

기본형 `perspective-origin: <x축 값> | <y축 값>;`

속성 값	설명
<x축 값>	웹 요소가 x축에서 어디에 위치하는지를 지정합니다. 사용할 수 있는 값은 길이 값이나 백분율, left, right, center입니다. 기본 값은 50%입니다.
<y축 값>	웹 요소가 y축에서 어디에 위치하는지를 지정합니다. 사용할 수 있는 값은 길이 값이나 백분율, top, center, bottom입니다. 기본 값은 50%입니다.

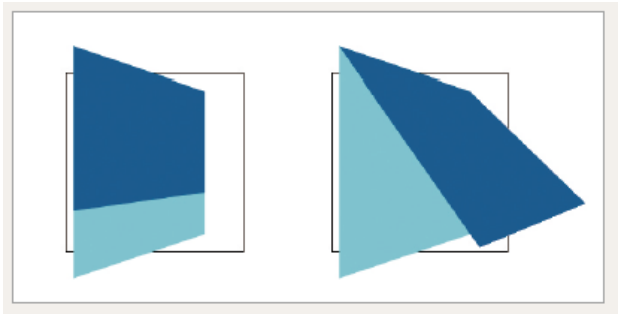
13-2 변형과 관련된 속성들

transform-style 속성

부모 요소에 적용한 3D 변형을 하위 요소에도 적용

기본형 `transform-style: flat | preserve-3d`

속성 값	설명
flat	하위 요소를 평면으로 처리합니다.
preserve-3d	하위 요소들에 3D효과를 적용합니다.



```
<style>
  .box1 {
    background:#82cbd8;
    transform: rotateY(45deg);
  }
  .box2 {
    background: #0d6097;
    transform-origin: left top;
    transform: rotateX(45deg);
  }
  #tr-style1 { transform-style:flat ; }
  #tr-style2 { transform-style:preserve-3d; }
</style>

<div class="container">
  <div class="box1" id="tr-style1">
    <div class="box2"></div>
  </div>
</div>
<div class="container">
  <div class="box1" id="tr-style2">
    <div class="box2"></div>
  </div>
</div>
```

13-2 변형과 관련된 속성들

backface-visibility 속성

요소의 뒷면, 즉 반대쪽 면을 표시할 것인지 결정

기본형 `backface-visibility : visible | hidden`

속성 값	설명
visible	뒷면을 표시합니다. 기본 값입니다.★
hidden	뒷면을 표시하지 않습니다.

두 개 이상의 변형 동시에 사용하려면

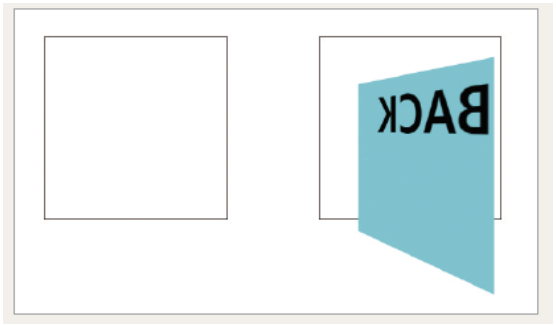
transform 속성에 변형 함수를 나열함.

예) 크기를 2배 확대하면서 x 축 기준으로 180도 회전.

```
transform: scale(2) perspective(120px) rotateX(180deg);
```

```
<style>
  .box {
    background:#82cbd8;
    transform: rotateY(135deg);
  }
  #back1 { backface-visibility: hidden; }
  #back2 { backface-vis : visible; }
</style>

<div class="container">
  <div class="box" id="back1"><h1>BACK</h1></div>
</div>
<div class="container">
  <div class="box" id="back2"><h1>BACK</h1></div>
</div>
```

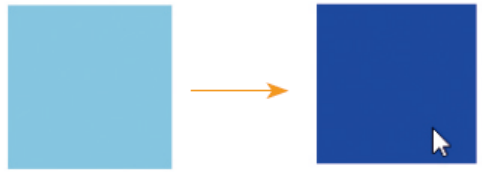


13-3 트랜지션

트랜지션이란

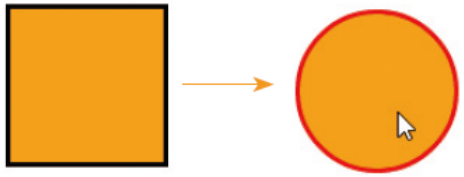
웹 요소의 스타일 속성이 조금씩 자연스럽게 바뀌는 것

예) 하늘색 도형 위로 마우스를 올려놓으면 도형이 하늘색에서 파란색으로 바뀌고 마우스를 치우면 원래 배경 색으로 되돌아감.



