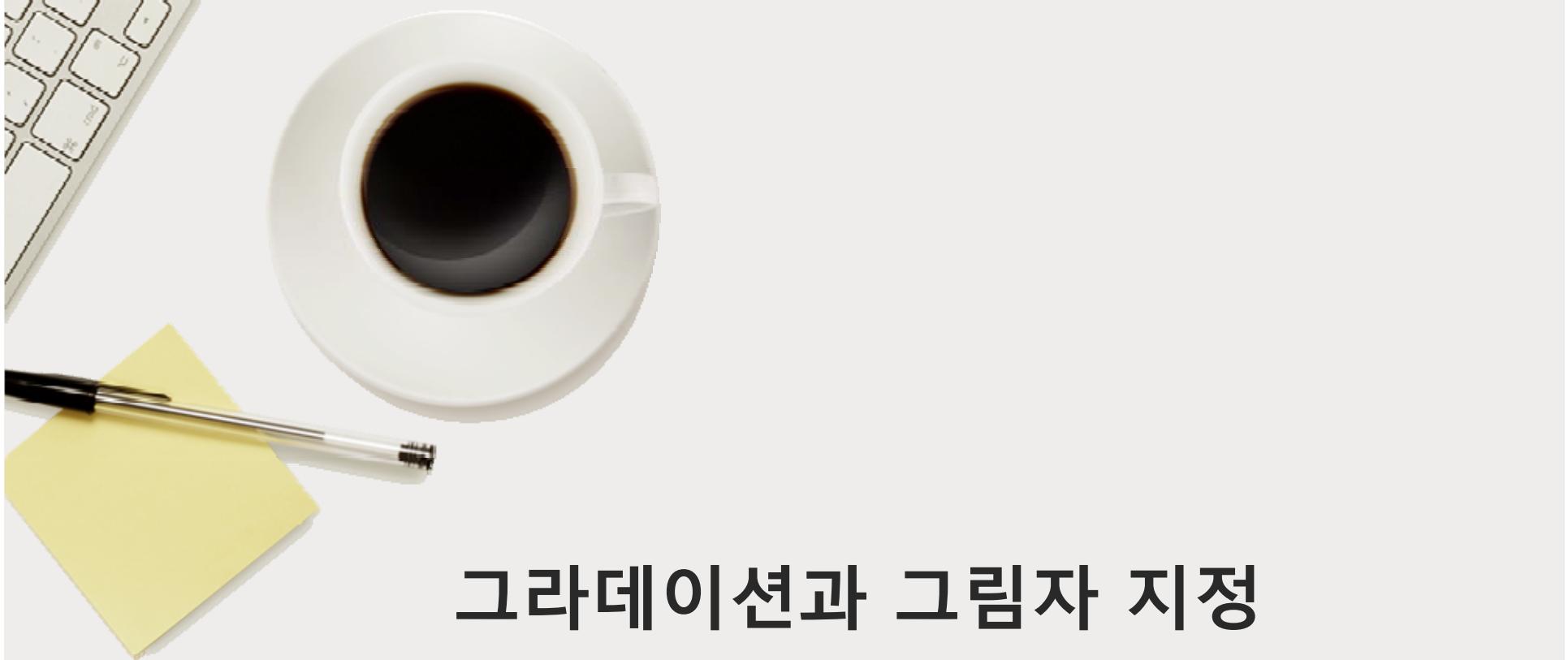




Chapter 8

캔버스(Canvas) – 2



그라데이션과 그림자 지정



■ **createLinearGradient()** 메서드

- 캔버스 내에 생성된 도형에 선형 그라데이션을 지정하기 위한 메서드
 - 그라데이션 생성 영역
 - : 그라데이션 컨텍스트를 생성해야 그라데이션 영역이 생성됨
 - : 그라데이션을 적용하기 위해서는 fillStyle 속성의 값으로 컨텍스트를 지정해야 함
 - : 색상의 지정은 addColorStop() 메서드를 사용
 - 그라데이션이 표시 영역
 - : fill(), fillRect() 메서드와 같이 영역을 지정하고 색을 채우는 메서드를 사용

```
context = createLinearGradient( 시작점x1, 시작점y1, 종료점x2, 종료점y2 )
```

속성 값	설명
시작점x1, 시작점y1	그라디언트가 시작되는 지점의 좌표
종료점x2, 종료점y2	그라디언트가 종료되는 지점의 좌표



그라데이션(gradation) 지정



▣ addColorStop() 메서드

- 그라데이션을 구성하는 색상을 지정하는 메서드
 - 하나 이상의 색상을 지정 가능
 - offset을 사용하여 색상이 나타날 위치를 지정함

```
context.addColorStop( offset, color )
```

속성 값	설명
offset	그라데이션 내에 지정한 색상이 나타나는 위치 지정 (0부터 1사이의 실수 값으로 지정)
color	색상 지정방식을 사용해 그라데이션 내에 표현될 색상을 지정



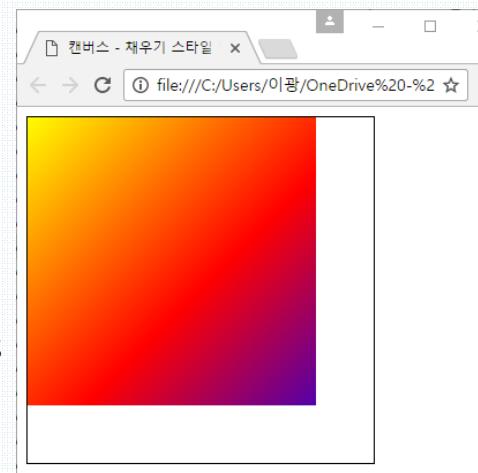
그라데이션(gradation) 지정



■ createLinearGradient() 메서드의 예 - 1

```
<!DOCTYPE html>
<html lang="Kor">
<head>
    <title>캔버스 - 채우기 스타일 연습</title>
    <script>
        function draw() {
            var canvas = document.getElementById("myCanvas");
            var context = canvas.getContext("2d");

            var rectstyle = context.createLinearGradient(0,0,300,300);
            rectstyle.addColorStop(0,"yellow");
            rectstyle.addColorStop(0.5,"red");
            rectstyle.addColorStop(1,"blue");
            context.fillStyle = rectstyle;
            context.fillRect(0,0,250,250);
        }
    </script>
</head>
<body onload="draw();">
    <canvas id="myCanvas" width="300" height="300" style="border: 1px solid #000000">
        캔버스 연습
    </canvas>
</body>
</html>
```

