13. CSS와 애니메이션

https://www.youtube.com/watch?v=SIkPFDTjjb4&list=PLG7te9eYUi7sxAaXX74J6lqiV8vtStuLr&index=19

Do it! HTML5+CSS3 웹코딩의 정석 개정 2판

13-1. 변형

일반적으로 **특정 요소의 크기나 형태가 변하는 것**을 ***변형*** 또는 ***트랜스폼 (transform)***이라고 합니다.
웹 요소도 예외가 아닙니다.
웹 문서에서 CSS를 이용하면 사용자의 동작에 반응해 텍스트나 이미지 등을 움직이게 할 수 있습니다.
이런 변형을 이용하면 사용자가 좀 더 흥미롭게 느끼겠죠?

transform과 변형함수

이미지를 회전시키거나 이동하는 등 웹 요소를 변형하려면 transform 속성을 사용해야 하는데 transform: 다음에 변형 함수를 함께 입력해 사용합니다.

예를 들어 .photo라는 클래스 선택자를 가진 웹 요소를 x축으로 50픽셀, y축으로 100픽셀 이동시키려면 웹 요소를 이동시키는 변형 함수 translate을 사용해 다음과 같이 스타일을 지정합니다.

.photo { transform:translate(50px,100px); }

translate 변형 함수 - 요소 이동시키기

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>
```

scale 변형 함수 - 요소 확대/축소하기

scale 함수

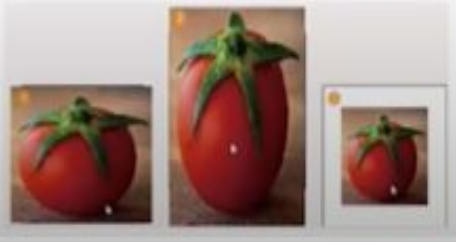
지정한 크기만큼 요소를 확대/축소

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 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>
```



1보다 크면
확대되고
1보다 작으면
축소됩니다.

rotate 변형 함수 - 요소 회전하기

2차원 함수 기본형 transform:rotate(각도)

3차원 함수 기본형 transform:rotate(rx, ry, 각도)
transform: rotate3d(rx, ry, rz, 각도)

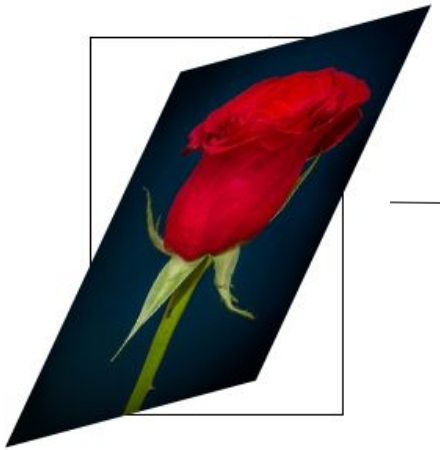
.rotate2:hover { transform: rotate(-40deg); }

.rotate1:hover { transform: rotate(20deg); }

skew 변형 함수 - 요소를 비틀어 왜곡하기

2차원 변형만 가능한 skew 변형 함수는 요소를 지정한 각도만큼 비틀어 왜곡합니다.
양쪽 방향이나 한쪽 방향으로만 비틀 수 있습니다.

기본형 transform:skew(ax, ay)
transform:skew(ax)
transform:skew(ay)



.skewxy:hover {transform: skew(-25deg,-15deg);}

x축으로 -25도, y축으로 -15도 비틀기

13-2. 변형과 관련된 속성들

앞에서 배운 변형 함수들에서 2차원 변형에 원근감을 추가하면 3차원 변형을 만들 수 있는데 이 때 단순한 Z축만 추가한다고 해서 원근감이 생기지는 않습니다.

다시 말해 변형할 때 기준이 되는 지점을 바꾸거나 요소의 원근감을 표현하기 위한 다른 속성도 필요합니다.

