```

console.log("js file has been called");




// Now this line will be the same size on the page

canvas = document.querySelector('#myCanvas');
var ctx = canvas.getContext('2d');
var width = 800;
var height = 600;
canvas.width = width;
canvas.height = height;

console.log(width);
console.log(height)
// rgb(0,0,0) rgb(153,153,153) rgb(255,255,255)
// rgb(204,0,0) rgb(255,204,51) rgb(51,51,255)
// rgb(255,102,102) rgb(255,255,153) rgb(0,153,204)
// draw rectangle
ctx.fillStyle='rgb(0,153,204)';
ctx.strokeStyle='rgb(0,0,0)';
ctx.lineWidth=10;
ctx.beginPath();
ctx.rect(10,10,100,100);
ctx.stroke();
ctx.fill();

// draw circle
ctx.fillStyle='rgb(255,204,51)';
ctx.strokeStyle='rgb(51,51,255)';
ctx.lineWidth=10;
ctx.beginPath();
ctx.arc(200,60, 50, 0, 2*Math.PI);
ctx.stroke();
ctx.fill();


// add text
ctx.fillStyle="rgb(0,0,255)";
var myFont= "bold 30px monospace";
ctx.font=myFont;
ctx.fillText("Hello World", 300,50);
/*
var BoxImg = new Image();   // Create new img element
BoxImg.src = 'image_test.png'; // Set source path
```

```
ctx.drawImage(BoxImg, 600,10,100,100);
*/

// draw line
ctx.strokeStyle="rgb(255,0,0)";
ctx.lineWidth=0.5;
ctx.beginPath();
ctx.moveTo(0, 200);
ctx.lineTo(750,200);
ctx.stroke();

//draw polyline with closure
ctx.strokeStyle="rgb(255,102,102)";
ctx.fillStyle="rgb(255,255,153)";
ctx.lineWidth=10;
ctx.beginPath();
ctx.moveTo(0, 250);
ctx.lineTo(500,250);
ctx.lineTo(700,300);
ctx.lineTo(400,300);
ctx.closePath();
ctx.stroke();
ctx.fill();

// draw shape with a gradient
var my_gradient=ctx.createLinearGradient(10,350,10,550);
my_gradient.addColorStop(0,"rgb(255,102,102)");
my_gradient.addColorStop(0.5,"rgb(255,255,153)");
my_gradient.addColorStop(1,"rgb(0,153,204)");
ctx.fillStyle=my_gradient;
ctx.beginPath()
ctx.rect(10,350, 200,200);
ctx.fill();
ctx.stroke();

// quadratic curves (bezier)
ctx.strokeStyle="rgb(255,0,0)";
ctx.beginPath();
ctx.moveTo(300,400);
ctx.lineWidth=10;
ctx.quadraticCurveTo(500, 550, 700, 400);
ctx.stroke();



// draw circle
ctx.fillStyle='rgb(255,204,51)';
ctx.strokeStyle='rgb(51,51,255)';
ctx.lineWidth=10;
ctx.beginPath();
ctx.arc(width/2,height/2, 50, 0, 2*Math.PI);
ctx.stroke();
ctx.fill();
```

