

Brooklyn Wakefield

Mrs.Awde

AOIT 2

1/4/2019

JavaScript Final Project

```
<!doctype html>
```

```
<html>
```

```
<head>
```

```
<meta charset="utf-8">
```

```
<link type="text/css" rel="stylesheet" href="style.css" />
```

```
<script type="text/javascript" src="controls.js"></script>
```

```
<script type="text/javascript" src="particles.js"></script>
```

```
<script type="text/javascript" src="ship.js"></script>
```

```
<script type="text/javascript" src="animation.js"></script>
```

```
<title>My Spaceship</title>
```

```
</head>
```

```
<div class="floatright">
```

```
<div align = "center">
```

```
<br>
```

```
<br>
```

```
<br>
```

```
<br>
```

```
<ul><h1> How to Play: </h1></ul>
```

```
<ul>Use the arrow keys to navigate the ship through the asteroid field</ul>
```

```
<ul>Avoid asteroids to maintain the ship's health</ul>
```

```
<ul>Get the highest time before the ship's health reaches 0%</ul>
```

```
<ul>Refresh to start over</ul>
```

```
<br>
```

```
<ul><h1>Your time: </h1></ul>
```

```
<ul><p><span id="countdowntimer">0</span> Seconds</p></ul>
```

```
</div>
```

```
</div>
```

```
<div class="wrap">
```

```
<div class="floatleft">
```

```
<body>
```

```
<canvas id="etchasketch" width = "800" height = "800">
```

```
</canvas>
```

```
</div>
```

```
</div>
```

```
</body>
```

```
</html>
```

Style.css

```
@charset "utf-8";
```

```
/* CSS Document */
```

```
body {
```

```
    background-image: url("bodyback.png");
```

```
    font-size: 20px;
```

```
    font-family: Impact;
```

```
    color: #ffffff;
```

```
}
```

```
#etchasketch{
```

```
border: 10px solid #ffffff;

background-image: url("newb.jpg");

}

.wrap {

width: 100%;

}

.floatleft {

float: left;

width: 50%;

height: 400px;

}

.floatright {

float: right;

height: 400px;

width: 38%;

}
```

Particles.js

```
function Particle(_x, _y, _vx, _vy, _radius)
```

```
{
```

```
//these are the attributes:
```

```
    this.x = _x;
```

```
    this.y = _y;
```

```
    this.radius = _radius;
```

```
    this.vy = _vy;
```

```
    this.vx = _vx;
```

```
    this.color = "rgb("+
```

```
        Math.round(Math.random() * 255)+
```

```
        "," +
```

```
        Math.round(Math.random() * 255)+
```

```
        "," +
```

```
        Math.round(Math.random() * 255)+
```

```
        ")";
```

```
this.startY = this.y;
```

```
this.startX = this.x;
```

```
//to store the distance from the particle to the ship
```

```
this.distance = 0;
```

```
this.draw = function()
```

```
{
```

```
    context.beginPath();
```

```
    context.fillStyle= this.color;
```

```
    var asteroid = new Image();
```

```
    asteroid.src='a.png';
```

```
    context.drawImage(asteroid,this.x,this.y,this.radius,this.radius);
```

```
    context.fill();
```

```
context.closePath();
```

```
}
```

```
this.move = function()
```

```
{
```

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

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

```
    this.reset();
```

```
}
```

```
this.reset = function ()
```

```
{  
    if (this.x <= 0)  
  
    {  
  
        this.x = this.startX;  
  
        //this.y = Math.round(Math.random() * canvas.height);  
  
        //a number between 5 and 14  
  
        //this.radius = Math.random() * 10 + 5;  
  
        //a number between 5 and 19  
  
        //this.vx = -Math.random() * 15 + 5;  
  
    }  
}  
  
//_obj is holding the space for the name of the ship  
  
this.collision = function(_obj)
```



```

{

    //Here we need to determine if there is collision.

    //We do this by calculating whether the distance between object A and
object B

    //is less than their combined radii. If so, they are touching.

    //So, what we need first is a smaller function that will give us the distance
between

    //the two objects (ship and particle).

    var dx = _obj.x - this.x;
    var dy = _obj.y - this.y;

    //Then we use the Pythagorean theorem to calculate the distance

    this.distance = Math.sqrt (dx * dx + dy * dy);

    //if the distance is less than the radius of the particle, then we have a collision

    if(this.distance < this.radius)

        {

