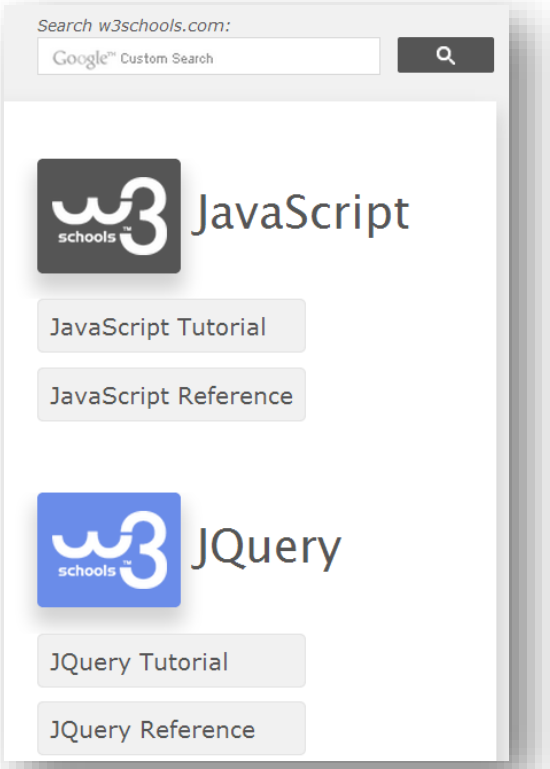
Mobile Simulation

2017-2

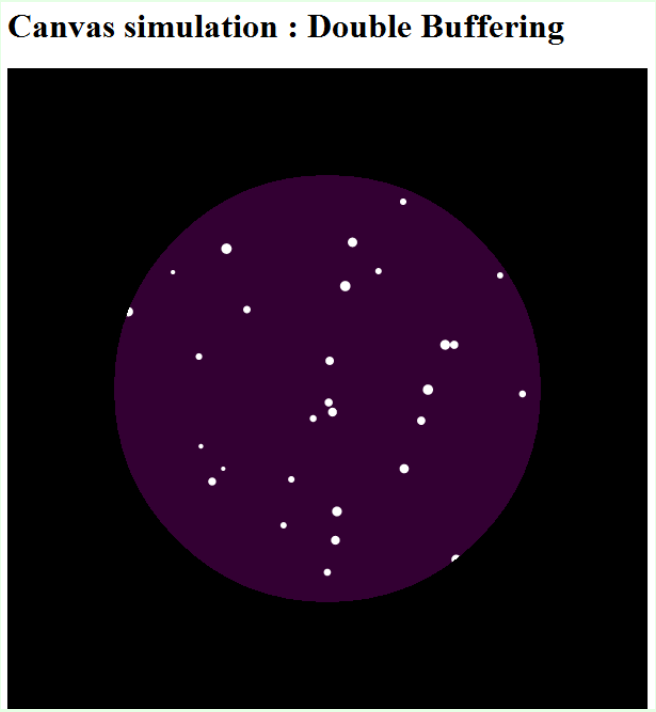


Weekly plan (HTML5, 1st semester 2017)

- **wk01 : Introduction to curriculum & current state of HTML5**
- **wk02 : Making HTML5 documents**
- **wk03 : Table, iframe and media**
- **wk04 : Semantic tag and Form**
- **wk05 : CSS3 I. Basic**
- **wk06 : CSS3 II. Advanced**
- **wk07 : CSS3 III. Animation**
- **wk08 : Mid-term Exam.**
- **wk09 : Javascript : Data types & operators**
- **wk10 : Javascript : Loop & functions**
- **wk11 : Javascript : Core objects**
- **wk12 : Javascript : DOM**
- **wk13 : Javascript : Event handling I**
- **wk14 : Javascript : Event handling II**
- **wk15 : Final exam.**

Weekly plan (Mobile Simulation, 2nd semester 2017)

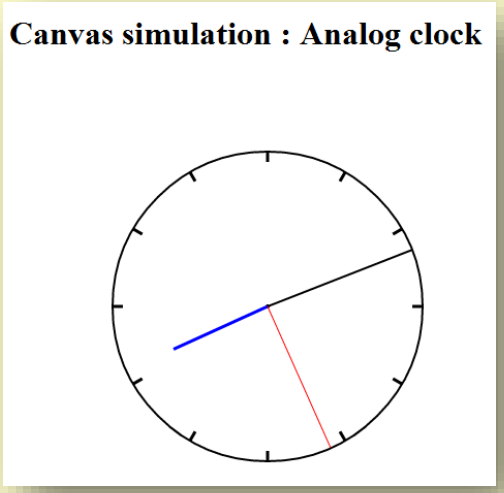
- **wk01 : Introduction to curriculum & current state**
- **wk02 : Browser Object Model (BOM), installing Brackets editor**
- **wk03 : Canvas graphics I. Basic**
- **wk04 : Canvas graphics II. Image & Transformation**
- **wk05 : Canvas graphics III. Animation**
- **wk06 :**
- **wk07 :**
- **wk08 : Mid-term Exam.**
- **wk09 :**
- **wk10 :**
- **wk11 :**
- **wk12 :**
- **wk13 :**
- **wk14 :**
- **wk15 : Final exam.**



과제03. msnn_rpt03.zip

[실습과제03] Analog clock on canvas

[1] 캔버스에 자바스크립트만을 사용해서 아날로그 시계 만들기
[2] Date 객체로부터 현재 시간을 읽어서 시간을 최대한 정확하게 표시.
[3] 시계 주변에 ID, 이름 표시.



파일명: msnn_rpt03.html

가점: Javascript 프로그래밍 응용 능력.

