

JavaScript

Canvas – 김근형 강사

Canvas

- Canvas
 - 웹 페이지에 동적인 이미지를 그리는 기술
 - Canvas로 비트맵 이미지를 그릴 수 있으며 직접적인 픽셀을 조작이 가능하다.
 - 캔버스는 2차원의 이미지를 그리는 2D 컨텍스트와 3차원의 이미지를 그리는 3D 컨텍스트 두 가지가 있다.
 - 여기서는 2D 컨텍스트에만 초점을 맞출 예정

Canvas

○ canvas 요소

```
<canvas width="300" height="150"></canvas>
```

- Canvas를 통해 그림을 그리기 위해서는 반드시 canvas 요소를 사용해야만 한다.
- canvas 요소에는 Canvas 영역의 가로를 정하는 width 속성과 세로를 나타내는 height 속성이 있다.
- 속성값은 숫자여야 하며 길이단위는 픽셀이다.
- 지정하지 않으면 기본값을 적용해서 width는 300, height는 150이다.
- 브라우저는 canvas 요소가 준비되면 투명한 검은색으로 채워진 상태로 랜더링하기 때문에 Canvas에 아무 것도 그리지 않는다면 페이지의 배경이 보인다.

Canvas

- 2d 컨텍스트 객체 얻기

- Canvas로 평면 이미지를 그리려면 2D 컨텍스트 객체를 얻어야 한다.

2D 컨텍스트 객체를 얻는 함수	설명
Context = canvas.getContext(contextId)	canvas 요소로부터 contextId에 지정한 타입의 컨텍스트 객체를 반환한다. 지정된 contextId를 지원하지 않으면 null을 반환한다.

- canvas 요소의 노드 객체에서 getContext() 함수를 호출하여 컨텍스트 객체를 얻을 수 있다.
 - 이 컨텍스트 객체를 통해서 Canvas 위에 그림을 그리는 함수나 속성을 사용할 수 있다.

Canvas

- 2d 컨텍스트 객체 얻기
 - getContext() 함수에는 종류를 나타내는 문자열을 매개변수로 지정해야 한다.
 - 사용 방법은 “2d”와 “3d” 가 존재한다.
 - [2d 컨텍스트]를 얻기 위해선 getContext안의 데이터로 “2d” 로 넣는다.

```
var canvas = document.querySelector("canvas");
var context = canvas.getContext("2d");
```

Canvas

- canvas의 좌표계
 - Canvas에서는 그림을 그릴 때 좌표를 지정해야 한다.
 - 좌표계의 원점은(0,0)을 기준으로 x 좌표 값은 좌에서 우로, y 좌표는 위에서 아래로 증가한다.



사각형 그리기

○ 직사각형 그리기

- 캔버스 API에서는 직사각형과 관련된 메서드로 직사각형을 그리는 메서드, 지우는 메서드, 칠하는 메서드 등 세 가지 메서드를 제공한다.

메서드	설명
context.clearRect(double x, double y, double w, double h)	특정 직사각형과 현재 클리핑 영역이 교차하는 곳에 있는 모든 픽셀을 지운다. 클리핑 영역은 캔버스 크기와 같으므로 클리핑 영역을 변경하지 않으면, 메서드의 인수에 명시된 픽셀만큼 지우게 된다.
context.strokeRect(double x, double y, double w, double h)	직사각형을 그린다. 직사각형을 그리는데 필요한 속성은 다음과 같은 속성을 통해 그릴 수 있다. 너비나 높이를 0으로 설정하면 fillRect() 메서드에서는 어떤 작업도 하지 않는다. -> strokeStyle, linewidth, lineJoin, miterLimit
context.fillRect(double x, double y, double w, double h)	fillStyle 속성에 따라 내부를 채운 직사각형을 그린다. 너비나 높이를 0으로 설정하면 fillRect() 메서드에서는 어떤 작업도 하지 않는다.

사각형 그리기

○ 색 지정하기

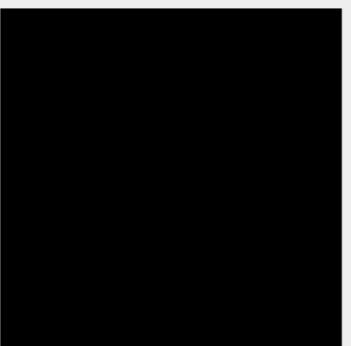
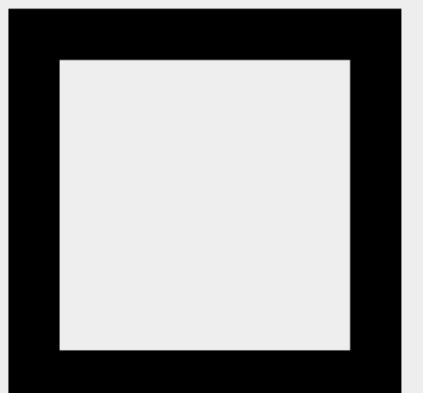
- Canvas에서 색을 지정이 가능하다.
- 칠하는 타겟은 윤곽선과 도형 안쪽의 색을 칠하는 것이 가능하다.

메서드	설명
context.strokeStyle	현재 윤곽선의 색을 지정한다. 값을 지정해서 윤곽선의 색을 변경할 수 있다.
context.fillStyle	현재 채우기 색을 지정한다. 값을 지정해서 채우기 색을 지정할 수 있다.

- 속성으로 지정할 수 있는 값은 CSS로 색을 지정할 때 사용하는 문자이다.
 - ex) "red", "#ff0000", "#f00", "rgb(255,0,0)", "rgba(255,0,0,0)"

사각형 그리기

○ 사각형 그리기 예제 - 1



```
<style type="text/css">
  body {
    background: #dddddd;
  }

  #canvas {
    background: #eeeeee;
    border: thin solid #aaaaaa;
  }
</style>
</head>
<body>
  <canvas id='canvas' width='600' height='400'>
    Canvas not supported
  </canvas>
  <script type="text/javascript">
    var canvas = document.getElementById('canvas'),
        context = canvas.getContext('2d');

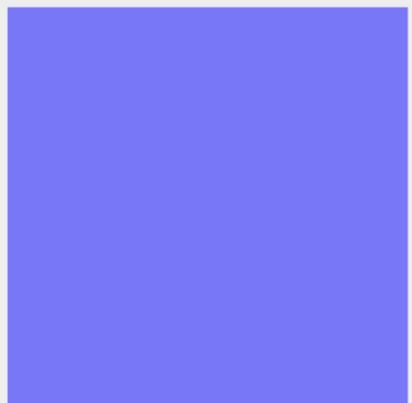
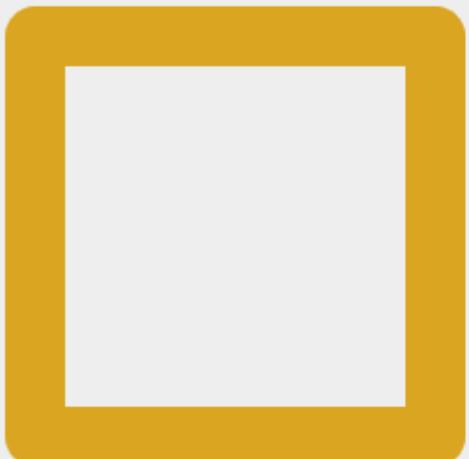
    context.lineWidth = 30;

    context.strokeRect(75, 100, 200, 200);
    context.fillRect(325, 100, 200, 200);

    context.canvas.onmousedown = function (e) {
      context.clearRect(0, 0, canvas.width, canvas.height);
    }
  </script>
</body>
```

사각형 그리기

○ 사각형 그리기 예제 - 2



```
<style type="text/css">
    body {
        background: #dddddd;
    }

    #canvas {
        background: #eeeeee;
        border: thin solid #aaaaaa;
    }
</style>
</head>
<body>
<canvas id='canvas' width='600' height='400'>
    Canvas not supported
</canvas>
<script type="text/javascript">
    var canvas = document.getElementById('canvas'),
        context = canvas.getContext('2d');

    context.lineJoin = 'round';
    context.lineWidth = 30;

    context.strokeStyle = 'goldenrod';
    context.fillStyle = 'rgba(0, 0, 255, 0.5)';

    context.strokeRect(75, 100, 200, 200);
    context.fillRect(325, 100, 200, 200);

    context.canvas.onmousedown = function (e) {
        context.clearRect(0, 0, canvas.width, canvas.height);
    };
</script>
```

사각형 그리기

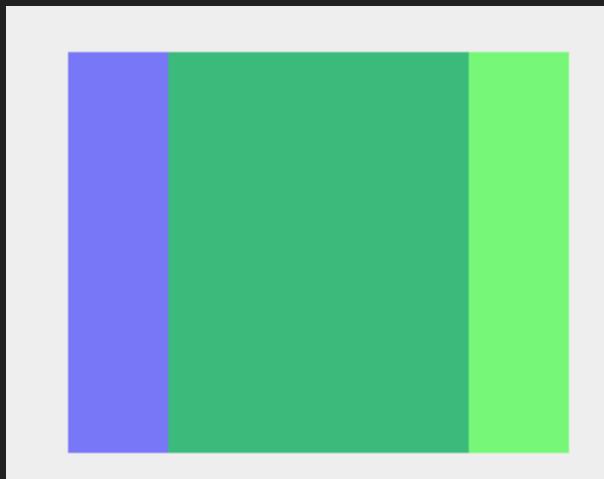
○ 투명도 지정하기

- Canvas에서 투명도를 지정하는 방법은 두가지 방법이 있다.
 - fillStyle 속성과 strokeStyle 속성에 rgba(255,0,0,50) 과 같은 방식으로 a에 값을 부여하는 방법
 - 다른 방법은 모든 이미지에 일정한 투명도를 적용하는 방법
- 후자의 방법에는 globalAlpha라는 속성을 사용하여 지정할 수 있다.

메서드	설명
context.globalAlpha	랜더링 처리에 적용할 알파 값을 반환한다. 값을 지정하여 알파 값을 변경할 수 있으며 지정할 수 있는 값은 0.0~1.0 사이의 소수이다. 0을 지정하면 투명해지고 1을 지정하면 완전히 불투명 해진다. 기본값은 1이다.