사각형의 배경 색이 바뀌는 트랜지션

예) 도형 위로 마우스를 올려놓으면 사각형의 테두리와 테두리색이 바뀌고 마우스를 치우면 원래 스타일로 되돌아감.



사각형의 모양과 테두리 색이 바뀌는 트랜지션

트랜지션의 속성

속성	설명
transition-property	트랜지션 대상을 설정합니다.
transition-duration	트랜지션 진행 시간을 설정합니다.
transition-timing-function	트랜지션 속도 곡선을 설정합니다.
transition-delay	트랜지션 지연 시간을 설정합니다.
transition	transition-property와 transition-duration, transition-timing-function, transition-delay 속성을 한꺼번에 설정합니다.

13-3 트랜지션

transition-property 속성

- 트랜지션을 적용할 속성 선택
- 이 속성을 지정하지 않으면 모든 속성이 트랜지션 대상이 됨.

기본형 `transition-property: all | none | <속성 이름>`

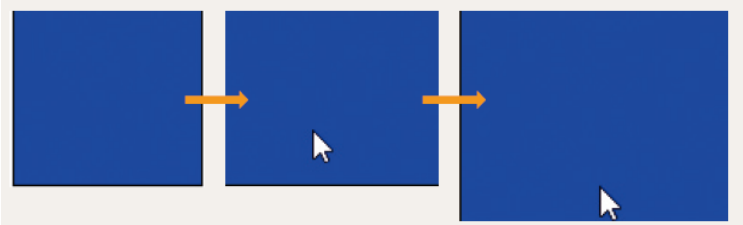
속성 값	설명
all	all 값을 사용하거나 transition-property를 생략할 경우, 요소의 모든 속성이 트랜지션 대상이 됩니다. 기본 값입니다.★
none	트랜지션 동안 아무 속성도 바뀌지 않습니다.
<속성 이름>	트랜지션 효과를 적용할 속성 이름을 지정합니다. 예를 들어 배경 색만 바꿀 것인지, width 값을 바꿀 것인지 원하는 대상만 골라 지정할 수 있습니다. 속성이 여러 개일 경우, 쉼표(,)로 구분해 나열합니다.

```
transition-property:all; /* 해당 요소의 모든 속성에 트랜지션 적용 */
transition-property:background-color; /* 해당 요소의 배경 색에 트랜지션 적용 */
transition-property:width, height; /* 해당 요소의 너비와 높이에 트랜지션 적용 */
```

transition-duration 속성

- 트랜지션 진행 시간 지정
- 시간 단위는 초(seconds) 또는 밀리초(milliseconds)
- 트랜지션이 여러 개라면 쉼표(,)로 구분해 진행 시간 지정

```
<style>
.tr1 {
  width: 100px;
  height: 100px;
  background-color: blue;
  border: 1px solid black;
  transition-property: width, height;
  transition-du : 2s, 1s;
}
.tr1:hover {
  width:200px;
  height:120px;
}
</style>
```



13-3 트랜지션

transition-timing-function 속성

트랜지션의 시작과 중간, 끝에서의 속도 지정

기본형 `transition-timing-function: linear | ease | ease-in | ease-out | ease-in-out | cubic-bezier(n,n,n,n)`

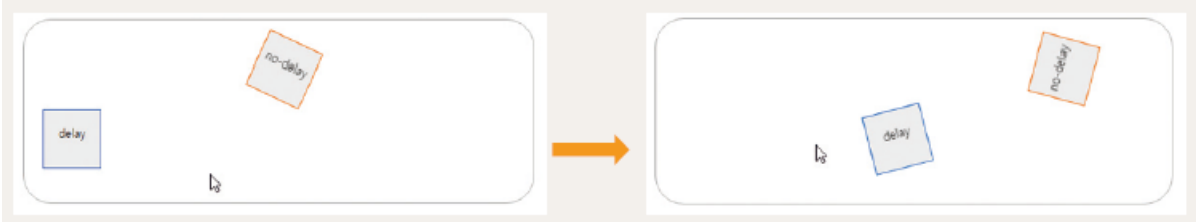
속성 값	설명
linear	시작부터 끝까지 똑같은 속도로 트랜지션을 진행합니다.
ease	처음에는 천천히 시작하고 점점 빨라지다가 마지막에는 천천히 끝냅니다. 기본 값입니다.★
ease-in	시작을 느리게 합니다.
ease-out	느리게 끝냅니다.
ease-in-out	느리게 시작하고 느리게 끝냅니다.
cubic-bezier(n,n,n,n)	베지에 함수를 직접 정의해 사용합니다. n에서 사용할 수 있는 값은 0~1입니다.

```
transition-timing-function: linear;
```

transition-delay 속성

- 트랜지션이 언제부터 시작될지 지연 시간 지정
- 시간 단위는 초(seconds) 또는 밀리초(milliseconds). 기본값 0

```
<style>
#no-delay {
  transition-duration: 3s;
}
#delay {
  transition-duration: 3s;
  transition-del : 1s;
}
</style>
```