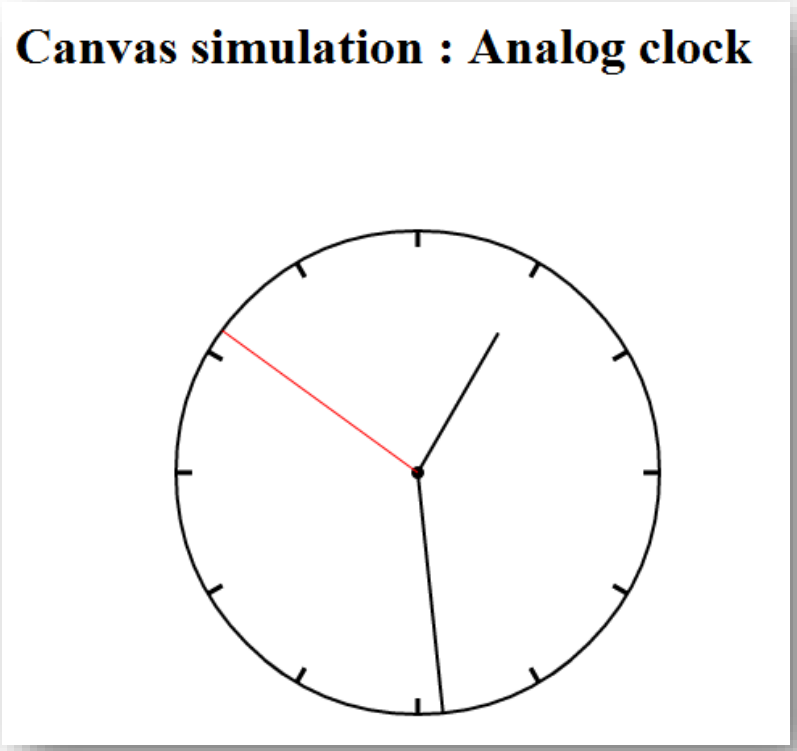
[제출파일] msnn_rpt03.zip

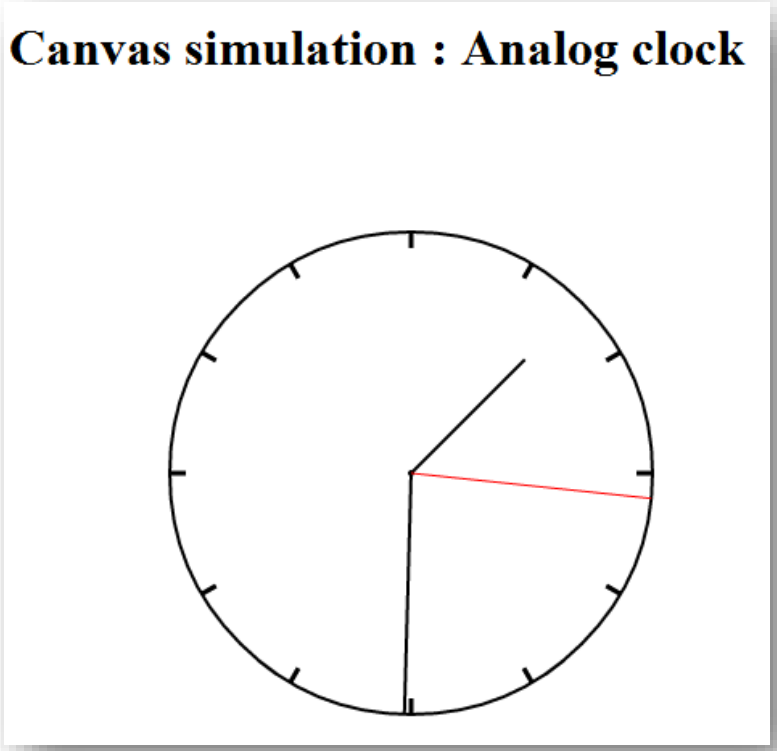
msnn_rpt03.html 과 사용된 그림을 이메일로 제출

Email : chaos21c@gmail.com

chaos21c@gmail.com

과제03. hint



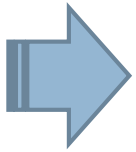


Which clock is running exactly in real time?

과제03. hint


```
// Get time
var date = new Date();

// Get current hour, minutes, seconds
var hours = date.getHours();
var minutes = date.getMinutes();
var seconds = date.getSeconds();
```

```
// second handle
ctx.strokeStyle = "red";
ctx.lineWidth = 1;
//seconds = 1;
drawHand(clockWidth / 2, seconds * 6);

// minute handle
ctx.strokeStyle = "black";
ctx.lineWidth = 2;
//minutes = 30;
drawHand(clockWidth / 2, minutes * 6);

// hour handle
ctx.strokeStyle = "black";
ctx.lineWidth = 3;
//hours = 3;
drawHand(clockWidth / 3, hours * 30);
```



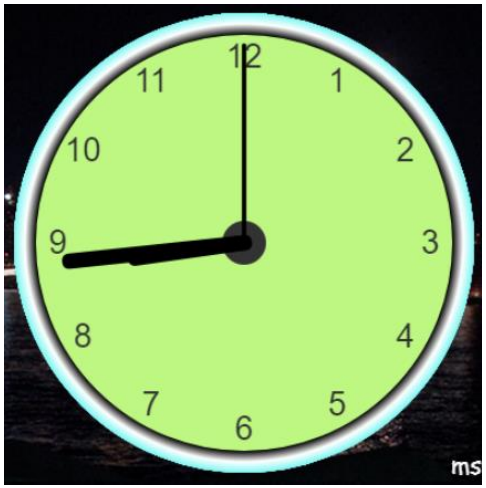
How can you make every handle correctly move?

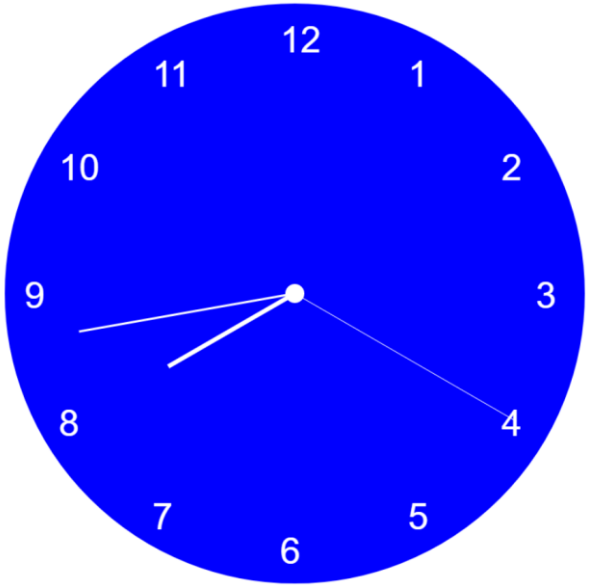
과제03. hint

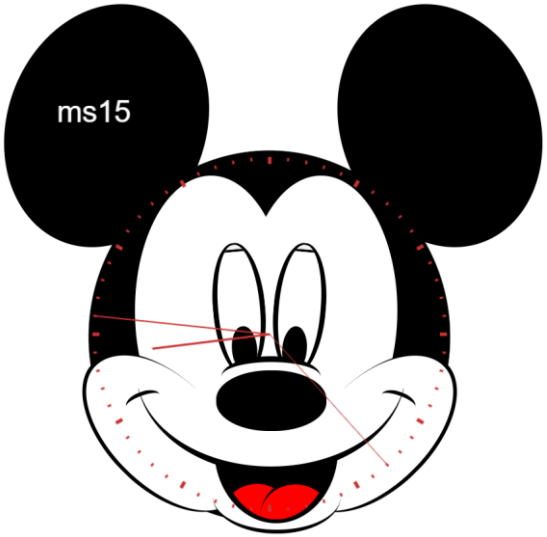

```
// Get current hour, minutes, seconds
var hours = date.getHours();
var minutes = date.getMinutes();
var seconds = date.getSeconds();

// Draw hour
ctx.strokeStyle = "black";
ctx.lineWidth = 3;
drawHand(clockWidth / 3, hours * 30 + minutes*30/60 + seconds*30/3600);

// Draw minutes
ctx.strokeStyle = "black";
ctx.lineWidth = 2;
drawHand(clockWidth / 2, minutes * 6 + seconds*6/60);

// Draw seconds
ctx.strokeStyle = "red";
ctx.lineWidth = 1;
drawHand(clockWidth / 2, seconds * 6);
```

