

# CSS 3 심화 과정



@ IT Korea 강사 이정훈

# 목차

1. transform 속성

2. animation 속성

3. 미디어쿼리

4. 그리드 레이아웃

5. 실습

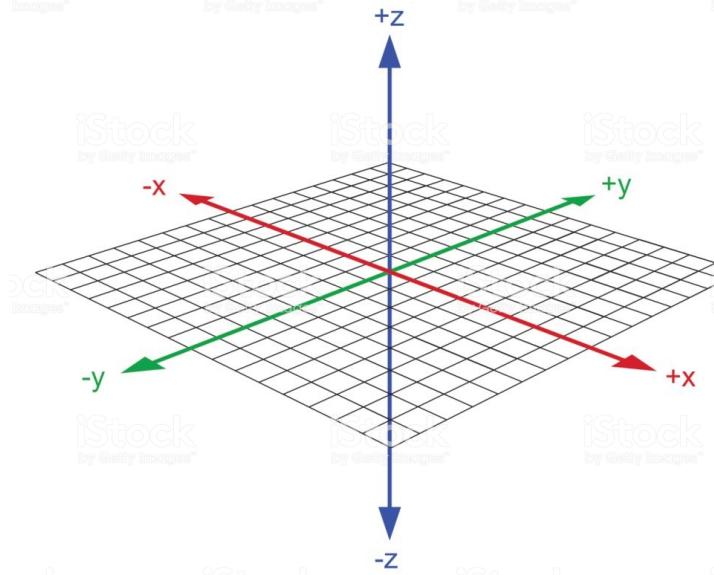
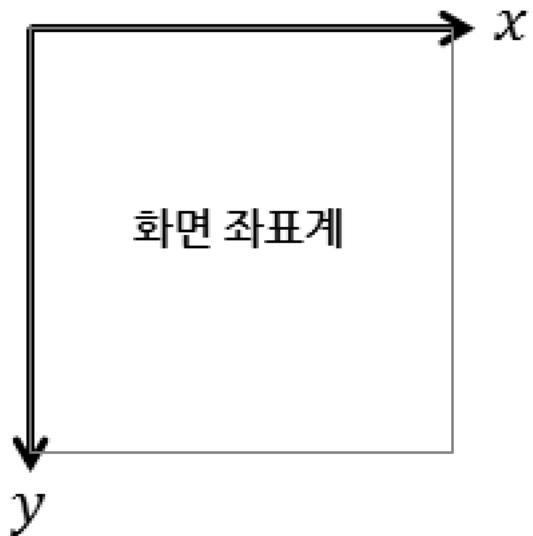
2개월차에 오신 여러분 환영합니다. 2개월차에서는 CSS3의 고급적인 기능을 다룰것입니다.

# 1. transform



**웹 요소의 위치를 옮기거나 크기를 조절하고 회전, 왜곡시키는것을 변형 (transform) 이라고 합니다. 수직 수평 이동뿐만이 아니고, 3차원 원근감도 표현 가능합니다.**

## 2차원, 3차원 좌표계



**2차원과 3차원의 좌표계입니다.  
지금부터 여러분의 공감각을 테스트할 시간입니다.**

## **transform 속성**

### **transform : translate( tx, ty)**

x축으로 tx, y축으로 ty 만큼 이동합니다. ty 값이 주어지지 않으면 0과 같다고 간주합니다.

### **transform : translate3d (tx , ty , tz)**

x축으로 tx, y축으로 ty, z축으로 tz만큼 이동합니다.

### **transform : translateX( tx )**    x축 방향으로 tx만큼 이동합니다.

### **transform : translateY( ty )**    y축 방향으로 ty만큼 이동합니다.

### **transform : translateZ( tz )**    z축 방향으로 tz만큼 이동합니다.

# translate - 요소 이동

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Translate3D</title>
<style>
img{border:1px solid black;}
div#original {
  margin-bottom:50px;
}
div.div1{
  float:left;
  margin:10px;
  width:152px;
  height:192px;
  background-color:rgba(0,0,255,0.3);
  perspective:200px;
  -webkit-perspective:200px;
  -moz-perspective:200px;
  transform-style:preserve-3d;
  -webkit-transform-style:preserve-3d;
  -moz-transform-style:preserve-3d;
}
.translatex img{
  transform:translateX(-10px);
  -webkit-transform:translateX(-10px);
  -moz-transform:translateX(-10px);
}
.translatey img{
  transform:translateY(20px);
  -webkit-transform:translateY(20px);
  -moz-transform:translateY(20px);
}
.translatez img{
  transform:translateZ(40px);
  -webkit-transform:translateZ(40px);
  -moz-transform:translateZ(40px);
}
.translatexyz img{
  transform:translate3d(10px, 20px, 10px);
  -webkit-transform:translate3d(10px, 20px, 10px);
  -moz-transform:translate3d(10px, 20px, 10px);
}
</style>
</head>
```