13-3 트랜지션

transition-timing-function 속성

트랜지션의 시작과 중간, 끝에서의 속도 지정

기본형 `transition-timing-function: linear | ease | ease-in | ease-out | ease-in-out | cubic-bezier(n,n,n,n)`

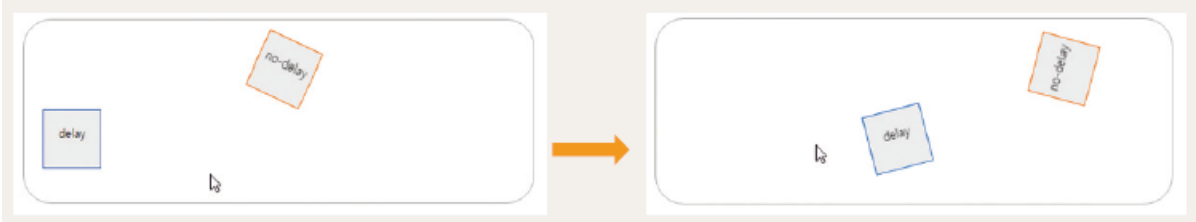
속성 값	설명
linear	시작부터 끝까지 똑같은 속도로 트랜지션을 진행합니다.
ease	처음에는 천천히 시작하고 점점 빨라지다가 마지막에는 천천히 끝냅니다. 기본 값입니다.★
ease-in	시작을 느리게 합니다.
ease-out	느리게 끝냅니다.
ease-in-out	느리게 시작하고 느리게 끝냅니다.
cubic-bezier(n,n,n,n)	베지에 함수를 직접 정의해 사용합니다. n에서 사용할 수 있는 값은 0~1입니다.

```
transition-timing-function: linear;
```

transition-delay 속성

- 트랜지션이 언제부터 시작될지 지연 시간 지정
- 시간 단위는 초(seconds) 또는 밀리초(milliseconds). 기본값 0

```
<style>
#no-delay {
  transition-duration: 3s;
}
#delay {
  transition-duration: 3s;
  transition-del : 1s;
}
</style>
```



13-4 애니메이션

CSS와 애니메이션

- 웹 요소에 애니메이션 추가
- 애니메이션을 시작해 끝내는 동안 원하는 곳 어디서든 스타일을 바꾸며 애니메이션을 정의할 수 있다.
- 키프레임(keyframe) : 애니메이션 중간에 스타일이 바뀌는 지점

인터넷 익스플로러 10 이상과 최신 모던 브라우저에서 지원하며 이전 모던 브라우저를 고려하려면 –webkit-, -moz- 접두사를 붙여야 한다.

CSS 애니메이션에서 사용하는 주요 속성

속성	설명
@keyframes	애니메이션이 바뀌는 지점을 설정합니다.
animation-delay	애니메이션 지연 시간을 지정합니다.
animation-direction	애니메이션 종료 후 처음부터 시작할지, 역방향으로 진행할지를 지정합니다.
animation-duration	애니메이션 실행 시간을 설정합니다.
animation-fill-mode	애니메이션이 종료되었거나 지연되어 애니메이션이 실행되지 않는 상태일 때 요소의 스타일을 지정합니다.
animation-iteration-count	애니메이션 반복 횟수를 지정합니다.
animation-name	@keyframes로 설정해 놓은 중간 상태의 이름을 지정합니다.
animation-play-state	애니메이션을 멈추거나 다시 시작합니다.
animation-timing-function	애니메이션의 속도 곡선을 지정합니다.
animation	animation 하위 속성들을 한꺼번에 묶어 지정합니다.

13-3 트랜지션

@keyframes 속성

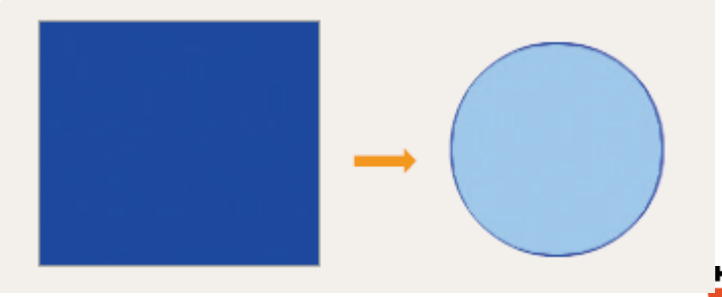
- 애니메이션의 시작과 끝을 비롯해 상태가 바뀌는 지점을 설정
- '이름'으로 애니메이션 구별

```
기본형   @keyframes  <이름> {
            <선택자> { <스타일> }
         }
```

