11

HTML5 캔버스 그래픽

강의 목표

1. HTML5에서 웹 페이지에 그래픽으로 그림을 그리는 방법을 안다.
2. <canvas> 태그로 웹 페이지에 캔버스를 만들 수 있다.
3. 캔버스에 직선, 사각형, 원호 등의 도형을 그릴 수 있다.
4. 경로(path)에 대한 개념을 이해하고 경로를 이용한 그리기를 할 수 있다.
5. 캔버스에 사각형, 원호 등의 닫힌 도형의 내부를 칠할 수 있다.
6. 캔버스에 텍스트를 출력할 수 있다.
7. 캔버스에 이미지를 출력할 수 있다.
8. 마우스 드래깅으로 캔버스에 그림을 그리는 응용프로그램을 작성할 수 있다.

HTML5와 캔버스

- 웹 페이지에 그래픽을 출력하는 기존의 2 방법
  - 사진이나 그림 : <img> 태그 이용
  - 그래픽 : 자바 애플릿이나 플래시 등 플러그인 이용
- HTML5에서 캔버스 도입
  - 도입 배경
    - 플러그인 없이 자바스크립트 코드로 웹 페이지에 자유롭게 그래픽
    - 모바일 단말기/PC를 포함하여 HTML5 표준 브라우저에서 작동
  - 그래픽 기능
    - 선, 원, 사각형, 곡선, 이미지, 2차원 문자
    - 이미지 합성 및 변환
  - 활용
    - 웹 페이지에 실시간 그래프, 애니메이션, 대화형 게임, 지도
  - 의미
    - 웹이 문서를 보여주는 수준을 넘어 응용 프로그램으로 진화하는 계기

<canvas> 태그

- 웹 페이지에 캔버스 영역 만들기

```
<canvas id="캔버스 객체 id"
        style="CSS3 스타일 시트"
        width="캔버스 영역의 폭"
        height="캔버스 영역의 폭">
    이 태그를 지원하지 않는 브라우저가 출력할 HTML 텍스트
</canvas>
```

- width, height : 캔버스가 만들어지는 영역의 크기(픽셀 단위)로 생략 가능. 각각 디폴트 300, 150 픽셀

  - 웹 페이지에 빈 캔버스 공간 할당
  - 예) 300x400 크기의 캔버스 공간 할당

```
<canvas id="grade" width="300" height="150">
        <p>canvas가 지원되지 않네요. 죄송합니다.</p>
</canvas>
```

예제 11-1 캔버스 만들기

<canvas> 태그를 이용하여 3 개의 캔버스를 가진 HTML 페이지를 만들어라.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>캔버스 만들기</title>
</head>
<body>
<h3>3 개의 캔버스 만들기</h3>
<hr>
<canvas id="canvas1" width="150" height="100"
   style="background-color:lightblue"></canvas>
<canvas id="canvas2" width="150" height="100"
   style="background-color:violet"></canvas><br>
<canvas id="canvas3" width="300" height="150"
   style="background-color:yellow"></canvas>
</body>
</html>
```

캔버스(canvas) 객체 다루기

- 캔버스 공간 할당

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

- 캔버스 객체 찾기

```
let canvas = document.getElementById("myCanvas");
```

- 캔버스 컨텍스트 얻어내기 : 캔버스에 그림 그리는 도구(컨텍스트) 얻어내기

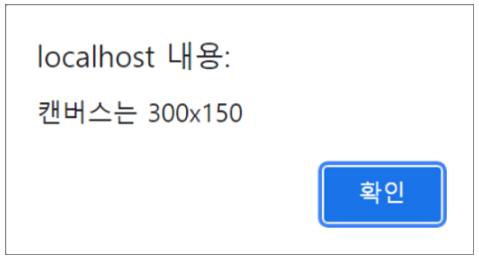
```
let context = canvas.getContext("2d");
```

- 캔버스에 사각형 그리기

```
context.rect(60, 60, 50, 50);      // context에 (60, 60)에서 50x50 크기의 사각형 그리기
context.strokeStyle = "blue";      // 선 색을 파란 색으로 설정
context.stroke();                  // context 내에 구성된 도형을 캔버스에 그린다.
```

- 캔버스의 크기, canvas의 width와 height 프로퍼티

```
let canvas = document.getElementById("myCanvas");
let width = canvas.width;
let height = canvas.height;
alert("캔버스는 " + width + "x" + height);
```

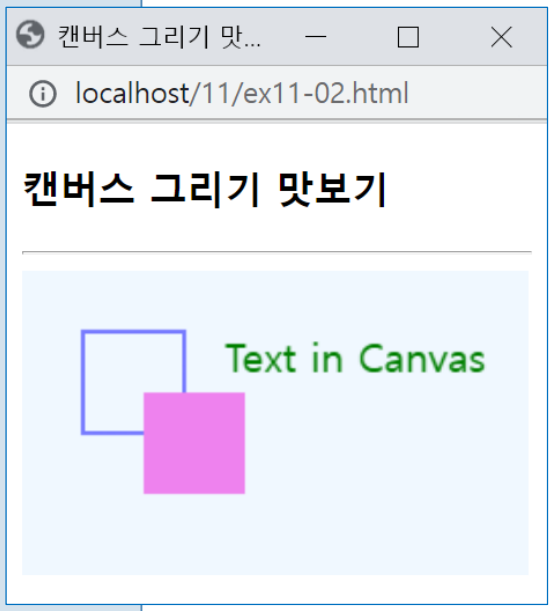


- 캔버스의 스타일 제어

```
canvas.style.backgroundColor ="yellowgreen";
```

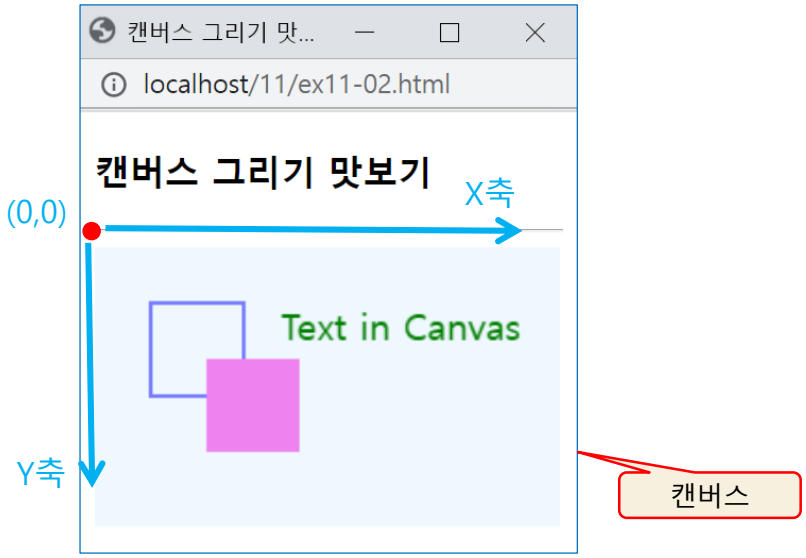
예제 11-2 캔버스 그리기 맛보기