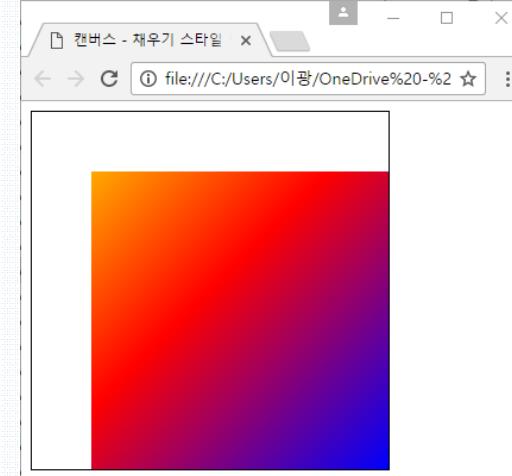


그라데이션(gradation) 지정



■ createLinearGradient() 메서드의 예 -2

```
function draw() {  
    var canvas = document.getElementById("myCanvas");  
    var context = canvas.getContext("2d");  
  
    var rectstyle = context.createLinearGradient(0,0,300,300);  
    rectstyle.addColorStop(0,"yellow");  
    rectstyle.addColorStop(0.5,"red");  
    rectstyle.addColorStop(1,"blue");  
    context.fillStyle = rectstyle;  
    context.fillRect(50,50,300,300);  
}
```



그라데이션(gradation) 지정



■ **createLinearGradient()** 메서드의 예 - 3

```
<!DOCTYPE html>
<html lang="Kor">
<head>
    <title>캔버스 - 채우기 스타일 연습</title>
    <script>
        function draw() {
            var canvas = document.getElementById("myCanvas");
            var context = canvas.getContext("2d");

            context.beginPath(); //두 번째 사각형
            rectstyle = context.createLinearGradient(0,150,300,150);
            rectstyle.addColorStop(0,"yellow");
            rectstyle.addColorStop(0.3,"red");
            rectstyle.addColorStop(1,"blue");
            context.fillStyle = rectstyle;
            context.fillRect(10,10,250,250);

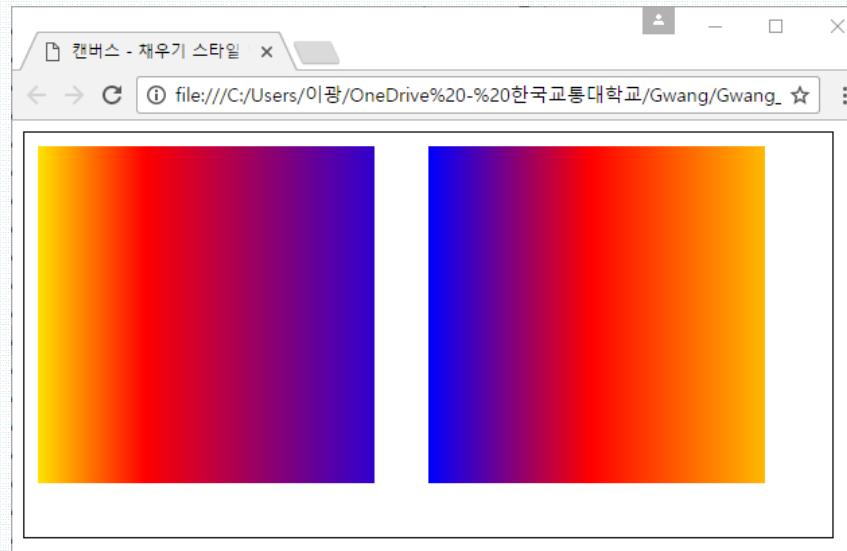
            context.beginPath(); //세 번째 사각형
            rectstyle = context.createLinearGradient(300,150,600,150);
            rectstyle.addColorStop(0,"blue");
            rectstyle.addColorStop(0.4,"red");
            rectstyle.addColorStop(1,"yellow");
            context.fillStyle = rectstyle;
            context.fillRect(300,10,250,250);
        }
    </script>
</head>
```



그라데이션(gradation) 지정



```
<body onload="draw();">
  <canvas id="myCanvas" width="600" height="300" style="border: 1px solid #000000">
    캔버스 연습
  </canvas>
</body>
</html>
```





▣ **createRadialGradient()** 메서드

- **캔버스 내에 생성된 도형에 원형 그라데이션을 지정하기 위한 메서드**
 - 그라데이션 생성 영역과 그라데이션이 표시 영역의 생성은 `createLinearGradient()` 메서드와 동일
 - 색상의 지정은 `addColorStop()` 메서드를 사용

```
context = createRadialGradient( 시작점x1, 시작점y1, 반지름r1, 종료점x2, 종료점y2, 반지름r2 )
```

속성 값	설명
시작점 x1, 시작점 y1	그라디언트가 시작되는 내부 원의 중심의 좌표
반지름 r1	내부 원의 반지름
종료점 x2, 종료점 y2	그라디언트가 시작되는 외부 원의 중심의 좌표
반지름 r2	외부 원의 반지름



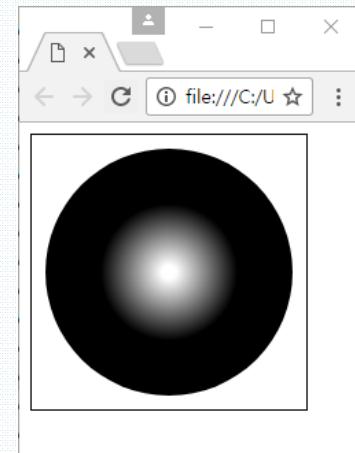
그라데이션(gradation) 지정



■ createRadialGradient() 메서드의 예 - 1

```
<!DOCTYPE html>
<html lang="Kor">
<head>
    <title>캔버스 - 채우기 스타일 연습</title>
    <script>
        function draw() {
            var canvas = document.getElementById("myCanvas");
            var context = canvas.getContext("2d");

            context.beginPath();
            var gradient = context.createRadialGradient( 100, 100, 5, 100, 100, 50 );
            gradient.addColorStop(0, "white");
            gradient.addColorStop(1, "black");
            context.fillStyle = gradient;
            context.arc(100, 100, 90, 0, 360*Math.PI/180, true);
            context.fill();
        }
    </script>
</head>
<body onload="draw();">
    <canvas id="myCanvas" width="200" height="200" style="border: 1px solid #000000">
        캔버스 연습
    </canvas>
</body>
</html>
```