사각형 그리기

○ 투명도 지정하기 예제 - 1



```
<canvas id='canvas' width='600' height='400'>
    Canvas not supported
</canvas>
<script type="text/javascript">
    var canvas = document.getElementById('canvas'),
        context = canvas.getContext('2d');

    context.lineJoin = 'round';
    context.lineWidth = 30;

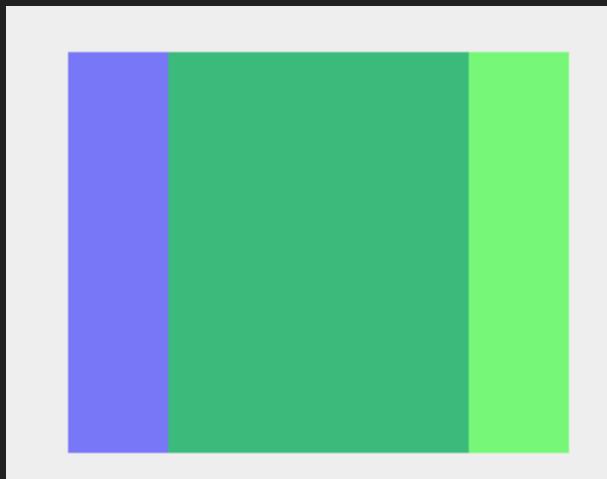
    context.strokeStyle = 'goldenrod';
    context.fillStyle = 'rgba(0, 0, 255, 0.5)';

    context.fillRect(75, 100, 200, 200);

    context.fillStyle = 'rgba(0, 255, 0, 0.5)';
    context.fillRect(125, 100, 200, 200);
</script>
```

사각형 그리기

○ 투명도 지정하기 예제 - 1



```
<canvas id='canvas' width='600' height='400'>
    Canvas not supported
</canvas>
<script type="text/javascript">
    var canvas = document.getElementById('canvas'),
        context = canvas.getContext('2d');

    context.lineJoin = 'round';
    context.lineWidth = 30;
    context.globalAlpha = 0.5

    context.strokeStyle = 'goldenrod';
    context.fillStyle = 'rgb(0, 0, 255)';

    context.fillRect(75, 100, 200, 200);

    context.fillStyle = 'rgb(0, 255, 0)';
    context.fillRect(125, 100, 200, 200);
</script>
```

사각형 그리기

○ 그라데이션

- Canvas에서는 두가지 타입의 그라데이션이 있다.
- 선형 그라데이션과 원형 그라데이션이 있으며 관련 메서드는 아래와 같다.

메서드	설명
context.createLinearGradient(x0, y0, x1, y1)	(x0,y0)와 (x1,y1)을 연결한 직선에 선형 그라데이션을 나타내는 CanvasGradient 객체를 반환한다.
context.createRadialGradient(x0, y0, r0, x1, y1, r1)	중심 좌표(x0, y0)에 반지름이 r0인 원에서 중심좌표(x1, y1)에 반지름이 r1인 원으로 원형 그라데이션을 나타내는 CanvasGradient 객체를 반환한다.
gradient.addColorStop(offset,color)	그라데이션을 나타내는 CanvasGradient 객체에 색을 지정한다. offset에는 0~1 사이의 수치를 지정한다. 0은 시작점, 1은 종료점을 나타낸다.

- CanvasGradient 객체에 addColorStop() 함수로 색을 지정한다.
- 윤곽선에 색을 채우려면 strokeStyle 속성에, 안에 색을 채우려면 fillStyle 속성에 CanvasGradient 객체를 지정한다.

사각형 그리기

○ 그라데이션 예제 - 1

```
<style type="text/css">
    #canvas {
        background: #eeeeee;
        border: thin solid cornflowerblue;
    }
</style>
</head>
<body>
    <canvas id="canvas" width="800" height="500"></canvas>
    <script type="text/javascript">
        var canvas = document.getElementById('canvas'),
            context = canvas.getContext('2d'),
            gradient = context.createLinearGradient(
                0, 0, 0, canvas.height);

        gradient.addColorStop(0, 'blue');
        gradient.addColorStop(0.25, 'white');
        gradient.addColorStop(0.5, 'purple');
        gradient.addColorStop(0.75, 'red');
        gradient.addColorStop(1, 'yellow');

        context.fillStyle = gradient;
        context.rect(0, 0, canvas.width, canvas.height);
        context.fill();
    </script>
```

사각형 그리기

○ 그라데이션 예제 - 2

```
<style type="text/css">
  #canvas {
    background: #eeeeee;
    border: thin solid cornflowerblue;
  }
</style>
</head>
<body>
  <canvas id="canvas" width="800" height="500"></canvas>
  <script type="text/javascript">
    var canvas = document.getElementById('canvas'),
        context = canvas.getContext('2d'),
        gradient = context.createLinearGradient(
          0, 0, canvas.width, 0);

    gradient.addColorStop(0, 'blue');
    gradient.addColorStop(0.25, 'white');
    gradient.addColorStop(0.5, 'purple');
    gradient.addColorStop(0.75, 'red');
    gradient.addColorStop(1, 'yellow');

    context.fillStyle = gradient;
    context.rect(0, 0, canvas.width, canvas.height);
    context.fill();
  </script>
```

사각형 그리기

○ 그라데이션 예제 - 3

```
<style type="text/css">
  #canvas {
    background: #eeeeee;
    border: thin solid cornflowerblue;
  }
</style>
</head>
<body>
  <canvas id="canvas" width="800" height="500"></canvas>
  <script type="text/javascript">
    var canvas = document.getElementById('canvas'),
        context = canvas.getContext('2d'),
        gradient = context.createLinearGradient(
          0, 0, canvas.width, canvas.height);

    gradient.addColorStop(0, 'blue');
    gradient.addColorStop(0.25, 'white');
    gradient.addColorStop(0.5, 'purple');
    gradient.addColorStop(0.75, 'red');
    gradient.addColorStop(1, 'yellow');

    context.fillStyle = gradient;
    context.rect(0, 0, canvas.width, canvas.height);
    context.fill();
  </script>
```

사각형 그리기

○ 그라데이션 예제 - 4

```
<style type="text/css">
  #canvas {
    background: #eeeeee;
    border: thin solid cornflowerblue;
  }
</style>
</head>
<body>
  <canvas id="canvas" width="800" height="500"></canvas>
  <script type="text/javascript">
    var canvas = document.getElementById('canvas'),
        context = canvas.getContext('2d'),
        gradient = context.createLinearGradient(
          canvas.width, canvas.height, 0, 0);

    gradient.addColorStop(0, 'blue');
    gradient.addColorStop(0.25, 'white');
    gradient.addColorStop(0.5, 'purple');
    gradient.addColorStop(0.75, 'red');
    gradient.addColorStop(1, 'yellow');

    context.fillStyle = gradient;
    context.rect(0, 0, canvas.width, canvas.height);
    context.fill();
  </script>
```

사각형 그리기

○ 그라데이션 예제 - 5

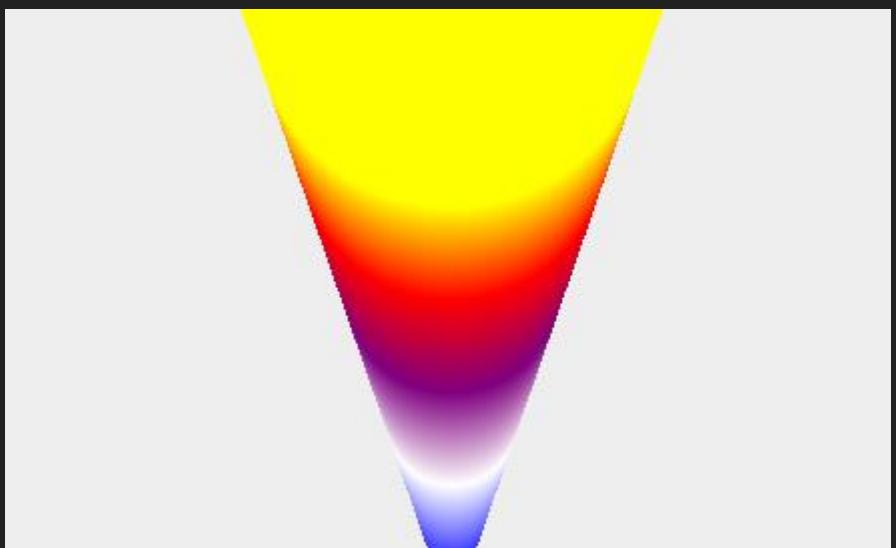
```
<style type="text/css">
  #canvas {
    background: #eeeeee;
    border: thin solid cornflowerblue;
  }
</style>
</head>
<body>
  <canvas id="canvas" width="800" height="500"></canvas>
  <script type="text/javascript">
    var canvas = document.getElementById('canvas'),
        context = canvas.getContext('2d'),
        gradient = context.createLinearGradient(
          0, 0, 0, canvas.height/2);

    gradient.addColorStop(0, 'blue');
    gradient.addColorStop(0.25, 'white');
    gradient.addColorStop(0.5, 'purple');
    gradient.addColorStop(0.75, 'red');
    gradient.addColorStop(1, 'yellow');

    context.fillStyle = gradient;
    context.rect(0, 0, canvas.width, canvas.height);
    context.fill();
  </script>
```

사각형 그리기

○ 그라데이션 예제 - 6



```
<style type="text/css">
    #canvas {
        background: #eeeeee;
        border: thin solid cornflowerblue;
    }
</style>
</head>
<body>
    <div display='inline'>
        <canvas id='canvas' width='450' height='275'>
            Canvas not supported
        </canvas>
    </div>
    <script type="text/javascript">
        var canvas = document.getElementById('canvas'),
            context = canvas.getContext('2d'),
            gradient = context.createRadialGradient(
                canvas.width/2, canvas.height, 10,
                canvas.width/2, 0, 100);

        gradient.addColorStop(0, 'blue');
        gradient.addColorStop(0.25, 'white');
        gradient.addColorStop(0.5, 'purple');
        gradient.addColorStop(0.75, 'red');
        gradient.addColorStop(1, 'yellow');

        context.fillStyle = gradient;
        context.rect(0, 0, canvas.width, canvas.height);
        context.fill();
    </script>
```

Path

○ Path

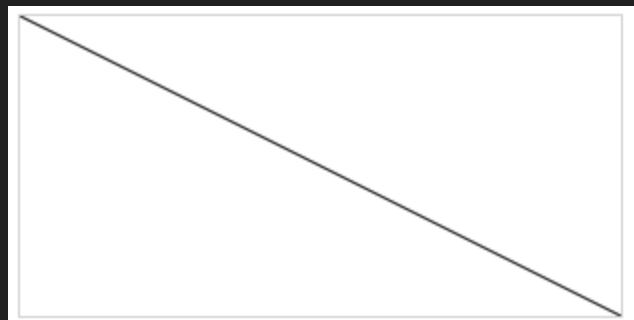
- 패스는 그림의 윤곽을 나타내는 선의 집합이다.
- Canvas에서는 여러가지 함수로 패스에서 선을 더할 수 있다.
- 직선, 곡선 전부 가능하며 패스를 따라 선을 그리거나 색을 채울 수도 있다.
- 다각형을 나타내는 패스를 정의하는 함수는 다음과 같다.

메서드	설명
context.beginPath()	패스를 재정의한다
context.moveTo()	지정된 좌표부터 시작하는 서브패스를 새롭게 만든다.
context.closePath()	서브패스의 종료 위치와 시작 위치를 직선으로 연결한다. 그리고 서브패스의 시작 위치, 즉, 서브패스를 닫을 때 그은 선의 종료 지점에 새로운 서브패스를 만든다.
context.lineTo(x,y)	서브패스의 마지막 위치에서 지정된 좌표로 직선을 더한다.
context.fill()	현재 채우기 스타일로 서브패스를 칠한다.
context.stroke()	현재 윤곽선 스타일로 서브패스에 따라 윤곽을 그린다.