```

```
//we use a boolean return to say that the collision occurred.
```

```
//this boolean will be used in the animation script
```

```
return true;
```

```
}
```

```
//if it doesn't collide, it will return a false
```

```
return false;
```

```
}
```

```
}
```

Ship.js

```
//declaring a function called Ship
```

```
//_x and _y are placeholder variables
```

```
//those values will be determined in the animation file, and then every _x and _y will be replaced  
with that number
```

```
function Ship(_x, _y)

{

//it has the following properties


this.x = _x;
this.y = _y;


//ax and ay are acceleration
this.ax = 1;
this.ay = 1;
this.vx = 0;
this.vy = 0;
this.radians = 0;
this.degrees = 0;
this.power = 1;


//it has the following methods

//it can move


this.move = function()

{
```

```
}
```

```
//and it can draw
```

```
this.draw = function ()
```

```
{
```

```
    //saves the current status of the context, so we can use the starting point later
```

```
    context.save();
```

```
    //takes the point of origin and moves it to the x and y
```

```
    //this code must be removed in order for the collision function to work
```

```
    //context.translate(this.x, this.y);
```

```
    //draw black line
```

```
    context.strokeStyle = "#000000";
```

```
//begin drawing
```

```
context.beginPath();
```

```
var ship = new Image();
```

```
ship.src='ship.gif';
```

```
context.drawImage(ship,this.x,this.y,100,50);
```

```
context.closePath();
```

```
context.stroke();
```

```
}
```

```
}
```

Controls.js

```
var up = false;
```

```
var down = false;
```

```
var left = false;
```

```
var right = false;
```

```
window.onkeydown = function(e){
```

```
    if(e.keyCode == 38)
```

```
    {
```

```
        up = true;
```

```
    }
```

```
    if(e.keyCode == 40)
```

```
    {
```

```
        down = true;
```

```
    }
```

```
    if(e.keyCode == 37)
```

```
    {
```

```
        left = true;
```

```
    }
```

```
        if(e.keyCode == 39)
        {
            right = true;
        }
    }

    window.onkeyup = function(e){
```

```
        if(e.keyCode == 38)
        {
            up = false;
        }
```

```
        if(e.keyCode == 40)
        {
            down = false;
        }
```

```
        if(e.keyCode == 37)
        {
            left = false;
```

```
    }

    if(e.keyCode == 39)
    {
        right = false;
    }

}
```

Animation.js

```
var canvas;
```

```
var context;
```

```
//create an instance of the Ship class
```

```
//the coordinates in this instance will pass into the _x and _y in the ship file
```

```
var ship = new Ship(100, 100);
```

```
//evaluate math lower in the program to see how these numbers will impact movement
```

```
//for moving shapes friction of .85 and power of 2 seem to work pretty well on average
```



```
var friction = .85;
```

```
ship.power = 2;
```

```
var count = 100;
```

```
//the higher the gravity number, the faster the object will fall
```

```
gravity = .2;
```

```
//creating a new array to contain the particles
```

```
//because we're playing a game and trying to destroy particles,
```

```
//we'll put fewer the array at a time so that there are fewer to hit
```

```
var asteroids = new Array();
```

```
//var amount is going to be used in the array instead of a maximum like 100
```

```
var amount = 30;
```

```
window.onload = function()
```

```
{
```

```
    canvas = document.getElementById("etchasketch");
```

