Canvas Tutorial

18.1.3

(https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API/Tutorial)

content

1. Basic usage

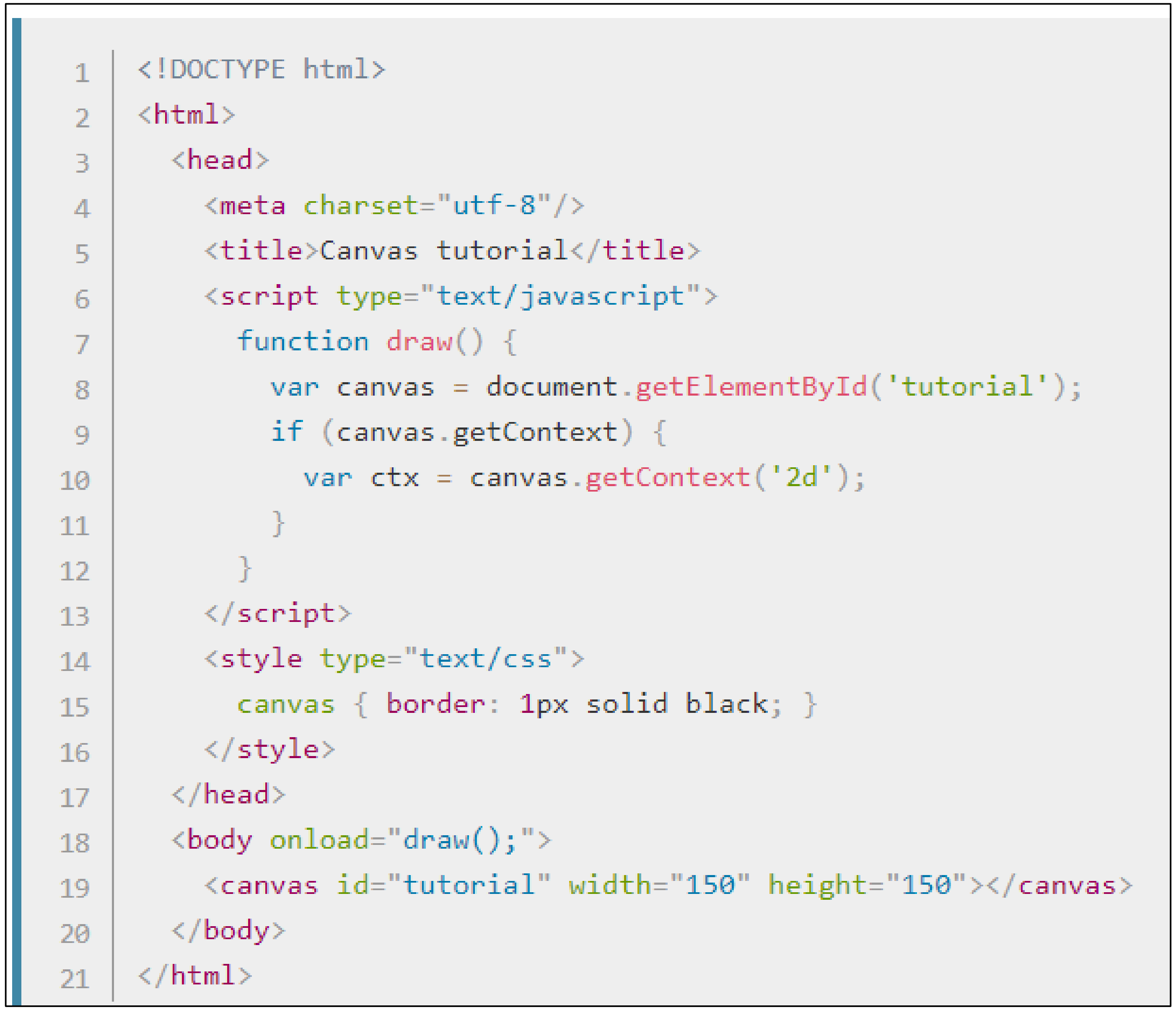
- **Start by looking at the <canvas> HTML element itself.**
- **Goal : how to set up a canvas 2D context & draw 1st example**

- **<canvas id = “tutorial” width =“150” height=“150”></canvas>**
- → canvas는 2개의 attribute (width, height)만 가지는데, 디폴트 value는 300 pixels wide, 150pixels high임
- → id attribute는 script에서 이를 찾기 쉽게 해주므로 넣어야 함
- → 다른 normal image처럼 취급 가능
- → Requird </canvas> tag