Path

○ 선 그리기

- 선 그리기는 `lineTo()`와 `Stroke`를 이용하여 그릴 수 있다.
- `lineTo()` 만 해서는 선이 그려지지 않으며 반드시 `Stroke()` 메서드를 사용해야만 한다.



```
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>

<script type="text/javascript">

var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.beginPath();
ctx.moveTo(0, 0);
ctx.lineTo(300, 150);
ctx.stroke();

</script>
```

Path

○ 선 그리기 예제

```
<canvas id='canvas' width='500' height='150'>
    Canvas not supported
</canvas>
<script type="text/javascript">
    var context = document.getElementById('canvas').getContext('2d');

    context.lineWidth = 1;
    context.beginPath();
    context.moveTo(50, 10);
    context.lineTo(450, 10);
    context.stroke();

    context.beginPath();
    context.moveTo(50.5, 50.5);
    context.lineTo(450.5, 50.5);
    context.stroke();
</script>
```

Path

○ 선 그리기 예제

```
<canvas id='canvas' width='500' height='150'>
    Canvas not supported
</canvas>
<script type="text/javascript">
    var context = document.getElementById('canvas').getContext('2d');

    context.lineWidth = 1;
    context.beginPath();
    context.moveTo(50, 10);
    context.lineTo(450, 10);
    context.stroke();

    context.beginPath();
    context.moveTo(50.5, 50.5);
    context.lineTo(450.5, 50.5);
    context.stroke();
</script>
```

Path

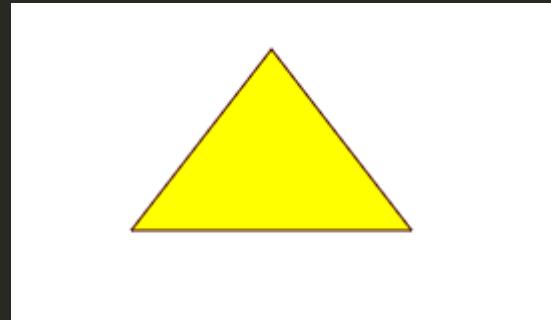
○ 선으로 도형 만들기 예제

```
<canvas id='canvas' width='800' height='500'>
    Canvas not supported
</canvas>
<script type="text/javascript">
    var context = document.getElementById('canvas').getContext('2d');

    context.beginPath();
    context.moveTo(150, 30);
    context.lineTo(220, 120);
    context.lineTo(80, 120);
    context.closePath();

    context.fillStyle = "#ffff00";
    context.fill();

    context.strokeStyle = "#440000";
    context.stroke();
</script>
```



Path

○ Serve-Path

- 서브패스는 패스의 부분집합이다.
- 서브패스는 선의 집합을 의미하며 패스는 수많은 서브패스로 이루어져 있다.
- 서브패스를 조절하여 원하는 모양을 만들어 내는 것이 캔버스 그리기의 핵심이다.

Path

○ Serve-Path 초기화 안 할 경우 예제

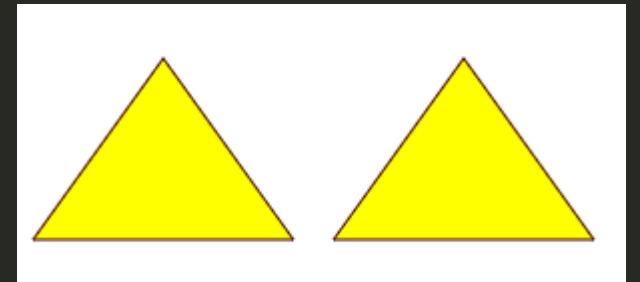
```
<canvas id='canvas' width='800' height='500'>
    Canvas not supported
</canvas>
<script type="text/javascript">
    var context = document.getElementById('canvas').getContext('2d');

    context.beginPath();
    context.moveTo(75, 30);
    context.lineTo(140, 120);
    context.lineTo(10, 120);
    context.closePath();

    context.moveTo(225, 30);
    context.lineTo(290, 120);
    context.lineTo(160, 120);
    context.closePath();

    context.fillStyle = "#ffff00";
    context.fill();

    context.strokeStyle = "#440000";
    context.stroke();
</script>
```



Path

- Serve-Path 초기화 할 경우 예제

```
<script type="text/javascript">
    var context = document.getElementById('canvas').getContext('2d');

    context.beginPath();
    context.moveTo(75, 30);
    context.lineTo(140, 120);
    context.lineTo(10, 120);
    context.closePath();

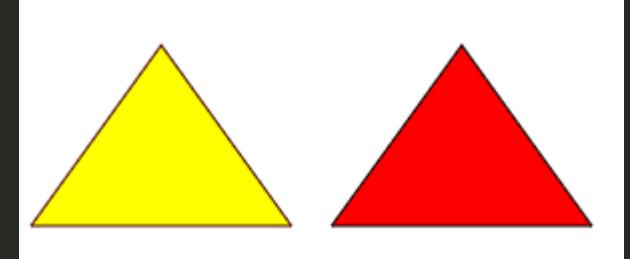
    context.fillStyle = "#ffff00";
    context.fill();

    context.strokeStyle = "#440000";
    context.stroke();

    context.beginPath();
    context.moveTo(225, 30);
    context.lineTo(290, 120);
    context.lineTo(160, 120);
    context.closePath();

    context.fillStyle = "#ff0000";
    context.fill();

    context.strokeStyle = "#000000";
    context.stroke();
</script>
```



Path

○ 원호 그리기

- Canvas 에는 원호를 그리기 위한 함수가 두가지 있다.

메서드	설명
context.arc(x, y, radius, startAngle, endAngle, anticlockwise)	원호를 그린다.

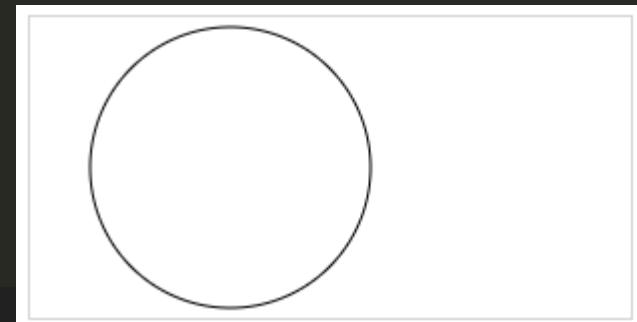
- arc() 메서드는 여섯개의 파라미터를 가지고 있다.
- x, y : 원에 대한 중심 좌표
- radius : 원의 반지름
- startAngle, endAngle : 원주를 따라 그려지는 호에 대한 시작 각도와 종료 각도
- counterclockwise : 기본값을 false로 설정하면 브라우저에서는 호를 시계 방향으로, true 일 경우 호를 반대 방향으로 그린다.

Path

○ 원호 그리기 예제

```
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>

<script type="text/javascript">
    var c = document.getElementById("myCanvas");
    var ctx = c.getContext("2d");
    ctx.beginPath();
    ctx.arc(100, 75, 70, 0, 2 * Math.PI, true);
    ctx.stroke();
</script>
```



Path

○ 원호 그리기 예제

```
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>

<script type="text/javascript">
    var c = document.getElementById("myCanvas");
    var ctx = c.getContext("2d");
    ctx.beginPath();
    ctx.arc(100, 75, 50, 1/2 * Math.PI, 2 * Math.PI, true);
    ctx.stroke();
</script>
```



Path

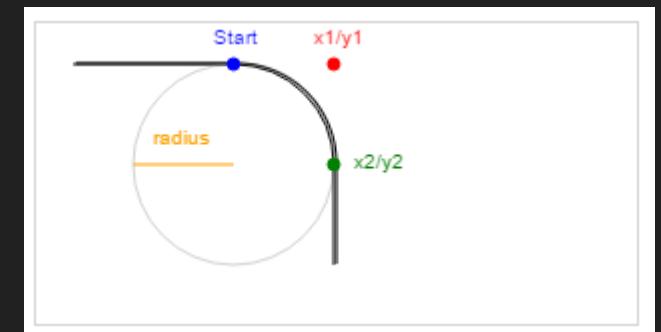
○ 원호 그리기

메서드	설명
context.arcTo(x1, y1, x2, y2, radius)	원호를 그린다.

- arc() 메서드와 비슷하지만 차이점은 현재 점(x1, x2)에서 다음 점(x2, y2) 까지 이어진 호를 그릴 수 있다.
- 주로 모서리가 둥근 도형을 만들 때 사용한다.

```
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>
```

```
<script>
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.beginPath();
ctx.moveTo(20, 20); // Create a starting point
ctx.lineTo(100, 20); // Create a horizontal line
ctx.arcTo(150, 20, 150, 70, 50); // Create an arc
ctx.lineTo(150, 120); // Continue with vertical line
ctx.stroke(); // Draw it
</script>
```



Path

○ 베지어 곡선

- 베지어 곡선은 차체를 디자인할 때 사용됐지만, 오늘날에는 어도비 일러스트레이터, 애플의 코코아, HTML5 캔버스 등과 같은 컴퓨터 그래픽 시스템에서 이용되고 있다.
- 베지어 곡선은 이차 곡선과 다항 곡선이 있다.
- 이차 곡선은 두개의 기준점과 한 개의 제어점으로 정의된다.
- 다항 베지어 곡선은 두개의 기준점과 두개의 제어점으로 정의된다.
- 캔버스에서는 이차곡선과 다항곡선 둘 다 지원하고 있다.

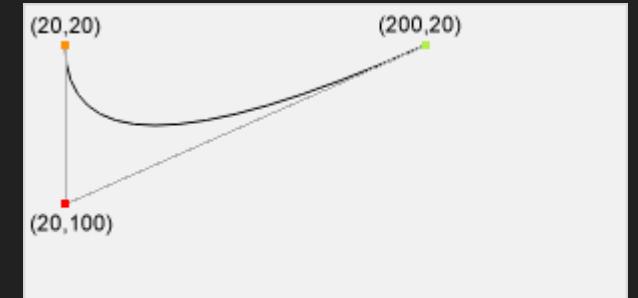
Path

- 이차 곡선

- 이차 베지어 곡선은 한 방향으로 곡선을 이루고 있는 간단한 곡선이다.
- 이차 베지어 곡선에 대한 메서드는 quadraticCurveTo()라는 메서드가 있으며 인수를 네 개를 쓴다.
- 아래와 같이 점을 설정할 경우 쓴다.

```
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">  
Your browser does not support the HTML5 canvas tag.</canvas>
```

```
<script>  
  var c = document.getElementById("myCanvas");  
  var ctx = c.getContext("2d");  
  ctx.beginPath();  
  ctx.moveTo(20, 20);  
  ctx.quadraticCurveTo(20, 100, 200, 20);  
  ctx.stroke();  
</script>
```



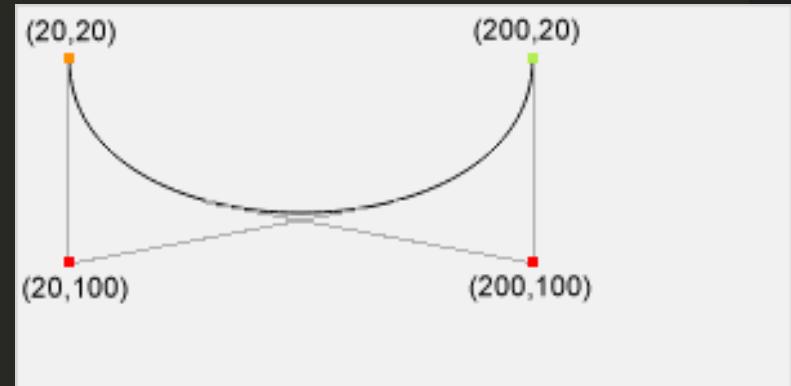
Path

- 다항 곡선

- 이차 곡선처럼 한 방향이 아닌 두 방향으로 곡선을 이루고 있는 곡선을 만들고 싶다면 다항 베지어 곡선을 쓰면 된다.
- 다항 곡선은 bezierCurveTo() 를 이용해 구현이 가능하다

```
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>
```

```
<script type="text/javascript">
  var c = document.getElementById("myCanvas");
  var ctx = c.getContext("2d");
  ctx.beginPath();
  ctx.moveTo(20, 20);
  ctx.bezierCurveTo(20, 100, 200, 100, 200, 20);
  ctx.stroke();
</script>
```



Path

○ 사각형 패스

- 사각형 패스를 설정하는 방법은 다음과 같다.