- 시작 위치는 0%, 끝 위치 100%로 놓고 위치 지정
- 시작과 끝 위치만 사용한다면 from, to 키워드 사용 가능
- @-webkit-keyframes나 @-moz-keyframes처럼 브라우저 접두사를 붙여함

```
<style>
div {
  width: 100px;
  height: 100px;
  background-color: blue;
  animation-name: change-bg;
  animation-duration: 3s;
}
@key change-bg {
  from {
    background-color: blue;
    border: 1px solid black;
  }
  to {
    background-color: #a5d6ff;
    border:1px solid blue;
    border-radius: 50%;
  }
}
</style>
```

시작할 때 파란색 사각형이었다가 끝날 때 옅은 파란색 원으로 바뀌는 애니메이션



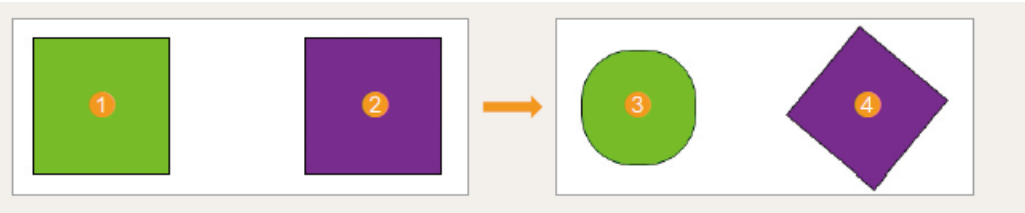
13-3 트랜지션

animation-name 속성

@keyframes 속성에서 만든 애니메이션 이름을 사용

@keyframes을 이용해 shape와 rotate라는 애니메이션 정의

#box1에는 animation-name:shape를,
#box2에는 animation-name:rotate를 사용해 애니메이션 실행




```
<style>
#box1 {
  background-color: #4cff00;
  border:1px solid black;
  animation-name: shape;
  animation-duration: 3s;
}
#box2 {
  background-color: #8f06b0;
  border: 1px solid black;
  animation-name : rotate;
  animation-duration: 3s;
}
@keyframes shape {
  from {
    border: 1px solid black;
  }
  to {
    border: 1px solid black;
    border-radius: 50%;
  }
}
@keyframes rotate {
  from {
    transform:rotate(0deg)
  }
  to {
    transform: rotate(45deg);
  }
}
</style>

<div id="box1"></div>
<div id="box2"></div>
```

13-3 트랜지션

animation-duration 속성

- 애니메이션 실행 시간 설정. 기본값 0
- 사용 가능한 값은 초(s)나 밀리초(ms)

기본형 `animation-duration: <시간>`

animation-direction 속성

애니메이션이 끝난 후 원래 위치로 돌아가거나 반대 방향으로 애니메이션 실행하도록 지정

기본형 `animation-direction: normal | alternate`

속성 값	설명
normal	애니메이션을 끝까지 실행하면 원래 있던 위치로 돌아갑니다. 기본 값입니다.★
alternate	애니메이션을 끝까지 실행하면 왔던 방향으로 되돌아가면서 애니메이션을 실행합니다.

animation-iteration-count 속성

애니메이션 반복 횟수 지정하기

기본형 `animation-iteration-count: <숫자> | infinite`

속성 값	설명
<숫자>	입력한 숫자만큼 반복합니다. 기본 값은 1입니다.
infinite	무한 반복합니다.

animation-timing-function 속성

애니메이션 속도 곡선 지정

기본형
```
animation-timing-function: linear | ease | ease-in | ease-out |
ease-in-out | cubic-bezier(n,n,n,n)
```

13-3 트랜지션

animation 속성

- 여러 개의 애니메이션 속성을 하나의 속성으로 줄여서 사용
- 지정하지 않은 속성은 기본 값 사용. 하지만 animation-duration 속성 값은 반드시 지정해야 함.

기본형

```
animation: <animation-name> | <animation-duration> |
           <animation-timing-function> | <animation-delay> |
           <animation-iteration-count> | <animation-direction>
```

```
.box {
  animation-name: moving;
  animation-duration: 3s;
  animation-timing-function:ease-in;
  animation-direction: alternate;
  animation-iteration-count: infinite;
  }
```

→

```
.box {
  animation: moving 3s alternate
  infinite ease-in;
}
```

```
<style>
.box {
  width:60px;
  height:60px;
  margin:60px;
  animation:rotate 1.5s infinite, background 1.5s infinite alternate;
}
@keyframes rotate {
  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 {
  from { background: red; }
  50% { background-color: green }
  to { background-color: blue; }
}
</style>

<div class="box"> </div>
```