1. Basic usage

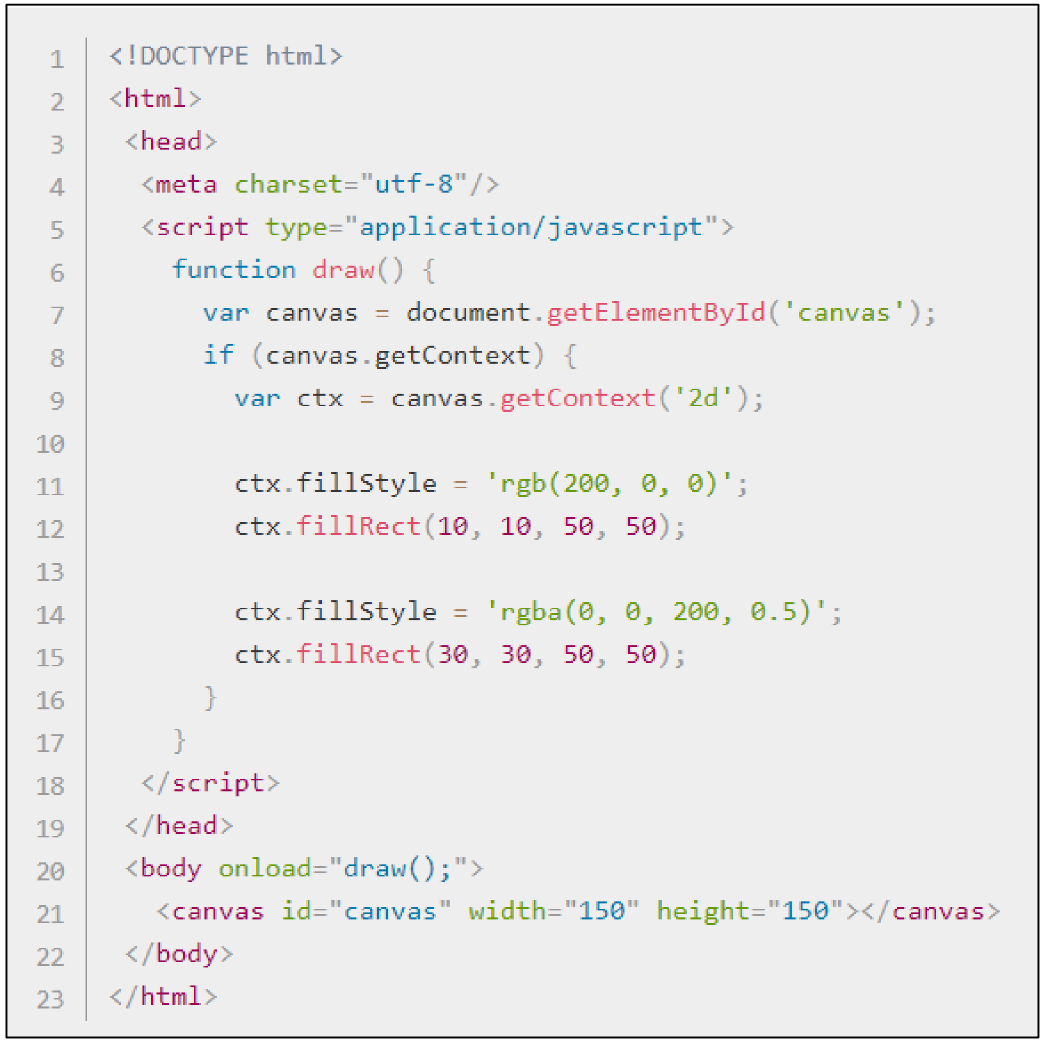
- <canvas> elemen는 getContext()라는 메소드를 가짐. (used to obtain the rendering context & its drawing functions.)

- A skeleton template → (starting point for later examples)

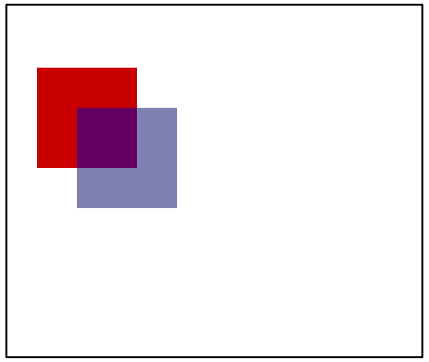


```html
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8"/>
    <title>Canvas tutorial</title>
    <script type="text/javascript">
      function draw() {
        var canvas = document.getElementById('tutorial');
        if (canvas.getContext) {
          var ctx = canvas.getContext('2d');
        }
      }
    </script>
    <style type="text/css">
      canvas { border: 1px solid black; }
    </style>
  </head>
  <body onload="draw();">
    <canvas id="tutorial" width="150" height="150"></canvas>
  </body>
</html>
```

1. Basic usage



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

        ctx.fillStyle = 'rgb(200, 0, 0)';
        ctx.fillRect(10, 10, 50, 50);