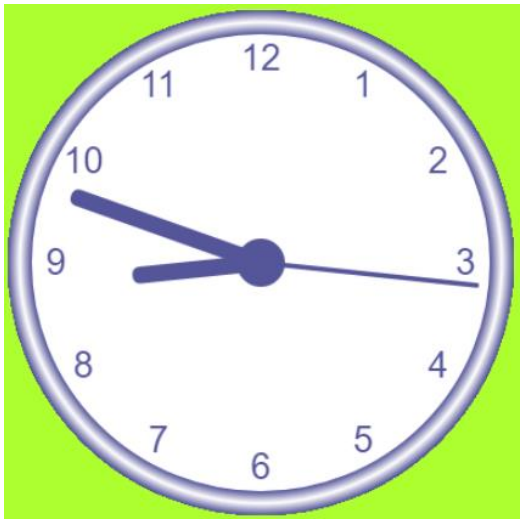
How can you make every handle correctly move?

과제03. Good results







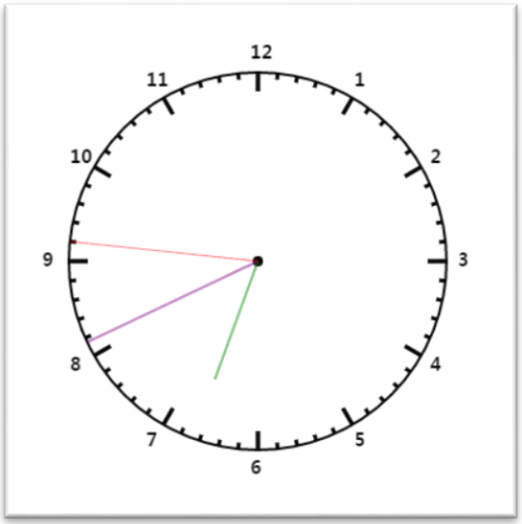


11

HTML5 캔버스 그래픽

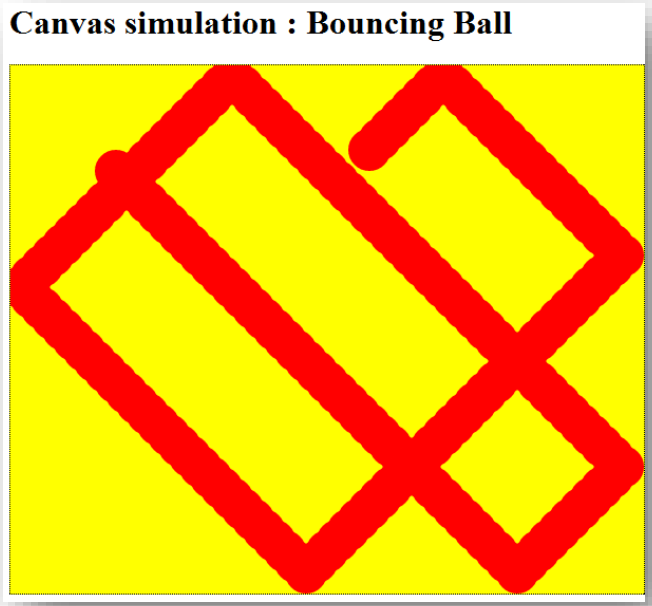
Canvas III. Animation

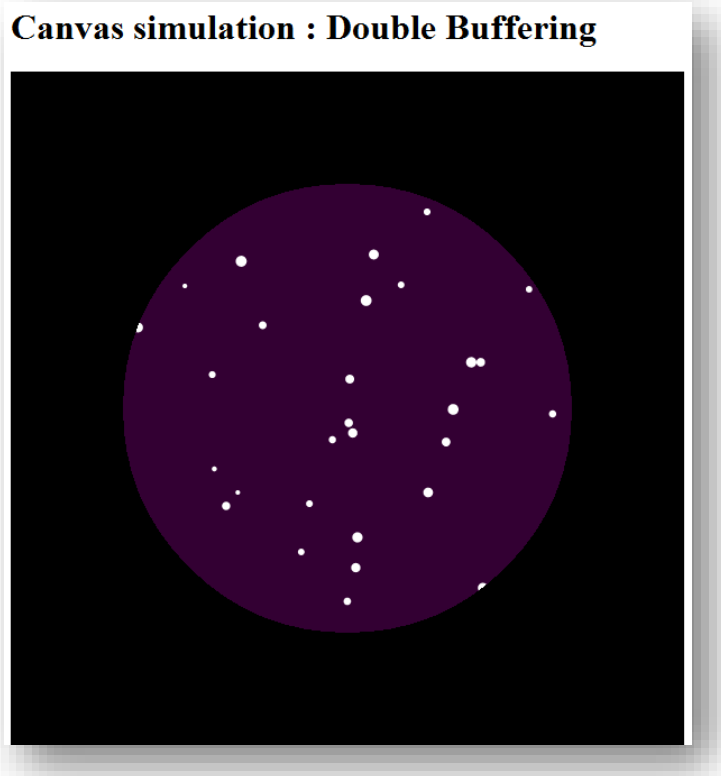
The <canvas> element is used to draw graphics, on the fly, on a web page.



Target:

Making HTML5 Animation using Canvas





Reference

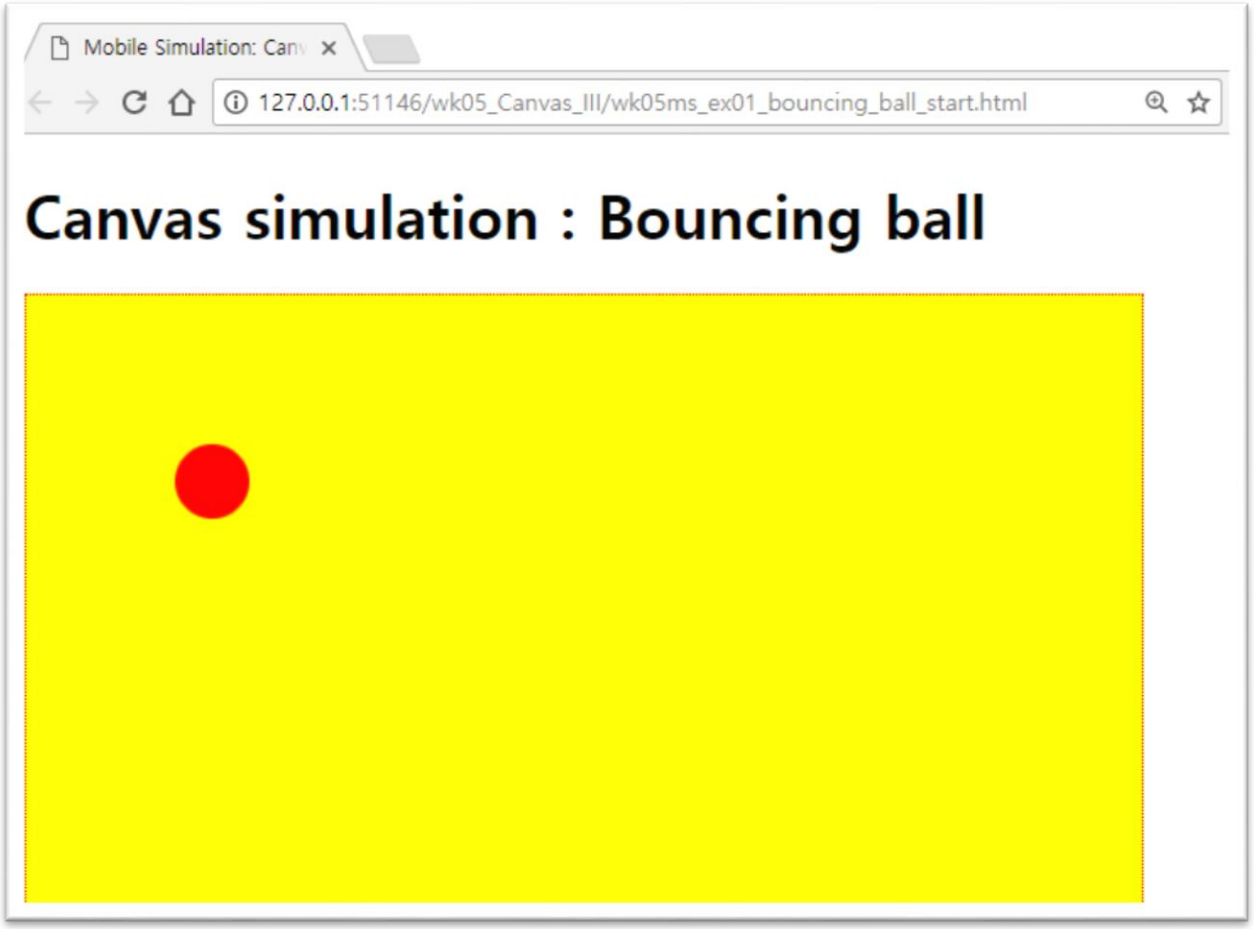
HTML5 Canvas
http://www.w3schools.com/html/html5_canvas.asp

HTML DOM Canvas Object
http://www.w3schools.com/jsref/dom_obj_canvas.asp

HTML Canvas Reference
http://www.w3schools.com/tags/ref_canvas.asp

애니메이션

- Bouncing Ball simulation



Bouncing Ball Simulation

```
<!DOCTYPE html>
<html>
<head>
   <meta charset="utf-8" />
   <title> Mobile Simulation: Canvas </title>
   <style type="text/css">
          canvas {
                 background: yellow;
                 border: 1px dotted red;
          }
  </style>
```

```
<body>

   <h1>Canvas simulation : Bouncing Ball </h1>
   <canvas id="myCanvas" width="600" height="500">  </canvas>

</body>
```

Bouncing Ball code

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

  var dx = 5; // velocity in the x-direction
  var dy = 5; // velocity in the y-direction
  var x = 100;
  var y = 100;

  function draw() {
          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>
```

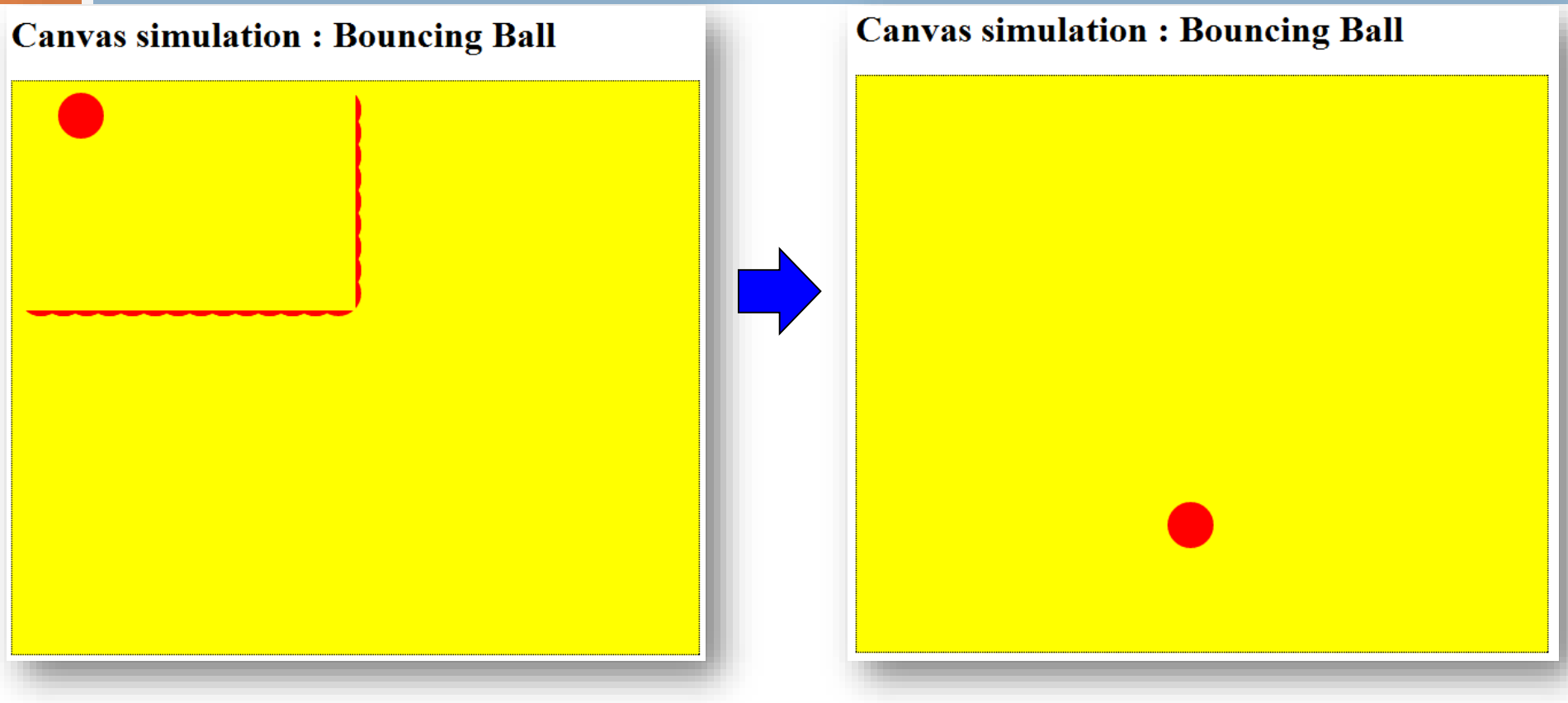
Bouncing Ball API reference

JavaScript syntax:	*context*.arc(*x*,*y*,*r*,*sAngle*,*eAngle*,*counterclockwise*);

Parameter Values

Parameter	Description
x	The x-coordinate of the center of the circle
y	The y-coordinate of the center of the circle
r	The radius of the circle
sAngle	The starting angle, in radians (0 is at the 3 o'clock position of the arc's circle)
eAngle	The ending angle, in radians
counterclockwise	Optional. Specifies whether the drawing should be counterclockwise or clockwise. False is default, and indicates clockwise, while true indicates counter-clockwise.

Bouncing Ball simulation: 결과



[DIY] 버그를 찾아서 버그 처리

Bouncing Ball update # 1

