[13차시 교안]

<자바스크립트와 캔버스로 게임만들기(2)>

1. 도형 변환

(1) 평행이동(translation) : 캔버스를 수평이나 수직으로 이동

```
<body>
    <canvas id="myCanvas" width="600" height="400"></canvas>
    <script>
        var canvas = document.getElementById('myCanvas');
        var context = canvas.getContext('2d');

        context.fillStyle = "blue";
        context.fillRect(0, 0, 100, 100);
        context.translate(50, 50);
        context.fillStyle = "red";
        context.fillRect(0, 0, 100, 100);
    </script>
</body>
```

(2) 신축(scaling) : 확대나 축소

```
<body>
    <canvas id="myCanvas" width="600" height="400"></canvas>
    <script>
        var canvas = document.getElementById('myCanvas');
        var context = canvas.getContext('2d');

        context.fillStyle = "blue";
        context.fillRect(0, 0, 100, 100);
        context.translate(50, 50);
        context.scale(0.5, 0.5)
        context.fillStyle = "red";
        context.fillRect(0, 0, 100, 100);
    </script>
</body>
```

(3) 회전(rotation) : 좌표 공간 회전

```
<body>
    <canvas id="myCanvas" width="600" height="400"></canvas>
    <script>
        var canvas = document.getElementById('myCanvas');
        var context = canvas.getContext('2d');

        context.fillStyle = "blue";
        context.fillRect(0, 0, 100, 100);
        context.translate(50, 50);
        context.rotate(Math.PI/4);
        context.fillStyle = "red";
        context.fillRect(0, 0, 100, 100);
    </script>
</body>
```

2. 애니메이션

캔버스를 만들고 여기에 자바스크립트로 그림을 순차적으로 그려주는 것

(1) 애니메이션을 작성하는 순서

가. 캔버스를 지운다.

① 캔버스를 지우는 방법

a. clearRect( ) : 프로그램에서 사용할 방법

b. 캔버스의 크기 조정

c. 캔버스 객체를 다시 생성하는 방법

② 캔버스를 지우지 않으면 이전 그림이 남아 있어서 현재 그림과 동시에 보임

```
context.clearRect(0, 0, 100, 100);
```

나. (x, y) 위치에 그림을 그린다.

① 웹 페이지 안에 캔버스 요소 생성하고 자바스크립트로 그림을 그림

② 공으로 사용할 빨간색 공

```
context.beginPath( );
context..fillStyle = “red”;
context.arc(x, y, 20, 0, Math.PI*2, true);
context.closePath( );
context.fill( );
```

다. 위치를 업데이트한다.

① 애니메이션은 시간에 따라서 변화되는 것

② 볼의 위치를 변경 시킴

```
x += dx;
y += dy;
```

a. dx, dy : 볼이 한번에 움직이는 거리

b. 바운스되는 볼을 만들기 위해서는 볼이 벽에 부딪치면 반사되도록 함

```
if (x < (0 + 20) || x > (300 - 20))
        dx = -dx;
if (y < (0 + 20) || y > (200 - 20))
        dy = -dy;
x += dx;
y += dy;
```

라. 위의 절차(1부터 3까지)를 반복한다.

① 애니메이션은 반복 수행됨

② 반복을 위해 사용 가능한 함수

a. setTimeout(doSomething, 500); // 500 밀리초 후에 doSomething( ) 호출

b. setInterval(doSomething, 500); //매 500밀리초마다 doSomething( ) 호출

③ 프로그램에서는 10밀리초마다 공이 호출(draw( ))되도록 함

a. setInterval(draw, 10);

3. Bouncing Ball 예제

```
<!DOCTYPE html>
<html>
<head>
  <title>Bouncing Ball Example</title>
  <style>
    canvas {
      background: yellow;
      border: 1px dotted black;
    }
  </style>
```

```
    <script>
        var context;
        var dx = 5;
        var dy = 5;
        var y = 100;
        var x = 100;
        function draw() {
            var canvas = document.getElementById('myCanvas');
            var context = canvas.getContext('2d');
            context.clearRect(0, 0, 300, 200);
            context.beginPath();
            context.fillStyle = "red";
            context.arc(x, y, 20, 0, Math.PI * 2, true);
            context.closePath();
            context.fill();
```

```
            if (x < (0 + 20) || x > (300 - 20))
                dx = -dx;
            if (y < (0 + 20) || y > (200 - 20))
                dy = -dy;
            x += dx;
            y += dy;
        }
        setInterval(draw, 10);
        </script>
</head>
<body>
    <canvas id="myCanvas" width="300" height="200"></canvas>
</body>
</html>
```

4. 게임 제작

(1) 게임 소개

가. 앵그리 버드와 유사한 다음과 같은 게임을 제작

나. '발사' 버튼을 누르면 왼쪽에 놓인 공이 발사됨

다. 속도와 각도에 의해 날아가다가 네트에 맞으면 점수가 올라감

(2) HTML 요소 생성하기

가. 그림을 그리기 위한 <canvas> 요소 생성

나. 사용자로부터 숫자를 입력 받을 수 있는 <input> 요소, 버튼

```
<html>
<head>
   <title>Javascript Game</title>
   <style>
      canvas {
         border: 1px dotted red;          /* 캔버스 경계선 */
         background-color: #fcff00;       /* 캔버스 배경색 */
      }
   </style>
   <body >
   <canvas id="canvas" width="500" height="300"></canvas>
   <div id="control">
      속도<input id="velocity" value="30" type="number" min="0" max="100" step="1" />
      각도<input id="angle" value="45" type="number" min="0" max="90" step="1" />
      <div id="score">점수 = 0</div>
      <button발사</button>
   </div>
</body>
</html>
```