속성	설명
rect(double x, double y, double width, double height)	직사각형의 서브패스를 생성한다. 이렇게 생성된 서브패스는 닫힌 패스로 항상 시계방향으로 생성된다.

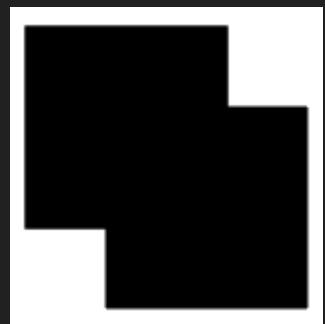
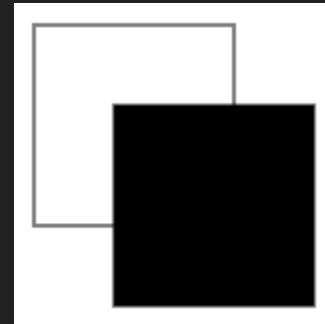
```
<canvas id='canvas' width='1050' height='700'>
    Canvas not supported
</canvas>
<script type="text/javascript">
    var context = document.getElementById('canvas').getContext('2d');

    context.beginPath();
    context.rect(10,10,100,100);
    context.stroke();

    // 쓰고 안쓰고의 차이
    context.beginPath();

    context.rect(50,50,100,100);
    context.stroke();
    context.fill()

</script>
```

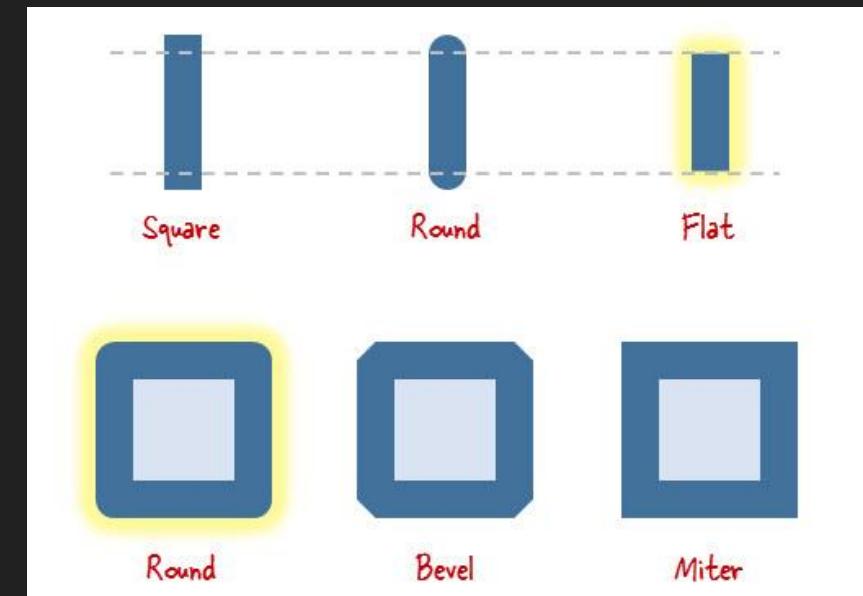


선 스타일 속성

○ 선 스타일 속성

- 선 스타일 속성은 다음과 같다.

속성	설명
context.lineWidth	선의 굵기를 반환한다. 값을 지정하여 선의 굵기를 변경할 수 있다.
context.lineCap	선 끝의 모양을 나타내는 문자열을 반환한다. 값을 지정하여 선 끝의 모양을 변경하는 것도 가능하다. 문자열에 "butt", "round", "square"를 지정해서 모양을 바꿀 수 있다.
context.lineJoin	연결 모양을 나타내는 문자열을 반환한다. 값을 지정하여 연결 모양을 변경할 수 있다. 이용 가능한 연결 모양을 나타내는 문자열은 "bevel", "round", "miter" 이다.
context.miterLimit	타이머의 한계비율을 반환한다. 값을 지정하여 타이머 한계비율을 변경할 수 있다.

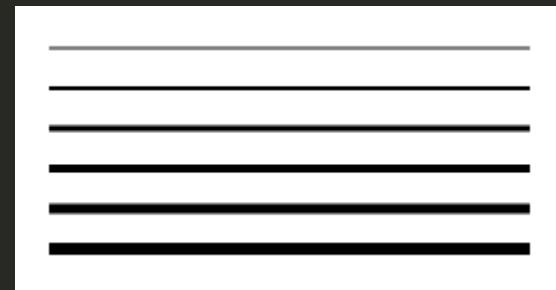


선 스타일 속성

○ 선 스타일 속성 예제 - 1

```
<canvas id='canvas' width='1050' height='700'>
    Canvas not supported
</canvas>
<script type="text/javascript">
    var context = document.getElementById('canvas').getContext('2d');

    for (var i = 1; i <= 6; i++) {
        context.beginPath();
        context.moveTo(30, 20*i);
        context.lineTo(270, 20*i);
        context.lineWidth = i;
        context.stroke();
    }
</script>
```



선 스타일 속성

○ 선 스타일 속성 예제 - 2

```
<p>The three different line caps:</p>
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>

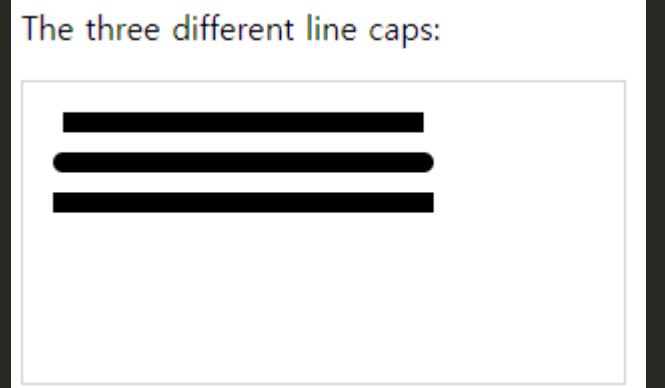
<script type="text/javascript">
  var c = document.getElementById("myCanvas");
  var ctx = c.getContext("2d");

  ctx.beginPath();
  ctx.lineWidth = 10;
  ctx.lineCap = "butt";
  ctx.moveTo(20, 20);
  ctx.lineTo(200, 20);
  ctx.stroke();

  ctx.beginPath();
  ctx.lineCap = "round";
  ctx.moveTo(20, 40);
  ctx.lineTo(200, 40);
  ctx.stroke();

  ctx.beginPath();
  ctx.lineCap = "square";
  ctx.moveTo(20, 60);
  ctx.lineTo(200, 60);
  ctx.stroke();
</script>
```

The three different line caps:

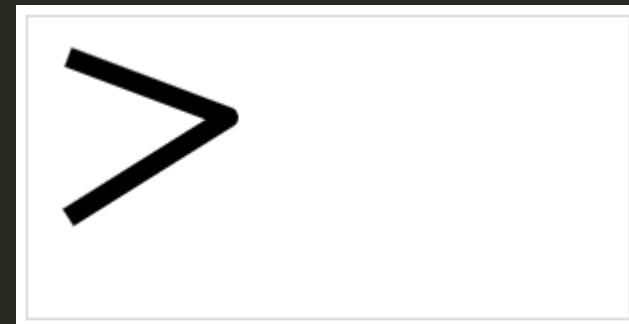


선 스타일 속성

○ 선 스타일 속성 예제 - 3

```
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>
```

```
<script type="text/javascript">
  var c = document.getElementById("myCanvas");
  var ctx = c.getContext("2d");
  ctx.beginPath();
  ctx.lineWidth = 10;
  ctx.lineJoin = "round";
  // ctx.lineJoin = "bevel";
  // ctx.lineJoin = "miter";
  ctx.moveTo(20, 20);
  ctx.lineTo(100, 50);
  ctx.lineTo(20, 100);
  ctx.stroke();
</script>
```

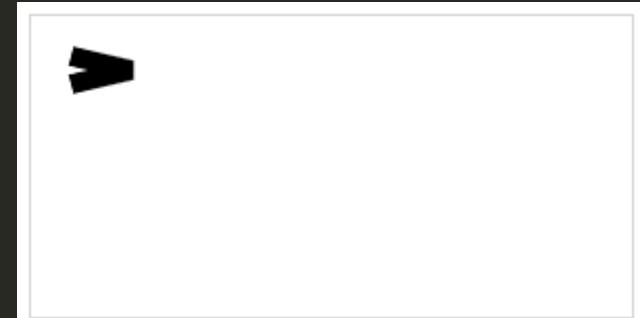


선 스타일 속성

○ 선 스타일 속성 예제 - 4

```
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>
```

```
<script type="text/javascript">
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.lineWidth = 10;
ctx.lineJoin = "miter";
ctx.miterLimit = 3;
ctx.moveTo(20, 20);
ctx.lineTo(50, 27);
ctx.lineTo(20, 34);
ctx.stroke();
</script>
```



텍스트

○ Canvas 의 텍스트 그리기

- Cnavas는 텍스트를 그릴 수 있다.
- Canvas 의 텍스트는 문자의 크기나 폰트 등의 CSS와 같이 스타일링이 가능하다.
- 텍스트를 그리기 위해서 사용하는 함수는 아래와 같다.

메서드/속성	설명
context.fillText(text, x, y [, maxWidth])	텍스트(text)를 좌표 (x, y) 위치에 채워지도록 그린다. 매개변수(maxWidth) 값을 지정하면 텍스트는 이 폭을 넘지 않는다.
context.strokeText(text, x, y [,maxWidth])	텍스트(text)를 좌표 (x, y) 위치에 윤곽선을 그린다. 매개변수(maxWidth) 값을 지정하면 텍스트는 이 폭을 넘지 않게 그린다.
context.font	현재 폰트의 설정을 반환한다. 값을 지정해서 폰트를 변경할 수 있다. 구문은 CSS의 font 속성과 같다. CSS 폰트 값으로 해석되지 않으면 무시된다.
context.measureText(text)	현재 스타일로 특정 텍스트가 그려질 때의 폭, 픽셀 등을 포함하는 TextMetrics 객체 리턴.

텍스트

○ 텍스트 예제 - 1

```
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>

<script type="text/javascript">
    var c = document.getElementById("myCanvas");
    var ctx = c.getContext("2d");

    ctx.font = "20px Georgia";
    ctx.fillText("Hello World!", 10, 50);
</script>
```

Hello World!

텍스트

○ 텍스트 예제 - 2

```
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">  
Your browser does not support the HTML5 canvas tag.</canvas>
```

```
<script type="text/javascript">  
    var c = document.getElementById("myCanvas");  
    var ctx = c.getContext("2d");  
  
    ctx.font = "30px Verdana";  
    // Create gradient  
    var gradient = ctx.createLinearGradient(0, 0, c.width, 0);  
    gradient.addColorStop("0", "magenta");  
    gradient.addColorStop("0.5", "blue");  
    gradient.addColorStop("1.0", "red");  
    // Fill with gradient  
    ctx.fillStyle = gradient;  
    ctx.fillText("Big smile!", 10, 50);  
</script>
```

Big smile!

텍스트

○ 텍스트 예제 - 3

```
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>

<script type="text/javascript">
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");

ctx.font = "20px Georgia";
ctx.strokeText("Hello World!", 10, 50);
</script>
```

Hello World!