이번장에서는 변형 관련 속성들을 알아보겠습니다.

transform-origin 속성 - 변형 기준점 설정하기

기본형 transform-origin: <x축> <y축> <z축>

transform-origin 속성을 이용하면 축이 아닌 특정 지점을 변형의 기준으로 설정할 수 있습니다.

<속성 값>

<x축> : 원점 기준의 x 좌표값으로 길이값이나 백분율, left, center, right 중에서 사용 가능
<y축> : 원점 기준의 y 좌표값으로 길이값이나 백분율, top, center, bottom 중에서 사용 가능
<z축> : 원점 기준의 z 좌표값으로 길이 값만 사용할 수 있습니다.

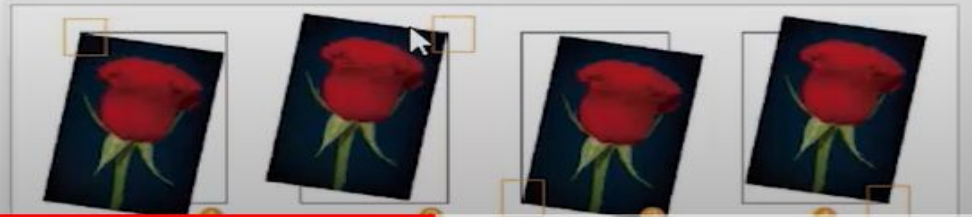
다음 예제는 네 개의 이미지에 transform:rotateZ(10deg)라는 변형을 똑같이 추가했습니다.

각 이미지마다 transform-origin 값을 다르게 주어 결과값을 비교해보겠습니다.

transform-origin 예제

```
<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>
```



13-3. 트랜지션

트랜지션이란 ?

웹 요소의 배경 색이 바뀌거나 도형의 테두리가 원형으로 바뀌는 것처럼 **스타일 속성이 바뀌는 것**을 의미합니다.

시간에 따라 웹 요소의 스타일 속성이 조금씩 바뀌는 것을 트랜지션이라고 합니다.

transition 속성

<속성 값>

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 : property, duration, timing-function, delay 속성을 한꺼번에 설정합니다.

transition-property 속성 - 트랜지션을 적용할 속성 지정

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

트랜지션을 어느 속성에 적용할것인지 선택하는 것입니다. all이 기본값이며 none은 트랜지션동안 아무 속성도 바뀌지않습니다. <속성 이름>은 트랜지션 효과를 적용할 속성 이름을 지정합니다.

예를들면,

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

기본형 transition-duration : <시간>

트랜지션 진행시간을 지정해야 그 시간 동안 속성이 자연스럽게 바뀌는 애니메이션 효과를 만들 수 있습니다.

기본단위는 ***초(seconds)*** 입니다.

트랜지션 대상 속성이 여러개라면 트랜지션 진행시간도 쉼표로 구분해 순서대로 ***여러개를 지정*** 할 수 있습니다.

transition-delay 속성 - 지연시간 설정하기

기본형 transition-delay : <시간>

transition-delay 속성은 트랜지션이 **언제부터 시작**할 것인지를 설정합니다.