(3) 배경 만들기

두 개의 이미지 사용 : 잔디밭, 그물망

```
<script>
  var image = new Image();      // 이미지 객체 생성
  image.src = "lawn.png";       // 이미지 파일 이름 설정
  var backimage = new Image();
  backimage.src = "net.png";

  function drawBackground() {     // 배경을 화면에 그린다.
    context.drawImage(image, 0, 270);
    context.drawImage(backimage, 450, 60);
  }
```

```
  function draw() {                  // 전체 화면을 그리는 함수
    context.clearRect(0, 0, 500, 300);     // 화면을 지운다.
    drawBackground();
  }
  function init() {
    context = document.getElementById('canvas').getContext('2d');
    draw();
  }
</script>
</head>
<body onload="init()">
```

(4) 움직이는 공 만들기

가. 공에 대한 변수

① var ballV : 공의 속도

② var ballVx : 공의 x 방향 속도

③ var ballVy : 공의 y방향 속도

④ var ballX : 공의 현재 x좌표

⑤ var ballY : 공의 현재 y 좌표

⑥ var ballRadius : 공의 반지름

나. 공의 움직임 계산 : 물리엔진 사용

① 물리학적인 법칙에 따라 여러 가지 계산을 해주는 자바스크립트 라이브러리

② 대표적인 물리 엔진 : Box2D

다. 프로그램에서는

① ballVx : 초기 속도에서 변하지 않음

② ballVy : 초기 속도에서 중력 가속도 만큼 느려짐

(5) 전체 프로그램

```
<html>
<head>
   <title>Javascript Game</title>
   <style>
      canvas {
         border: 1px dotted red;          /* 캔버스에 경계선을 그려준다. */
         background-color: #fcff00;       /* 캔버스의 배경색을 지정한다. */
      }
   </style>
   <script>
      var context;                        /* 컨텍스트 객체*/
      var velocity;                       /* 사용자가 입력한 공의 초기속도 */
      var angle;                          /* 사용자가 입력한 공의 초기각도 */
      var ballV;                          /* 공의 현재 속도 */
      var ballVx;                         /* 공의 현재 x방향 속도 */
      var ballVy;                         /* 공의 현재 y방향 속도 */
      var ballX = 10;                     /* 공의 현재 x방향 위치 */
      var ballY = 250;                    /* 공의 현재 y방향 위치 */
      var ballRadius = 10;                /* 공의 반지름 */
      var score = 0;                      /* 점수 */
```

```
var image = new Image();                    /* 이미지 객체 생성 */
image.src = "lawn.png";                     /* 이미지 파일 이름 설정 */
var backimage = new Image();
backimage.src = "net.png";
var timer;                                          /* 타이머 객체 변수 */

/* 공을 화면에 그린다. */
function drawBall() {
   context.beginPath();
   context.arc(ballX, ballY, ballRadius, 0, 2.0 * Math.PI, true);
   context.fillStyle = "red";
   context.fill();
}

/* 배경을 화면에 그린다. */
function drawBackground() {
   context.drawImage(image, 0, 270);
   context.drawImage(backimage, 450, 60);
}
```

```
/* 전체 화면을 그리는 함수 */
function draw() {
   context.clearRect(0, 0, 500, 300);       /* 화면을 지운다. */
   drawBall();
   drawBackground();
}

/* 초기화를 담당하는 함수 */
function init() {
   ballX = 10;
   ballY = 250;
   ballRadius = 10;
   context = document.getElementById('canvas').getContext('2d');
   draw();
}
```

```
        /* 사용자가 발사 버튼을 누르면 호출된다. */
        function start() {
            init();
            velocity = Number(document.getElementById("velocity").value);   //속도
            angle = Number(document.getElementById("angle").value);     //각도
            var angleR = angle * Math.PI / 180;   // 각도를 도에서 라디언으로 변환

            ballVx = velocity * Math.cos(angleR);  //초기 x방향 속도
            ballVy = -velocity * Math.sin(angleR);  //초기 y방향 속도

            draw();   //전체 화면을 그리는 함수
            timer = setInterval(calculate, 100);
            return false;           (공의 현재 속도와 위치 업데이트 함수)
        }
```

```
/* 공의 현재 속도와 위치를 업데이트한다. */
        function calculate() {
            ballVy = ballVy + 1.98;

            ballX = ballX + ballVx;
            ballY = ballY + ballVy;

            /* 공이 목표물에 맞았으면 */
            if ((ballX >= 450) && (ballX <= 480) && (ballY >= 60) && (ballY <= 210)) {
                score++;
                document.getElementById("score").innerHTML = "점수=" + score;
                clearInterval(timer);
            }
            /* 공이 경계를 벗어났으면 */
            if (ballY >= 300 || ballY < 0) {
                clearInterval(timer);
            }
            draw();
        }
    </script>
</head>
```

```
<body onload="init();">
    <canvas id="canvas" width="500" height="300"></canvas>
    <div id="control">
        속도<input id="velocity" value="30" type="number" min="0" max="100" step="1" />
        각도<input id="angle" value="45" type="number" min="0" max="90" step="1" />
        <div id="score">점수 = 0</div>
        <button onclick="start()">발사</button>
    </div>
</body>
</html>
```