        ctx.fillStyle = 'rgba(0, 0, 200, 0.5)';
        ctx.fillRect(30, 30, 50, 50);
      }
    }
  </script>
 </head>
 <body onload="draw();">
   <canvas id="canvas" width="150" height="150"></canvas>
 </body>
</html>
```

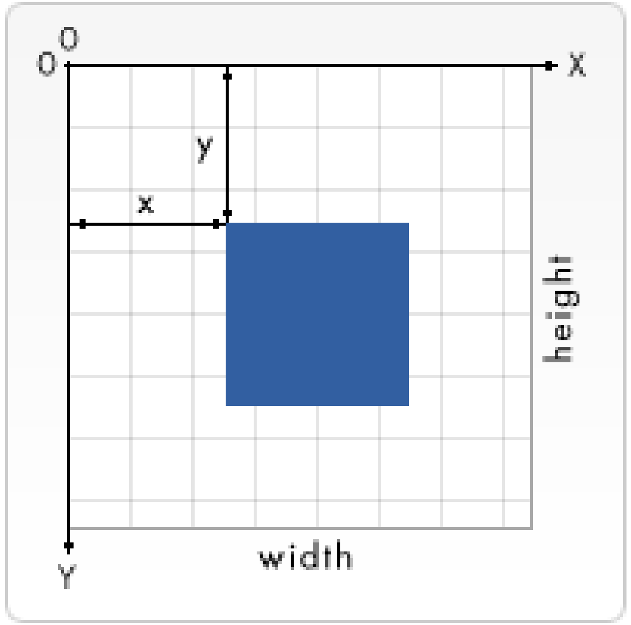


- A simple example → draws two intersecting rectangles, one of which has alpha transparency.

2. Drawing shapes

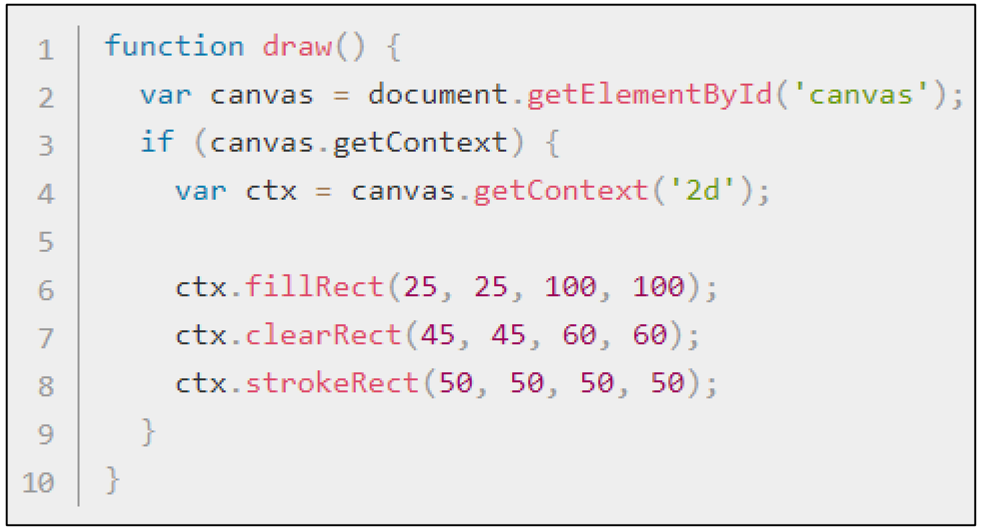
- Get into the details of how to draw on the canvas.
- Goals : how to draw rectangles, triangles, lines, arcs and curves, providing familiarity with some of the basic shapes.

- The grid.(→)
  - Top left가 (0,0). 모든 unit은 pixel단위. 150 픽셀, 150픽셀이 디폴트인것.

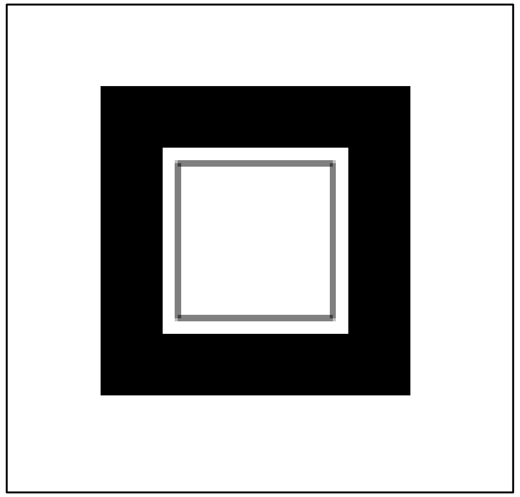


2. Drawing shapes

- Drawing rectangles.
  - <canvas>는 한 가지 primitive shape만 제공 : rectangles.
  - Canvas에 rectangle을 그리는 function은 3개 존재
    - fillRect(x,y,width,height)
    - strokeRect(x,y,width,height)
    - clearRect(x,y,width,height)

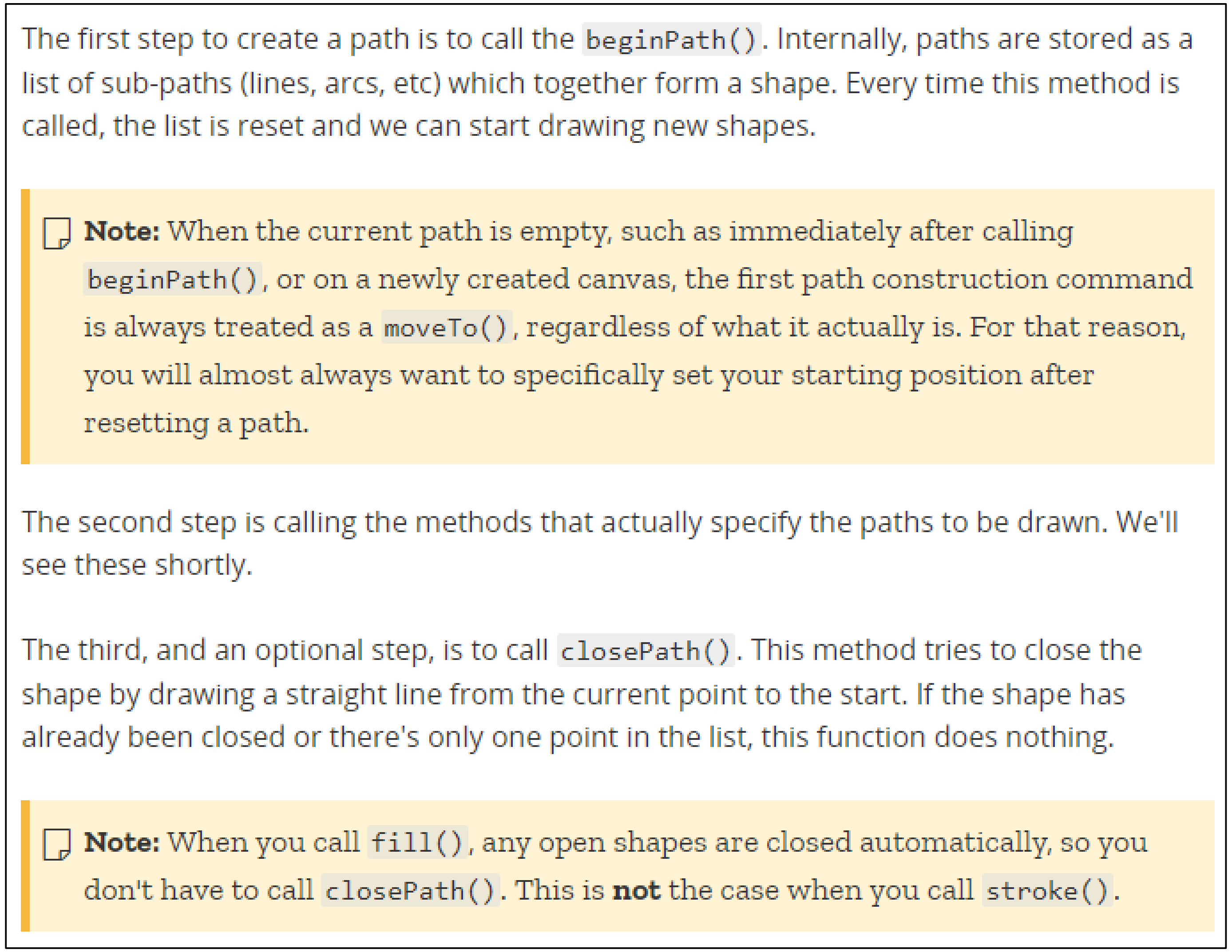