Listing 1: Python example

1 Functions

```javascript
/**
 * Draw a rectangle
 *
 * @param {number} x corner x
 * @param {number} y corner y
 * @param {number} w width
 * @param {number} h height
 * @param {string} fillcolour rgb string
 * @param {string} strokecolour rgb string.
 * @param {number} strokewidth x coordinate of second point.
 * @return {null}
 */
function drawRect(x,y,w,h, fillcolour, strokecolour, strokewidth){
    ctx.fillStyle = fillcolour;
    ctx.strokeStyle = strokecolour;
    ctx.lineWidth = strokewidth;
    ctx.beginPath()
    ctx.rect(x,y,w,h)
    ctx.fill();
    ctx.stroke();
}
// call the function to make a rectangle
drawRect(700,100,250, 450, "rgb(240, 100, 80)", "rgb(0, 100, 80)", 3)
/**
 * Draw a circle
 *
 * @param {number} x corner x
 * @param {number} y corner y
 * @param {number} r radius
 * @param {string} fillcolour rgb string
 * @param {string} strokecolour rgb string.
 * @param {number} strokewidth x coordinate of second point.
 * @return {null}
 */
 function drawCircle(x,y,r, fillcolour, strokecolour, strokewidth){
    ctx.fillStyle = fillcolour;
    ctx.strokeStyle = strokecolour;
    ctx.lineWidth = strokewidth;
    ctx.beginPath()
    ctx.arc(x,y,r, 0, 2*Math.PI)
    ctx.fill();
    ctx.stroke();
}
drawCircle(700,500,50, "rgb(0, 255, 80)", "rgb(0, 100, 255)", 8)
/**
```

```
 * Draw a white line between two points
 *
 * @param {number} x_1 x coordinate of first point.
 * @param {number} y_1 y coordinate of first point.
 * @param {number} x_2 x coordinate of second point.
 * @param {number} y_2 y coordinate of second point.
 * @return {null}
 */
function draw_line(x_1, y_1, x_2,y_2){
    ctx.strokeStyle="rgb(255,255,255)";
    ctx.lineWidth=0.25;
    ctx.beginPath();
    ctx.moveTo(x_1, y_1);
    ctx.lineTo(x_2,y_2);
    ctx.stroke();
}
draw_line(0,400, 600,100)
//use the drawline method to make a grid
/**
 * Draw a grid  line between two points
 *
 * @param {number} n width and height of each grid square
 * @return {null}
 */
function draw_grid(n){
    var grid_interval = n;
    for(var i=0; i< width/grid_interval; i++){
        draw_line(i*grid_interval,0,i*grid_interval,height);
    }
    for(var i=0; i< height/grid_interval; i++){
        draw_line(0,i*grid_interval,width,i*grid_interval);
    }
}
// call the function and draw the grid
draw_grid(50);
/**
 * Draw a white line between two points
 *
 * @param {number} x_1 x coordinate of first point.
 * @param {number} y_1 y coordinate of first point.
 * @param {number} x_2 x coordinate of second point.
 * @param {number} y_2 y coordinate of second point.
 * @return {null}
 */
function text_box(x,y,w,h, bCol, tCol, message){
    ctx.fillStyle=bCol;
    ctx.strokeStyle='rgb(255,255,255)';
    ctx.lineWidth=1;
    //create and fill-draw the rectangle
    ctx.beginPath();
    ctx.rect(x,y,w,h);
    ctx.fill();
    ctx.stroke();
    // reset the context for the text color
    ctx.fillStyle=tCol;
    var myFont= "bold 25px monospace";
    // position and draw text in middle of box
    ctx.font=myFont;
    ctx.textBaseline = 'middle';
    ctx.textAlign = "center";
    var output = message;
    ctx.fillText(output, x+w/2,y+h/2);
}
// create one text box
text_box(0,0,300,50, "rgb(100,200,0)", "rgb(255,255,255)", "Little Text");
// create a set using an array and a loop
box_list = ["hello", "goodbye", "see you"]
box_height = 50;
for(var i =0 ; i<box_list.length; i++){
text_box(300,200+i*box_height,300,box_height, "rgb(0,0,100)", "rgb(255,255,255)",
    box_list[i]);
}

/**
 * Draw a rectangle with rounded edges
 *
 * @param {number} x_1 x coordinate of first point.
 * @param {number} y_1 y coordinate of first point.
 * @param {number} x_2 x coordinate of second point.
 * @param {number} y_2 y coordinate of second point.
 * @return {null}
 */
function rounded_rectangle(x,y,w,h, bCol = "rgb(0,0,255)"){
    console.log("function called")
    ctx.fillStyle=bCol;
    ctx.lineWidth=1;
    // corner radius cannot be more than half the height
    var rad = 100;
    if(rad > h/2){
        rad = h/2;
    }
    ctx.beginPath();
    // draw in order the 4 quater circles of the rounded rectangle edges
    // straight lines will autaomatically connect them
    ctx.arc(x+rad,y+rad, rad, Math.PI,3*Math.PI/2 );
    ctx.arc(x+w-rad,y+rad, rad, 3*Math.PI/2,0 );
    ctx.arc(x+w-rad,y+h-rad, rad,0,Math.PI/2 );
    ctx.arc(x+rad,y+h-rad, rad,Math.PI/2,Math.PI );
    ctx.closePath();
    ctx.fill();
    ctx.stroke();
}
rounded_rectangle(50,100,200,50);
```

Listing 2: Python example

2 Draggable Point

```javascript
console.log("point js file has been called");
class Point{
// class Point x,y,r, stroke, fill, over, canvas
constructor(x,y,r, stroke, fill, over){
    //basic position, size and colours
    this.x = x;
    this.y = y;
    this.r = r;
    this.stroke = stroke;
    this.fill = fill;
    this.over = over;
    //set true if mouse inside point circle
    this.inBounds = false;
    //cointinually registered mouse position
    this.xMouse = 0;
    this.yMouse = 0;
    //listeners
    canvas.addEventListener('mousedown', this.mDown.bind(this));
    canvas.addEventListener('mousemove', this.mMove.bind(this));
    canvas.addEventListener('mouseup', this.mUp.bind(this));
}
mDown(e){
    // if the mouse is pressed (goes down) and the mouse is inside the point
   circle,
    // set the this object as taken
    if(this.inBounds){
        Point.taken = this;
    }
}
mMove(e){
    // event registered every time the mouse moves
    // object variables updated with current mouse position
    this.xMouse = e.offsetX;
    this.yMouse = e.offsetY;
    //update boundary boolean
    this.inBounds = this.boundsCheck(this.xMouse, this.yMouse, this.x, this.y,
   this.r);
}
mUp(e){
    //when mouse goes up set taken point as nothing
    //hence deselect this point
    Point.taken = "";
}
/**
 * called from animation loop
 */
update(){
// make x,y coordinates of the point the same as the mouse position
// if the point has been taken
    if(Point.taken == this){
        this.x=this.xMouse;
        this.y=this.yMouse;
    }
    this.draw();
}
draw(){
    // change fill state if mouse is over or the point is selected
    if(this.inBounds || Point.taken == this){
    ctx.fillStyle= this.over;
    }else{
        ctx.fillStyle= this.fill;
    }
    ctx.strokeStyle = this.stroke;
    ctx.lineWidth = 2;
    ctx.beginPath()
    ctx.arc(this.x,this.y, this.r, 0, 2*Math.PI);
    ctx.fill();
    ctx.stroke();
}
/**
 * Pythagoras distance check
 * @param  x,y,positions of mouse and of point circle and radius of point circle
   (number)
 * @return boolean
 */
boundsCheck(x_1, y_1, x_2, y_2, r){
        var d = Math.sqrt( Math.pow(x_2 - x_1, 2) + Math.pow(y_2 - y_1, 2) );
        if(d<r){
            return true;
        }else{
            return false;
        }
}
/**
 * Make x, y coordinates of point available outside of object
 * @return number
 */
getX(){
    return this.x;
}
getY(){
    return this.y;
}
}
// static variable available to all Point objects
// the same for all Point objects
// means only one Point can be selected and moveable
Point.taken="";
```

Listing 3: Python example