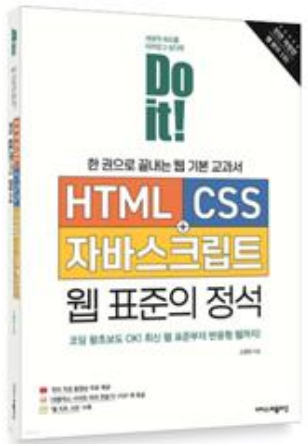
14. 트랜지션과 애니메이션



14-1 트랜스폼 알아보기

14-2 트랜지션 알아보기

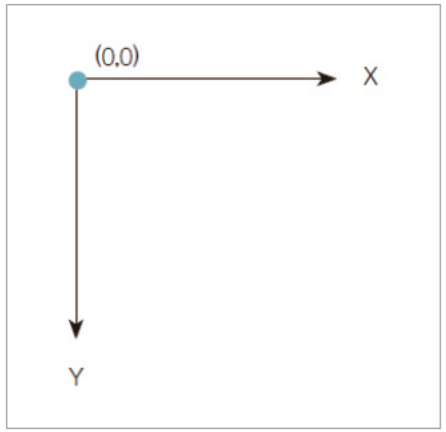
14-3 애니메이션 알아보기

트랜스폼 알아보기

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

2차원 트랜스폼

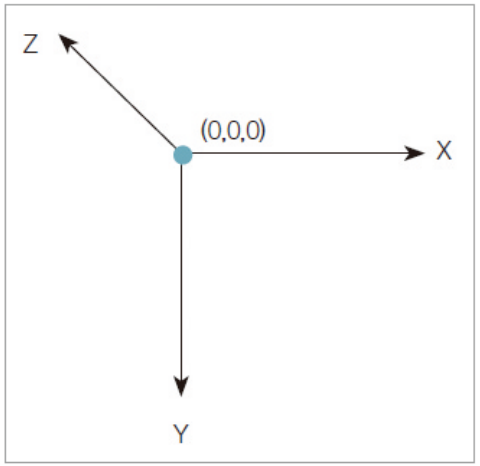
- 수평이나 수직으로 웹 요소 변형
- 크기나 각도만 지정하면 됨
- 2차원 좌표 사용



2차원 좌표계

3차원 트랜스폼

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



3차원 좌표계

트랜스폼 알아보기

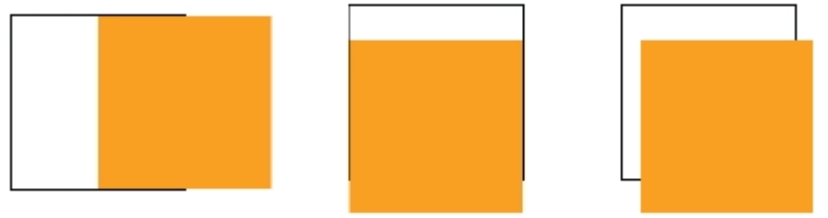
translate 함수

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

```
<style>
 ......
  #movex:hover { transform: translateX(50px); }
  #movey:hover { transform: translateY(20px); }
  #movexy:hover { transform: translate(10px, 20px); }
 .origin > div {
    width:100px;
    height:100px;
    background-color:orange;
    transition: 1s; /* 부드럽게 움직이게 하려면 */
  }
</style>
```



트랜스폼 알아보기

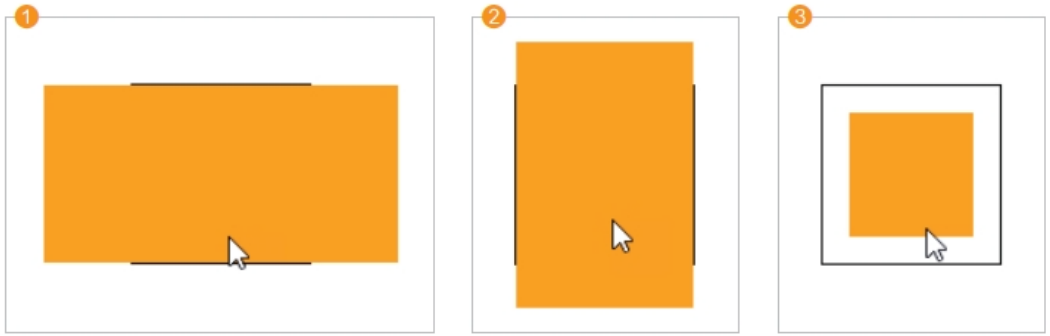
scale 함수

지정한 크기만큼 요소를 확대/축소

```
기본형  transform: scale(sx, sy)
        transform: scale3d(sx, sy, sz)
        transform: scaleX(sx)
        transform: scaleY(sy)
        transform: scaleZ(sz)
```