```
function draw() {
  var canvas = document.getElementById('canvas');
  if (canvas.getContext) {
    var ctx = canvas.getContext('2d');

    ctx.fillRect(25, 25, 100, 100);
    ctx.clearRect(45, 45, 60, 60);
    ctx.strokeRect(50, 50, 50, 50);
  }
}
```



2. Drawing shapes

- Drawing paths.
  - 필요해졌을 때 나중에 정리



The first step to create a path is to call the `beginPath()`. Internally, paths are stored as a list of sub-paths (lines, arcs, etc) which together form a shape. Every time this method is called, the list is reset and we can start drawing new shapes.

Note: When the current path is empty, such as immediately after calling `beginPath()`, or on a newly created canvas, the first path construction command is always treated as a `moveTo()`, regardless of what it actually is. For that reason, you will almost always want to specifically set your starting position after resetting a path.

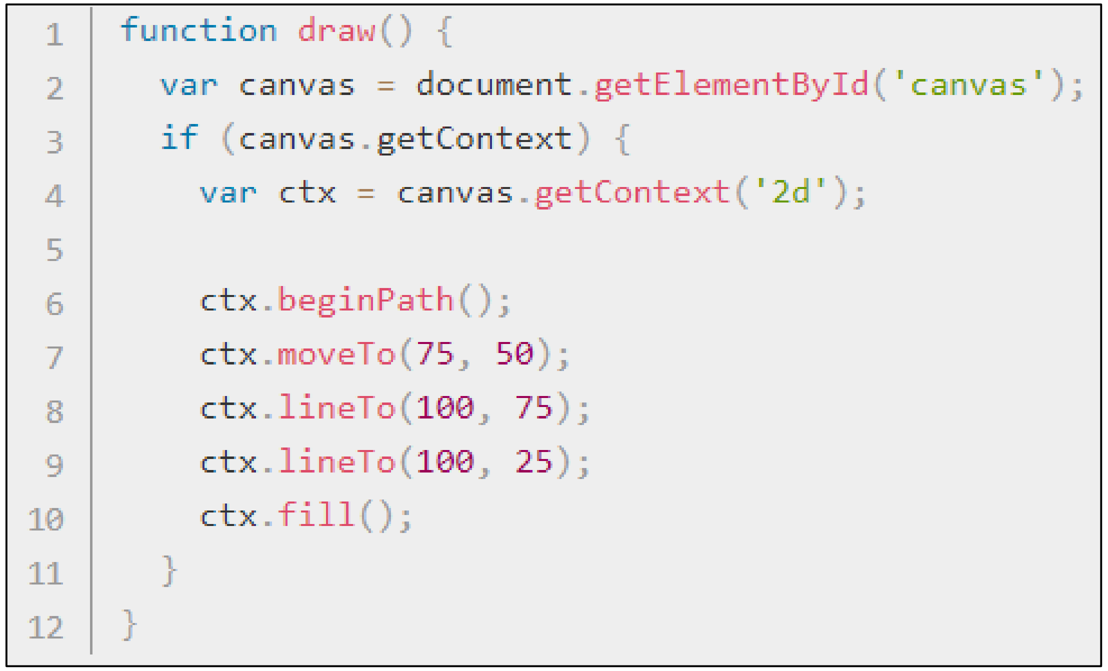
The second step is calling the methods that actually specify the paths to be drawn. We'll see these shortly.

The third, and an optional step, is to call `closePath()`. This method tries to close the shape by drawing a straight line from the current point to the start. If the shape has already been closed or there's only one point in the list, this function does nothing.

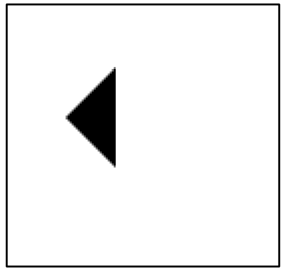
Note: When you call `fill()`, any open shapes are closed automatically, so you don't have to call `closePath()`. This is **not** the case when you call `stroke()`.

2. Drawing shapes

- Drawing a triangle

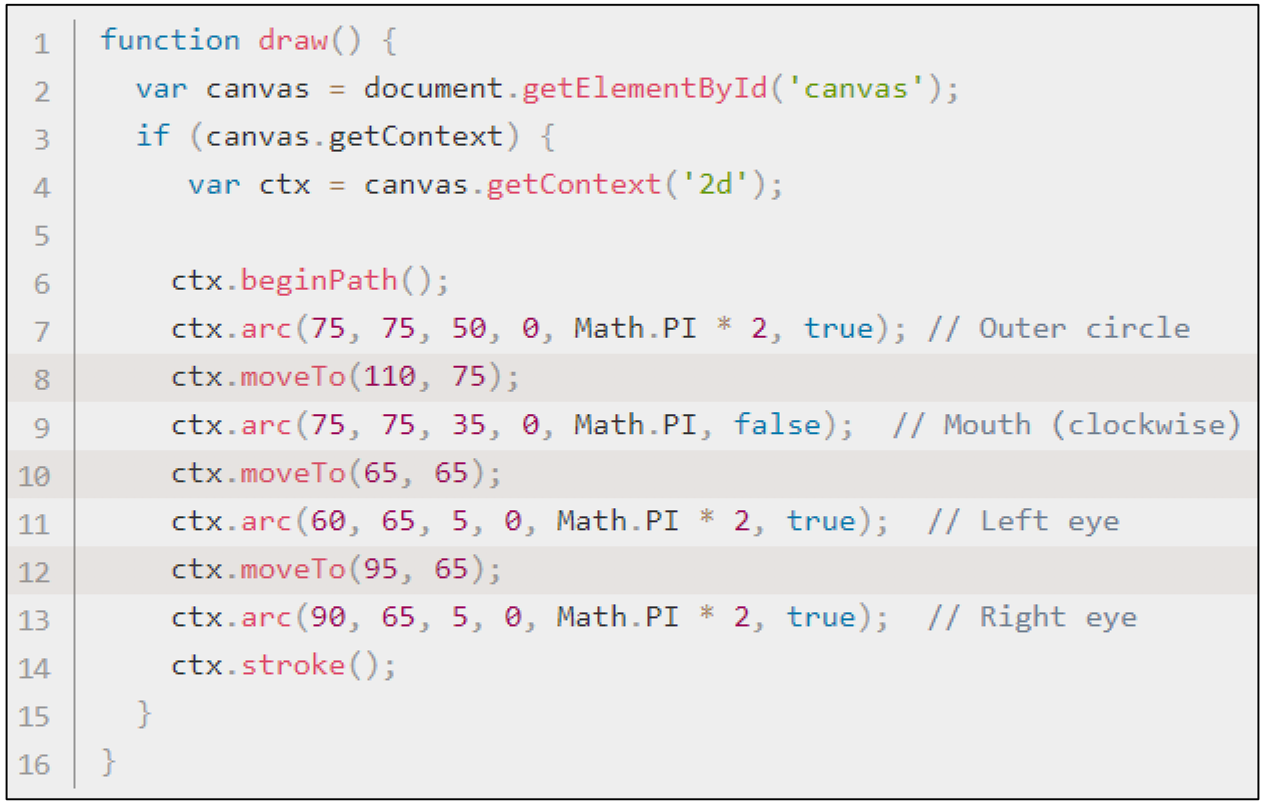