텍스트

○ 텍스트 예제 - 4

```
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>

<script type="text/javascript">
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");

ctx.font = "30px Verdana";

// Create gradient
var gradient = ctx.createLinearGradient(0, 0, c.width, 0);
gradient.addColorStop("0", "magenta");
gradient.addColorStop("0.5", "blue");
gradient.addColorStop("1.0", "red");

// Fill with gradient
ctx.strokeStyle = gradient;
ctx.strokeText("Big smile!", 10, 50);
</script>
```

Big smile!

텍스트

○ 텍스트 예제 - 5

```
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>

<script type="text/javascript">
  var c = document.getElementById("myCanvas");
  var ctx = c.getContext("2d");
  ctx.font = "30px Arial";
  var txt = "Hello World"
  ctx.fillText("width:" + ctx.measureText(txt).width, 10, 50);
  ctx.fillText(txt, 10, 100);
</script>
```

width:154.4970703129

Hello World

텍스트

- Canvas 의 폰트 속성

- 정렬을 지정하는 폰트 속성은 다음과 같다.

메서드/속성	설명
context.textAlign	현재의 텍스트 정렬 상태를 반환한다. 값을 지정하여 정렬 상태를 변경할 수 있다.
context.textBaseline	현재의 베이스라인 정렬 설정값을 반환한다. 값을 지정하여 베이스라인 정렬을 변경할 수 있다.

```
<canvas width="500" id="canvas" height="300"></canvas>
<script type="text/javascript">
  const canvas = document.getElementById('canvas');
  const ctx = canvas.getContext('2d');

  ctx.font = 'bold 48px serif';
  ctx.strokeText('Hello world', 50, 100);
</script>
```

Hello world

텍스트

○ Canvas 의 폰트 속성 예제 - 1

```
<canvas id='canvas' width='850' height='440'>
    Canvas not supported
</canvas>

<script type="text/javascript">
    var canvas = document.getElementById('canvas'),
        context = canvas.getContext('2d'),

        LEFT_COLUMN_FONTS = [
            '2em palatino',                      'bolder 2em palatino',
            'lighter 2em palatino',                'italic 2em palatino',
            'oblique small-caps 24px palatino',   'bold 14pt palatino',
            'xx-large palatino',                  'italic xx-large palatino'
        ],
        RIGHT_COLUMN_FONTS = [
            'oblique 1.5em lucida console',       'x-large fantasy',
            'italic 28px monaco',                 'italic large copperplate',
            '36px century',                     '28px tahoma',
            '28px impact',                      '1.7em verdana'
        ],
        LEFT_COLUMN_X = 50,
        RIGHT_COLUMN_X = 500,
        DELTA_Y = 50,
        TOP_Y = 50,
        y = 0;
</script>
```

```
function drawBackground() {
    var STEP_Y = 12,
        i = context.canvas.height;

    context.strokeStyle = 'rgba(0,0,200,0.225)';
    context.lineWidth = 0.5;

    context.save();
    context.restore();

    while(i > STEP_Y*4) {
        context.beginPath();
        context.moveTo(0, i);
        context.lineTo(context.canvas.width, i);
        context.stroke();
        i -= STEP_Y;
    }

    context.save();

    context.strokeStyle = 'rgba(100,0,0,0.3)';
    context.lineWidth = 1;

    context.beginPath();

    context.moveTo(35,0);
    context.lineTo(35,context.canvas.height);
    context.stroke();

    context.restore();
}
```

텍스트

○ Canvas 의 폰트 속성 예제 - 1

```
drawBackground();

context.fillStyle = 'blue',

LEFT_COLUMN_FONTS.forEach( function (font) {
    context.font = font;
    context.fillText(font, LEFT_COLUMN_X, y += DELTA_Y);
});

y = 0;

RIGHT_COLUMN_FONTS.forEach( function (font) {
    context.font = font;
    context.fillText(font, RIGHT_COLUMN_X, y += DELTA_Y);
});
```

2em palatino	<i>oblique 1.5em lucida con</i>
bolder 2em palatino	x-large fantasy
lighter 2em palatino	<i>italic 28px monaco</i>
<i>italic 2em palatino</i>	<i>italic large copperplate</i>
<i>OBLIQUE SMALL-CAPS 24PX PALATINO</i>	36px century
bold 14pt palatino	28px tahoma
<i>xx-large palatino</i>	28px impact
<i>italic xx-large palatino</i>	1.7em verdana

텍스트

○ Canvas 의 폰트 속성 예제 - 2

```
<canvas id="myCanvas" width="300" height="200" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>

<script type="text/javascript">
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");

// Create a red line in position 150
ctx.strokeStyle = "red";
ctx.moveTo(150, 20);
ctx.lineTo(150, 170);
ctx.stroke();

ctx.font = "15px Arial";

// Show the different textAlign values
ctx.textAlign = "start";
ctx.fillText("textAlign=start", 150, 60);
ctx.textAlign = "end";
ctx.fillText("textAlign=end", 150, 80);
ctx.textAlign = "left";
ctx.fillText("textAlign=left", 150, 100);
ctx.textAlign = "center";
ctx.fillText("textAlign=center", 150, 120);
ctx.textAlign = "right";
ctx.fillText("textAlign=right", 150, 140);
</script>
```

textAlign=start
textAlign=end
textAlign=left
textAlign=center
textAlign=right

텍스트

○ Canvas 의 폰트 속성 예제 - 3

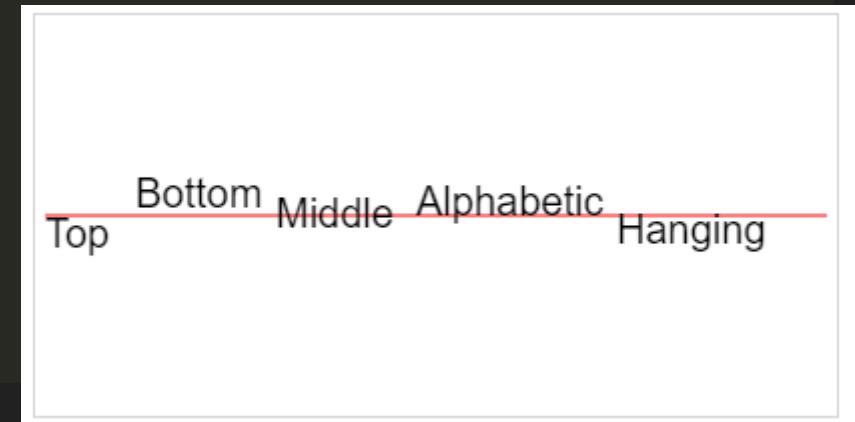
```
<canvas id="myCanvas" width="400" height="200" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>

<script type="text/javascript">
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");

//Draw a red line at y=100
ctx.strokeStyle = "red";
ctx.moveTo(5, 100);
ctx.lineTo(395, 100);
ctx.stroke();

ctx.font = "20px Arial"

//Place each word at y=100 with different textBaseline values
ctx.textBaseline = "top";
ctx.fillText("Top", 5, 100);
ctx.textBaseline = "bottom";
ctx.fillText("Bottom", 50, 100);
ctx.textBaseline = "middle";
ctx.fillText("Middle", 120, 100);
ctx.textBaseline = "alphabetic";
ctx.fillText("Alphabetic", 190, 100);
ctx.textBaseline = "hanging";
ctx.fillText("Hanging", 290, 100);
</script>
```



Shadow

○ 쉐도우 기능

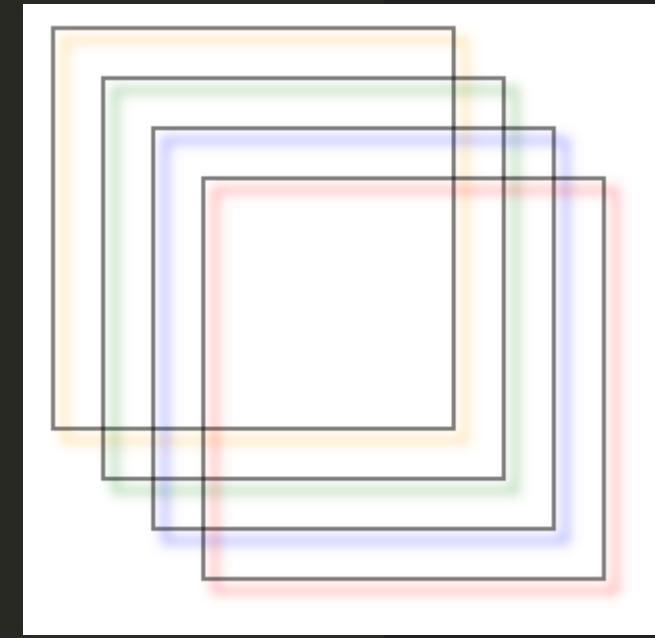
- Canvas에 그려지는 도형에는 그림자를 그릴 수 있다.
- 그림자 관련 속성은 다음과 같다.

메서드/속성	설명
context.shadowColor	현재의 그림자 색을 반환한다. 값을 지정하여 그림자 색을 변경할 수 있다. CSS의 색을 나타내는 구문으로 해석되지 않는 값은 무시된다. 기본값은 투명한 검은색이다.
context.shadowOffsetX context.shadowOffsetY	현재 그림자의 오프셋을 반환한다. 값을 지정하여 그림자의 오프셋을 변경할 수 있다 기본값은 모두 0 이다.
context.shadowBlur	그림자에 적용할 번짐 효과의 단계를 반환한다. 값을 지정하여 blur 단계를 변경할 수 있다. 기본 값은 0 이다.

Shadow

○ 쉐도우 예제 - 1

```
<canvas id="DemoCanvas" width="500" height="600"></canvas>
<script type="text/javascript">
    var canvas = document.getElementById("DemoCanvas");
    if (canvas.getContext) {
        var ctx = canvas.getContext('2d');
        ctx.shadowColor = "black";
        ctx.shadowBlur = 6;
        ctx.shadowOffsetX = 6;
        ctx.shadowOffsetY = 6;
        ctx.shadowColor = "orange";
        ctx.strokeRect(25, 25, 200, 200);
        ctx.shadowColor = "green";
        ctx.strokeRect(50, 50, 200, 200);
        ctx.shadowColor = "blue";
        ctx.strokeRect(75, 75, 200, 200);
        ctx.shadowColor = "red";
        ctx.strokeRect(100, 100, 200, 200);
    }
</script>
```



Shadow

○ 쉐도우 예제 - 2

```
<canvas id="DemoCanvas" width="500" height="600"></canvas>
<script>
var canvas = document.getElementById("DemoCanvas");
if (canvas.getContext) {
    var ctx = canvas.getContext('2d');
    ctx.rect(100, 40, 200, 100);
    ctx.fillStyle = 'green';
    ctx.shadowColor = '#898';
    ctx.shadowBlur = 20;
    ctx.shadowOffsetX = 20;
    ctx.shadowOffsetY = 20;
    ctx.fill();
}
</script>
```



```
<canvas id="DemoCanvas" width="500" height="600"></canvas>
<script>
var canvas = document.getElementById("DemoCanvas");
if (canvas.getContext) {
    var ctx = canvas.getContext('2d');
    ctx.rect(100, 40, 200, 100);
    ctx.fillStyle = 'green';
    ctx.shadowColor = '#898';
    ctx.shadowBlur = 20;
    ctx.shadowOffsetX = -20;
    ctx.shadowOffsetY = -20;
    ctx.fill();
}
</script>
```