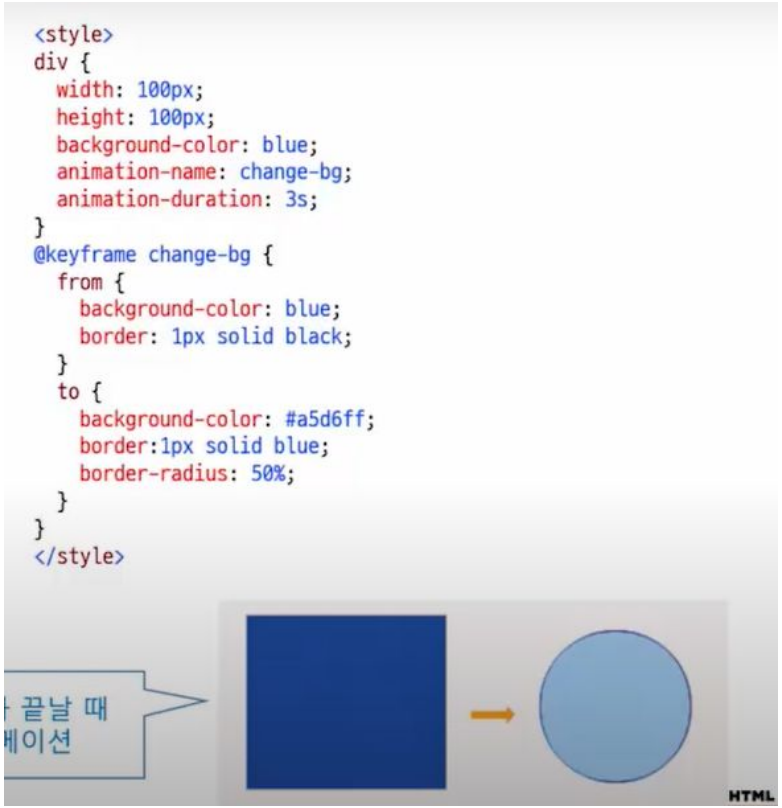
13-4. 애니메이션

CSS 애니메이션은 어떤 면에서는 트랜지션과 비슷하고 어떤면에서는 다릅니다.

CSS 애니메이션은 시작해 끝내는 동안 원하는 곳 어디서든 스타일을 바꾸며 애니메이션을 정의 할 수 있는점이 트랜지션과 다릅니다.

이 때 애니메이션 **중간에 스타일이 바뀌는 지점**을 *키프레임(keyframe)*이라고 부릅니다.

@keyframes 속성 - 애니메이션 지점 설정하기




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

애니메이션에서 중간 지점을 추가하려면 시작 위치를 0%, 끝 위치를 100%로 놓고 50% 위치에 키프레임을 하나 더 추가하면 됩니다.

시작과 끝 위치만 사용하겠다면 0%와 100% 값 대신 from과 to라는 키워드를 사용해도 됩니다.

예시를 보면 시작은 파란색 사각형이었다가 끝날 때 하늘색으로 바뀌는 애니메이션입니다.

애니메이션이 끝나면 처음의 파란색 사각형으로 돌아갑니다.

animation-duration 속성 - 애니메이션 실행시간 설정

기본형 animation-duration: <시간>

animation-duration 속성은 애니메이션을 얼마동안 재생할것인지 설정합니다.

animation-duration 속성을 정하지 않으면 애니메이션은 일어나지 않습니다.

예를 들어 애니메이션 진행 시간을 3초로 지정하려면 아래와 같이 작성합니다.

```
div {
…..
animation-duration: 3s;
}
```

animation-direction 속성 - 애니메이션 방향 지정

기본형 animation-direction: normal | alternate

기본적으로 애니메이션이 한번 실행되면 원래 위치로 되돌아가는데 animation-direction 속성을 이용하면 원래 위치로 되돌아가거나 반대방향으로 애니메이션을 한번 더 실행 할 수 있습니다.

<속성 값>

normal : 애니메이션을 끝까지 실행하면 원래 위치로 돌아갑니다. 기본값입니다.

alternate : 애니메이션을 끝까지 실행하면 왔던 방향으로 되돌아가면서 애니메이션을 실행합니다.

animation-iteration-count 속성 - 반복 횟수 지정

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

기본적으로 애니메이션은 한번만 실행하고 끝나지만 animation-iteration-count 속성을 사용해 반복 횟수를 지정할 수 있습니다.

animation 속성 - 애니메이션 관련 속성 한꺼번에 표기

기본형 animation: <animation-name> | <animation-duration> | <animation-delay> | <animation-iteration-count> | <animation-direction>

예시)

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

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