```
<!DOCTYPE html>
<html><head><meta charset="utf-8"><title>캔버스 그리기 맛보기</title></head>
<body>
<h3>캔버스 그리기 맛보기</h3>
<hr>
<canvas id="myCanvas" style="background-color:aliceblue"
    width="250" height="150"></canvas>
<script>
   let canvas = document.getElementById("myCanvas");
   let context = canvas.getContext("2d");

   // 파란선으로 사각형 그리기
   context.beginPath(); // 빈 경로 만들기
   context.strokeStyle = "blue"; // 선 색  설정
   context.rect(30, 30, 50, 50); // (30,30)에서 50x50 크기 사각형을 경로에 삽입
   context.stroke(); // 경로에 있는 모든 도형의 외곽선 그리기

   // violet 색으로 채운 사각형 그리기
   context.beginPath(); // 빈 경로 만들기
   context.fillStyle = "violet"; // 채우기 색
   context.rect(60, 60, 50, 50); // (60,60)에서 50x50 크기 사각형을 경로에 삽입
   context.fill(); // 경로에 있는 모든 도형의 내부만 채워 그리기

   // green 색으로 텍스트 내부만 그리기
   context.font = "20px Gothic";
   context.fillStyle = "green"; // 채우기 색
   context.fillText("Text in Canvas", 100, 50); // 텍스트를 경로에 넣지 않고 바로  그리기
</script></body></html>
```



캔버스 그래픽 좌표와 그래픽 기능

- 캔버스 그래픽은 픽셀 단위의 좌표와 크기



- 그래픽 기능
  - 도형 그리기와 칠하기
  - 글자 그리기
  - 이미지 그리기
  - 이미지 변환
  - 클리핑

도형 그리기

- 도형 그리기
  - 캔버스가 지원하는 도형의 종류
    - 직선, 사각형, 원호
    - 외곽선 그리기, 내부색으로 채워 그리기 모두 가능
  - **도형 그리는 과정**
    - 경로(path) 만들기
    - 캔버스에 경로에 담긴 도형 모두 그리기
- 경로(path) 만들기
  - 그리고자 하는 도형들을 컨텍스트 내 경로에 담는 과정
- 캔버스에 도형 그리는 순서
  - beginPath() - 새로운 빈 경로 만들기
  - moveTo(), lineTo(), rect(), arc() - 경로에 도형 담기
  - stroke() – 경로 속의 도형을 캔버스에 그리기

도형 그리기 사례

- 경로 만들기

```
context.beginPath();                // 빈 경로 구성
```

- 시작점 설정

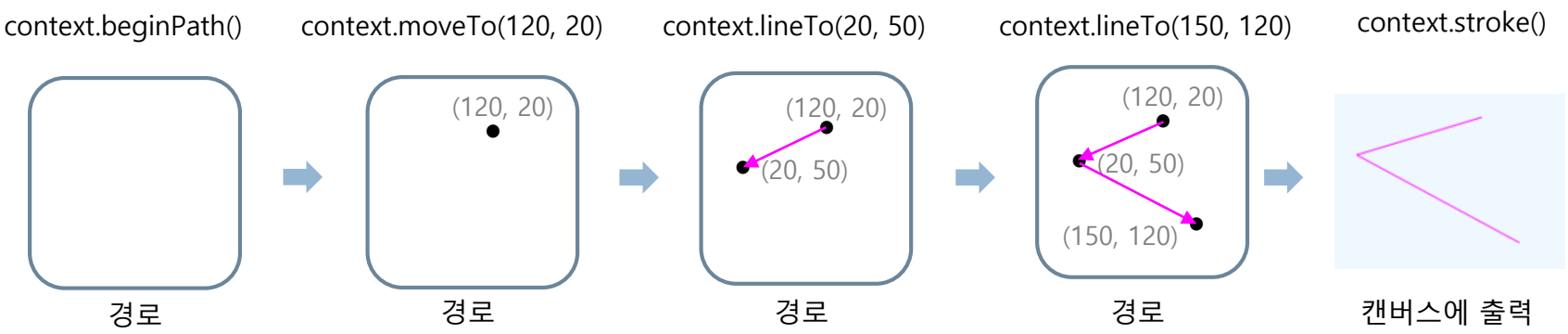
```
context.moveTo(120, 20);            // (120, 20)을 시작점으로 설정
```

- 경로에 도형 추가

```
context.lineTo(20, 50);             // (120, 20)에서 (20, 50)까지의 직선을 경로에 추가
context.lineTo(150, 120);           // (20, 50)에서 (150, 120)까지의 직선을 경로에 추가
```

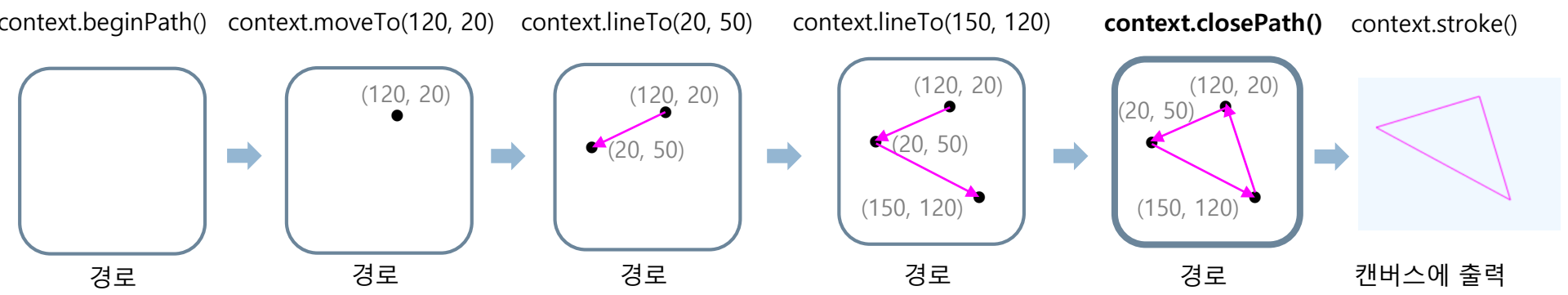
- 캔버스에 그리기