```
function draw() {
  var canvas = document.getElementById('canvas');
  if (canvas.getContext) {
    var ctx = canvas.getContext('2d');

    ctx.beginPath();
    ctx.moveTo(75, 50);
    ctx.lineTo(100, 75);
    ctx.lineTo(100, 25);
    ctx.fill();
  }
}
```

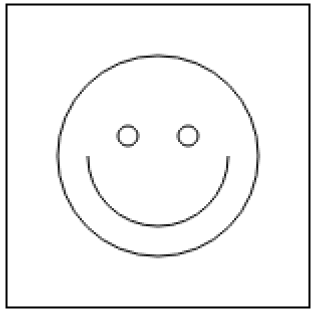


2. Drawing shapes

- moveTo(x,y)
  - Moves the pen to the coordinates specified by x and y
  - beginPath() 호출한 다음에 쓰거나 또는, unconnected path들을 그리는 데에도 쓸 수 있다 →



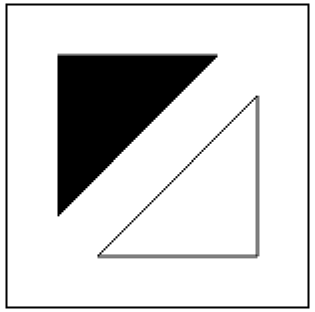
```
function draw() {
  var canvas = document.getElementById('canvas');
  if (canvas.getContext) {
     var ctx = canvas.getContext('2d');