```
var dx = 10;  // velocity in the x-direction
var dy = 10;  // velocity in the y-direction
var y = 100;
var x = 100;
var r = 20;
x_max = context.canvas.width;
y_max = context.canvas.height;

function draw() {
    //var canvas = document.getElementById('myCanvas');
    //var context = canvas.getContext('2d');
    //context.clearRect(0, 0, x_max, y_max);
    context.beginPath();
    context.fillStyle = "red";
    context.arc(x, y, r, 0, Math.PI * 2, true);
    context.closePath();
    context.fill();
    if (x < (0 + r) || x > (x_max - r - dx))
        dx = -dx;
    if (y < (0 + r) || y > (y_max - r - dy))
        dy = -dy;
    x += dx;
    y += dy;
}

setInterval(draw, 10);
```

[Tip !!] x_max - r - dx

Bouncing Ball update # 2(모듈화-1)

```
<script>
var canvas = null;
var context = null;

var dx = 10; // velocity in the x-direction
var dy = 10; //Math.random()*20+10;  // velocity in the y-direction
var x = 100;
var y = 100;
var r = 20;
var x_max = 0;
var y_max = 0;

function init() {
  canvas = document.getElementById('myCanvas');
  context = canvas.getContext("2d");
  x_max = context.canvas.width;
  y_max = context.canvas.height;

  blank();

  context.beginPath();
  context.fillStyle = "red";
  context.arc(x, y, r, 0, Math.PI * 2, true);
  context.closePath();
  context.fill();
   // start animation
  setInterval(draw, 10);
}
```

Bouncing Ball update # 2(모듈화-2)

```
function blank() {
  context.fillStyle = "yellow";
  context.fillRect(0,0,context.canvas.width, context.canvas.height);
}

function draw() {

    //blank();

    if (x < (0 + r) || x > (x_max - r - dx))
    dx = -dx;
    if (y < (0 + r) || y > (y_max - r - dy))
    dy = -dy;
    x += dx;
    y += dy;

    context.beginPath();
    context.fillStyle = "red";
    context.arc(x, y, r, 0, Math.PI * 2, true);
    context.closePath();
    context.fill();
}
```

=== Project 1. ===

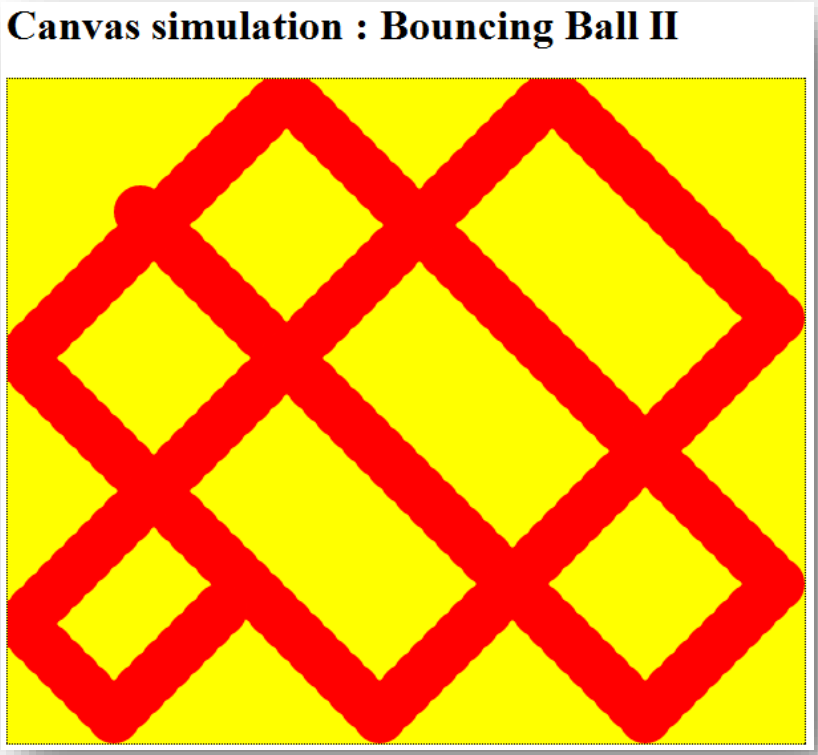
[1] 벽에 충돌 후 가로 및 세로 방향 속도가 무작위로 변하면서 상자 내에서 운동하도록 코드를 수정.
[2] Canvas 밖에 버튼을 만들어서 운동 종료 기능 추가.
[3] 각자 아이디어 추가.

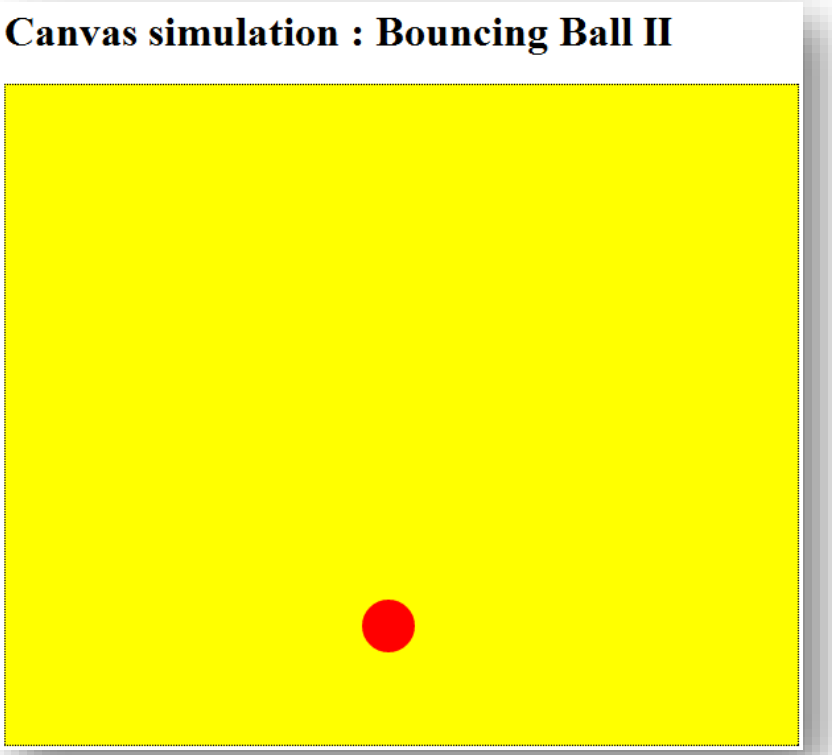
Save as

msnn_bouncing.html

Bouncing Ball update # 2(모듈화-3)

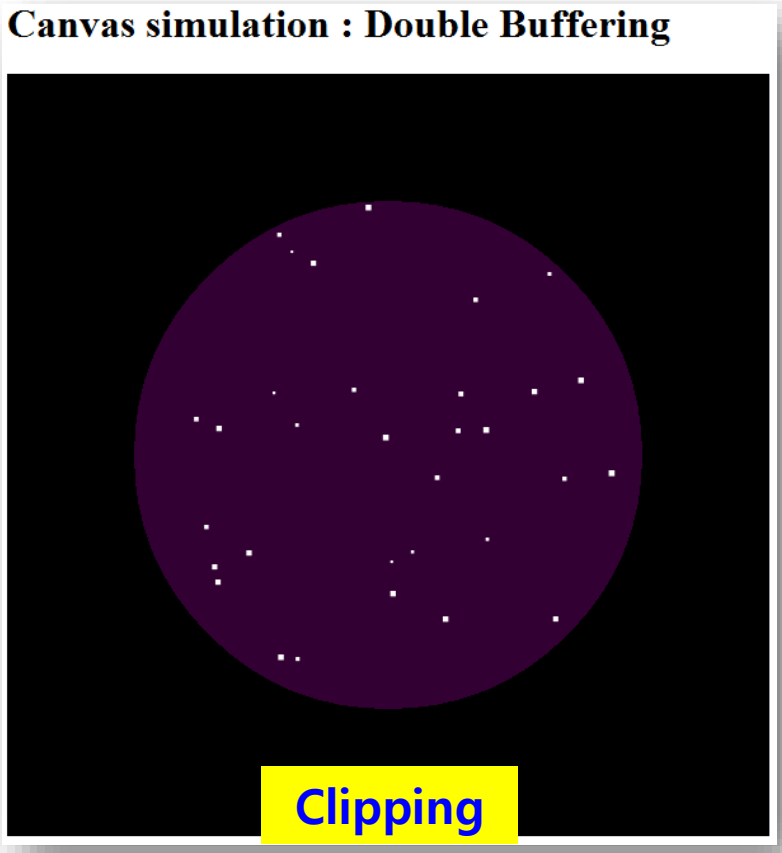
```
<body onload="init()">
  <h1>Canvas simulation : Bouncing Ball II</h1>
  <canvas id="myCanvas" width="600" height="500"> </canvas>
```





Animation Practice

Animation using Double Buffering (DB)



Snowy night (DB) #0

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <title> Mobile Simulation: Canvas </title>
    <style type="text/css">

    </style>

</head>

<body onload="init()">
    <h1>Canvas simulation : Double Buffering </h1>
    <canvas id="myCanvas" width="600" height="600"> </canvas>

    <script>


    </script>

</body>

</html>
```