```
<body>
<div id="wrapper">
  <div id="original">
    <h4>원본 이미지</h4>
    
  </div>
  <div class="div1">
    <div class="translatex"></div>
  </div>
  <div class="div1">
    <div class="translatey"></div>
  </div>
  <div class="div1">
    <div class="translatez"></div>
  </div>
  <div class="div1">
    <div class="translatexyz"></div>
  </div>
</div>
</body>
</html>
```

원본 이미지



## **scale - 요소 확대 / 축소**

### **transform : scale( sx , sy )**

x축으로 sx, y축으로 sy 만큼 확대합니다. sy 값이 주어지지 않으면 sx 값과 같다고 간주합니다. 즉, scale(2)는 scale(2,2)와 같습니다.

### **transform : scale3d (sx , sy , sz)**

x축으로 sx, y축으로 sy, z축으로 sz만큼 확대합니다.

### **transform : scaleX( sx )**    x축 방향으로 sx만큼 확대합니다.

### **transform : scaleY( sy )**    y축 방향으로 sy만큼 확대합니다.

### **transform : scaleZ( sz )**    z축 방향으로 sz만큼 확대합니다.

```

<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Scale3D</title>
<style>
img{border:1px solid black;}
div#original {
  margin-bottom:50px;
}
div.div1 {
  float:left;
  margin:10px;
  width:152px;
  height:192px;
  background-color:rgba(0,0,255,0.3);
  perspective:200px;
  -webkit-perspective:200px;
  -moz-perspective:200px;
  transform-style:preserve-3d;
  -webkit-transform-style:preserve-3d;
  -moz-transform-style:preserve-3d;
}
.scaler img{
  transform:scaleX(1.2);
  -webkit-transform:scaleX(1.2);
  -moz-transform:scaleX(1.2);
}
.scalesy img{
  transform:scaleY(1.5);
  -webkit-transform:scaleY(1.5);
  -moz-transform:scaleY(1.5);
}
.scalesz img{
  transform:scaleZ(0.7);
  -webkit-transform:scaleZ(0.7);
  -moz-transform:scaleZ(0.7);
}
.scalexyz img{
  transform:scale3d(1.2, 1.5, 0.7);
  -webkit-transform:scale3d(1.2, 1.5, 0.7);
  -moz-transform:scale3d(1.2, 1.5, 0.7);
}
</style>
</head>

```

```

<body>
<div id="wrapper">
  <div id="original">
    <h4>원본 이미지</h4>
    
  </div>
  <div class="div1">
    <div class="scalesx"></div>
  </div>
  <div class="div1">
    <div class="scalesy"></div>
  </div>
  <div class="div1">
    <div class="scalesz"></div>
  </div>
  <div class="div1">
    <div class="scalexyz"></div>
  </div>
</div>
</body>
</html>

```

원본 이미지



**scale은 확대 축소를 위한 속성입니다.**

## **rotate - 요소 회전하기**

**transform : rotate( sx , sy , 각도)**

x,y 값이 주어졌기 때문에, 주어진 각도만큼 2차원 회전합니다.

**transform : rotate3d (sx , sy , sz, 각도)**

방향 벡터(rx, ry, rz)와 같은 직선을 기준으로 주어진 각도만큼 회전합니다.

**transform : rotateX( 각도 )** x축 방향으로 주어진 각도 만큼 회전합니다.

**transform : rotateY( 각도 )** y축 방향으로 주어진 각도 만큼 회전합니다.

**transform : rotateZ( 각도 )** z축 방향으로 주어진 각도 만큼 회전합니다.