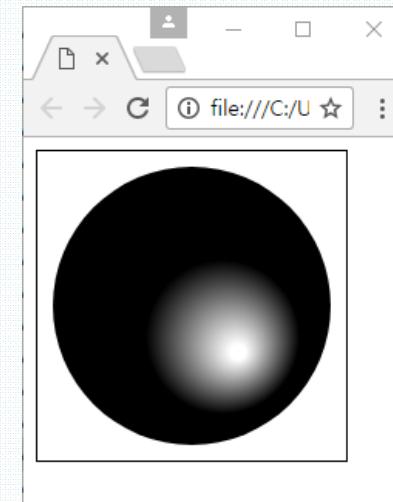




■ createRadialGradient() 메서드의 예 - 2

```
<script>
    function draw() {
        var canvas = document.getElementById("myCanvas");
        var context = canvas.getContext("2d");

        context.beginPath();
        var gradient = context.createRadialGradient( 130, 130, 5, 120, 120, 50 );
        gradient.addColorStop(0, "white");
        gradient.addColorStop(1, "black");
        context.fillStyle = gradient;
        context.arc(100, 100, 90, 0, 360*Math.PI/180, true);
        context.fill();
    }
</script>
```





■ shadowColor 속성

- 도형 그림자의 색상을 지정하기 위한 속성

```
shadowColor = color
```

- 색상은 색상명, 16진수 표현, rgb()를 사용할 수 있음

■ shadowBlur 속성

- 그림자 경계선의 흐림 정보를 지정하는 속성

```
shadowblue = 값
```

- 값이 클수록 그림자의 경계선은 흐려짐 (그림자 경계선의 영역이 넓어짐)
- 기본값은 0





■ shadowOffsetX 속성

- 그림자 대상을 기준으로 한 도형 그림자 위치의 X축 옵셋을 지정

```
shadowOffsetX = offset
```

- offset은 픽셀을 사용하며, 기본값은 0임

■ shadowOffsetY 속성

- 그림자 대상을 기준으로 한 도형 그림자 위치의 Y축 옵셋을 지정

```
shadowOffsetY = offset
```

- offset은 픽셀을 사용하며, 기본값은 0임



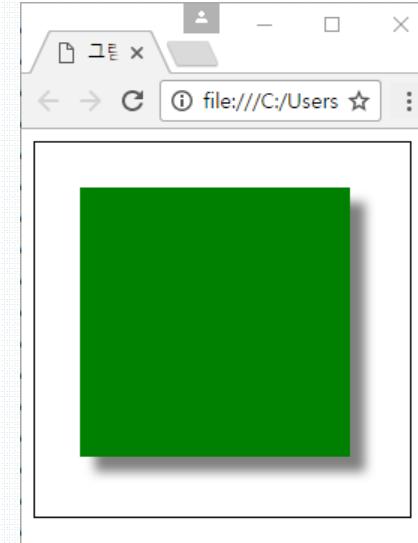
도형 그림자 지정



▣ 도형 그림자 지정의 예

```
<!DOCTYPE html>
<html>
<head>
    <title> 그림자 </title>
    <script type="text/javascript">
        function rect()
        {
            var canvas = document.getElementById('canvas');
            context = canvas.getContext('2d');
            context.shadowOffsetX = 10;
            context.shadowOffsetY = 10;
            context.shadowColor = 'gray';
            context.shadowBlur = '10';
            context.fillStyle="green";
            context.fillRect(30, 30, 180, 180);
        }
    </script>
</head>

<body onload="rect();">
    <canvas id="canvas" width="250" height="250" style="border:1px solid #000000">
        canvas 사용하기
    </canvas>
</body>
</html>
```





이미지와 패턴 삽입



■ drawImage() 메서드

- 캔버스 내에 이미지를 삽입하기 위한 메서드로 세 가지의 생성자를 제공하고 있음
 - `drawImage(image, dx, dy)`
 - `drawImage(image, dx, dy, dw, dh)`
 - `drawImage(image, sx, sy, sw, sh, dx, dy, dw, dh)`

`drawImage(image, dx, dy)`

- 이미지를 원래의 크기로 나타냄
 - : `image` : 출력할 이미지
 - : `dx, dy` : 이미지가 삽입될 좌표 (이미지의 왼쪽 상단 모서리 좌표)

`drawImage(image, dx, dy, dw, dh)`

- 이미지를 지정한 크기로 나타냄
 - : `image` : 출력할 이미지
 - : `dx, dy` : 이미지가 삽입될 좌표 (이미지의 왼쪽 상단 모서리 좌표)
 - : `dw, dh` : 출력할 이미지의 너비와 높이



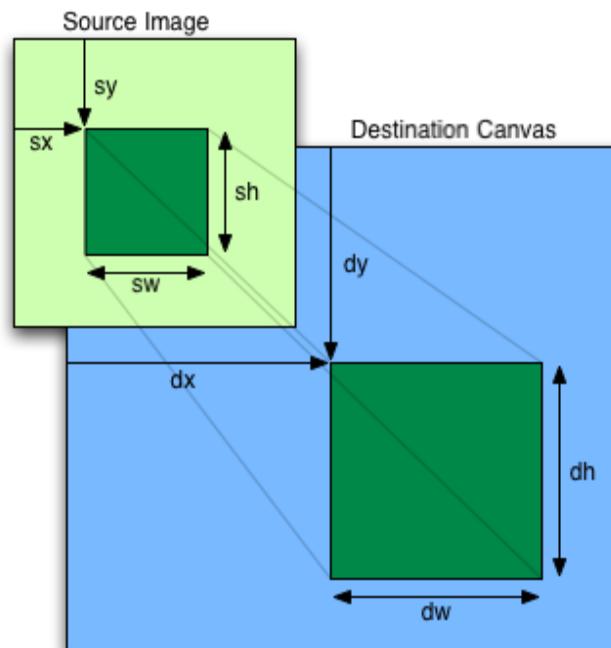
이미지 삽입



`drawImage(image, sx, sy, sw, sh, dx, dy, dw, dh)`

- 원래 이미지의 일부를 추출해 지정한 크기로 출력함

- : image : 출력할 이미지
- : sx, sy : 원래 이미지에서 추출할 영역의 시작 좌표(추출할 영역의 좌측 상단 좌표)
- : sw, sh : 추출할 영역의 너비와 높이
- : dx, dy : 이미지가 삽입될 좌표 (이미지의 왼쪽 상단 모서리 좌표)
- : dw, dh : 삽입될 이미지의 너비와 높이