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

```
    //generating amount (10) new particles and storing them in the array 0-amount minus - 1
```

```
    //each iteration generates a new series of properties of x, y, velocity, radius, etc.
```

```
    for(var i = 0; i < amount; i++)
```

```
    {
```

```
        var x = canvas.width + Math.random() * canvas.width;
```

```
        var y = Math.random() * canvas.height;
```

```
        //going left which is negative
```

```
        var vx = -5;
```

```
        //no movement down the screen. Particles are only moving from right to left
```

```
        var vy = 0;
```

```
        //generating a variety of particle sizes
```

```
var radius = Math.random() * 50 + 10;
```

```
//taking these properties that have been generated and passing them through to the  
particles class creating a new particle
```

```
asteroids[i] = new Particle(x, y, vx, vy, radius);
```

```
}
```

```
var interval = setInterval ("animate()", 1000/30);
```

```
}
```

```
//Timer
```

```
var timeleft = 0;
```

```
var downloadTimer = setInterval(function(){
```

```
timeleft++;
```

```
document.getElementById("countdowntimer").textContent = timeleft;
```

```
//pause at 0 health
```

```
if(count == 0)
```

```
clearInterval(downloadTimer);  
},1000);
```

```
function animate ()
```

```
{
```

```
//starting in upper left corner (0,0) it clears the entire canvas to its height and width
```

```
//it clears the screen every time the animation function is called and then redraws the ship and  
asteroids at their new location
```

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

```
if(right == true)
```

```
{
```

```
//power is how hard are you stepping on the gas
```

```
ship.vx += ship.ax * ship.power;
```

```
}
```

```
if(left == true)

{

    //negative acceleration is to go in the left direction


    ship.vx += ship.ax * -ship.power;


}


if(up == true)

{

    ship.vy += -ship.ay * 1.5 * ship.power;

}


if(down == true)

{

    ship.vy += ship.ay * ship.power;

}


if (ship.x > canvas.width)
```

```
{  
    ship.x = canvas.width;  
}
```

```
if (ship.x < canvas.width - 800)  
{  
    ship.x = canvas.width - 800;  
}
```

```
if (ship.y > canvas.height )  
{  
    ship.y = canvas.width ;  
}
```

```
if (ship.y < canvas.width - 800)  
{  
    ship.y = canvas.width - 800;  
}
```

```
ship.vx *= friction;
```

```
ship.vy *= friction;
```

```
ship.vy += gravity;
```

```
ship.x += ship.vx;
```

```
ship.y += ship.vy;
```

```
ship.draw();
```

```
for(var i = 0; i < amount; i++)
```

```
{
```

```
    asteroids[i].move();
```

```
    asteroids[i].draw();
```

```
    //if this collision function has a true value will increment count and remove
```

```
particle
```

```
    //if not, it won't
```

```

if (asteroids[i].collision(ship))

    {

        //if it is true, the counting variable increments

        //and the particle clears the screen and returns to startX


        count=count-10;

        asteroids[i].x = asteroids[i].startX;


    }

}

//////////

if(count == 0)

    {

        context.fillStyle = "#FF0000";

        context.font = "70px Impact";

        context.fillText ("Game Over", canvas.width/2, canvas.height/2);


    }

    else if (count < 0)

```



```
{  
    count =0;  
}  
  
//printing to screen in color 00ffff  
  
context.fillStyle = "#FF0000";  
  
//printing to screen in 20px font of Georgia  
  
context.font = "20px Impact";  
  
//print the string "Score" to screen beginning at x= 10 and y = 90  
  
context.fillText ("Health =", 10, 70);  
  
//right after the string "Score", print the contents of the count variable  
//which increments each time that there is a collision  
//if no collision has taken place, the collision function returns a value  
//of false and the counting variable does not increment  
  
context.fillText(count, 88, 70);  
context.fillText("%",118, 70);
```

}