```

<style>
  img{border:1px solid black;}

  #wrapper div{
    float:left;
    margin:10px;
  }
  .origin div{
    width:152px;
    height:192px;
    background-color:rgba(0,0,255,0.3);
    perspective:200px;
    -webkit-perspective:200px;
    -moz-perspective:200px;
    transform-style:preserve-3d;
    -webkit-transform-style:preserve-3d;
    -moz-transform-style:preserve-3d;
  }
  .rotatex img{
    transform:rotateX(45deg);
    -webkit-transform:rotateX(45deg);
    -moz-transform:rotateX(45deg);
  }
  .rotatey img{
    transform:rotateY(45deg);
    -webkit-transform:rotateY(45deg);
    -moz-transform:rotateY(45deg);
  }
  .rotatez img{
    transform:rotateZ(45deg);
    -webkit-transform:rotateZ(45deg);
    -moz-transform:rotateZ(45deg);
  }
  .rotatexyz img{
    transform:rotate3d(2.5, 1.2, -1.5, 45deg);
    -webkit-transform:rotate3d(2.5, 1.2, -1.5, 45deg);
    -moz-transform:rotate3d(2.5, 1.2, -1.5, 45deg);
  }
</style>
</head>

```

```

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

```



**rotate는 회전을 위한 속성입니다.**

## Skew - 요소 비틀어 왜곡하기

### **transform : skew( 각도, 각도 )**

첫 번째 각도는 x축에서의 왜곡 각도를 나타내고, 두 번째 각도는 y축에서의 왜곡 각도입니다. 두 번째 값이 주어지지 않으면, y축에 대한 왜곡 각도를 0도로 간주하여 y축으로 왜곡이 생기지 않습니다.

**transform : skewX( 각도 )**    x축에서만 주어진 각도만큼 왜곡합니다.

**transform : skewY( 각도 )**    y축에서만 주어진 각도만큼 왜곡합니다.

```

<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Skew</title>
<style>
img{
    border:1px solid black;
}
div#original {
    margin-bottom:50px;
}
div.div1{
    float:left;
    margin:30px;
    width:152px;
    height:192px;
    background-color:rgba(0,0,255,0.3);
    perspective:200px;
    -webkit-perspective:200px;
    -moz-perspective:200px;
    transform-style:preserve-3d;
    -webkit-transform-style:preserve-3d;
    -moz-transform-style:preserve-3d;
}
.skewx img{
    transform:skewX(30deg);
    -webkit-transform:skewX(30deg);
    -moz-transform:skewX(30deg);
}
.skewy img{
    transform:skewY(15deg);
    -webkit-transform:skewY(15deg);
    -moz-transform:skewY(15deg);
}
.skew img{
    transform:skew(-25deg, -15deg);
    -webkit-transform:skew(-25deg, -15deg);
    -moz-transform:skew(-25deg, -15deg);
}
</style>
</head>

```

```

<body>
<div id="wrapper">
    <div id="original">
        <h4>원본 이미지</h4>
        
    </div>
    <div class="div1">
        <div class="skewx"></div>
    </div>
    <div class="div1">
        <div class="skewy"></div>
    </div>
    <div class="div1">
        <div class="skew"></div>
    </div>
</div>

```



# transform-origin

```
<style>
img{
    transform-origin: left top;
    transform-origin: right bottom;
}
</style>
```

**x축** : 길이값, 백분율, left, center, right

**y축** : 길이값, 백분율, top, center, bottom

**z축** : 길이 값만 사용가능

**transfom-origin**는 지정한 요소의 변형 원점을 설정한것으로  
transform 속성과 함께 사용합니다.

```

<style>
img{border:1px solid black;}
#wrapper div{
    float:left;
    margin:10px;
}
.ex div{
    width:152px;
    height:180px;
    background-color:rgba(0,0,255,0.3);
    perspective:200px;
    -webkit-perspective:200px;
    -moz-perspective:200px;
    transform-style:preserve-3d;
    -webkit-transform-style:preserve-3d;
    -moz-transform-style:preserve-3d;
}
.ex img{
    transform:rotateZ(10deg);
    -webkit-transform:rotateZ(10deg);
    -moz-transform:rotateZ(10deg);
}
.ltop img{
    transform-origin:left top;
    -webkit-transform-origin:left top;
    -moz-transform-origin:left top;
}
.rtop img{
    transform-origin:right top;
    -webkit-transform-origin:right top;
    -moz-transform-origin:right top;
}
.lbt img{
    transform-origin:left bottom;
    -webkit-transform-origin:left bottom;
    -moz-transform-origin:left bottom;
}
.rbt img{
    transform-origin:right bottom;
    -webkit-transform-origin:right bottom;
    -moz-transform-origin:right bottom;
}
</style>