    ctx.beginPath();
    ctx.arc(75, 75, 50, 0, Math.PI * 2, true); // Outer circle
    ctx.moveTo(110, 75);
    ctx.arc(75, 75, 35, 0, Math.PI, false);  // Mouth (clockwise)
    ctx.moveTo(65, 65);
    ctx.arc(60, 65, 5, 0, Math.PI * 2, true);  // Left eye
    ctx.moveTo(95, 65);
    ctx.arc(90, 65, 5, 0, Math.PI * 2, true);  // Right eye
    ctx.stroke();
  }
}
```



2. Drawing shapes

- Lines
- lineTo(x,y)
  - draws a line from the current drawing position to the position specified by x and y (x,y : line’s end point)
  - Starting point는 ‘end point of the previous path is the starting point for the following.’, Starting point는 moveTo() method를 써서도 변하게 할 수 있음.

2. Drawing shapes



Note: When you call `fill()`, any open shapes are closed automatically, so you don't have to call `closePath()`. This is **not** the case when you call `stroke()`.

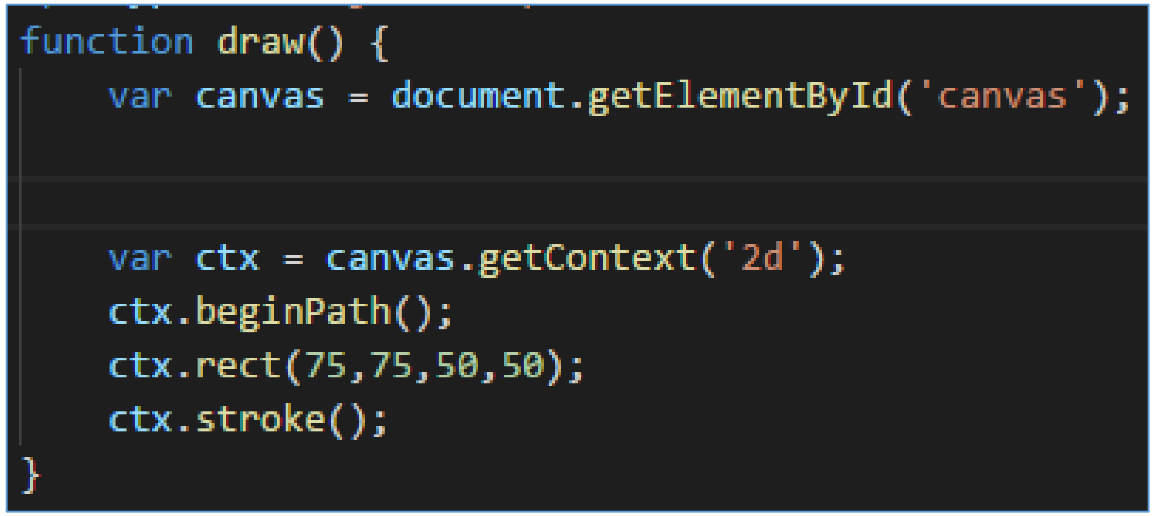
```
function draw() {
  var canvas = document.getElementById('canvas');
  if (canvas.getContext) {
    var ctx = canvas.getContext('2d');

    // Filled triangle
    ctx.beginPath();
    ctx.moveTo(25, 25);
    ctx.lineTo(105, 25);
    ctx.lineTo(25, 105);
    ctx.fill();

    // Stroked triangle
    ctx.beginPath();
    ctx.moveTo(125, 125);
    ctx.lineTo(125, 45);
    ctx.lineTo(45, 125);
    ctx.closePath();
    ctx.stroke();
  }
}
```

2. Drawing shapes

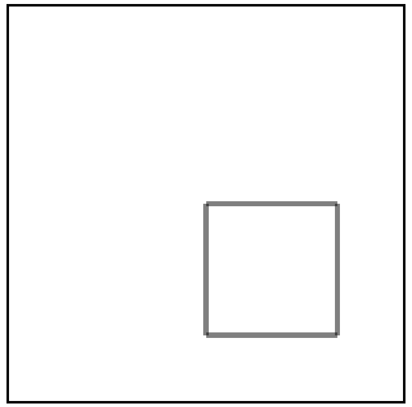
- Arc
  - 생략
- Bezier and quadratic curves
  - 생략

2. Drawing shapes

- Rectangles
  - Rect(x,y,width,height) – Draws a rectangle whose top-left corner is specified by (x,y) with the specified width and height.
  - 위 메소드가 실행될 때 moveTo() 메소드는 자동적으로 (0,0)매개변수와 함께 호출됨. → 현재 펜의 위치가 자동적으로 디폴트 좌표로 리셋되는 것



```
function draw() {
    var canvas = document.getElementById('canvas');