```
context.stroke();                   // context의 경로 속 도형들을 캔버스에 모두 그린다.
```



경로 닫기

- closePath()
  - 경로의 마지막 점과 시작점을 연결하는 직선 추가
  - 더 이상 경로에 새로운 도형을 추가할 수 없음
    - beginPath()를 호출하면 새로운 경로 시작



선 그리기와 사각형 그리기

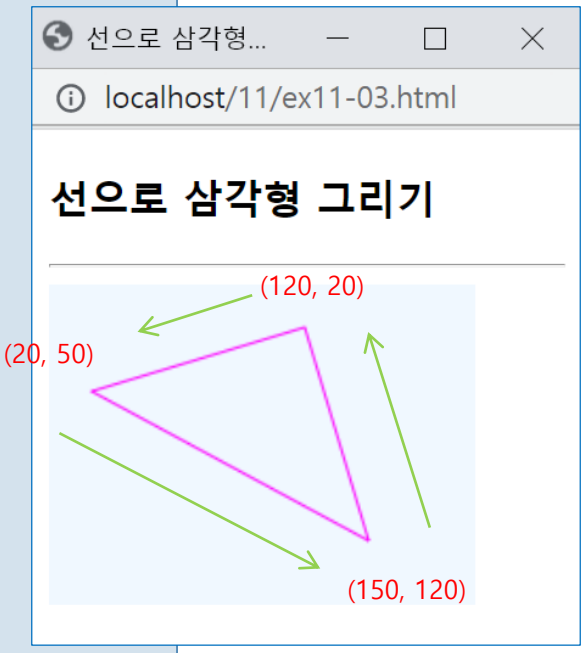
- 선 그리는 컨텍스트 메소드

moveTo(x, y)	경로에 담긴 도형은 그대로 두고, 점 (*x*, *y*)를 새 시작점으로 삽입한다.
lineTo(x, y)	경로의 끝 점에서 (*x*, *y*)까지 직선을 경로에 추가한다.

- 선그리기
  - moveTo()를 이용하여 시작점을 설정하고, lineTo()로 선을 연결해 나간다.
  - lineTo(x, y)에 지정한 (x,y)가 끝점이 된다.

예제 11-3 선으로 삼각형 그리기


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>선으로 삼각형 그리기</title>
</head>
<body>
<h3>선으로 삼각형 그리기</h3>
<hr>
<canvas id="myCanvas"
        style="background-color:aliceblue"
        width="200" height="150"></canvas>
<script>
   let canvas = document.getElementById("myCanvas");
   let context = canvas.getContext("2d");

   context.beginPath(); // 빈 경로 만들기
   context.moveTo(120, 20); // (120, 20)을 시작점으로 설정
   context.lineTo(20, 50); // 경로에 (120, 20)에서 (20, 50)까지 직선 추가
   context.lineTo(150, 120); // 경로에 (20, 50)에서 (150, 120)까지 직선 추가
   context.lineTo(120, 20); // 경로에 (150, 120)에서 (120, 20)까지 직선 추가
   context.strokeStyle="magenta"; // 선의 색
   context.stroke(); // 경로를 캔버스에 그린다
</script>
</body>
</html>
```



원호 그리기

- 원호를 그리는 arc() 메소드

arc(x, y, radius, startAngle, endAngle, anticlockwise)

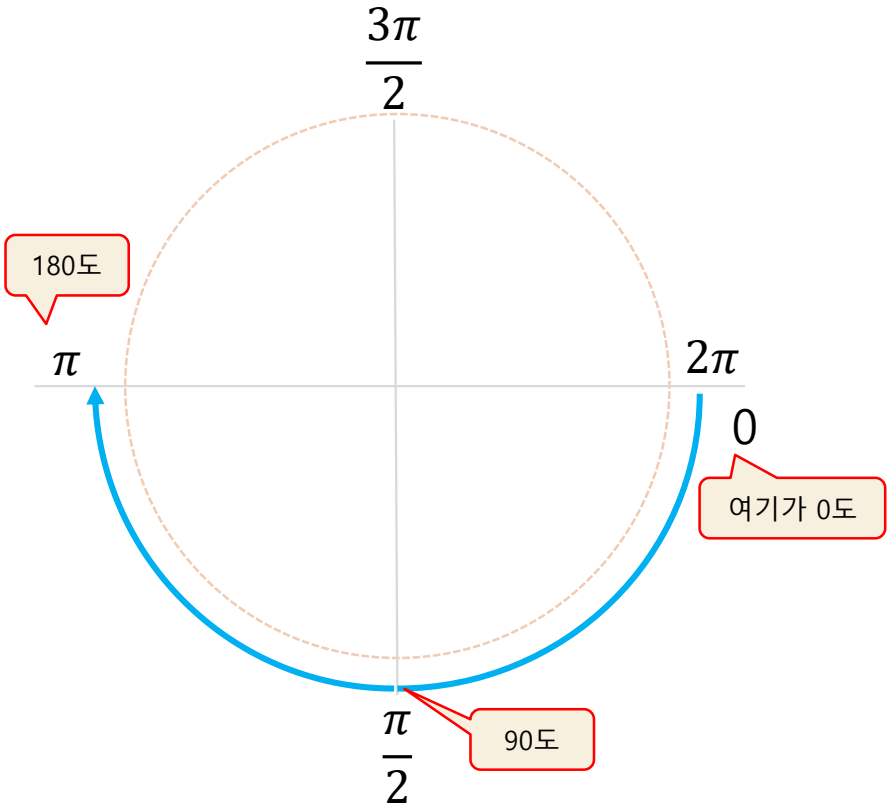
- x, y, radius : (x, y)는 원호의 중심이고 radius는 반지름
- startAngle : 원호의 시작 각도. 3시를 기점으로 시계방향으로 각도 계산
- endAngle : 원호의 끝 각도. 3시를 기점으로 시계방향으로 각도 계산
- anticlockwise : true이면 반시계방향, false이면 시계방향으로 원호 그리기. 생략가능하며 디폴트는 시계방향(false)

(x, y)를 중심으로 반지름이 radius이고, startAngle(시작 각도) 지점과 endAngle(끝 각도) 지점을 연결하는 원호를 그린다. 원호는 anticlockwise의 값에 따라 반시계 방향이나 시계방향 중 하나로 그려진다.