<body>


</div>



</div>



</div>



</div>


</body>


```



# perspective

```
.origin div{  
width:152px;  
height:192px;  
background-color:rgba(0,0,255,0.3);  
perspective:200px;  
-webkit-perspective:200px;  
-moz-perspective:200px;  
transform-style:preserve-3d;  
-webkit-transform-style:preserve-3d;  
-moz-transform-style:preserve-3d;  
}  
.rotateX img{  
transform:rotateX(45deg);  
-webkit-transform:rotateX(45deg);  
-moz-transform:rotat  
}  
References:  
https://www.w3.org/...
```

**perspective**는 원근감에 대한 속성입니다.

원래 위치에서 사용자 방향 또는 반대 방향으로 잡아당기거나 밀어서 원근감을 표시합니다.

# **perspective-origin**

**perspective-origin : x축값 | y축값;**

**x축** : 3D요소가 x축에서 어디에 위치하는지 지정합니다. 사용가능 값은 길이값, 백분율, left, right, center, top, bottom이고, 기본값은 50%입니다.

**y축** : 3D요소가 y축에서 어디에 위치하는지 지정합니다. 사용가능 값은 길이값, 백분율, top, center, bottom이고, 기본값은 50%입니다.

이 속성은 3D요소의 bottom 부분의 위치를 지정하는데,  
좀 더 높은곳에서 원근을 지정하는 느낌을 만들 수 있습니다.  
이 속성을 사용하려면, perspective 속성과 함께 사용해야합니다.

```

<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Perspective</title>
|
<style>
  div#original {
    margin-bottom:50px;
  }
  div#noper{
    position:absolute;
    left:30px;
    top:250px;
    width:152px;
    height:180px;
  }
  div#pers{
    position:absolute;
    left:300px;
    top:250px;
    width:152px;
    height:180px;
    background-color:rgba(0,0,255,0.3);
    perspective:200px;
    -webkit-perspective:200px;
  }
  img{
    border:1px solid black;
  }
  .ex{
    transform:rotateX(50deg);
    -webkit-transform:rotateX(50deg);
  }
</style>
</head>

```

```

<body>
  <div id="original">
    <h4>원본 이미지</h4>
    
  </div>
  <div id="noper">
    <div class="ex"></div>
  </div>
  <div id="pers">
    <div class="ex"></div>
  </div>
</body>
</html>

```

원본 이미지



# transform-style

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Transform style</title>
<style>
img {border:1px solid black;}
.flat{
    width:152px;
    height:192px;
    background-color:rgba(0,0,255,0.3);
    transform:rotateY(50deg);
    -webkit-transform:rotateY(50deg);
    -moz-transform:rotateY(50deg);
    transform-style:flat;
    -webkit-transform-style:flat;
}
.ex img{
    transform-origin:right top;
    -webkit-transform-origin:right top;
    transform:rotateX(25deg);
    -webkit-transform:rotateX(25deg);
}
</style>
</head>

<body>
<div class="flat">
    <div class="ex"></div>
</div>
</body>
</html>
```

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Transform style</title>
<style>
img {border:1px solid black;}
.preserve{
    width:152px;
    height:180px;
    background-color:rgba(0,0,255,0.3);
    transform:rotateY(50deg);
    -webkit-transform:rotateY(50deg);
    -moz-transform:rotateY(50deg);
    transform-style:preserve-3d;
    -webkit-transform-style:preserve-3d;
}
.ex img{
    transform-origin:right top;
    -webkit-transform-origin:right top;
    transform:rotateX(25deg);
    -webkit-transform:rotateX(25deg);
}
</style>
</head>

<body>
<div class="preserve">
    <div class="ex"></div>
</div>
</body>
</html>
```

# 시간에 따른 변화 트랜지션(애니메이션 효과)

## transition-property

```
<style type="text/css">
#ex div{
    transition-property: all;
    transition-property: none;
    transition-property: background-color;
    transition-property: background-color, border;
}
</style>
```

트랜지션이란 한 스타일에서 다른스타일로 바뀌는것을 말합니다.

Flash 또는Javascript 을 사용하지 않고, CSS로만 애니메이션효과는 내는 첫 단계는 어떤 속성에 적용시킬지 입니다. 그리고 property가 그것을 결정합니다.

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Transition timing function</title>
|
<style>
#ex div{
    width:100px;
    height:100px;
    background-color:red;
    border-radius:0px;
    transition-property:background-color, border-radius;
    -webkit-transition-property:background-color, border-radius;
    -moz-transition-property:background-color, border-radius;
}
#ex:hover div{
    background-color:blue;
    border-radius:50px;
}
</style>
</head>

<body>
<p> 도형 위로 마우스 포인터를 올려보세요</p>
<div id="ex">
    <div></div>
</div>
</body>
</html>
```