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

```
<style>
  #scalex { transform: scaleX(2); } /* x축으로 2배 확대 */
  #scaley { transform: scaleY(1.5); } /* y축으로 1.5배 확대 */
  #scale { transform: s (0.7); } /* x, y축으로 0.7배 확대 */
</style>
```



트랜스폼 알아보기

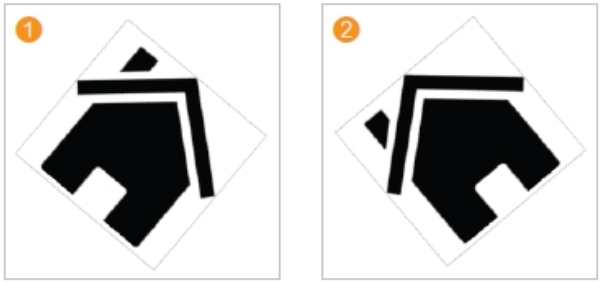
rotate 함수

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

2차원 rotate( ) 함수

기본형 transform: rotate(각도)

```
<style>
  #rotate1:hover { transform: rotate(40deg); } /* 오른쪽으로 40도 회전 */
  #rotate2:hover { transform: rotate(-40deg); } /* 왼쪽으로 40도 회전 */
</style>
```



트랜스폼 알아보기

3차원 rotate( ) 함수

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

perspective 속성

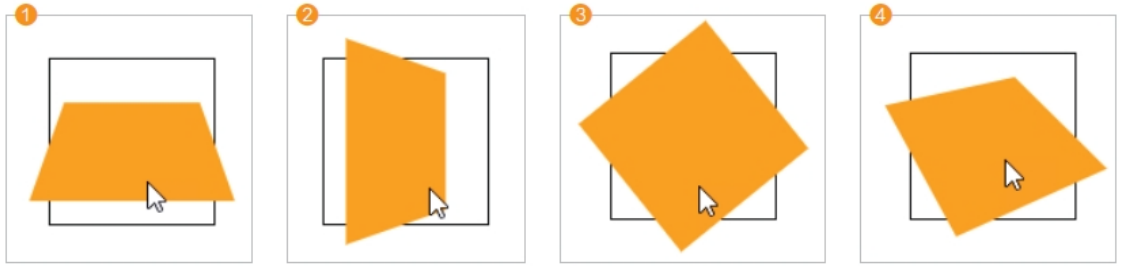
- 원근감을 표현하기 위해 사용하는 속성
- 원래 있던 위치에서 사용자가 있는 쪽으로 얼마나 이동하는지 나타냄.
- 값(픽셀 단위)은 0보다 커야 하며 값이 클수록 사용자로부터 멀어짐.
- perspective 속성은 변형하는 요소의 부모 요소에 정의해야 한다.

```
<style>
  .rotatex:hover { transform: rotateX(50deg); } /* x축으로 50도 회전 */
  #pers { perspective: 300px; } /* 원근감 추가 */
</style>
.....
<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>
```

트랜스폼 알아보기

3차원 rotate( ) 함수

```
<style>
  .origin {
     ......
    perspective: 200px; /* 원근감 추가 */
  }
  .origin > div {
   ......
    transition: all 3s; /* 3초 동안 회전하도록 트랜지션 적용 */
  }
  #rotatex:hover { transform: rotateX(55deg); } /* x축으로 55도 회전 */
  #rotatey:hover { transform: rotateY(55deg); } /* y축으로 55도 회전 */
  #rotatez:hover { transform: r (55deg); } /* z축으로 55도 회전 */
  #rotatexyz:hover {transform: rotate3d(2.5, 1.2, -1.5, 55deg); } /* x, y, z축에 방향 벡터를 지정하고 55도 회전 */
</style>
```