Snowy night (DB) #1

```
var canvas = null;
var context = null;
var bufferCanvas = null;
var bufferCanvasCtx = null;
var flakeArray = [];
var flakeTimer = null;
var maxFlakes = 200;

function init() {
    canvas = document.getElementById('myCanvas');
    context = canvas.getContext("2d");

    bufferCanvas = document.createElement("canvas");
    bufferCanvasCtx = bufferCanvas.getContext("2d");
    bufferCanvasCtx.canvas.width = context.canvas.width;
    bufferCanvasCtx.canvas.height = context.canvas.height;

    // initialize the rects, make snow flakes

    flakeTimer = setInterval(addFlake, 200);

    Draw();

    setInterval(animate, 30);
}
```

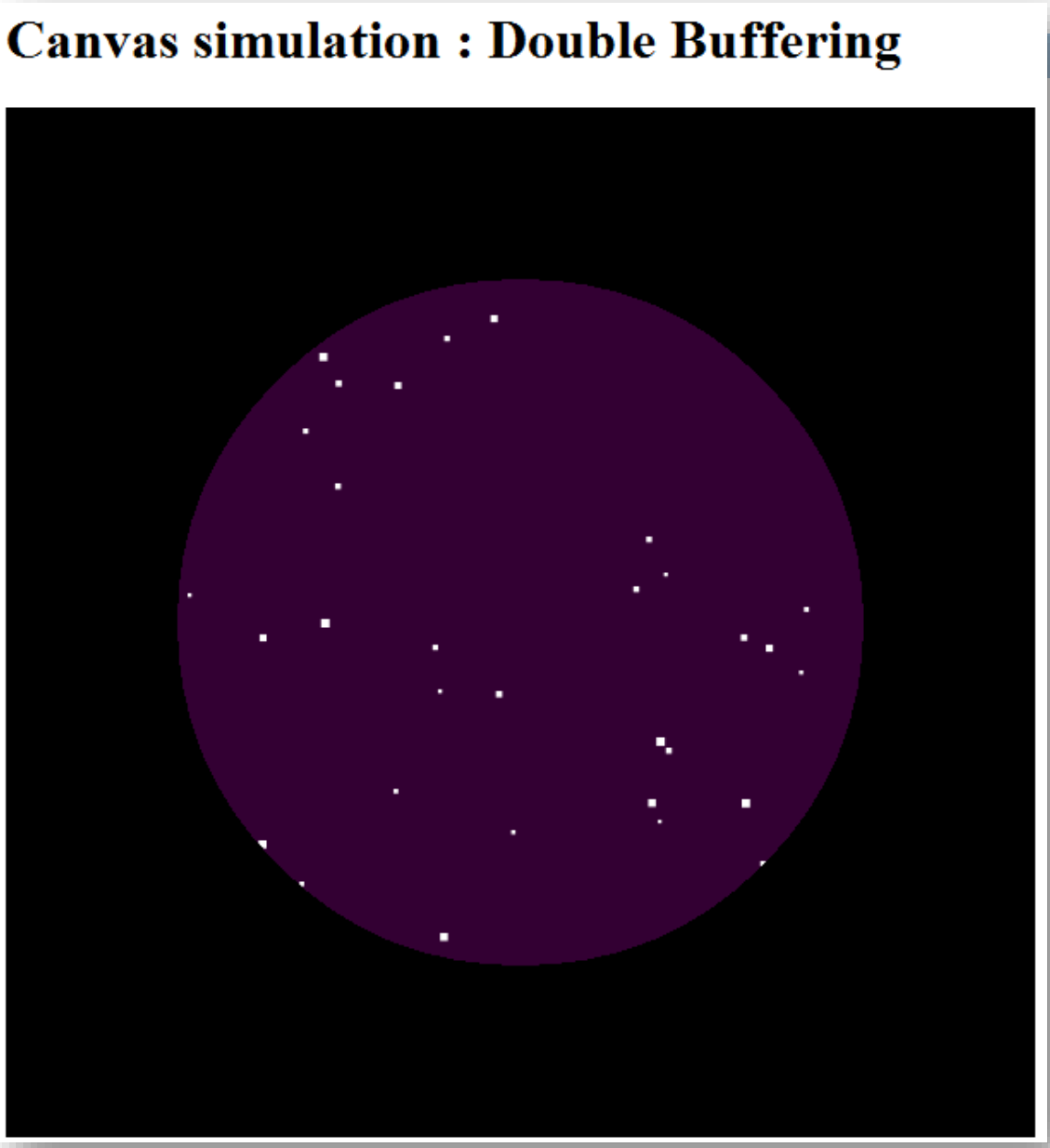
Snowy night (DB) #2

```
// Properties of snowflakes
function Flake() {
    this.x = Math.round(Math.random() * context.canvas.width);
    this.y = -10;
    this.drift = Math.random();
    this.speed = Math.round(Math.random() * 5) + 1;
    this.width = (Math.random() * 3) + 2; // size of snow
    this.height = this.width;
}

// make snowflakes
function addFlake() {
    flakeArray[flakeArray.length] = new Flake();
    if (flakeArray.length == maxFlakes)
        clearInterval(flakeTimer);
}

// Clear buffer canvas
function blank() {
    bufferCanvasCtx.fillStyle = "black"; //"#330033";
    bufferCanvasCtx.fillRect(0, 0, bufferCanvasCtx.canvas.width,
    bufferCanvasCtx.canvas.height);
}

// animate snowflakes
function animate() {
    update();
    draw();
}
```