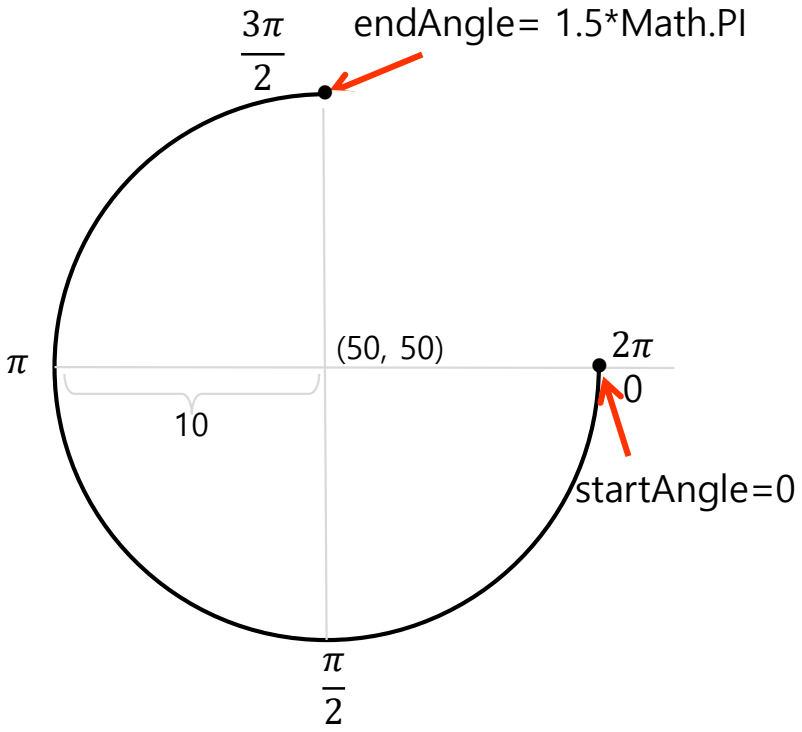
- 각도는 3시에서 0도 시작
- 각도는 원주율로 표현
  - 360°는 2π, 90°는 $\pi/2$이고, 180°는 π
  - 270° -> $3\pi/2$ 는 자바스크립트 코드 1.5*Math.PI 로 표현

원주률 **r = n*Math.PI/180**, 여기서 n은 각도(0~360)

startAngle(시작 각도)와 endAngle(끝 각도)



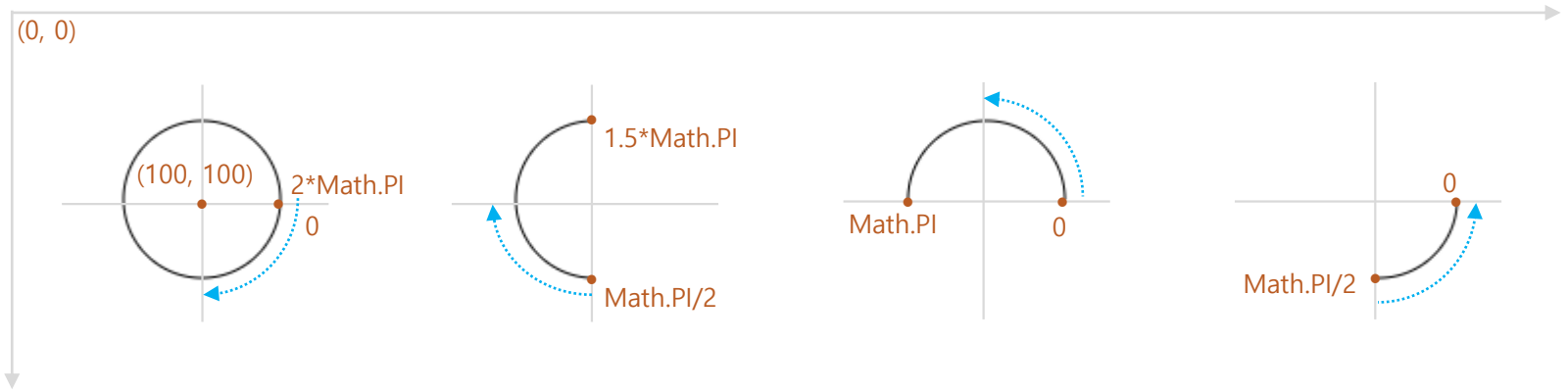
(a) 각도는 3시에서 시계방향으로 계산




```
context.arc(50, 50, 10, 0, 1.5*Math.PI, false);
context.stroke(); // 원호를 그린다.
```

(b) 원호 그리기 사례

arc() 메소드로 그린 원호 사례



context.arc(100, 100, 40, 0, 2*Math.PI, **false**);

context.arc(300, 100, 40, Math.PI/2, 1.5*Math.PI);

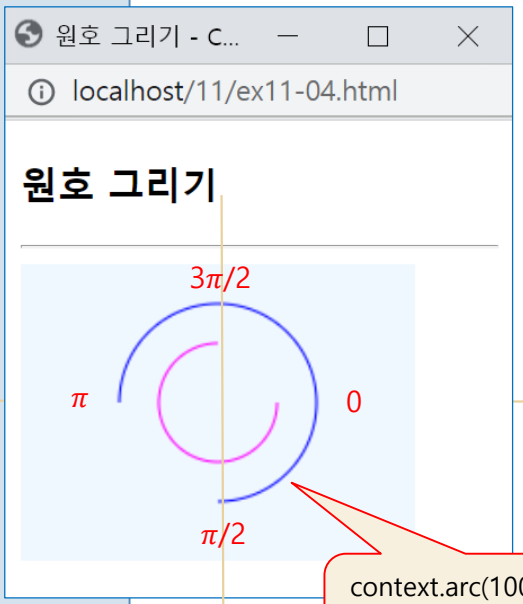
context.arc(500, 100, 40, 0, Math.PI, **true);**

context.arc(700, 100, 40, Math.PI/2, 0, **true);**

* true 이면, 시계방향, false이면 반시계 방향. 생략되면 true

예제 11-4 원호 그리기


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>원호 그리기</title>
</head>
<body>
<h3>원호 그리기</h3>
<hr>
<canvas id="myCanvas" style="background-color:aliceblue"
    width="200" height="150"></canvas>
<script>
   let canvas = document.getElementById("myCanvas");
   let context = canvas.getContext("2d");

   context.beginPath(); // 빈 경로 구성
   context.strokeStyle="magenta";
   context.arc(100, 70, 30, 0, 1.5*Math.PI, false); // 시계 방향
   context.stroke(); // 경로에 있는 원호를 그린다