Shadow

○ 쉐도우 예제 - 3

```
<canvas id="DemoCanvas" width="500" height="600"></canvas>
<script type="text/javascript">
    var canvas = document.getElementById("DemoCanvas");
    if (canvas.getContext) {
        var ctx = canvas.getContext('2d');
        ctx.fillStyle = 'green';
        ctx.shadowColor = '#898';
        ctx.shadowBlur = 20;
        ctx.shadowOffsetX = 20;
        ctx.shadowOffsetY = 20;
        //ctx.fill();
        ctx.font = 'italic 32px sans-serif';
        ctx.fillText('HTML5 Canvas Tutorial', 10, 50);

        ctx.fillStyle = 'red';
        ctx.shadowColor = '#998';
        ctx.shadowBlur = 1;
        ctx.shadowOffsetX = 9;
        ctx.shadowOffsetY = 7;
        //ctx.fill();
        ctx.font = 'italic 32px sans-serif';
        ctx.fillText('HTML5 Canvas Tutorial', 10, 150);
    }
</script>
```

HTML5 Canvas Tutorial

HTML5 Canvas Tutorial

Pattern

- 패턴

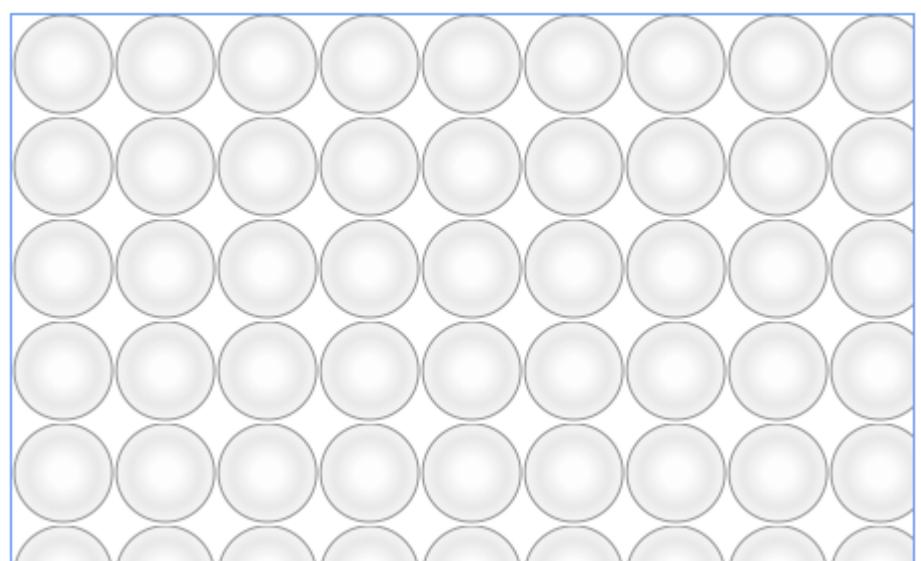
- Canvas에서는 특정 도형이나 지정한 이미지를 배경으로 채울 수 있다.

메서드/속성	설명
context.createPattern(image, repetition)	매개변수 image가 나타내는 이미지를 매개변수 repetition에 지정된 방향으로 반복해서 그리기 위한 CanvasPattern 객체를 반환한다. 매개변수(image)에는 img 요소, canvas 요소 video 요소의 노드 객체를 지정할 수 있다. 매개변수(repetition)에는 "repeat"(양방향), "repeat-x"(수평 방향만), "repeat-y"(수직 방향만), "no-repeat"(없음)를 지정할 수 있다. 기본값은 "repeat" 이다.

Pattern

○ 패턴 예제 - 1

repeat repeat-x repeat-y no repeat



```
<div id='radios'>
    <input type='radio' id='repeatRadio' name='patternRadio' checked>repeat
    <input type='radio' id='repeatXRadio' name='patternRadio'>repeat-x
    <input type='radio' id='repeatYRadio' name='patternRadio'>repeat-y
    <input type='radio' id='noRepeatRadio' name='patternRadio'>no repeat
</div>

<canvas id="canvas" width="450" height="275">Canvas not supported</canvas>

<script type="text/javascript">
    var canvas = document.getElementById('canvas'),
        context = canvas.getContext('2d'),
        repeatRadio = document.getElementById('repeatRadio'),
        noRepeatRadio = document.getElementById('noRepeatRadio'),
        repeatXRadio = document.getElementById('repeatXRadio'),
        repeatYRadio = document.getElementById('repeatYRadio'),
        image = new Image();

    function fillCanvasWithPattern(repeatString) {
        var pattern = context.createPattern(image, repeatString);
        context.clearRect(0, 0, canvas.width, canvas.height);
        context.fillStyle = pattern;
        context.fillRect(0, 0, canvas.width, canvas.height);
        context.fill();
    }

    repeatRadio.onclick = function (e) {fillCanvasWithPattern('repeat'); };
    repeatXRadio.onclick = function (e) {fillCanvasWithPattern('repeat-x'); };
    repeatYRadio.onclick = function (e) {fillCanvasWithPattern('repeat-y'); };
    noRepeatRadio.onclick = function (e) {fillCanvasWithPattern('no-repeat'); };

    image.src = 'redball.png';
    image.onload = function (e) {fillCanvasWithPattern('repeat');};
```

이미지 및 비디오

- 이미지 조작

- 캔버스 컨텍스트에서는 이미지를 그리고 조작하는 작업을 위해 다음과 같이 네가지 메서드를 제공하고 있다.

- drawImage()
 - getImageData()
 - putImageData()
 - createImageData()

이미지 및 비디오

- drawImage()
 - drawImage() 메서드는 이미지, 캔버스 또는 비디오를 캔버스에 그린다.
 - 이미지의 일부를 그리거나 이미지 크기를 늘리거나 줄일 수도 있다.
 - 이미지가 로드되기 전에 drawImage() 메서드를 호출 할 수 없다. 이미지가 로드되도록 하려면 window.onload() 또는 document.getElementById("imageID").onload에서 drawImage()를 호출 할 수 있다.

이미지 및 비디오

- drawImage()
 - drawImage() 메서드는 다음과 같이 3가지 형태로 사용할 수 있으며 인자에 들어가는 내용은 다음과 같다.

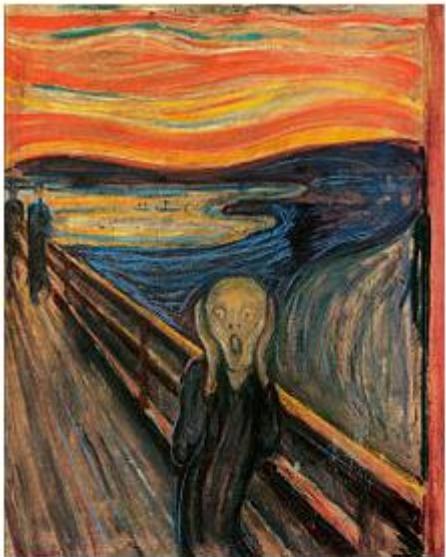
```
drawImage(img, x, y);  
drawImage(img,x,y,width,height);  
drawImage(img,sx,sy,swidth,sheight,x,y,width,height);
```

이미지 및 비디오

○ drawImage() 인자 요소

인자	설명
img	사용해야 할 이미지, 캔버스, 비디오 요소를 지정함
sx	클리핑을 시작할 x축 지점
sy	클리핑을 시작할 y축 지점
swidth	클리핑 할 이미지 폭
sheight	클리핑 할 이미지 높이
x	캔버스에 위치할 이미지의 x 위치
y	캔버스에 위치할 이미지의 y 위치
width	사용할 이미지의 폭(확대 아니면 축소)
height	사용할 이미지의 높이(확대 아니면 축소)

Image to use:



Canvas:



이미지 및 비디오

○ drawImage() 예제(1)

```
<p>Image to use:</p>


<p>Canvas:</p>
<canvas id="myCanvas" width="240" height="297" style="border:1px solid #d3d3d3;">
    Your browser does not support the HTML5 canvas tag.
</canvas>

<script type="text/javascript">
    window.onload = function() {
        var c = document.getElementById("myCanvas");
        var ctx = c.getContext("2d");
        var img = document.getElementById("scream");
        ctx.drawImage(img, 10, 10);
    }
</script>
```

이미지 및 비디오

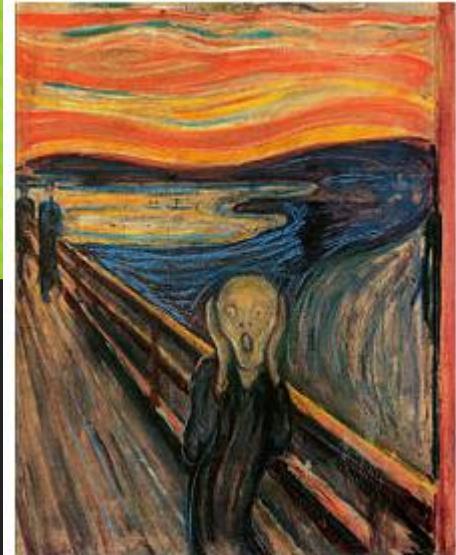
○ drawImage() 예제(2)

```
<p>Image to use:</p>


<p>Canvas:</p>
<canvas id="myCanvas" width="240" height="297" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.
</canvas>

<script type="text/javascript">
window.onload = function() {
    var c = document.getElementById("myCanvas");
    var ctx = c.getContext("2d");
    var img = document.getElementById("scream");
    ctx.drawImage(img, 10, 10, 150, 180);
}
</script>
```

Image to use:



Canvas:

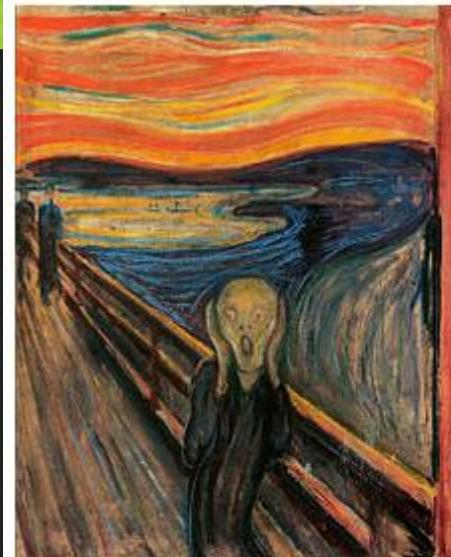


이미지 및 비디오

○ drawImage() 예제(3)

```
<p>Image to use:</p>
![The Scream](img_the_scream.jpg)
```

Image to use:



Canvas:



ImageData 가져오기/넣기/생성하기

- ImageData란?
 - ImageData는 픽셀 데이터를 담고 있는 객체
 - ImageData는 단순히 그림이 아니며 캔버스의 일부 (직사각형)를 지정하며 해당 사각형 내부의 모든 픽셀 정보를 보유한다.
 - 픽셀정보는 RGBA 형태를 배열로 보유하고 있으며 각 배열의 인덱스와 픽셀의 순서는 아래와 같다.
 - index 0 => Red
 - index 1 => Green
 - index 2 => Blue
 - index 3 => Opacity

ImageData 가져오기/넣기/생성하기

- `ImageData`를 쓰는 함수
 - `ImageData`를 쓰는 함수는 아래와 같다.