트랜스폼 알아보기

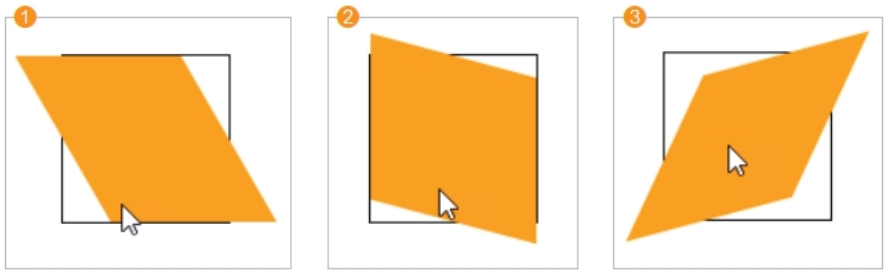
skew 함수

요소를 지정한 각도만큼 비틀어 왜곡

```
기본형  transform: skew(x각도, y각도)
        transform: skewX(x각도)
        transform: skewY(y각도)
```

- **transform:skewX(ax)** – x축을 따라 당김.
- **transform:skewY(ay)** – y축을 따라 당김.
- **transform:skew(ax, ay)** – 첫 번째 각도는 x축을 따라 당기는 각도이고 두 번째 각도는 y축을 따라 당기는 각도. 두 번째 값이 주어지지 않으면 y축에 대한 각도를 0으로 간주함.

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

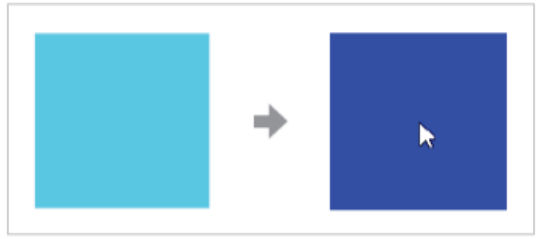


트랜지션 알아보기

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.



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

트랜지션의 속성

종류	설명
transition-property	트랜지션의 대상을 지정합니다.
transition-duration	트랜지션을 실행할 시간을 지정합니다.
transition-timing-function	트랜지션의 실행 형태를 지정합니다.
transition-delay	트랜지션의 지연 시간을 지정합니다.
transition	transition-property, transition-duration, transition-timing-function, transition-delay 속성을 한꺼번에 정합니다.

트랜지션 알아보기

transition-property 속성

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

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

종류	설명
all	all값을 사용하거나 transition-property를 생략할 경우 요소의 모든 속성이 트랜지션 대상이 됩니다. 기본값입니다.
none	트랜지션을 하는 동안 아무 속성도 바뀌지 않습니다.
속성 이름	트랜지션 효과를 적용할 속성을 지정합니다. 예를 들어 배경색만 바꿀 것인지, width값을 바꿀 것인지 원하는 대상만 골라 지정할 수 있습니다. 속성이 여럿일 경우 쉼표(,)로 구분하여 나열합니다.

```
transition-property: all; /* 해당 요소의 모든 속성에 트랜지션 적용 */
transition-property: background-color; /* 해당 요소의 배경색에 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

기본형 `transition-duration: <시간>`

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)`

종류	설명
ease	처음에는 천천히 시작하고 점점 빨라지다가 마지막엔 천천히 끝납니다. 기본값입니다.
linear	시작부터 끝까지 똑같은 속도로 진행합니다.
ease-in	느리게 시작합니다.
ease-out	느리게 끝납니다.
ease-in-out	느리게 시작하고 느리게 끝납니다.
cubic-bezier(n, n, n, n)	베지에 함수를 정의해서 사용합니다. 이때 n값은 0~1 사이만 사용할 수 있습니다.

트랜지션 알아보기

transition-delay 속성

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

기본형 `transition-delay: <시간>`

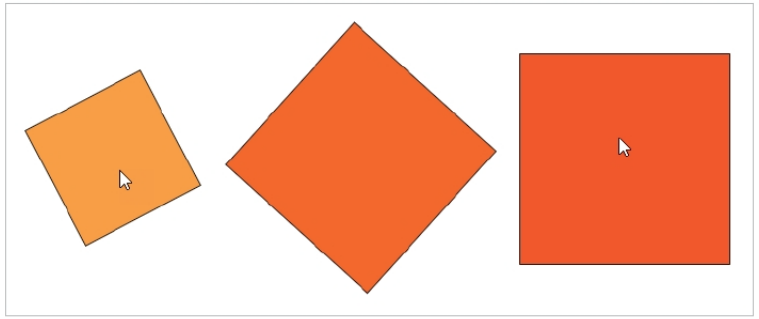
transition 속성

- 트랜지션 관련 속성을 한꺼번에 지정
- 시간 값 속성이 2개이므로, 앞에 오는 시간값은 transition-duration, 뒤에 오는 시간 값은 transition-delay 속성으로 간주

기본형
```
transition: <transition-property값> ¦ <transtion-duration값>
    ¦ <transition-timing-function값> ¦ <transition-delay값>
```