이미지 삽입



- **drawImage() 메서드를 사용하기 위해서는 반드시 Image() 객체를 먼저 생성해야 함**
 - Image() 객체의 src 속성에 이미지의 주소를 지정
- **onload 이벤트와 콜백함수를 사용해야 함**

```
var obj = new Image();
obj.src = 이미지 이름;

obj.onload = function(e) {
    context.drawImage(obj, x, y);
}
```

```
var pic = new Image();
pic.src = 'image.jpg';

pic.onload = function(e) {
    context.drawImage(pic, 10, 10);
}
```





■ drawImage() 메서드의 예 - 1

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<script type="text/javascript">
function draw()
{
    var canvas = document.getElementById('canvas');
    context = canvas.getContext('2d');

    var golf_1 = new Image();
    golf_1.src = 'images/image.jpg';
    golf_1.onload = function(e) {
        context.drawImage(golf_1, 10, 10);
    }
    var golf_2 = new Image();
    golf_2.src = 'images/image.jpg';
    golf_2.onload = function(e) {
        context.drawImage(golf_2, 350, 10, 140, 100);
    }
}
</script>
</head>
<body onload="draw()">
    <canvas id="canvas" width=600 height=250 style="border:1px solid #000000">
        canvas 사용하기
    </canvas>
</body>
</html>
```





■ drawImage() 메서드의 예 - 2

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <script type="text/javascript">
        function draw()
        {
            var canvas = document.getElementById('canvas');
            context = canvas.getContext('2d');

            var golf_1 = new Image();
            golf_1.src = 'images/image.jpg';
            golf_1.onload = function(e) {
                context.drawImage(golf_1, 10, 10);
            }

            var golf_2 = new Image();
            golf_2.src = 'images/image.jpg';
            golf_2.onload = function(e) {
                context.drawImage(golf_2, 100,100, 100, 100, 350, 10, 150, 150);
            }
        }
    </script>
</head>
<body onload="draw();">
    <canvas id="canvas" width="600" height="250" style="border:1px solid #000000">canvas 사용하기</canvas>
</body>
</html>
```





■ createPattern() 메서드

- 특정 이미지를 반복해서 사용하는 패턴을 생성하고자 할 경우 사용

createPattern(pattern, 반복여부)

- pattern : 반복하고자 하는 패턴의 이미지
- 반복 여부 : repeat, repeat-x, repeat-y, no-repeat

```
var obj = new Image();
obj.src = 이미지 이름;

obj.onload = function(e) {
    var patrn = context.createPattern(obj, 반복선택)
    context.fillStyle=patrn
    패턴 출력 영역 지정
}
```

```
var smile = new Image();
smile.src = 'images/smile.jpg';

smile.onload = function(e) {
    var pattern = context.createPattern(smile, 'repeat');
    context.fillStyle = pattern;
    context.fillRect(0, 0, 200, 200);
}
```



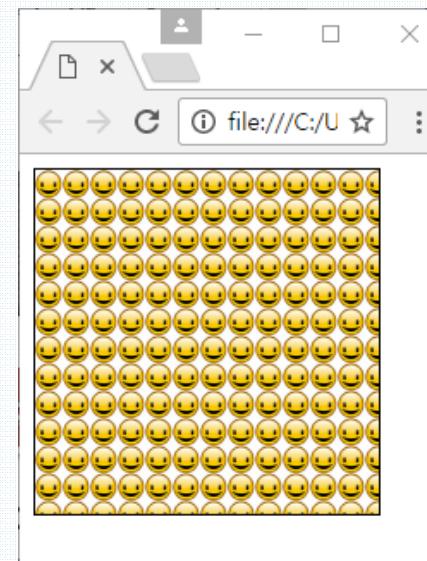


■ createPattern() 메서드의 예

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <script type="text/javascript">
        function rect()
        {
            var canvas = document.getElementById('canvas');
            context = canvas.getContext('2d');
            var smile = new Image();
            smile.src = 'images/smile.jpg';

            smile.onload = function(e) {
                var pattern = context.createPattern(smile, 'repeat');
                context.fillStyle = pattern;
                context.fillRect(0, 0, 200, 200);
            }
        }
    </script>
</head>