인자	설명
<code>getImageData()</code>	캔버스에서 지정된 사각형의 픽셀 데이터를 복사하는 <code>ImageData</code> 객체를 반환한다.
<code>putImageData()</code>	지정된 <code>ImageData</code> 객체의 이미지 데이터를 다시 캔버스에 넣는다.
<code>createImageData()</code>	새로운 빈 <code>ImageData</code> 객체를 만듭니다. 새 객체의 픽셀 값은 기본적으로 투명한 검은 색이다.

ImageData 가져오기/넣기/생성하기

- `getImageData()`

- `getImageData()` 메서드는 다음과 같이 사용할 수 있으며 인자의 내용은 다음과 같다.

```
getImageData(x, y, width, height)
```

인자	설명
x	이미지를 복사할 상단 좌측의 x 좌표
y	이미지를 복사할 상단 좌측의 y 좌표
width	x로부터 복사할 가로 길이
height	y로부터 복사할 세로 길이

ImageData 가져오기/넣기/생성하기

○ getImageData() 예제

```
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">  
Your browser does not support the HTML5 canvas tag.</canvas>
```

```
<script>  
var c = document.getElementById("myCanvas");  
var ctx = c.getContext("2d");  
ctx.fillStyle = "red";  
ctx.fillRect(10, 10, 50, 50);  
  
var imgData = ctx.getImageData(30, 30, 50, 50);  
red = imgData.data[0];  
green = imgData.data[1];  
blue = imgData.data[2];  
alpha = imgData.data[3];  
alert(red + " " + green + " " + blue + " " + alpha);  
</script>
```

이 페이지 내용:

255 0 0 255

확인

ImageData 가져오기/넣기/생성하기

- putImageData()

- putImageData() 메서드는 다음과 같이 사용할 수 있으며 인자의 내용은 다음과 같다.

```
putImageData ( imgData, x, y, dirtyX, dirtyY, dirtyWidth, dirtyHeight )
```

인자	설명
imgData	canvas에 놓을 imageData 객체.
x	imageData 객체의 좌측 상단의 x 좌표
y	imageData 객체의 좌측 상단의 y 좌표
dirtyX	캔버스에 이미지를 위치할 x 축
dirtyY	캔버스에 이미지를 위치할 y 축
dirtyWidth	캔버스에 이미지를 위치할 시 이미지의 폭
dirtyHeight	캔버스에 이미지를 위치할 시 이미지의 높이

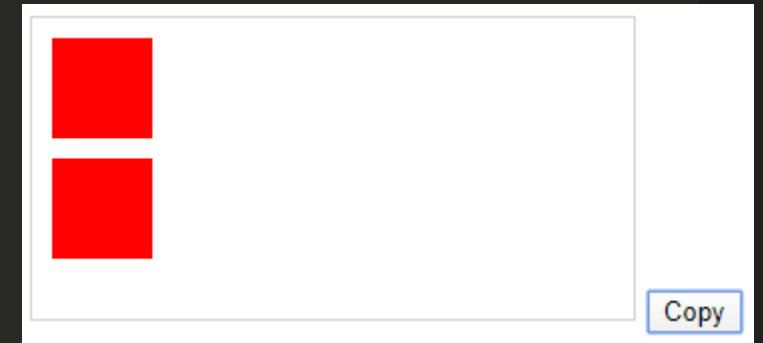
ImageData 가져오기/넣기/생성하기

○ putImageData() 예제 - 1

```
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">  
Your browser does not support the HTML5 canvas tag.</canvas>
```

```
<script type="text/javascript">  
var c = document.getElementById("myCanvas");  
var ctx = c.getContext("2d");  
ctx.fillStyle = "red";  
ctx.fillRect(10, 10, 50, 50);  
  
function copy() {  
    var imgData = ctx.getImageData(10, 10, 50, 50);  
    ctx.putImageData(imgData, 10, 70);  
}  
</script>
```

```
<button onclick="copy()">Copy</button>
```



ImageData 가져오기/넣기/생성하기

- `createImageData()`
 - `createImageData()` 메서드는 다음과 같이 사용할 수 있으며 인자의 내용은 다음과 같다.

```
createImageData(width,height);  
createImageData(imageData);
```

인자	설명
<code>width</code>	새로 선언한 <code>ImageData</code> 객체의 가로길이
<code>height</code>	새로 선언한 <code>ImageData</code> 객체의 세로길이
<code> imageData</code>	다른 <code>ImageData</code> 객체

ImageData 가져오기/넣기/생성하기

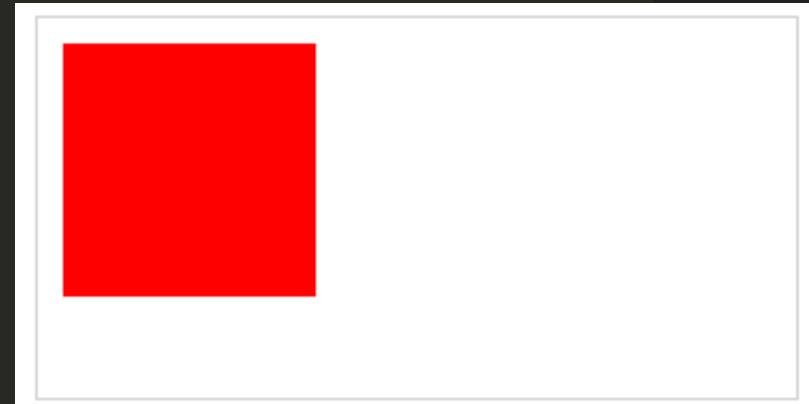
○ createImageData() 예제 - 1

```
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>
```

```
<script type="text/javascript">
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
var imgData = ctx.createImageData(100, 100);

var i;
for (i = 0; i < imgData.data.length; i += 4) {
    imgData.data[i+0] = 255;
    imgData.data[i+1] = 0;
    imgData.data[i+2] = 0;
    imgData.data[i+3] = 255;
}

ctx.putImageData(imgData, 10, 10);
</script>
```



합성

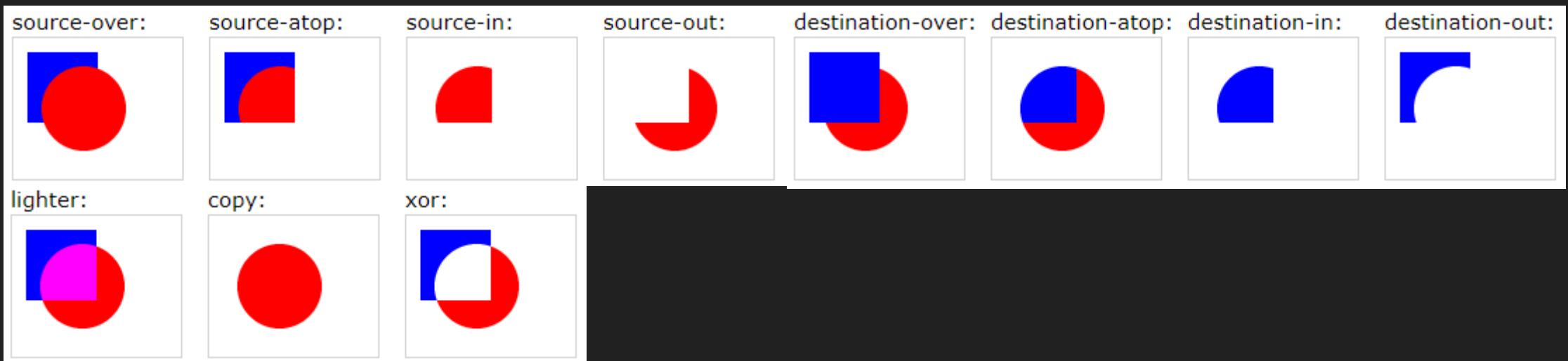
○ Porter-Duff 합성

- Porter-Duff 합성은 2개의 이미지의 합성 방법을 13종류의 패턴으로 분류한다.
- 그리고 알파 채널이라는 개념을 도입하여 픽셀 단위로 어느 지점에 알파 값을 얼마나 줄 건지 제시한다.
- 캔버스에서의 이미지 합성에서도 Porter-Duff 합성을 사용할 수 있다.
- 캔버스 컨텍스트의 globalCompositeOperation 속성을 통해 위의 효과를 설정할 수가 있다.

메서드/속성	설명
context.globalCompositeOperation	현재의 합성 패턴을 나타내는 무자열을 반환한다. 값을 지정하여 합성 패턴을 변경할 수 있다 알 수 없는 값은 무시한다.

합성

- glabalCompositeOperation
 - glabalCompositeOperation 속성 종류는 다음과 같다.

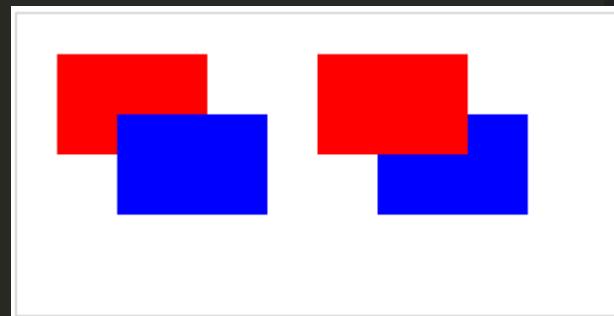


합성

○ globalCompositeOperation 예제 - 1

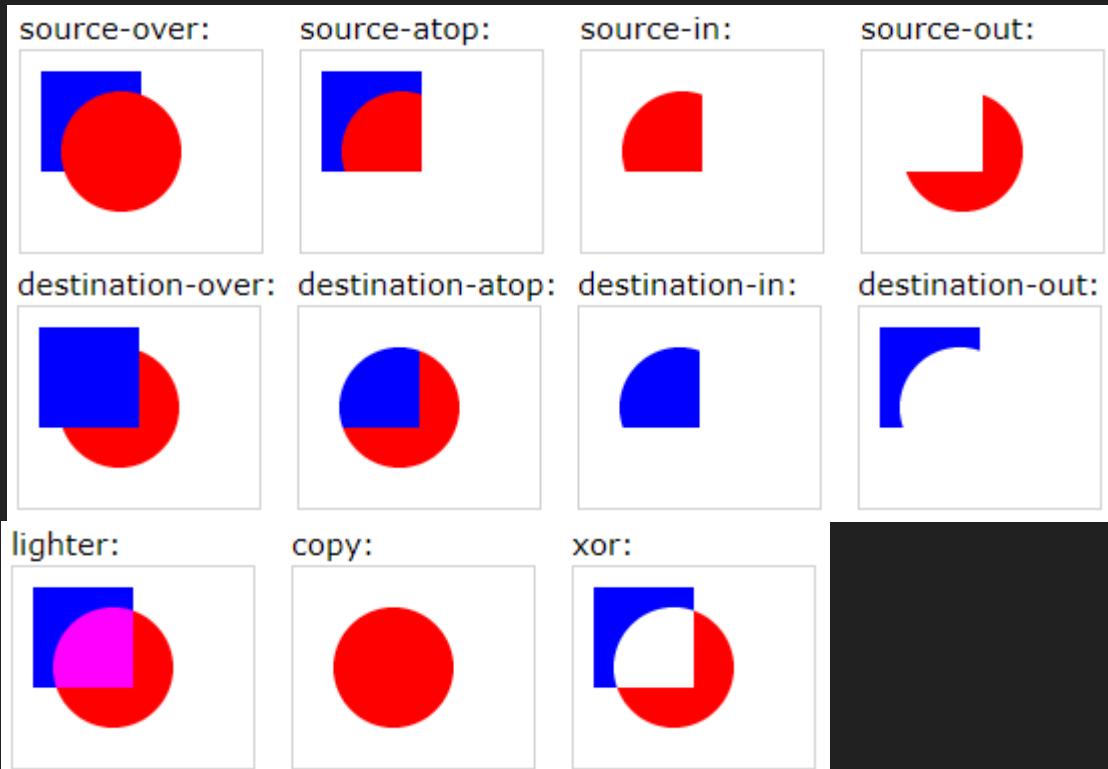
```
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>
```

```
<script type="text/javascript">
  var c = document.getElementById("myCanvas");
  var ctx = c.getContext("2d");
  ctx.fillStyle = "red";
  ctx.fillRect(20, 20, 75, 50);
  ctx.fillStyle = "blue";
  ctx.globalCompositeOperation = "source-over";
  ctx.fillRect(50, 50, 75, 50);
  ctx.fillStyle = "red";
  ctx.fillRect(150, 20, 75, 50);
  ctx.fillStyle = "blue";
  ctx.globalCompositeOperation = "destination-over";
  ctx.fillRect(180, 50, 75, 50);
</script>
```