Snowy night (DB) #3 - clipping & DB

```
// set position and speed of snowflakes
function update() {
    for (var i = 0; i < flakeArray.length; i++) {
        if (flakeArray[i].y < context.canvas.height) {
            flakeArray[i].y += flakeArray[i].speed;
            if (flakeArray[i].y > context.canvas.height)
                flakeArray[i].y = -5;
            flakeArray[i].x += flakeArray[i].drift;
            if (flakeArray[i].x > context.canvas.width)
                flakeArray[i].x = 0;
        }
    }
}

function draw() {
    context.save();
    // create a clipping region on buffer canvas
    bufferCanvasCtx.beginPath();
    bufferCanvasCtx.fillStyle="black";
    bufferCanvasCtx.fillRect(0,0,bufferCanvas.width,bufferCanvas.height);
    //bufferCanvasCtx.fillStyle="#330033";
    bufferCanvasCtx.arc(bufferCanvas.width/2,
    bufferCanvas.height/2,bufferCanvas.height/2,0,2*Math.PI);
    bufferCanvasCtx.fill();
    bufferCanvasCtx.clip();

    blank();
    // draw all snowflakes on buffer canvas
    for (var i = 0; i < flakeArray.length; i++) {
        bufferCanvasCtx.beginPath();
        bufferCanvasCtx.fillStyle = "white"; //"skypink";
        bufferCanvasCtx.fillRect(flakeArray[i].x, flakeArray[i].y, flakeArray[i].width,
        flakeArray[i].height);
    }
    // Double buffering
    // copy the entire rendered image from the buffer canvas to the visible one
    context.drawImage(bufferCanvas, 0, 0, bufferCanvas.width, bufferCanvas.height);

    context.restore();
}
```

Snowy night : final result

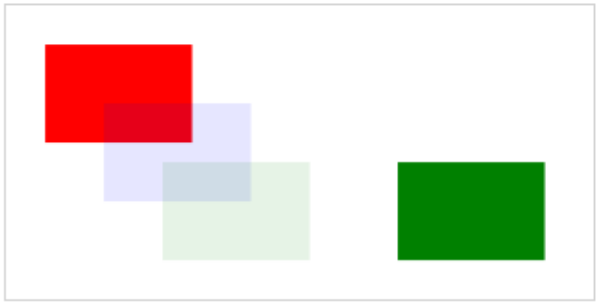


[복습] Context save() & restore().

Edit This Code: See Result »

```
<!DOCTYPE html>
<html>
<body>

<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(20, 20, 75, 50);
ctx.save();

//Turn transparency on
ctx.globalAlpha = 0.1;
ctx.fillStyle = "blue";
ctx.fillRect(50, 50, 75, 50);
ctx.fillStyle = "green";
ctx.fillRect(80, 80, 75, 50);

ctx.restore();
ctx.fillStyle = "green";
ctx.fillRect(200, 80, 75, 50);

</script>

</body>
</html>
```

Result:



globalAlpha : 0.1 1.0

과제04. msnn_rpt04.zip

[실습과제04] Christmas card
[1] Bouncing ball 완성. – msnn_bouncing.html
[2] 캔버스와 자바스크립트만을 사용해서 크리스마스 카드 만들기
- 사각형 눈송이의 모양을 원형으로 변경
- 크리스마스 관련 그림을 활용 (배경 또는 애니메이션 추가)
- msnn_rpt04,html
**** html 파일 및 관련 파일(image 등..)을
MSnn_Rpt04.zip 으로 압축해서 제출하시오.

파일명: msnn_bouncing.html, msnn_rpt04.html

가점: Javascript 프로그래밍 응용 능력.