   context.beginPath(); // 여기서 반드시 빈 경로로 시작해야 함
   context.strokeStyle="blue";
   context.arc(100, 70, 50, Math.PI/2, Math.PI, true); // 반시계 방향
   context.stroke(); // 경로에 있는 한 개의 원호를 캔버스에 그린다
</script>
</body>
</html>
```



context.arc(100, 70, 50, Math.PI/2, Math.PI, true);

사각형 그리기

- 사각형 그리는 컨텍스트 메소드

rect(x, y, w, h)	*(x, y)*에서 *w×h* 크기의 사각형을 경로에 삽입한다.
strokeRect(x, y, w, h)	*(x, y)*에서 *w×h* 크기의 사각형을 경로에 삽입하지 않고 캔버스에 직접 그린다.

  - 예) (10, 10)에서 100x100 크기의 사각형 그리기

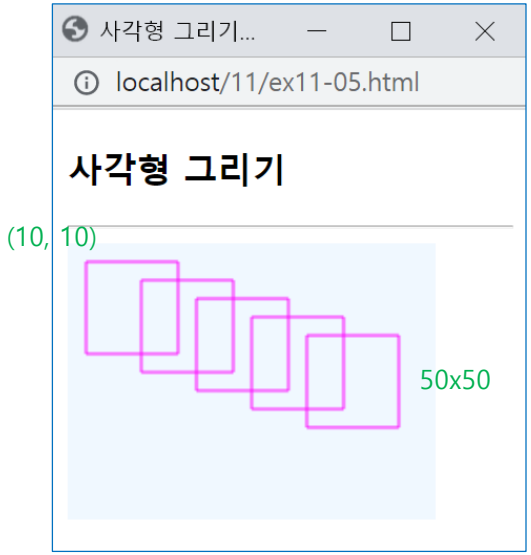
```
context.rect(10, 10, 100, 100);   // (10, 10)에서 100x100 크기의 사각형을 경로에 추가
context.stroke();                 // context에 구성된 사각형을 캔버스에 그린다.
```

  - 예) strokeRect() 메소드 이용 – 캔버스에 바로 그리기

```
context.strokeRect(10, 10, 100, 100);     // (10, 10)에서 100x100 크기의 사각형 그리기
```

예제 11–5 사각형 그리기


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>사각형 그리기</title>
</head>
<body>
<h3>사각형 그리기</h3>
<hr>
<canvas id="myCanvas" style="background-color:aliceblue"
   width="200" height="150"></canvas>
<script>
   let canvas = document.getElementById("myCanvas");
   let context = canvas.getContext("2d");

   context.beginPath(); // 빈 경로 구성
   for(let i=0; i<5; i++) {
      context.rect(10+i*30,10+i*10, 50,50);
   }
   context.strokeStyle="magenta"; // 선의 색
   context.stroke(); // 사각형을 캔버스에 그린다
</script>
</body>
</html>
```



캔버스 지우기

- 캔버스에 그리진 그래픽을 모두 지울 때

```
context.clearRect(0, 0, canvas.width, canvas.height);
```

  - 이 코드는 경로를 지우지는 못함

- 캔버스의 그래픽과 경로를 모두 지울 때

```
context.clearRect(0, 0, canvas.width, canvas.height);
context.beginPath();
```

도형 꾸미기

- 도형 꾸미기
  - 선, 원호, 사각형, 글자 등의 색이나 굵기 조절
  - 선 색 : strokeStyle 프로퍼티 이용

```
context.strokeStyle = "blue";
context.strokeStyle = "#0000FF";
context.strokeStyle = "rgb(0, 0, 255)";
```

  - 채우기 색 : fillStyle 프로퍼티

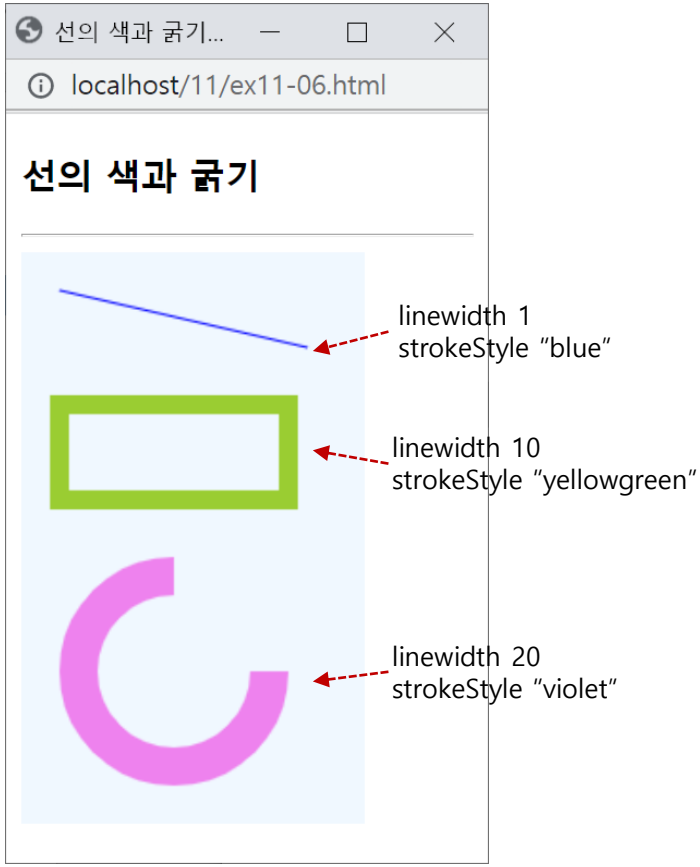
```
context.fillStyle = "red";
```

  - 선 굵기 : lineWidth 프로퍼티

```
context.lineWidth = 20;          // 선 굵기를 20픽셀로 지정
```

예제 11-6 선의 색과 굵기


```
<!DOCTYPE html>
<html>
<head><meta charset="utf-8"><title>선의 색과 굵기</title></head>
<body>
<h3>선의 색과 굵기</h3>
<hr>
<canvas id="myCanvas" style="background-color:aliceblue"
            width="180" height="300"></canvas>
<script>
    let canvas = document.getElementById("myCanvas");
    let context = canvas.getContext("2d");

     // 1 픽셀의 blue 직선 그리기
    context.beginPath();
    context.moveTo(20, 20);
    context.lineTo(150, 50);
    context.strokeStyle = "blue";
    context.stroke();

    // 10 픽셀 yellowgreen 사각형 그리기
    context.beginPath();
    context.rect(20, 80, 120, 50);
    context.lineWidth = 10; // 선 굵기 10픽셀
    context.strokeStyle = "yellowgreen"; // 선 색
    context.stroke();

    // 20 픽셀의 violet 색 원호 그리기
    context.beginPath();
    context.arc(80, 220, 50, 0, 1.5*Math.PI, false);
    context.lineWidth = 20; // 선 굵기 20픽셀
    context.strokeStyle = "violet"; // 선 색
    context.stroke();
</script>
</body>
</html>
```