합성

○ globalCompositeOperation 예제 - 2



```
<script type="text/javascript">
    var gco = new Array();
    gco.push("source-atop");
    gco.push("source-in");
    gco.push("source-out");
    gco.push("source-over");
    gco.push("destination-atop");
    gco.push("destination-in");
    gco.push("destination-out");
    gco.push("destination-over");
    gco.push("lighter");
    gco.push("copy");
    gco.push("xor");

    var n;
    for (n = 0; n < gco.length; n++) {
        document.write("<div id='p_" + n
            + "' style='float:left;'>" + gco[n] + ":<br>");
        var c = document.createElement("canvas");
        c.width = 120;
        c.height = 100;
        document.getElementById("p_" + n).appendChild(c);
        var ctx = c.getContext("2d");
        ctx.fillStyle = "blue";
        ctx.fillRect(10, 10, 50, 50);
        ctx.globalCompositeOperation = gco[n];
        ctx.beginPath();
        ctx.fillStyle = "red";
        ctx.arc(50, 50, 30, 0, 2 * Math.PI);
        ctx.fill();
        document.write("</div>");

    }
</script>
```

화면에 맞춰 자르기

- clip()
 - clip() 메서드는 원본 캔버스에서 모든 모양과 크기의 영역을 자른다.
 - 영역이 잘리면 이후의 모든 도면이 잘린 영역으로 제한된다 (캔버스의 다른 영역에 액세스 할 수 없음).
 - 그러나 clip() 메소드를 사용하기 전에 save() 메소드를 사용하여 현재 캔버스 영역을 저장하고 나중에 언제든지 restore() 메소드로 복원이 가능하다.

메서드/속성	설명
context.clip()	이 함수를 호출하면 그 시점에 패스로 정해진 영역을 클리핑 영역으로 지정한다.

화면에 맞춰 자르기

○ clip() 예제 - 1

```
<span>Without clip():</span>
<canvas id="myCanvas" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>
```

```
<script>
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
// Draw a rectangle
ctx.rect(50, 20, 200, 120);
ctx.stroke();
// Draw red rectangle
ctx.fillStyle = "red";
ctx.fillRect(0, 0, 150, 100);
</script>
```

```
<span>With clip():</span>
<canvas id="myCanvas2" width="300" height="150" style="border:1px solid #d3d3d3;">
Your browser does not support the HTML5 canvas tag.</canvas>
```

```
<script>
var c = document.getElementById("myCanvas2");
var ctx = c.getContext("2d");
// Clip a rectangular area
ctx.rect(50, 20, 200, 120);
ctx.stroke();
ctx.clip();
// Draw red rectangle after clip()
ctx.fillStyle = "red";
ctx.fillRect(0, 0, 150, 100);
</script>
```



변형

○ 변형을 하기 전 알아둬야 할 메서드

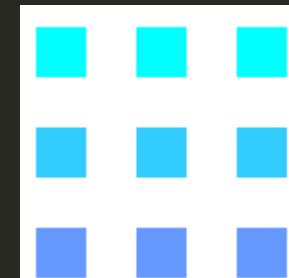
- save() : canvas의 모든 상태를 저장한다.
- restore() : 가장 최근에 저장된 canvas 상태를 복원한다.
- Canvas 상태는 스택(stack)에 쌓인다. save() 메소드가 호출될 때마다 현재 drawing 상태가 스택 위로 푸시되는데 푸시되는 값은 다음과 같다.
 - strokeStyle, fillStyle, globalAlpha, lineWidth, lineCap, lineJoin, miterLimit, lineDashOffset, shadowOffsetX, shadowOffsetY, shadowBlur, shadowColor, globalCompositeOperation, font, textAlign, textBaseline, direction, imageSmoothingEnabled.
- 앞으로 설명하게 되는 변형 또한 save() 메서드가 호출될 때 저장된다.

변형

- translate(x, y)

- 그리드에서 canvas와 그 원점을 이동한다. x는 이동시킬 수평 거리를 가리키고, y는 그리드에서 수직으로 얼마나 멀리 떨어지는지를 표시한다.

```
<canvas id="canvas" width="500" height="300"></canvas>
<script type="text/javascript">
    function draw() {
        var ctx = document.getElementById('canvas').getContext('2d');
        for (var i = 0; i < 3; i++) {
            for (var j = 0; j < 3; j++) {
                ctx.save();
                ctx.fillStyle = 'rgb(' + (51 * i) + ', ' + (255 - 51 * i) + ', 255)';
                ctx.translate(10 + j * 50, 10 + i * 50);
                ctx.fillRect(0, 0, 25, 25);
                ctx.restore();
            }
        }
        draw();
    }
</script>
```



변형

- `rotate(angle)`

- canvas를 현재 원점을 기준으로 라디안의 angle 숫자만큼 시계방향으로 회전시킨다.
- 각도의 단위는 도(degree)가 아닌 라디안(radian)이다. 변환하려면 $\text{radians} = (\text{Math.PI}/180)*\text{degrees}$. 를 사용한다.

```
<canvas id="canvas" width="800" height="500"></canvas>
<script type="text/javascript">
    function draw() {
        var ctx = document.getElementById('canvas').getContext('2d');

        // 좌측 사각형, canvas 원점에서 회전하기
        ctx.save();
        // 파란 사각형
        ctx.fillStyle = '#0095DD';
        ctx.fillRect(30, 30, 100, 100);
        ctx.rotate((Math.PI / 180) * 25);
        // 회색 사각형
        ctx.fillStyle = '#4D4E53';
        ctx.fillRect(30, 30, 100, 100);
        ctx.restore();

        // 우측 사각형, 사각형 중심에서 회전하기
        // 파란 사각형 그리기
        ctx.fillStyle = '#0095DD';
        ctx.fillRect(150, 30, 100, 100);

        ctx.translate(200, 80); // 사각형 중심으로 이동하기
                                // x = x + 0.5 * width
                                // y = y + 0.5 * height
        ctx.rotate((Math.PI / 180) * 25); // 회전
        ctx.translate(-200, -80); // 예전 위치로 이동하기

        // 회색 사각형 그리기
        ctx.fillStyle = '#4D4E53';
        ctx.fillRect(150, 30, 100, 100);
    }
draw();
```



변형

- scale(x, y)
 - canvas 단위를 수평으로 x만큼, 수직으로 y만큼 크기를 확대·축소한다.
 - 둘의 매개 변수는 실수이다.
 - 1.0보다 작은 값이면 단위의 크기를 축소하고, 1.0보다 큰 값이면 단위의 크기를 확대한다. 값이 1.0이면 단위의 크기는 그대로다.

```
<canvas id="canvas" width="800" height="500"></canvas>
<script type="text/javascript">
function draw() {
  var ctx = document.getElementById('canvas').getContext('2d');

  // 간단하지만 확대·축소 비율을 적용한 사각형 그리기
  ctx.save();
  ctx.scale(10, 3);
  ctx.fillRect(1, 10, 10, 10);
  ctx.restore();

  // 수평으로 대칭하기
  ctx.scale(-1, 1);
  ctx.font = '48px serif';
  ctx.fillText('MDN', -135, 120);
}
draw();
</script>
```



변형

- transform(a, b, c, d, e, f)
 - 확대, 축소, 비틀기, 이동을 전부 가능하게 하는 메서드

transform(a, b, c, d, e, f)

- a : 수평으로 확대·축소하기
- b : 수평으로 비틀기
- c : 수직으로 비틀기
- d : 수직으로 확대·축소하기
- e : 수평으로 이동하기
- f : 수직으로 이동하기

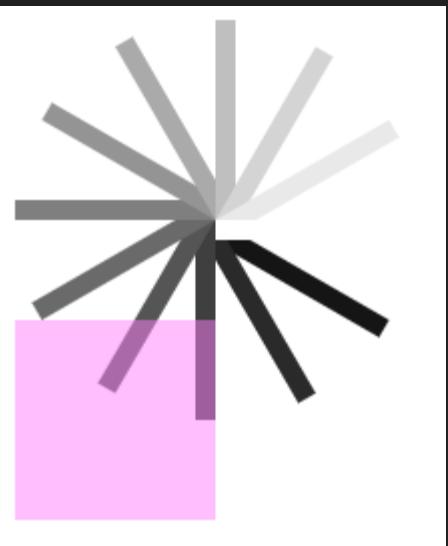
변형

- transform(a, b, c, d, e, f)
 - 관련 메서드는 다음과 같다.

속성	설명
setTransform(a, b, c, d, e, f)	현재의 변형을 무효로 한 후에 명시된 변형으로 바뀐다.
resetTransform()	transform 초기화. setTransform(1, 0, 0, 1, 0, 0)

변형

○ transform(a, b, c, d, e, f) 예제



```
<canvas id="canvas" width="1000" height="700"></canvas>
<script type="text/javascript">
function draw() {
    var ctx = document.getElementById('canvas').getContext('2d');

    var sin = Math.sin(Math.PI / 6);
    var cos = Math.cos(Math.PI / 6);
    ctx.translate(100, 100);
    var c = 0;
    for (var i = 0; i <= 12; i++) {
        c = Math.floor(255 / 12 * i);
        ctx.fillStyle = 'rgb(' + c + ', ' + c + ', ' + c + ')';
        ctx.fillRect(0, 0, 100, 10);
        ctx.transform(cos, sin, -sin, cos, 0, 0);
    }

    ctx.setTransform(-1, 0, 0, 1, 100, 100);
    ctx.fillStyle = 'rgba(255, 128, 255, 0.5)';
    ctx.fillRect(0, 50, 100, 100);
}
draw();
</script>
```

도형의 내부, 외부 구별

○ 도형 내,외부 구별

- Canvas 에서는 패스로 그려진 도형이라면 특정 좌표가 그 패스로 만들어진 도형의 안쪽에 위치하는지 바깥쪽에 위치하는지 간단하게 판별할 수 있다.

메서드/속성	설명
context.isPointInPath(x, y)	매개변수로 지정된 좌표(x,y)가 현재 패스 안에 있으면 true 그렇지 않으면 false를 반환한다.

- isPointInPath() 함수는 바꿔 말하면 지정한 좌표가 현재의 패스를 이용해서 fill() 함수를 호출했을 때 색이 칠해지는 영역이라면 true를, 그렇지만 false를 반환 한다.