CSE 3342

Quest6: JavaScript Game UI

Due: Monday Apr 2 @ midnight

Download the code to jumpstart this Quest from Canvas.

Download from Canvas:

- Battlefield.html -- sets up the web page and loads the file drawcanvas.js
- drawcanvas.js -- contains the JavaScript code for drawing on the screen
- sprites.json - contains a json array of sprite object definitions
- jquery.getdata.html - contains jquery script to GET data from web

Part A.

A1. Modify drawcanvas.js so that when drawSprite(team,x,y,dir) is called a sprite is drawn in the correct position on the screen. In graphics systems, 0,0 is the upper left hand corner of the graphics screen. Team values should be 1 or 2. Make team 1 =blue and team 2 =red. You are free to draw as a rectangle or any shape but it must be contained within the grid at your x,y position.

A2. Define a JS function constructor called Sprite that can used to instantiate Sprite objects with the following properties:

```
team  : int
id    : int
x     : int
y     : int
dir   : int
speed : int
weight: int
health: int
```

```
Create an array and store 10 Sprite instances, 5 on each team.
Modify the code to display your 10 sprites on the screen. Use blue for
one team and red for the other team.
When the html page loads, you should see your sprites at their x,y
positions and colored based on their team colors.
```

Part B. Data Entry

The Battlefield.html code includes a form that can be used to get data from the user. We want the form to help interactively define sprites.

Add a button labeled LOAD SPRITE to the HTML page.

Modify the code so that when the user presses LOAD SPRITE, a new sprite, based on the form data, is added to your sprite list and gets displayed on the screen.

Deploy your HTML and JS files on the Lyle server under your account. Test your deployed code. Be sure to set permissions to allow Others to READ files.

Part C. Load data from WEB

- Use the code in jquery.getdata.html to study how jquery works. Rewrite as needed and put the jQuery code into your script file. (You can have script code in your HTML file but it is good practice to separate JS from HTML).
- Here we will use jQuery to load a JSON file from the web.
- Deploy the file sprites.json on the Lyle server and set permissions.
- Add a textfield and a button labeled LOAD SPRITES FROM WEB
- to the web page.
- Add code so that when the LOAD SPRITES button is pressed, your code uses jQuery to download the file. This will give you a JavaScript array of objects. Use this array to build new sprites to add to your sprite array.
- Display the new sprites on the board.

Submit to Canvas:

- Your HTML page
- Your JavaScript
- A text file with a link to your deployed web page.

Drawing in JavaScript

https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D

Battlefield.html

```
<!DOCTYPE html>
<html>
<head>
 <script src="drawcanvas.js"></script>
</head>

<body onload="draw()">

<h2>Gismo</h2>

<canvas id="myCanvas" width="300" height="225"></canvas>
<h3>Data Entry</h3>

<form id="frm1">
  team: <input type="text" name="team" maxlength="1" size="2">
  x:    <input type="text" name="x" value="0" maxlength="3" size="3">
  y:    <input type="text" name="y" value="0" maxlength="3" size="3">
</form>

<p>Click "Load Data" to get data from Form </p>

<button onclick="processForm()">Load Data</button>

<p id="demo"></p>

<button type="button" onclick="drawSprite(1,10,10,90)">drawSprite</button>
<button type="button" onclick="drawLine()">drawLine</button>


</body>
```

drawcanvas.js

```
1   //Global variables for drawing the grid on a JS Canvas
2   var gXmax               = 500;
3   var gYmax               = 375;
4   var gGridsize           = 20;
5
6   function draw() {
7
8            var b_canvas = document.getElementById("myCanvas");
9            var ctx = b_canvas.getContext("2d");
10
11           ctx.beginPath();
12
13           for (var x = 0.5; x < gXmax; x+=gGridsize) {
14                  ctx.moveTo(x,0);
15                  ctx.lineTo(x,gYmax);
16           }
17
18           for (var y = 0.5; y < 375; y+=gGridsize) {
19                  ctx.moveTo(0,y);
20                  ctx.lineTo(gXmax,y);
21           }
22           ctx.strokeStyle = "#e2e";
23           ctx.stroke();
24           ctx.closePath();
25
26           console.log("Grid drawn");
27  }
28
29  function drawLine() {
30          var b_canvas = document.getElementById("myCanvas");
31          var ctx = b_canvas.getContext("2d");
32
33           ctx.beginPath();
34           ctx.moveTo(40,40);
35           ctx.lineTo(200,200);
36           ctx.strokeStyle = "#000";
37           ctx.stroke();
38           ctx.closePath();
39  }
40
41  function drawSprite(team, x, y, dir) {
42       var b_canvas = document.getElementById("myCanvas");
43           var ctx = b_canvas.getContext("2d");
44
45       // todo: modify so team1 = blue and team2 = red
46           ctx.fillStyle= 'rgb(200,0,0)';
47
```

```
48        // todo: modify so a rectangle appears at appropriate x,y position
49            ctx.fillRect(10,10, gGridsize, gGridsize);
50    }
51
52    function processForm() {
53            // todo: rewrite so form data is used to create new sprite
54            //       add to spritelist and display
55        var text = "";
56
57        // obtain values from the web form
58        var x = document.forms['frm1'].elements['x'].value;
59        var y = document.forms['frm1'].elements['y'].value;
60
61        // add values to text string
62        text += "x=" + x + " y=" + y
63
64        // update web page element id='demo'
65        document.getElementById("demo").innerHTML = text;
66    }
67
```

jquery.getdata.html

```
<!DOCTYPE html>
<html>
<head>

<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

<script>
$(document).ready(function(){
    $("button").click(function(){
        $.get("sprites.json", function(dataObject, status){
            // remove alert boxes in production code
            alert("Data: " + dataObject + "\nStatus: " + status);

            processSpriteData(dataObject);
        });
    });
});

function processSpriteData(spriteData) {
    console.log("Loaded SpriteData for first sprite " + spriteData[0].x );
}

</script>
</head>

<body>

<button>Send an HTTP GET request to a page and get the result back</button>
<p id="demo"></p>

</body>
</html>
```

sprites.json

```
 [ {"x":6, "y":3, "team":1, "dir":45, "weight":3, "speed":1},
  {"x":12, "y":3, "team":1, "dir":90, "weight":3, "speed":1},
  {"x":10, "y":3, "team":1, "dir":90, "weight":3, "speed":1},
  {"x":8,  "y":3, "team":1, "dir":90, "weight":3, "speed":1},
  {"x":4, "y":3, "team":1, "dir":90, "weight":3, "speed":1},
  {"x":6, "y":13, "team":2, "dir":45, "weight":3, "speed":1},
  {"x":12, "y":13, "team":2, "dir":90, "weight":3, "speed":1},
  {"x":10, "y":13, "team":2, "dir":90, "weight":3, "speed":1},
  {"x":8,  "y":13,  "team":2, "dir":90, "weight":3, "speed":1},
  {"x":4, "y":13, "team":2, "dir":90, "weight":3, "speed":1}
]
```