칠하기

- 도형 내부를 칠하는 기능
  - 원호 내부, 사각형 내부, 텍스트 내부 칠하기
- 칠하는 여러 방법
  - fillStyle 프로퍼티 : 원호나 사각형, 텍스트의 내부를 칠할 색 지정

```
context.fillStyle = "violet";
```

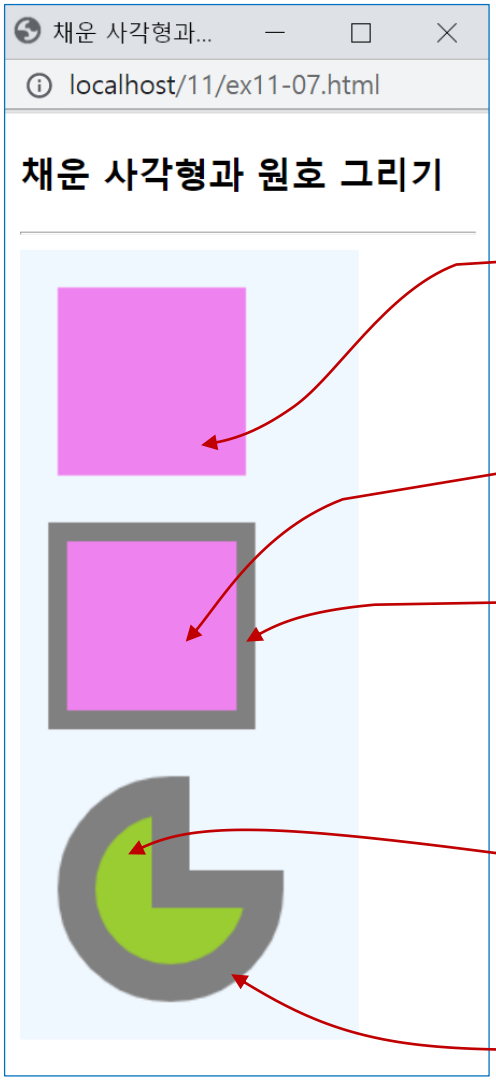
  - 캔버스에 바로 칠하기 : fillRect()
    - fillStyle의 색으로 사각형 내부 채우기

```
context.fillStyle = "violet";
context.fillRect(20, 20, 100, 100);
```

  - 경로에 닫힌 도형 칠하기 : fill()
    - fill()은 사각형과 원호 모두 적용

```
context.fillStyle = "violet";
context.rect(20, 20, 100, 100);   // 경로에 사각형 삽입
context.fill();                   // 경로내 도형 내부 칠하기
context.strokeStyle = "gray";
context.lineWidth = 10;
context.stroke();                 // 경로내 도형 그리기(외곽선)
```

예제 11-7 칠하기




```
<!DOCTYPE html>
<html>
<head><meta charset="utf-8">
<title>채운 사각형과 원호 그리기</title></head>
<body>
<h3>채운 사각형과 원호 그리기</h3>
<hr>
<canvas id="myCanvas" style="background-color:aliceblue"
        width="180" height="420"></canvas>
<script>
    let canvas = document.getElementById("myCanvas");
    let context = canvas.getContext("2d");

    // fillRect()로 외곽선 없이 색으로 채운 사각형 그리기
    context.fillStyle = "violet";
    context.fillRect(20, 20, 100, 100); // 채운 사각형 그리기

    // fill()로 사각형 내부 칠하기
    context.beginPath();
    context.rect(20, 150, 100, 100); // 경로에 사각형 삽입
    context.fillStyle = "violet";
    context.fill(); // 사각형 내부 칠하기

    // 사각형 외곽선 그리기
    context.strokeStyle = "gray";
    context.lineWidth = 10;
    context.stroke(); // 사각형 외곽선 그리기

    // fill()로 원호 내부 칠하기
    context.beginPath();
    context.moveTo(80, 340); // 원호의 중심을 시작점으로 설정
    context.arc(80, 340, 50, 0, 1.5*Math.PI); // 경로에 원호 삽입
    context.closePath(); // 원호의 끝점과 경로 시작점(원호중심)을
                         // 연결하는 직선 자동 추가
    context.fillStyle = " yellowgreen " ;
    context.fill(); // 원호 내부 칠하기

    // 원호 외곽선 그리기
    context.strokeStyle = " gray " ;
    context.lineWidth = 20;
    context.stroke(); // 원호 외곽선 그리기
</script>
</body></html>
```

텍스트 그리기

- 캔버스에 텍스트 그리기
  - 비트맵 이미지로 캔버스에 글자 그리기(출력)
  - 텍스트 그리기는 2가지 방법
    - 텍스트의 외곽선만 그리기 - strokeText()
    - 텍스트 외곽선 없이 내부를 채워 그리기 - fillText()

strokeText(text, x, y [, maxWidth])
fillText(text, x, y [, maxWidth])

- text : 출력하고자 하는 문자열 텍스트
- x, y : 텍스트가 출력되는 시작 점 (x, y)
- maxWidth : 텍스트가 출력되는 최대 폭. text가 이 값보다 크면 자동으로 다른 폰트로 대체됨

strokeText()는 strokeStyle 색으로 (x,y) 위치에 text의 외곽선만 그리며, fillText()는 외곽선없이 텍스트 내부를 fillStyle 색으로 칠한다.

텍스트 그리기 사례

- 폰트 설정 : font 프로퍼티 이용
  - 예) context.font = “50px arial”;
- 정렬 설정 : textAlign 프로퍼티 이용
  - 예) context.textAlign =“center”;