<body onload="rect();">
    <canvas id="canvas" width="200" height="200" style="border:1px solid #000000">
        canvas 사용하기
    </canvas>
</body>
</html>
```





도형의 합성과 변환



▣ 합성(composition)

- 캔버스에서 객체(도형이나 이미지) 위에 다른 도형이나 이미지가 겹쳐질 때, 두 객체 간의 겹치는 부분의 처리를 결정하는 것

▣ globalCompositeOperation 속성

- 합성을 수행하는 속성

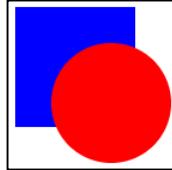
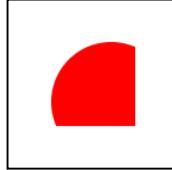
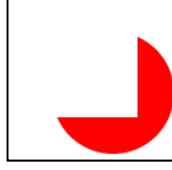
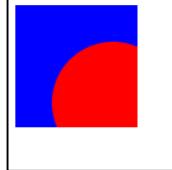
globalCompositeOperation = 값

- 값은 source와 destination으로 구성됨
- **source**
 - : 앞으로 그리고자 하는 도형을 의미 (나중에 위로 겹쳐질 도형을 의미)
- **destination**
 - : 이미 존재하는 도형을 의미 (먼저 그려진 도형을 의미)





▣ 도형 합성하기 (원이 source이고, 사각형이 destination인 경우)

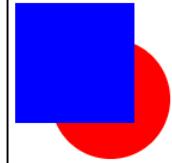
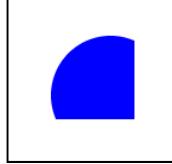
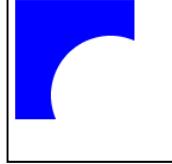
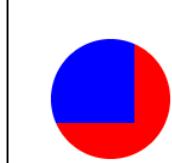
속성 값	예	설명
source-over		<ul style="list-style-type: none">▪ 두 도형이 모두 출력되나 나중에 그린 도형이 먼저 그린 도형 위에 존재▪ 기본값
source-in		<ul style="list-style-type: none">▪ 먼저 그린 도형과 나중에 그린 도형이 겹치는 부분만 표시▪ 나중에 그린 도형이 출력되며, 나머지 부분은 투명하게 처리됨
source-out		<ul style="list-style-type: none">▪ 먼저 그린 도형과 나중에 그린 도형이 겹치지 않는 부분 중에서 나중에 그린 도형의 영역만 출력▪ 나머지 부분은 투명하게 처리
source-atop		<ul style="list-style-type: none">▪ 먼저 그린 도형의 겹치지 않는 부분과 나중에 그린 도형의 겹치는 부분이 출력되며, 나머지 부분은 투명처리▪ 나중에 그린 도형이 위에 존재



도형의 합성



▣ 도형 합성하기 (원이 source이고, 사각형이 destination인 경우)

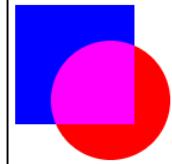
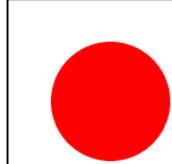
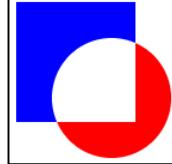
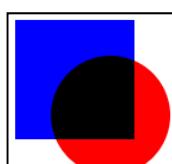
속성 값	예	설명
destination-over		<ul style="list-style-type: none">▪ 두 도형이 모두 출력되나 먼저에 그린 도형이 위에 출력
destination-in		<ul style="list-style-type: none">▪ 먼저 그린 도형과 나중에 그린 도형이 겹치는 부분만 표시▪ 먼저에 그린 도형이 출력되며, 나머지 부분은 투명하게 처리
destination-out		<ul style="list-style-type: none">▪ 먼저 그린 도형과 나중에 그린 도형이 겹치지 않는 부분 중에서 먼저 그린 도형의 영역만 출력▪ 나머지 부분은 투명하게 처리
destination-atop		<ul style="list-style-type: none">▪ 먼저 그림 도형의 겹치지 않는 부분과 나중에 그린 도형의 겹치는 부분이 출력되며, 나머지 부분은 투명처리▪ 먼저에 그린 도형이 위에 존재



도형의 합성



▣ 도형 합성하기 (원이 source이고, 사각형이 destination인 경우)

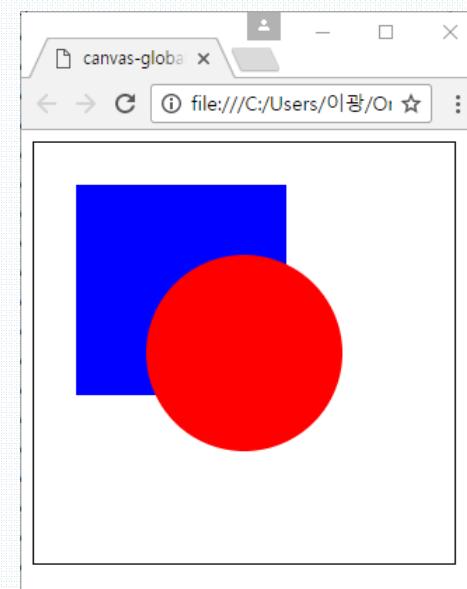
속성 값	예	설명
lighter		두 도형이 모두 출력되거나 겹치는 부분은 두 도형의 색이 결합되어 출력
copy		나중에 그린 도형만 출력 (나머지는 모두 투명 처리)
xor		두 도형이 겹치는 부분만 투명 처리되고 모두 출력
darker		두 도형이 모두 출력되거나 겹치는 부분은 두 도형의 색의 차이가 출력





■ globalCompositeOperation 속성의 예

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <script type="text/javascript">
        function draw()
        {
            var canvas = document.getElementById('canvas');
            context = canvas.getContext('2d');
            context.fillStyle = 'blue';
            context.fillRect(30, 30, 150, 150);
            context.globalCompositeOperation = 'source-over';
            context.fillStyle = 'red';
            context.arc(150, 150, 70, 0, 360*Math.PI/180, true);
            context.fill();
        }
    </script>
</head>
<body onload="draw();">
    <canvas id="canvas" width="300" height="300" style="border:1px solid #000000">
        canvas 사용하기
    </canvas>
</body>
</html>
```



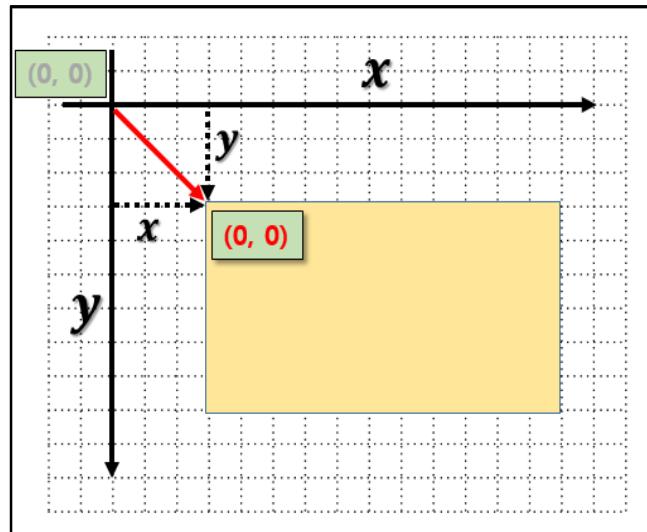


■ translate() 메서드

- 캔버스 내에서 좌표계를 인자로 주어진 값 만큼 이동하는 기능 수행

```
translate(x, y)
```

- x, y 는 각각 수평이동좌표(x)와 수직이동좌표(y)를 의미함





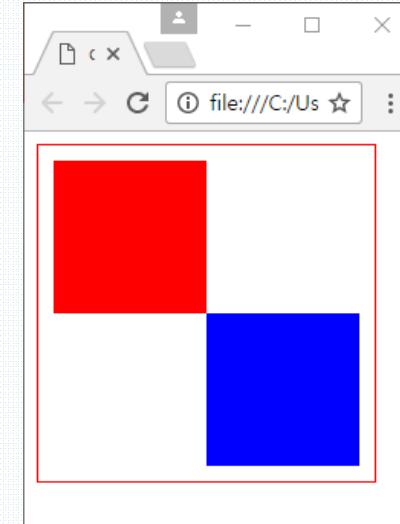
■ translate() 메서드의 예

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <script>
        function Translate() {
            var canvas = document.getElementById('myCanvas');
            var context = canvas.getContext('2d');