# transition-duration

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Transition</title>

<style>
#ex div{
    width:100px;
    height:100px;
    background-color:red;
    border-radius:0px;
    transition-property:background-color, border-radius;
    -webkit-transition-property:background-color, border-radius;
    -moz-transition-property:background-color, border-radius;
    transition-duration:1s;
    -webkit-transition-duration:1s;
    -moz-transition-duration:1s;
}
#ex:hover div{
    background-color:blue;
    border-radius:50px;
}
</style>
</head>

<body>
<p> 도형 위로 마우스 포인터를 올려보세요</p>
<div id="ex">
    <div></div>
</div>
</body>
</html>
```

이 속성은 진행시간을 지정합니다. 초 또는 밀리초로 지정합니다. 0보다 작을수 없습니다.

# transition-timing-function

**linear** : 시작부터 끝까지 똑같은 속도로 트랜지션합니다.

**ease** : 처음에는 천천히 시작해서 점점 빨라지다 마지막에 천천히 끝납니다. 기본값입니다.

**ease-in** : 트랜지션 시작을 느리게합니다.

**ease-out** : 트랜지션을 느리게 끝냅니다.

**ease-in-out** : 느리게 시작해서 느리게 끝납니다.

**cubic-bezier(n,n,n,n)** : 직접 함수를 정의해서 사용. n은 0~1입니다.

트랜지션 효과의 시작과 중간, 그리고 끝에서 속도 곡선을 선택하는 속성입니다.

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Transition timing function</title>
<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{
    -webkit-transition:3s ease;
    -moz-transition:3s ease;
    -o-transition:3s ease;
    -ms-transition:3s ease;
}
#ex .linear{
    -webkit-transition:3s linear;
    -moz-transition:3s linear;
    -o-transition:3s linear;
    -ms-transition:3s linear;
}
#ex .ease-in{
    -webkit-transition:3s ease-in;
    -moz-transition:3s ease-in;
    -o-transition:3s ease-in;
    -ms-transition:3s ease-in;
}
#ex .ease-out{
    -webkit-transition:3s ease-out;
    -moz-transition:3s ease-out;
    -o-transition:3s ease-out;
    -ms-transition:3s ease-out;
}
#ex .ease-in-out{
    -webkit-transition:3s ease-in-out;
    -moz-transition:3s ease-in-out;
    -o-transition:3s ease-in-out;
    -ms-transition:3s ease-in-out;
}
</style>
</head>

<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>
```

# transition-delay

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Transition</title>

<style>
#ex div{
    width:100px;
    height:100px;
    background-color:red;
    border-radius:0px;
    transition-property:background-color, border-radius;
    -webkit-transition-property:background-color, border-radius;
    -moz-transition-property:background-color, border-radius;
    transition-duration:1s;
    -webkit-transition-duration:1s;
    -moz-transition-duration:1s;
    transition-delay:2s;
    -webkit-transition-delay:2s;
    -moz-transition-delay:2s;
}
#ex:hover div{
    background-color:blue;
    border-radius:50px;
}
</style>
</head>

<body>
<p> 도형 위로 마우스 포인터를 올려보세요</p>
<div id="ex">
    <div></div>
</div>
</body>
</html>
```

delay 속성은 자연시간을  
설정하는 속성입니다.  
즉, 지정하는 시간만큼 기다렸다가  
트랜지션이 시작됩니다.  
시간값은 초나 밀리초를 사용합니다.  
기본값은 0초 입니다.

# transition 속성

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Transition</title>

<style>
#ex div{
    width:100px;
    height:100px;
    border-radius:0px;
    border:2px solid black;
    background:url(f1.png) no-repeat center center padding-box;
    transition:all 1s ease-in 0.3s;
    -moz-transition:all 1s ease-in 0.3s;
    -webkit-transition:all 1s ease-in 0.3s;
}
#ex:hover div{
    border-radius:50px;
    border:2px solid blue;
    background:url(f2.png) no-repeat center center padding-box;
}
</style>
</head>

<body>
<p> 도형 위로 마우스 포인터를 올려보세요</p>
<div id="ex">
    <div></div>
</div>
</body>
</html>
```

다른 많은 경우처럼 트랜지션 속성으로 여러가지 값을 함께 사용할수있습니다.