- 텍스트 외곽선 그리기

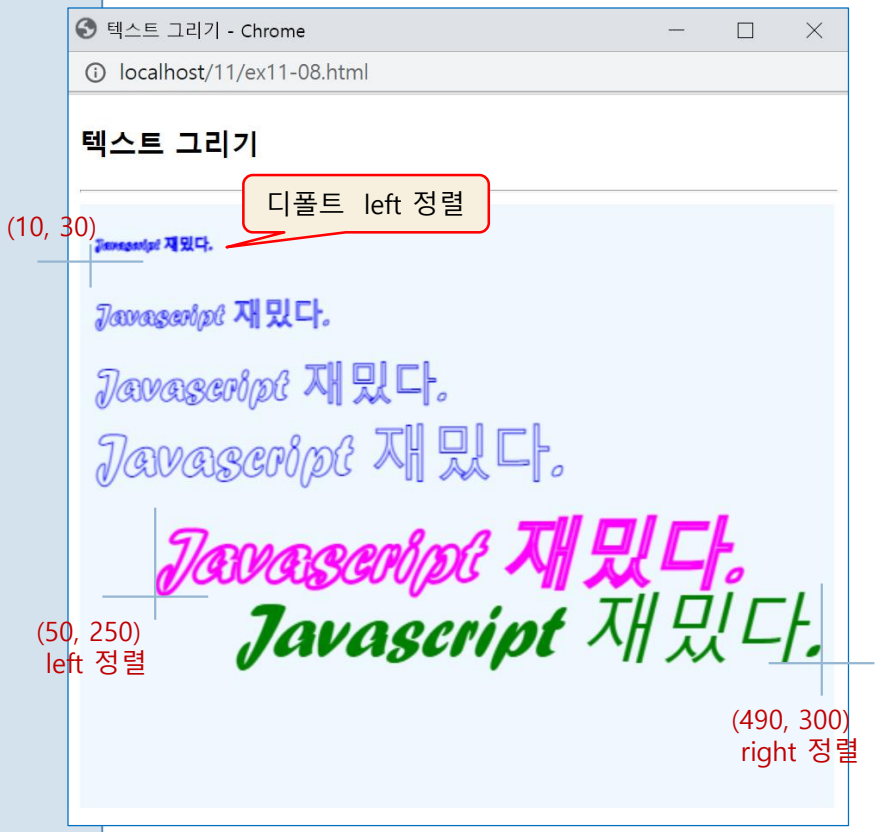
```
context.font = "50px arial";
context.strokeStyle = "blue";
context.lineWidth = 1;
context.strokeText("Javascript", 30, 100);
```

- 텍스트 채워 그리기

```
context.font = "50px arial";
context.fillStyle = "green";
context.fillText("Javascript", 30, 200);
```

예제 11-8 텍스트 그리기


```
<!DOCTYPE html>
<html>
<head><meta charset="utf-8"><title>텍스트 그리기</title></head>
<body>
<h3>텍스트 그리기</h3>
<hr>
<canvas id="myCanvas" style="background-color:beige"
    width="500" height="400"></canvas>
<script>
    let canvas = document.getElementById("myCanvas");
    let context = canvas.getContext("2d");
    context.strokeStyle = "blue";

    // font 프로퍼티로 다양한 크기와 서체 활용
    for(let i=0; i<4; i++) {
        context.font = (10 + i*10) + "px forte";
        context.strokeText("Javascript 재밌다.", 10, 30+i*50);
    }

    // 텍스트 외곽선만 그리기
    context.font = "italic 50px forte";
    context.strokeStyle = "magenta";
    context.lineWidth = 3;
    context.textAlign = "left";
    context.strokeText("Javascript 재밌다.", 50, 250);

    // 텍스트 채워 그리기
    context.fillStyle = "green";
    context.textAlign = "right";
    context.fillText("Javascript 재밌다.", 490, 300);
</script>
</body>
</html>
```



이미지 그리기

- 이미지 객체 생성
  - 파일에서 읽은 이미지를 담을 객체 필요

```
let img = new Image();
```

- 이미지 그리는 과정
  - 이미지 로딩과 onload
    - 이미지 파일의 로딩이 완료된 후 이미지를 그린다.

```
1  img.onload = function () {
2          ... // img 객체에 로드된 이미지를 그린다.
3  }
4  img.src = "test.png"; // img 객체에 test.png 파일로부터 이미지의 로딩 시작
```

  - 이미지 그리기
    - 컨텍스트 객체의 drawImage() 메소드 이용

drawImage()로 이미지 그리기 사례

- (20, 20) 위치에 원본 크기로 그리기

```
let img = new Image();
img.onload = function () {
   context.drawImage(img, 20, 20);
}
img.src = "test.png";
```

- (20, 20) 위치에 100×200 크기로 그리기

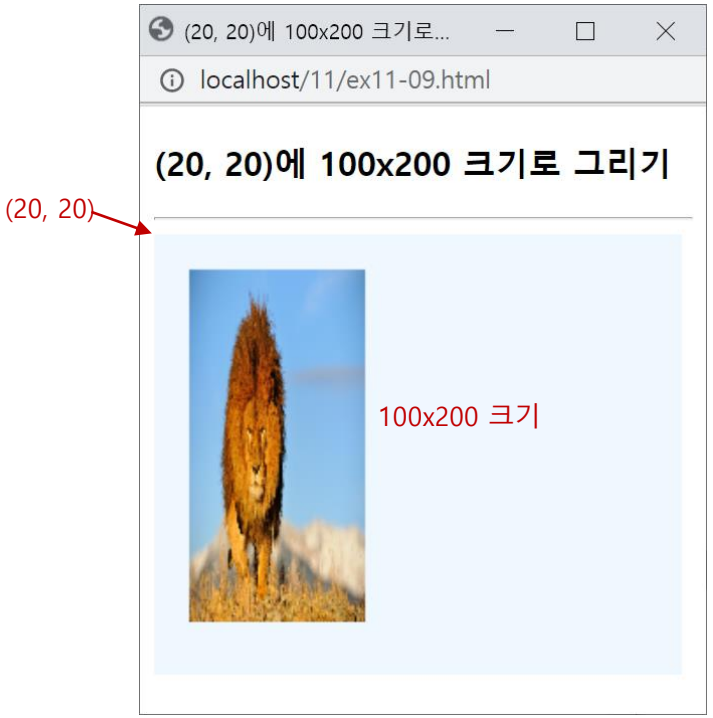
```
let img = new Image();
img.onload = function () {
   context.drawImage(img, 20, 20, 100, 200);
}
img.src = "test.png";
```

- 캔버스에 꽉 차게 이미지 그리기

```
let img = new Image();
img.onload = function () {
   context.drawImage(img, 0, 0, canvas.width, canvas.height);
}
img.src = "test.png";
```