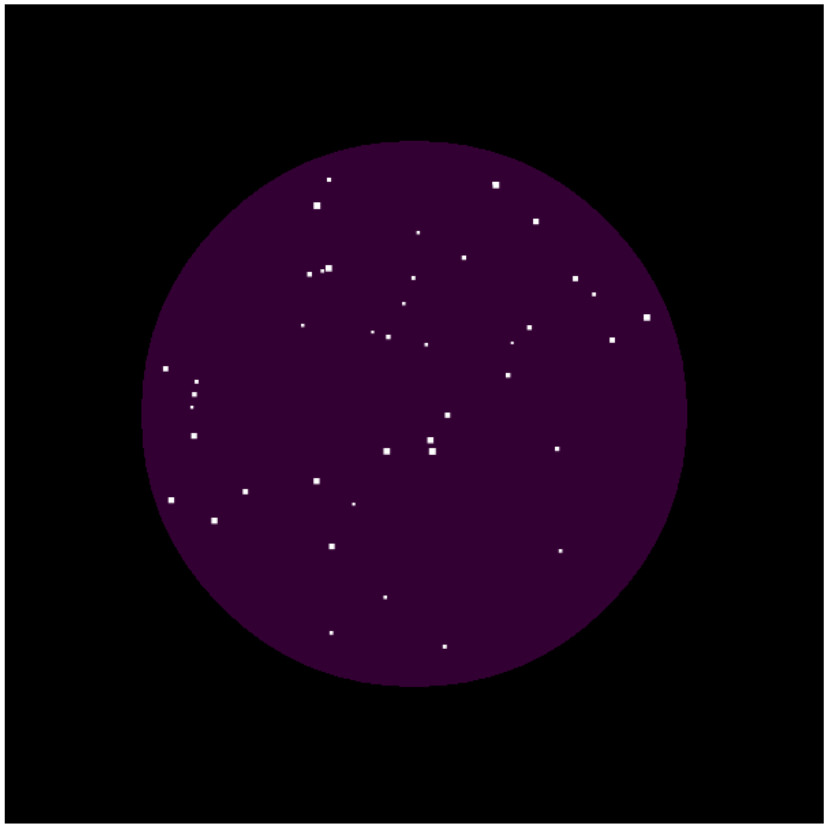
[제출파일] msnn_rpt04.zip

두 개의 html 파일과 사용된 그림을 이메일로 제출

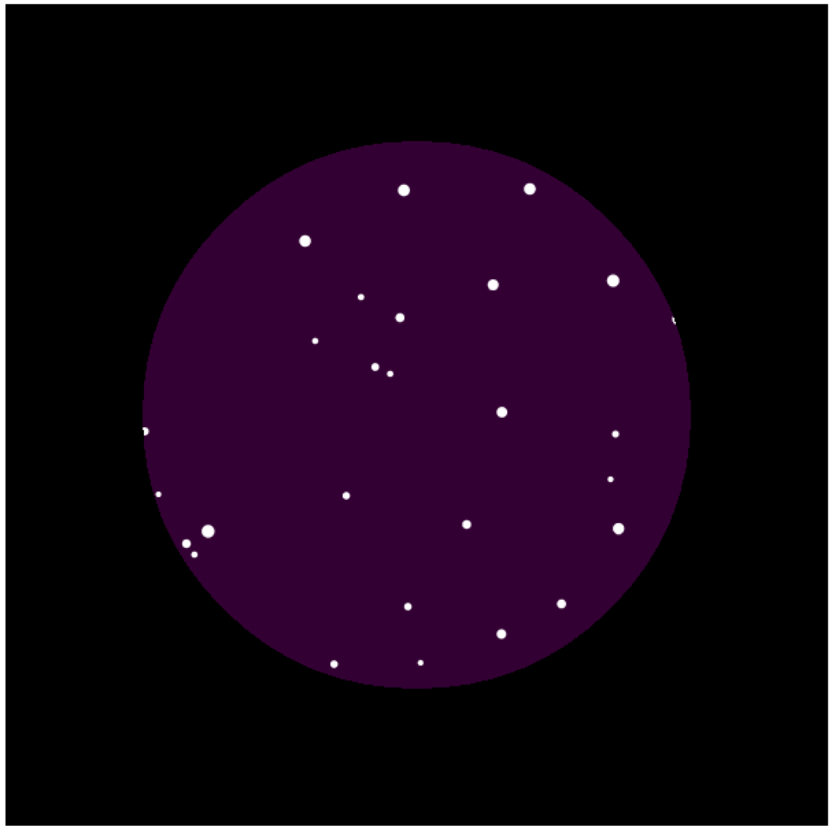
Email : chaos21c@gmail.com

과제04. result

Canvas simulation : Double Buffering



Canvas simulation : Double Buffering



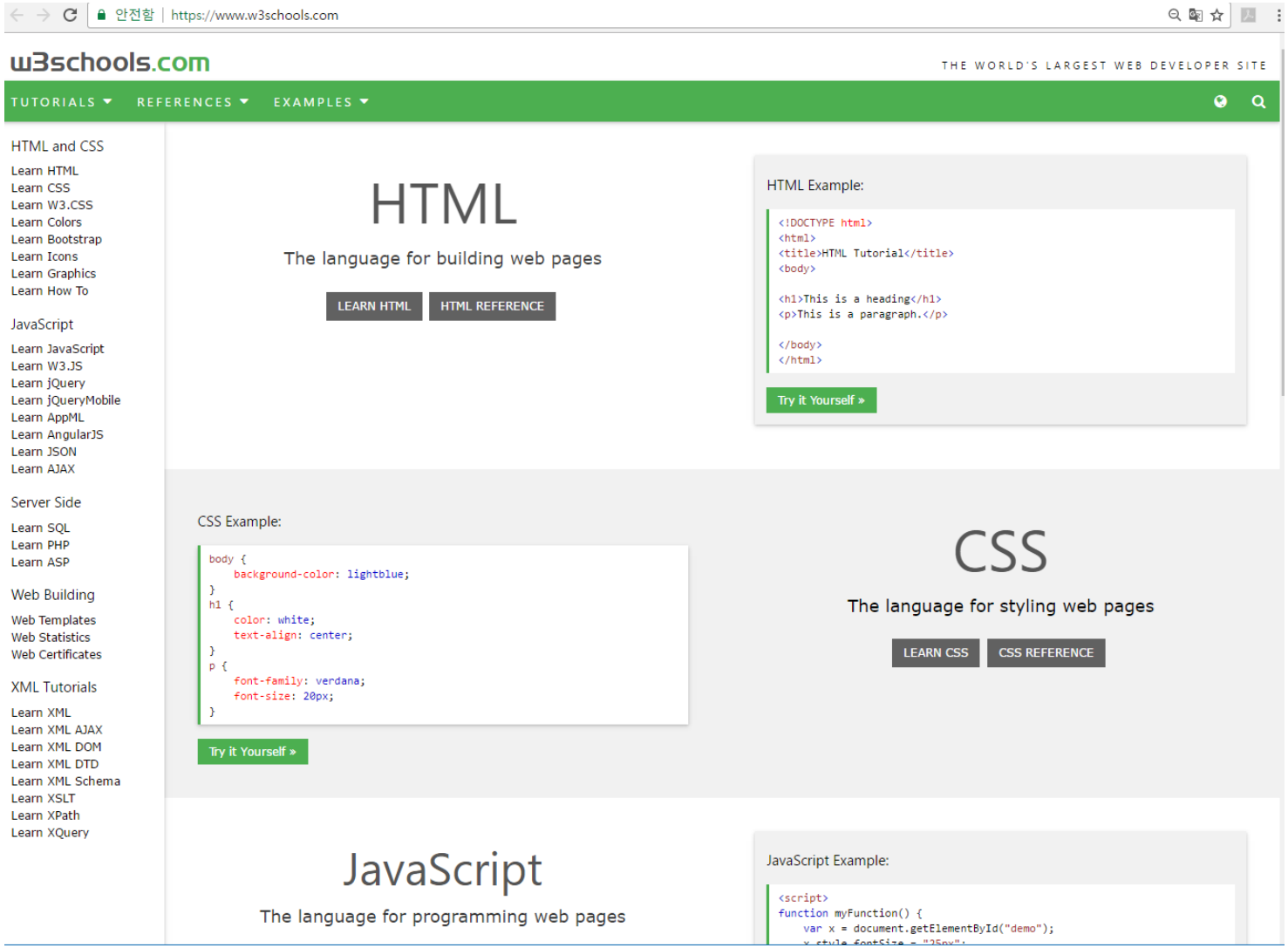
Round snowflakes and clipping effect

과제04. possible results

교재 WEB 강의 소개

http://webprogramming.co.kr

관련 WEB 강의 소개 – w3schools.com



http://www.w3schools.com