Javascript Lab 07

Canvas:- Pong Game

1. Create an HTML page called “pong.html” and put in a canvas tag

```
<canvas id="board" width="400" height="400">does not support canvas</canvas>
```

2. Write a script that will draw a rectangle in the canvas

```
var canvas = document.getElementById("board");
var ctx = canvas.getContext("2d");
ctx.fillRect(20, 20, 30, 50);
```

3. Create a function called drawAll(), that can be called whenever we need to redraw the canvas. TEST putting in the rectangle code and calling it.

```
function drawAll(){
        var canvas = document.getElementById("board");
        var ctx = canvas.getContext("2d");
        ctx.fillRect(20, 20, 30, 50);
}
drawAll();
```

4. Create a function called paddleDraw(ctx) that takes in the context and draws the rectangle, and call it from drawAll().

```
function drawAll(){
        var canvas = document.getElementById("board");
        var ctx = canvas.getContext("2d");
        paddleDraw(ctx)
}
function paddleDraw(ctx){
    ctx.fillRect(20, 20, 30, 50);
}
```

5. Create a paddle1 object that will store all the attributes for player1 including the paddleDraw function: eg
    i. x,y coordinates
    ii. vx, vy speed
    iii. width and height
    iv. the draw Function
    v. the tick Function
    vi. the handleKeyDown function
    vii. the handleKeyUp function
    viii. the up and down keyCodes

```
var paddle1 = {x:10,y:100,vx:0,vy:0,w:10,h:40,
    tick:undefined,
    draw:paddleDraw,
    handkeKeyDown: undefined,
    handleKeyUp: undefined,
    downKey:83,
    upKey:87};
```

6. Modify drawAll() so that it uses the paddle1 object to call draw function.

```
paddle1.draw(ctx);
```

7. Modify paddleDraw (ctx) to take the attributes from this object (as opposed to being hard coded as before.

```
ctx.fillRect(this.x, this.y, this.w, this.h);
```

8. Create a function called tickAll(), have it get the canvas and call drawAll();

```
function tickAll(){
    var canvas = document.getElementById("board");
    drawAll();
}
```

9. Use interval to call this function 60 times a second (1000/60). Test this code by putting a console.log into the tickAll() function. (make sure this code is going to be called when the page is loaded, ie it is not in any other function).

```
setInterval(tickAll,1000/60);
```

10. Create a function called paddleTick(canvas) that take in the canvas as a parameter, put it into the paddle1 object as the tick function, call it from the tickAll() function. **TEST IT.**

```
function paddleTick(canvas){
    //console.log(this.vy);
}
```

```
tick:paddleTick,
```

```
paddle1.tick(canvas);
```

11. In the paddleTick(canvas) function put in the code that will increment the this.y by this.vy.

```
this.y += this.vy;
```

TEST IT;- by putting in a value into vy and running the code the paddle should move.

12. Did it behave in the way you expected? Do you need to put in a clearRect into the drawAll().
13. How will you handle when the paddle reaches the sides of the canvas?

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

```
if (this.y < 0-this.h){this.y= canvas.height;}
if (this.y > canvas.height){this.y= 0- this.h;}
```

Dealing with user input

14. Write the code that will handle a keyDown event for the window (**TEST IT**:- with a console.log()).

```
window.addEventListener("keydown",function(event){
  console.log(event.keyCode);
});
```

15. Write a function called paddleKeyDown(keyCode), that takes in a keyCode. This function should detect if the keyCode matches this paddles up or down keycodes, and set vy accordingly.

```
function paddleKeyDown(key){
    if (key == this.downKey){this.vy = options.paddleY;}
    if (key == this.upKey){this.vy = -options.paddleY;}
}
```

16. Put this function into the paddle1 object and call it from keyDown event handler.

```
handleKeyDown:paddleKeyDown,
```

```
paddle1.handleKeyDown(key);
```

17. Do the same for keyUp.
    a. Make the key up handler

```
window.addEventListener("keyup",function(event){
  console.log(event.keyCode);
});
```

    b. Make the paddleKeyUp(keyCode) function

```
function paddleKeyUp(key){
    if (key == this.downKey){this.vy = 0;}
    if (key == this.upKey){this.vy = 0;}
}
```

    c. Put it into the object

```
handleKeyUp:paddleKeyUp,
```

    d. Call it from the handler

```
paddle1.handleKeyUp(key);
```

18. Make a second paddle object called paddle2 with different attributes and put it into the:
    a. drawAll() function
    b. tickAll() function
    c. keyDown handler
    d. keyUpHandler
19. **TEST IT**

Make the Ball

20. Create a Ball object that the following attributes
    i. x,y coordinates
    ii. vx, vy speed (set the them to something)
    iii. radius
    iv. the draw Function
    v. the tick Function

```
var ball={x:200,y:200,r:20,vx:1,vy:1.5,
    tick:undefined,
    draw:ballDraw};
```

21. Create a function called ballDraw(ctx) that draws the ball, and put it into the ball object

```
function ballDraw(ctx){
    ctx.beginPath();
    ctx.arc(this.x,this.y,this.r,0,2*Math.PI);
    ctx.fill();
}
```

22. Call the draw Function for the ball from drawAll(). **TEST IT**

```
ball.draw(ctx);
```

23. Create a function called tickBall(canvas), put it into the ball object as the tick function and call it from the tickAll(). **TEST IT**:- by putting a console.log into it.

```
function ballTick(canvas){
   console.log("ball ticking");
}
```

```
tick:ballTick,
```

```
ball.tick(canvas);
```

24. OK! Ball functionality, in the ballTick() function:-

    a. Move the ball

```
this.x+=this.vx;
this.v+= this.vv;
```

    b. Have the ball bounce of the top and bottom of the canvas

```
if (this.y < this.r){this.vy = Math.abs(this.vy);}
if (this.y >(canvas.height- this.r)){this.vy = -Math.abs(this.vy);}
```

    c. If the ball goes over the sides put it back in the middle.

```
if (this.x < 0 - this.r){
    this.x= 200;
    this.y= 200;
    this.vx = Math.abs(this.vx);
}
if (this.x > (canvas.width + this.r)){
    this.x= 200;
    this.y= 200;
    this.vx = -Math.abs(this.vx);
}
```

    d. Check if the ball hits the paddle, and “bounce it” if it does.

```
if (checkCollision(this, paddle2)){
    this.vx = -Math.abs(this.vx)
}
if (checkCollision(this, paddle1)){
    this.vx = Math.abs(this.vx)
}
```

```
function checkCollision(ball, paddle){
    var bx = ball.x;
    var by = ball.y;
    var r = ball.r;
    var px = paddle.x;
    var py  = paddle.y;
    var w = paddle.w;
    var h = paddle.h;

    if ((bx> px-r)&& (bx<px+w+r)){
        if ((by>py-r)&&(by<(py+h+r))){
            return true;
        }
    }

    return false;
}
```

Score

25. Make two score objects that have the attributes
    i. x,y coordinates
    ii. score
    iii. the draw Function called drawScore

```
var score1 = {x:20, y:10,score:0,
    draw:scoreDraw};
var score2 = {x:320, y:10,score:0,
    draw:scoreDraw};
```

26. Make the drawScore(ctx) function and put it in and call it from drawAll()

```
function scoreDraw(ctx){
    ctx.fillText(""+this.score, this.x, this.y)
}
```

```
score1.draw(ctx);
score2.draw(ctx);
```

27. Modify ballTick() so that when the ball goes out the left side it increments the score2 and score1 when it goes out the right side.

```
score2.score++;
```

28. How would you improve this game?

Happy playing