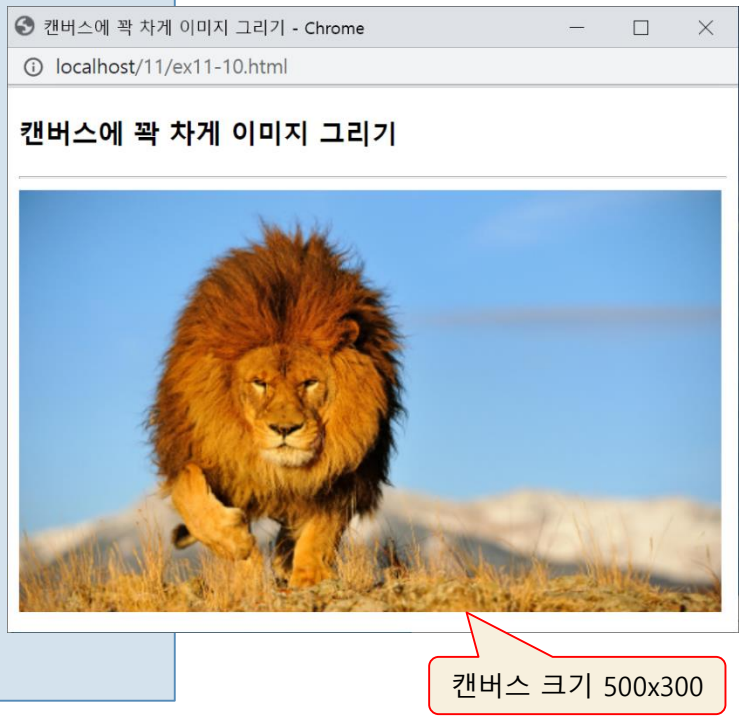
예제 11-9 캔버스의 (20, 20)에 100x200 크기로 변형하여 그리기


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>(20, 20)에 100x200 크기로 그리기</title>
</head>
<body>
<h3>(20, 20)에 100x200 크기로 그리기</h3>
<hr>
<canvas id="myCanvas"
            style="background-color:aliceblue"
            width="300" height="250"></canvas>
<script>
   let canvas = document.getElementById("myCanvas");
   let context = canvas.getContext("2d");

   let img = new Image();
   img.onload = function () {
      context.drawImage(img, 20, 20, 100, 200);
   }
   img.src = "media/lion.png";
</script>
</body>
</html>
```



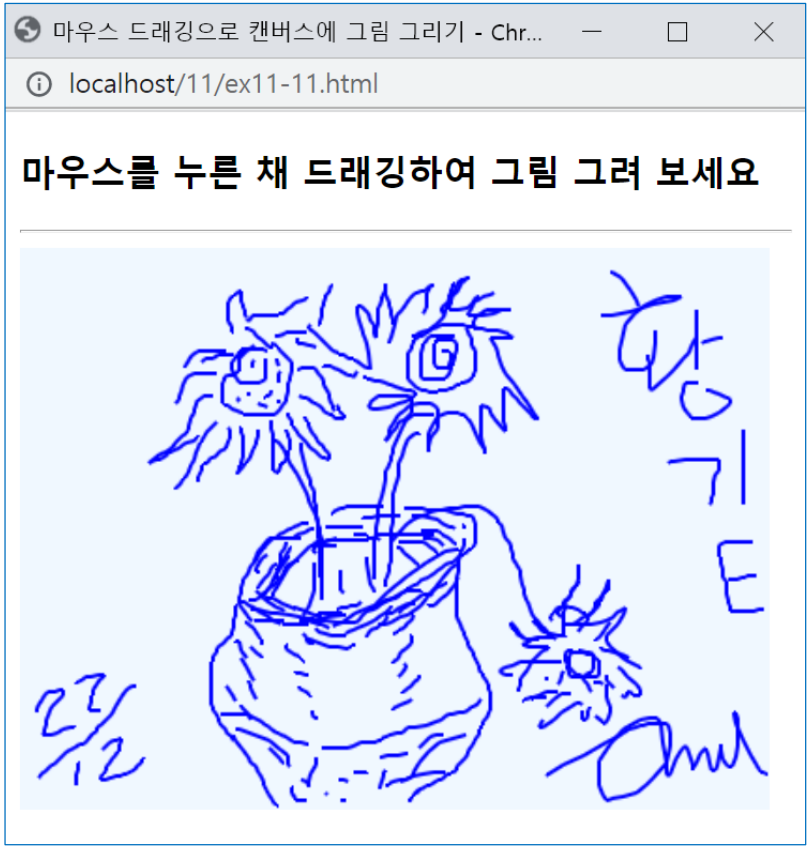
예제 11-10 캔버스에 꽉 차게 이미지 그리기


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>캔버스에 꽉 차게 이미지 그리기</title>
</head>
<body>
<h3>캔버스에 꽉 차게 이미지 그리기</h3>
<hr>
<canvas id="myCanvas"
            style="background-color:aliceblue"
            width="500" height="300"></canvas>
<script>
   let canvas = document.getElementById("myCanvas");
   let context = canvas.getContext("2d");

   let img = new Image();
   img.onload = function () {
      context.drawImage(img, 0, 0, canvas.width, canvas.height);
   }
   img.src = "media/lion.png";
</script>
</body>
</html>
```



canvas 객체와 마우스 이벤트 활용

- 사용자가 마우스로 드래깅하여 캔버스 위에 자유롭게 그림을 그리는 자바스크립트 응용 프로그램 작성



예제 11-11 마우스 드래깅으로 캔버스에 그림 그리기


```
<!DOCTYPE html>
<html>
<head><meta charset="utf-8"><title>마우스 드래깅으로 캔버스에 그림 그리기</title></head>
<body onload="init()">
<h3>마우스를 누른 채 드래깅하여 그림 그려 보세요</h3>
<hr>
<canvas id="myCanvas" style="background-color:aliceblue" width="400" height="300">
</canvas>
<script>
let canvas, context;
function init() {
    canvas = document.getElementById("myCanvas");
    context = canvas.getContext("2d");

    context.lineWidth = 2; // 선 굵기 2
    context.strokeStyle = "blue";

    canvas.addEventListener("mousemove", function (e) { move(e) }, false);
    canvas.addEventListener("mousedown", function (e) { down(e) }, false);
    canvas.addEventListener("mouseup", function (e) { up(e) }, false);
    canvas.addEventListener("mouseout", function (e) { out(e) }, false);
}

let startX=0, startY=0; // 드래깅동안, 처음 마우스가 눌러진 좌표
let dragging=false;
function draw(curX, curY) {
    context.beginPath();
    context.moveTo(startX, startY);
    context.lineTo(curX, curY);
    context.stroke();
}
```

마우스 리스너 등록.
e는 MouseEvent 객체

```
function down(e) {
    startX = e.offsetX; startY = e.offsetY; dragging = true;
}
function up(e) { dragging = false; }
function move(e) {
    if(!dragging) return; // 마우스가 눌러지지 않았으면 리턴
    let curX = e.offsetX, curY = e.offsetY;
    draw(curX, curY);
    startX = curX; startY = curY;
}
function out(e) { dragging = false; }
</script>
</body>
</html>
```