## 2. animation

animation속성은 transition속성과 대부분이 비슷합니다.

하지만 큰 차이점이 있으니.... 그것은 바로!

### @keyframe

요 녀석의 차이인데요, 이 녀석도 속성중에 한가지입니다.

이 속성은 애니메이션의 시작부터 끝날때까지 어느 지점이든

이 속성을 이용해 애니메이션을 정의할 수 있습니다.

# animation-duration / animation-name

```
<style>

    @keyframes myani{
        0% { left: 10px; }
        100% { left: 500px; }
    }

    @keyframes myani{
        from { left:10px; }
        to { left:500px; }
    }

</style>
```

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Animation</title>
<style>
    #myball {
        position: relative;
        width: 100px;
        height: 100px;
        border-radius: 50px;
        border: 2px solid black;
        background: url(f1.png) no-repeat center center padding-box;
        animation-duration: 3s;
        animation-name: myani;
        -webkit-animation-duration: 3s;
        -webkit-animation-name: myani;
        -moz-animation-duration: 3s;
        -moz-animation-name: myani;
    }
    @keyframes myani {
        0%{
            left: 10px;
        }
        100% {
            left:500px;
        }
    }
    @-webkit-keyframes myani {
        0% {
            left: 10px;
        }
        100% {
            left:500px;
        }
    }
    @-moz-keyframes myani {
        0% {
            left: 10px;
        }
        100% {
            left:500px;
        }
    }
</style>
</head>
```

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

애니메이션을 만들때 `@keyframes`속성을 이용해 여러개의 애니메이션을 정의하는데,  
특정요소에 어떤 애니메이션을 적용할것인지 정의하는 속성이 `name` 속성입니다.

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Animation</title>
<style>
#myball {
    position: relative;
    width: 100px;
    height: 100px;
    border-radius: 50px;
    border: 2px solid black;
    background: url(f1.png) no-repeat center center padding-box;
    animation-duration: 3s;
    animation-name: myani;
    -webkit-animation-duration: 3s;
    -webkit-animation-name: myani;
    -moz-animation-duration: 3s;
    -moz-animation-name: myani;
}
@-webkit-keyframes myani {
    0% {
        left: 10px;
    }
    40% {
        top: 50px;
        background: url(f2.png) no-repeat center center;
    }
    100% {
        left: 500px;
    }
}
@-moz-keyframes myani {
    0% {
        left: 10px;
    }
    40% {
        top: 50px;
        background: url(f2.png) no-repeat center center;
    }
    100% {
        left: 500px;
    }
}
</style>
</head>
<body>
<div id="myball"></div>
</body>
</html>
```

# animation-iteration-count / animation-direction

```
<html>
<head>
<meta charset="utf-8">
<title>Animation</title>
<style>
    #myball {
        position: relative;
        width: 100px;
        height: 100px;
        border-radius: 50px;
        border: 2px solid black;
        background: url(f1.png) no-repeat center center padding-box;
        animation-duration: 3s;
        animation-name: myani;
        animation-direction: alternate;
        animation-iteration-count: infinite;
        -webkit-animation-duration: 3s;
        -webkit-animation-name: myani;
        -webkit-animation-direction: alternate;
        -webkit-animation-iteration-count: infinite;
        -moz-animation-duration: 3s;
        -moz-animation-name: myani;
        -moz-animation-direction: alternate;
        -moz-animation-iteration-count: infinite;
    }
}
```

**iteration-count는 반복횟수를 나타냅니다.**

**특정 숫자 혹은 무한 반복인 infinite 속성이 있습니다.**

**direction 속성은 말 그대로 방향입니다. 즉 진행방향의 반대방향으로 돌아가게 합니다.**

**normal 은 기본값, alternate는 역실행을 하게합니다.**

# animation 속성

```
<html>
<head>
<meta charset="utf-8">
<title>Animation</title>
<style>
    #myball {
        position: relative;
        width: 100px;
        height: 100px;
        border-radius: 50px;
        border: 2px solid black;
        background: url(f1.png) no-repeat center center padding-box;
        animation: myani 3s infinite alternate;
        -webkit-animation: myani 3s infinite alternate;
        -moz-animation: myani 3s infinite alternate;
    }

```

transition속성과 마찬가지 4가지를 뭉쳐서 작성할수있습니다.

순서대로 name, duration, timing-function, delay, iteration-count, direction

# 실습 시간!!