    var ctx = canvas.getContext('2d');
    ctx.beginPath();
    ctx.rect(75,75,50,50);
    ctx.stroke();
}
```

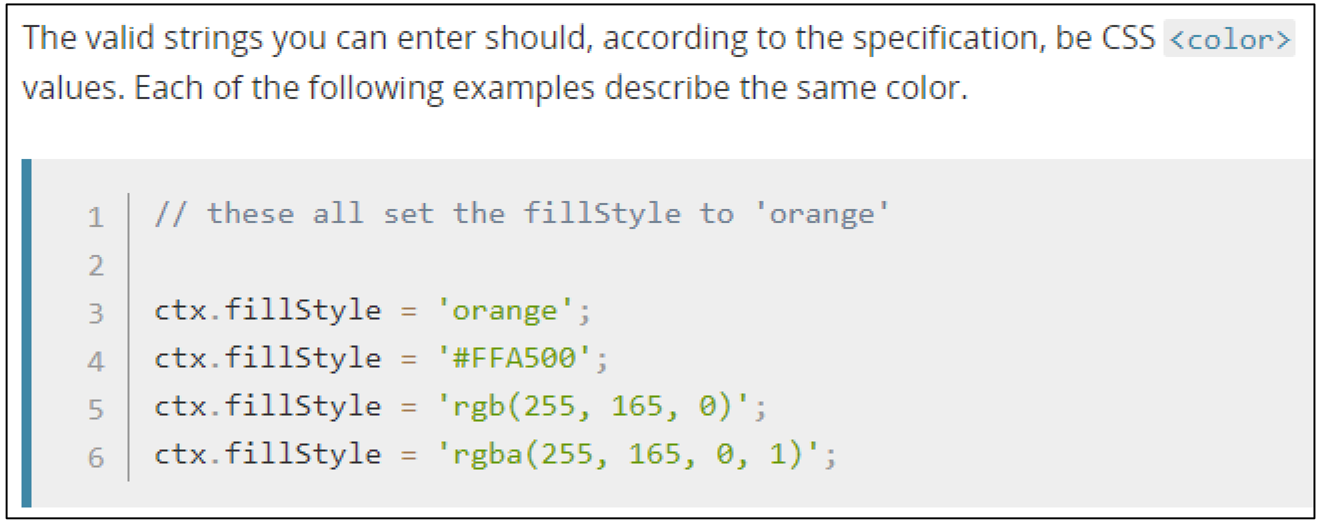


2. Drawing shapes

- Path2D example : 생략

3. Applying styles and colors

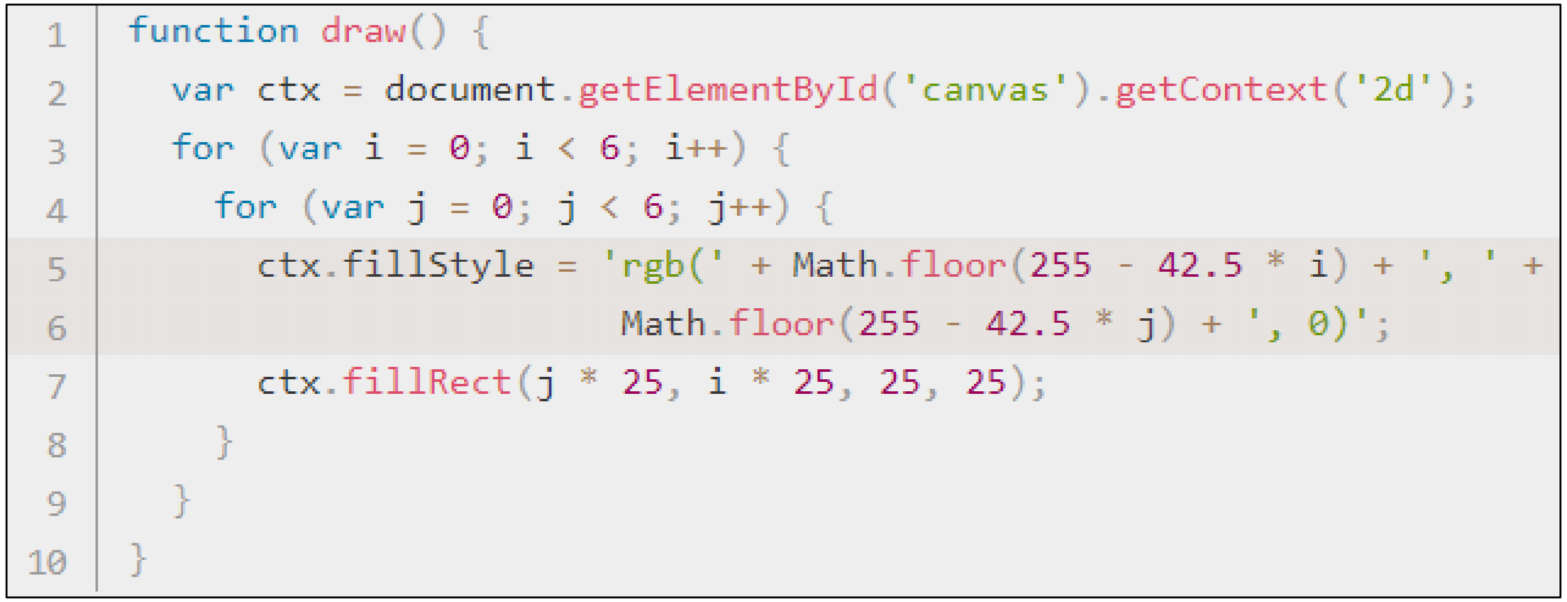
- 이전까지는 디폴트 line/fill 스타일을 사용함.
- Shape에 color를 적용하기 위해서는 2가지 중요한 프로퍼티가 있다
  - fillStyle / strokeStyle
  - fillStyle = color (Sets the style used when filling shapes.)
  - strokeStyle = color (Sets the style for shapes' outlines.)


The valid strings you can enter should, according to the specification, be CSS `<color>` values. Each of the following examples describe the same color.

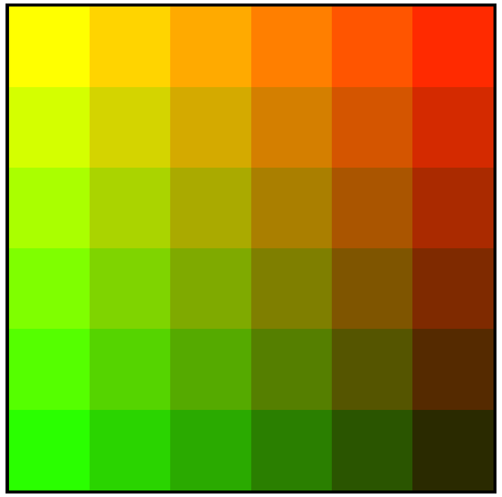
```
// these all set the fillStyle to 'orange'