```
<style>
  .box {
    ......
    transition: 2s ease-in;
  }
  .box:hover {
    width: 200px;
    height: 200px;
    background-color: #f50;
    transform: rotate(270deg);
  }
</style>
```

transition-property: 기본값 all
transition-duration: 2s
transition-timing-function: ease-in
transition-delay: 기본값 0



애니메이션 알아보기

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

애니메이션 관련 속성

종류	설명
@keyframes	애니메이션이 바뀌는 지점을 지정합니다.
animation-delay	애니메이션의 시작 시간을 지정합니다.
animation-direction	애니메이션을 종료한 뒤 처음부터 시작할지, 역방향으로 진행할지 지정합니다.
animation-duration	애니메이션의 실행 시간을 지정합니다.
animation-iteration-count	애니메이션의 반복 횟수를 지정합니다.
animation-name	@keyframes로 설정해 놓은 중간 상태를 지정합니다.
animation-timing-function	키프레임의 전환 형태를 지정합니다.
animation	animation 속성을 한꺼번에 묶어서 지정합니다.

애니메이션 알아보기

@keyframes 속성

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

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

- @keyframes의 선택자에서 속성값이 바뀌는 지점을 가리킴
- 시작 위치는 0%, 끝 위치 100%로 놓고 위치 지정
- 시작과 끝 위치만 사용한다면 from, to 키워드 사용 가능

animation-name 속성

- 어떤 애니메이션을 사용할지 구별
- @keyframes 속성에서 만든 애니메이션 '이름'을 사용

```
기본형 animation-name: <키프레임 이름> | none
```

animation-duration 속성

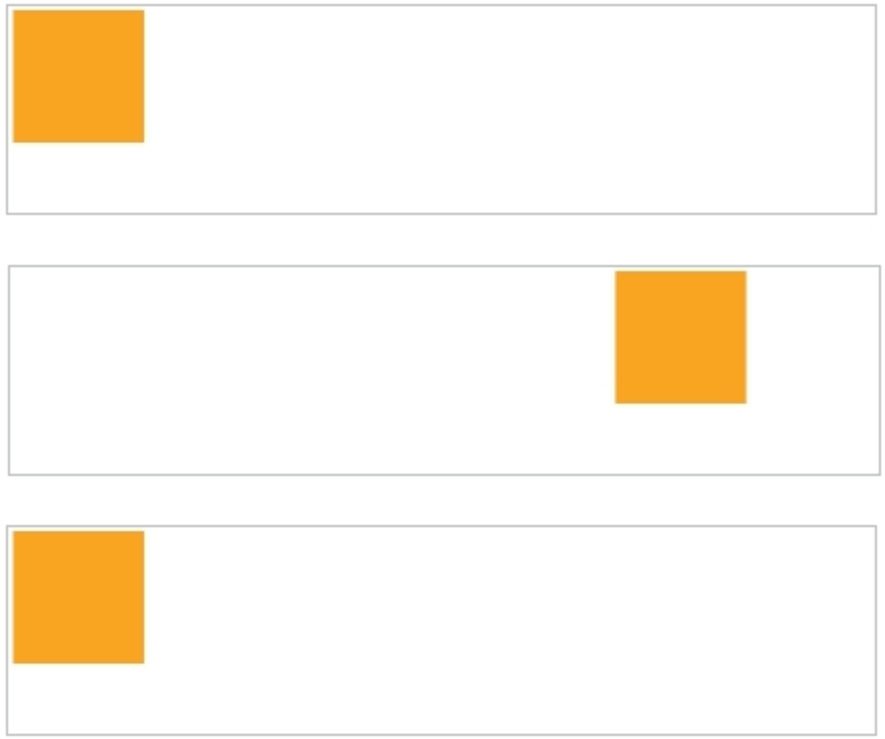
- 애니메이션 실행 시간 설정. 기본값 0
- 사용 가능한 값은 초(s)나 밀리초(ms)