            context.fillStyle = "red";
            context.fillRect(10, 10, 100, 100);

            context.translate (100, 100);

            context.fillStyle = "blue";
            context.fillRect (10, 10, 100, 100);
        }
    </script>
</head>
<body onload="Translate();">
    <canvas id="myCanvas" width="220" height="220" style="border: 1px solid red">
        캔버스 연습
    </canvas>
</body>
```



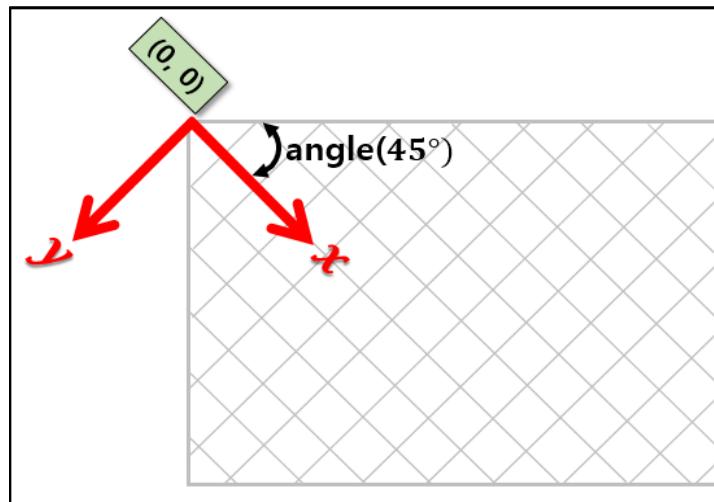


■ rotate() 메서드

- 캔버스 내에서 주어진 각도만큼 좌표계를 시계방향으로 회전시키는 기능을 수행

rotate(회전각)

- 회전각은 라디안을 사용함





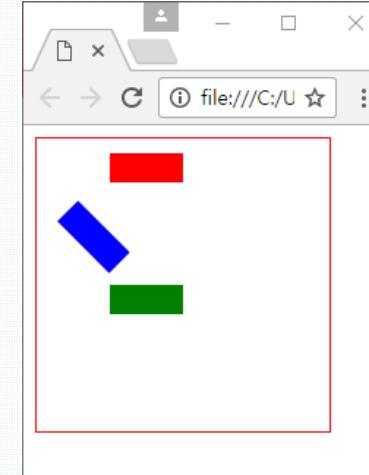
■ rotate() 메서드의 예

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<script>
    function Translate() {
        var canvas = document.getElementById('myCanvas');
        var context = canvas.getContext('2d');

        context.fillStyle = "red";
        context.fillRect(50, 10, 50, 20);

        context.rotate(45*Math.PI/180); 좌표계를 45도 회전
        context.fillStyle = "blue";
        context.fillRect (50, 10, 50, 20);

        context.rotate(315*Math.PI/180); 좌표계를 -45도 회전
        context.fillStyle = "green";
        context.fillRect (50, 100, 50, 20);
    }
</script>
</head>
<body onload="Translate();">
<canvas id="myCanvas" width="200" height="200" style="border: 1px solid red">
    캔버스 연습
</canvas>
</body>
```



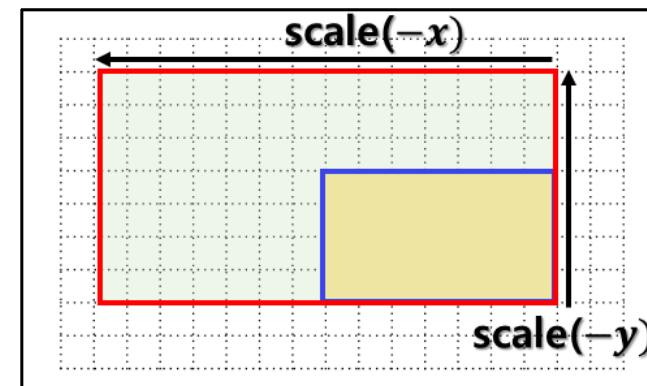
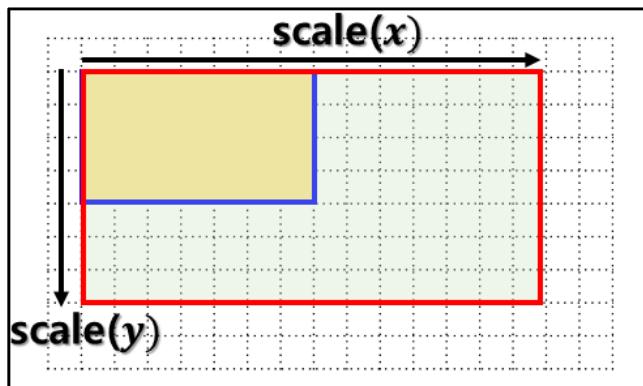


■ scale() 메서드

- 캔버스 내에서 좌표계를 확대하거나 축소하는 기능을 수행

`scale(x, y)`

- x는 수평방향 확대/축소 비율을 의미하며, y는 수평방향 확대/축소 비율을 의미
- x와 y는 각각 음수로도 표현 가능함



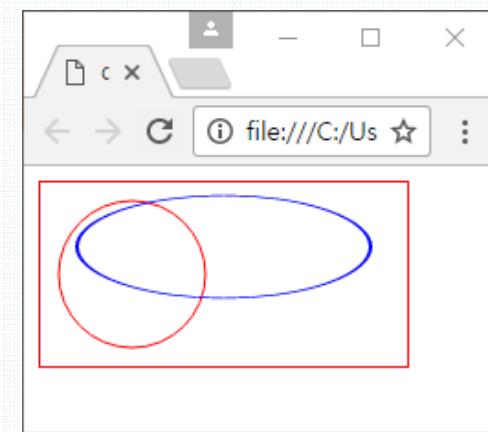


■ scale() 메서드의 예

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <script>
        function Translate() {
            var canvas = document.getElementById('myCanvas');
            var context = canvas.getContext('2d');
            context.beginPath();
            context.strokeStyle = "red";
            context.arc(50, 50, 40, 0, 2 * Math.PI, false);
            context.stroke();