ctx.fillStyle = 'orange';
ctx.fillStyle = '#FFA500';
ctx.fillStyle = 'rgb(255, 165, 0)';
ctx.fillStyle = 'rgba(255, 165, 0, 1)';
```

3. Applying styles and colors

- A fillStyle example

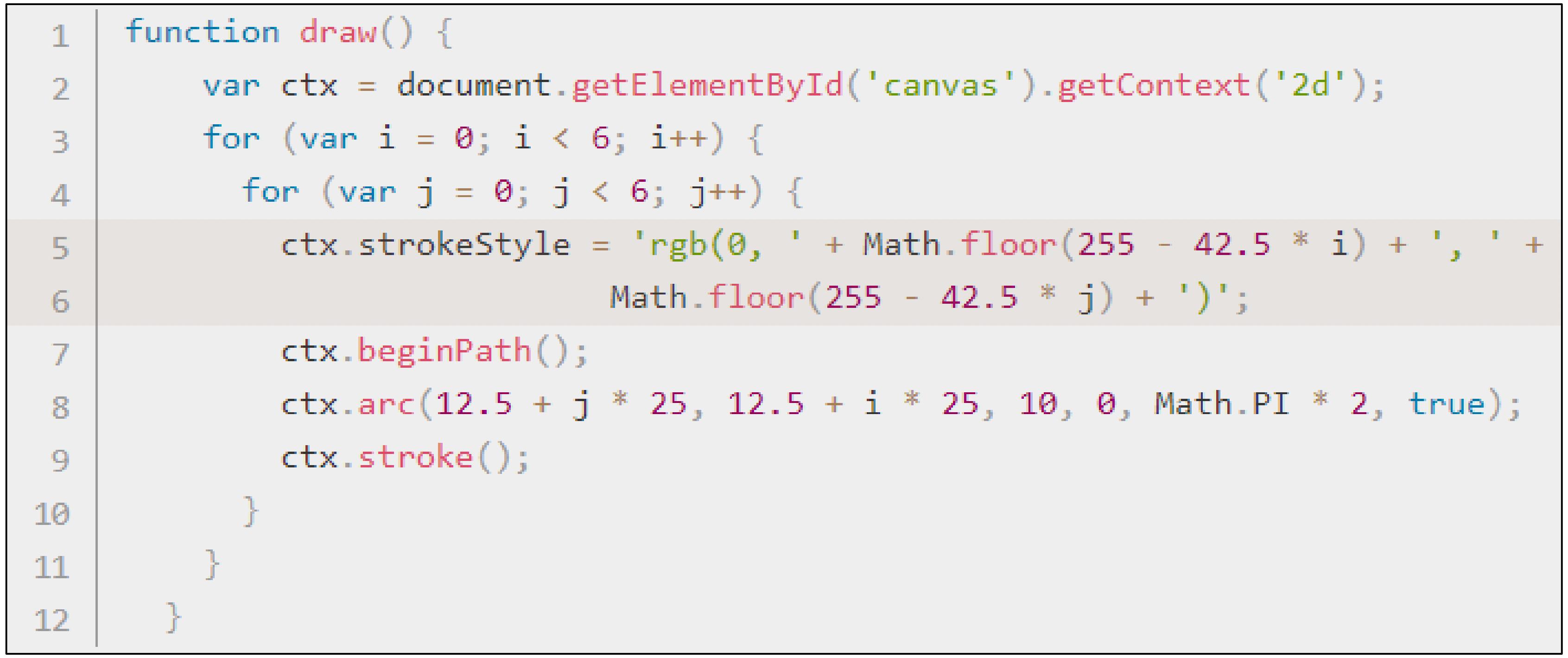


```
function draw() {
  var ctx = document.getElementById('canvas').getContext('2d');
  for (var i = 0; i < 6; i++) {
    for (var j = 0; j < 6; j++) {
      ctx.fillStyle = 'rgb(' + Math.floor(255 - 42.5 * i) + ', ' +
                       Math.floor(255 - 42.5 * j) + ', 0)';
      ctx.fillRect(j * 25, i * 25, 25, 25);
    }
  }
}
```

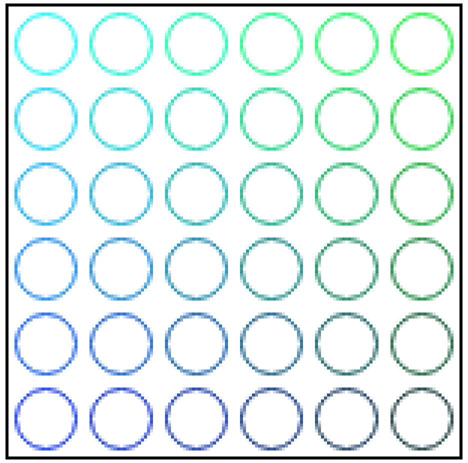


3. Applying styles and colors

- A strokeStyle example



```
function draw() {
    var ctx = document.getElementById('canvas').getContext('2d');
    for (var i = 0; i < 6; i++) {
      for (var j = 0; j < 6; j++) {
        ctx.strokeStyle = 'rgb(0, ' + Math.floor(255 - 42.5 * i) + ', ' +
                          Math.floor(255 - 42.5 * j) + ')';
        ctx.beginPath();
        ctx.arc(12.5 + j * 25, 12.5 + i * 25, 10, 0, Math.PI * 2, true);
        ctx.stroke();
      }
    }
  }
```



3. Applying styles and colors

- Transparency 부터 생략?