```
기본형 animation-duration: <시간>
```

애니메이션 알아보기

```
<style>
  .box {
    width: 100px;
    height: 100px;
    background-color: #fa0;
    animation-name: slideRight;
    animation-duration: 2s;
  }

  @keyframes slideRight {
    from { transform: translateX(0); }
    to { transform: translateX(500px); }
  }
</style>
```



애니메이션 알아보기

animation-direction 속성

애니메이션이 끝난 후 원래 위치로 돌아가거나 반대 방향으로 실행하도록 지정

기본형 `animation-direction: normal ¦ reverse ¦ alternate ¦ alternate-reverse`

종류	설명
normal	애니메이션을 from에서 to로 진행합니다. 기본값입니다.
reverse	애니메이션을 to에서 from으로, 원래 방향과는 반대로 진행합니다.
alternate	홀수 번째는 normal로, 짝수 번째는 reverse로 진행합니다.
alternate-reverse	홀수 번째는 reverse로, 짝수 번째는 normal로 진행합니다.

```
<style>
  .box {
    ......
      animation-name: slideRight;
      animation-duration: 2s;
      animation-iteration-count: infinite;
      animation-direction: alternate;
  }
......
</style>
```

animation-iteration-count 속성

애니메이션 반복 횟수 지정하기

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

종류	설명
숫자	애니메이션의 반복 횟수를 정합니다.
infinite	애니메이션을 무한 반복합니다.

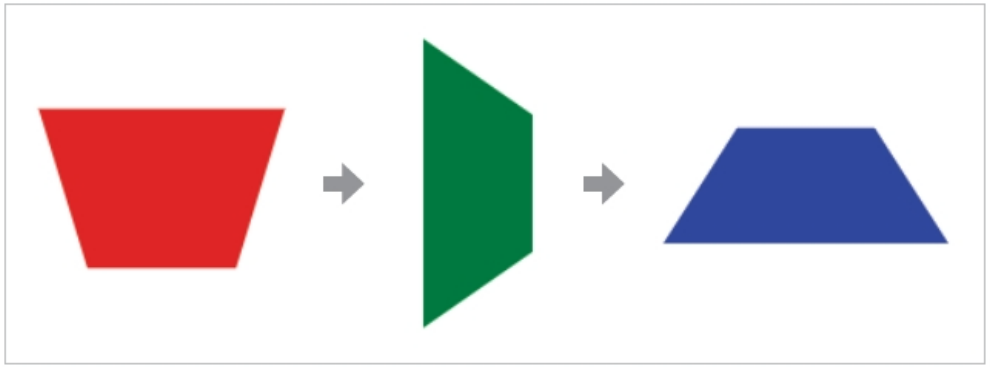
애니메이션 알아보기

animation-timing-function 속성

애니메이션 속도 곡선 지정

animation 속성

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



2개의 애니메이션 한꺼번에 지정하기

```
<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 */
    0% { transform: perspective(120px) rotateX(0deg) rotateY(0deg); }
    50% { transform: perspective(120px) rotateX(-180deg) rotateY(0deg); }
    100% { transform: perspective(120px) rotateX(-180deg) rotateY(-180deg); }
  }
  @keyframes background {
    0% { background-color: red; } /* 시작 배경색은 빨강 */
    50% { background-color: green; } /* 중간(50%) 배경색은 초록 */
    100% { background-color: blue; } /* 마지막(100%) 배경색은 파랑 */
  }
</style>
```