            context.scale(2, 0.7); //좌표 공간을 수평으로 2배 확대, 수직으로 0.7배 축소 수행

            context.beginPath();
            context.strokeStyle = "blue";
            context.arc(50, 50, 40, 0, 2 * Math.PI, false);
            context.stroke();
        }
    </script>
</head>
<body onload="Translate();">
    <canvas id="myCanvas" width="200" height="100" style="border: 1px solid red">
        캔버스 연습
    </canvas>
</body>
</html>
```





텍스트 작성



▣ **strokeText() 메서드, fillText() 메서드**

strokeText(text, x, y, [length])

- 지정된 위치에 텍스트를 입력하고 테두리를 그리는 메서드

fillText(text, x, y, [length])

- 지정된 위치에 텍스트를 입력하고 내부를 채우는 메서드

인자	설명
text	- 입력하고자 하는 텍스트
x, y	- 텍스트가 입력되는 좌표 - 좌표는 텍스트의 왼쪽 상당 모서리를 의미함
length	- 입력되는 텍스트의 전체 길이 - 입력되는 텍스트의 기본 길이보다 짧은 길이로 지정 가능 (입력되는 텍스트의 기본 길이보다 긴 길이로 텍스트를 입력할 수는 없음)





▣ font() 메서드

- 캔버스 내에 작성되는 글자의 속성, 글자 크기, 글자체를 일괄 지정

```
font = “글자속성, 글자크기, 글자체” ;
```

인자	설명
글자 속성	- 글자의 모양 등 글자의 속성을 의미함
글자 크기	- 글자의 크기를 의미함 - 주로 픽셀(px)이나 포인트(pt)를 사용함
글자체	- 글자체를 지정

- 인자의 순서는 중요하지 않음
- 세 개의 인자를 모두 지정해야 하는 것은 아님
 - : 글자의 크기를 지정하지 않을 경우 - 10px로 지정
 - : 글자체를 지정하지 않을 경우 - sans-serif로 표현
 - : 글자의 속성을 지정하지 않을 경우 - normal로 지정



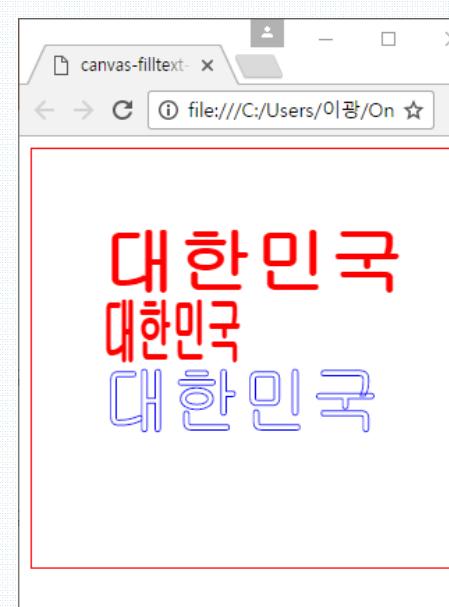


▣ 텍스트 작성의 예 - 1

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <script type="text/javascript">
        function draw()
        {
            var canvas = document.getElementById('myCanvas');
            context = canvas.getContext('2d');

            context.fillStyle='red';
            context.strokeStyle='blue';
            context.font = 'bold 40pt 굴림체';
            context.fillText("대한민국", 50, 100);
            context.fillText("대한민국", 50, 150, 100);
            context.strokeText("대한민국", 50, 200, 200);

        }
    </script>
</head>
<body onload="draw();">
    <canvas id="myCanvas" width="300" height="300" style="border: 1px solid red">
        캔버스 연습
    </canvas>
</body>
</html>
```





▣ 텍스트 작성의 예 - 2

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<script type="text/javascript">
function text()
{
    var canvas = document.getElementById('myCanvas');
    context = canvas.getContext('2d');

    context.beginPath();
    var rectstyle = context.createLinearGradient(100,50,600,300);
    rectstyle.addColorStop(0,"blue");
    rectstyle.addColorStop(1,"red");
    context.fillStyle = rectstyle;

    context.shadowOffsetX = 5;
    context.shadowOffsetY = 5;
    context.shadowColor = 'gray';
    context.shadowBlur = 10;

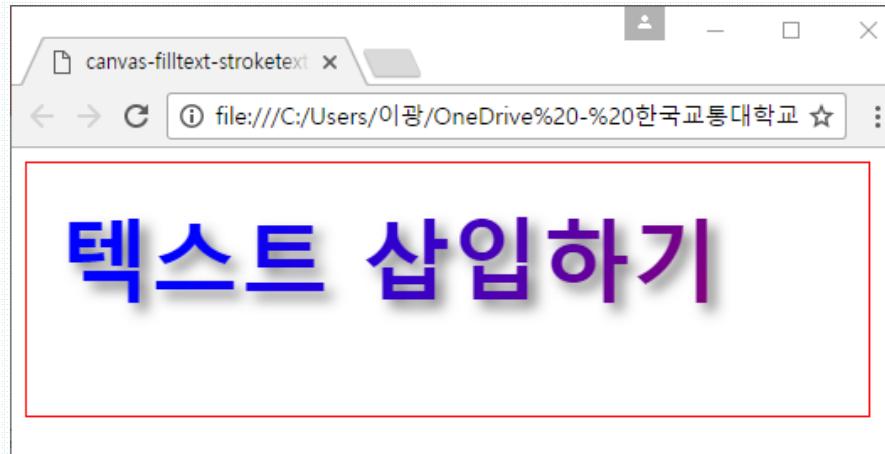
    context.font = "bold 40pt 고딕";
    context.textAlign = 'left';
    context.textBaseline = 'top';
    context.fillText("텍스트 삽입하기",20,20);
}
</script>
```



텍스트 작성



```
</head>
<body onload="text();">
  <canvas id="myCanvas" width="500" height="150" style="border: 1px solid red">
    캔버스 연습
  </canvas>
</body>
</html>
```





▣ textAlign() 메서드

- 텍스트에 대한 수평 정렬을 지정하기 위한 기능 수행

textAlign = start | end | left | right | center

인자	설명
start	캔버스 요소의 방향성에 따라서 앵커 포인트는 다음과 같이 결정됨 - ltr(left-to-right) 왼쪽에서 오른쪽 방향 : 텍스트의 왼쪽 가장자리가 기준이 됨 - rtl(right-to-left) 오른쪽에서 왼쪽 방향 : 텍스트의 오른쪽 가장자리가 기준이 됨
end	캔버스 요소의 방향성에 따라서 앵커 포인트는 다음과 같이 결정됨 - ltr(left-to-right) 왼쪽에서 오른쪽 방향 - 텍스트의 오른쪽 가장자리가 기준이 됨 - rtl(right-to-left) 오른쪽에서 왼쪽 방향 - 텍스트의 왼쪽 가장자리가 기준이 됨
left	앵커 포인트는 텍스트의 왼쪽 가장자리가 기준이 됨
right	앵커 포인트는 텍스트의 오른쪽 가장자리가 기준이 됨
center	앵커 포인트는 텍스트의 중심에 있음





▣ textAlign() 메서드의 예

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <script>
        function text() {
            var canvas = document.getElementById('myCanvas');
            var context = canvas.getContext('2d');
            var startX = 90;

            context.lineWidth = 1;
            context.strokeStyle = "red";

            context.beginPath();
            context.moveTo(90, 10);
            context.lineTo(90, 200);
            context.stroke();

            context.font = "20px 'Courier New'";
            context.fillStyle = "black";

            context.textAlign = "start";
            context.fillText ( "start", 90, 30);

            context.textAlign = "end";
            context.fillText ( "end", 90, 70);
        }
    </script>
</head>
<body>
<div style="border: 1px solid black; width: 100px; height: 100px; position: relative; margin: 10px auto; background-color: #f0f0f0; overflow: hidden; border-radius: 10px; font-family: 'Courier New'></div>
</body>

```



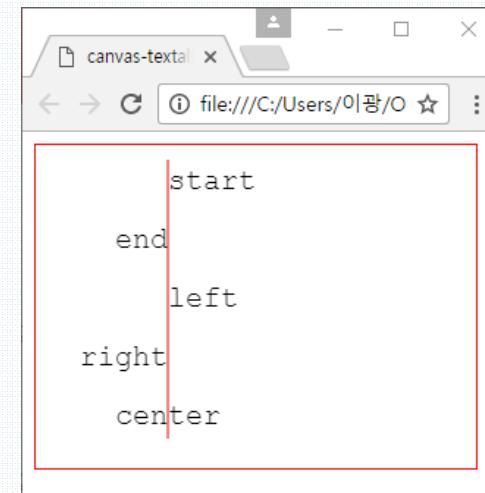
텍스트 정렬



```
context.textAlign = "left";
context.fillText ( "left", 90, 110);

context.textAlign = "right";
context.fillText ( "right", 90, 150);

context.textAlign = "center";
context.fillText ( "center", 90, 190);
}
</script>
</head>
<body onload="text();">
<canvas id="myCanvas" width="300" height="300" style="border: 1px solid red">
    캔버스 연습
</canvas>
</body>
</html>
```





▣ **textBaseline()** 메서드

- 텍스트에 대한 수직 정렬을 지정하기 위한 기능 수행

```
textBaseline = top | hanging | middle| alphabetic | ideographic | bottom
```

인자	설명
top	세로 상단을 기준으로 함
hanging	hanging 기준선을 기준으로 한다(주로 인도의 다양한 언어에서 사용된다).
middle	세로 가운데를 기준으로 함
alphabetic	영어 알파벳 글자의 기준선을 기준으로 함
ideographic	표의 문자의 기준선을 기준으로 함 (주로 일본어와 중국어 등의 언어에서 사용)
bottom	세로 바닥을 기준으로 함





▣ textBaseline() 메서드의 예

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
<script>
    function text() {
        var canvas = document.getElementById('myCanvas');
        var context = canvas.getContext('2d');

        context.lineWidth = 1;
        context.strokeStyle = "red";

        context.beginPath();
        context.moveTo(5, 50);
        context.lineTo(595, 50);
        context.stroke();

        context.font = "15px 'Courier New'";
        context.fillStyle = "black";

        context.textBaseline = "top";
        context.fillText("top", 10, 50);

        context.textBaseline = "middle";
        context.fillText("middle", 50, 50);
    }
}
```



텍스트 정렬



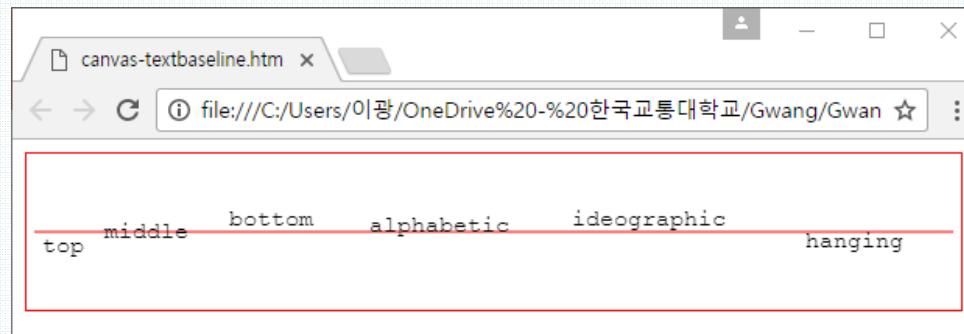
```
context.textBaseline = "bottom";
context.fillText("bottom", 130, 50);

context.textBaseline = "alphabetic";
context.fillText("alphabetic", 220, 50);

context.textBaseline = "ideographic";
context.fillText("ideographic", 350, 50);

context.textBaseline = "hanging";
context.fillText("hanging", 500, 50);
}

</script>
</head>
<body onload="text()">
    <canvas id="myCanvas" width="600" height="100" style="border: 1px solid red">
        캔버스 연습
    </canvas>
</body>
</html>
```





캔버스 상태의 저장과 복원



▣ **save()** 메서드

- 현재 드로잉 상태의 복사본을 스택에 저장

```
save()
```

▣ **restore()** 메서드

- 스택에 저장된 드로잉 상태를 복원

```
restore()
```





▣ save() 메서드, restore() 메서드의 예

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <script>
        function save_restore() {
            var canvas = document.getElementById("myCanvas");
            var context = canvas.getContext("2d");

            var colors = new Array ( "red", "blue", "green");
            var alphas = new Array (0.2, 0.5, 0.8);

            for (var i = 0; i < 3; i++) {
                context.fillStyle = colors[i];
                context.globalAlpha = alphas[i];
                context.save();
            }

            for (var i = 0; i < 3; i++) {
                context.restore();
                context.beginPath();
                context.arc((i+1) * 120, 120, 100, 0, Math.PI * 2, false);
                context.fill();
            }
        }
    </script>
</head>
```



캔버스 상태의 저장과 복원



```
<body onload="save_restore();>
  <canvas id="myCanvas" width="500" height="300" style="border: 1px solid red">
    캔버스 연습
  </canvas>
</body>
</html>
```

