COM S/SE 319 : Software Construction and User Interfaces
Fall 2018

HW 3

[Total Points: 50]

Assignment Due: Wednesday, September 26, 2018, 11:59 PM

[N.B.:5% penalty per day up to maximum of 7 days from **September 26, 2018**]

This assignment is focused on UI and event driven programming and Event Handling

Task 1: UI and Event Driven Programming: (30 points)

Objectives:

Learn to use Javascript objects, functions, and closures to implement UI and event driven programming.

Warm-up:

NOTE 1: One suggestion (to help you play with javascript) is to use online Javascript code tool like http://codepen.io/pen/ or https://jsbin.com. They are very useful for trying javascript examples as you can change the html or javascript directly on the website, and you can immediately see the results of your changes.

NOTE 2: You will need to also learn how to use the available tools for JS debugging.

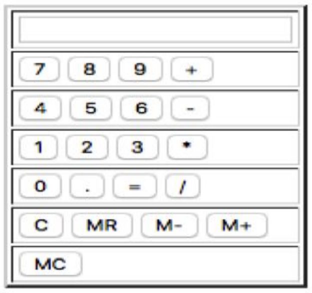
Firefox has tools->WebDeveloper->Debugger,

Chrome has Tools->Developer Tools (ctrl-shift-I).

NOTE 3: Play with each of the given examples (in the examples directory). Open them using a text editor of your choice and modify parts of the html or js files to learn how the different instructions work.

Task:

A complete example of another program (Matching game) is provided in folder **SampleProgram**. Please take a look at that one first. A starting template is provided in folder **ExerciseHelp**. Your assignment is to use this template to **create a simple decimal calculator programs** using objects, functions, and closures. This calculator should look approximately like the below picture.



You can look at a normal calculator to figure out the functionality of M+, M-, MR and MC.

For **decimal calculator**,

1. " + ", " - ", " * " , " / " should be used respectively for addition, subtraction, multiplication and division. (**2x4 = 8 points**)
2. " **.** " should be used for operation with decimals. (**3 points**)
3. Negative number operations e.g., "(-2)-3 = -5" (**3 points**)
4. Assume that the calculator does not need to calculate complex operations such as 5 + 5 * 5. Instead, expect users to press "=" operator after a basic operation. So, press 5 + 5 followed by =. At this point it should show 10. Then, press "*" and then 5 followed by "=". At this point show 50. When an operator button is pressed, the **operator button**'s font becomes **red**. In other words, assume that we are expecting user to enter only "operand1 ***operator*** operand2 = ". However, we can use the results of the previous operation as the first operand for the next operation.

Check list:

[ ] Your javascript file should be named "**calculator.js**".

[ ] Use relative path in all of your files.

[ ] Name your Objects based on their purpose. Do the same with your JavaScript functions.

[ ] Show UI Display for decimal calculator correctly. (**3 points**)

[ ] **MR** (shows memory value on screen) (**2 points**)

[ ] **MC** (clears memory value) (**2 points**)

[ ] **M+** (Whatever is on screen gets added to memory) (**2 points**)

[ ] **M-** (Whatever is on screen gets subtracted from memory) (**2 points**)

[ ] **C** (clears screen value, clear the last operation, press "=" will not repeat the last operation) (**2points**)

[ ] = (shows results of an operation) and highlight the last button (any digit/ operator) clicked (**3 points**)

[ ] Make sure that your variables are not global (so that if someone includes some other js files with same names for variables, then your code still works ok).

Task 2: Event Handling (15 points)

Write a Javascript and HTML code (named snake.html and snake.js) to implement the functionality shown in '**Problem2Output.mp4**' included in the zip file.

Note:

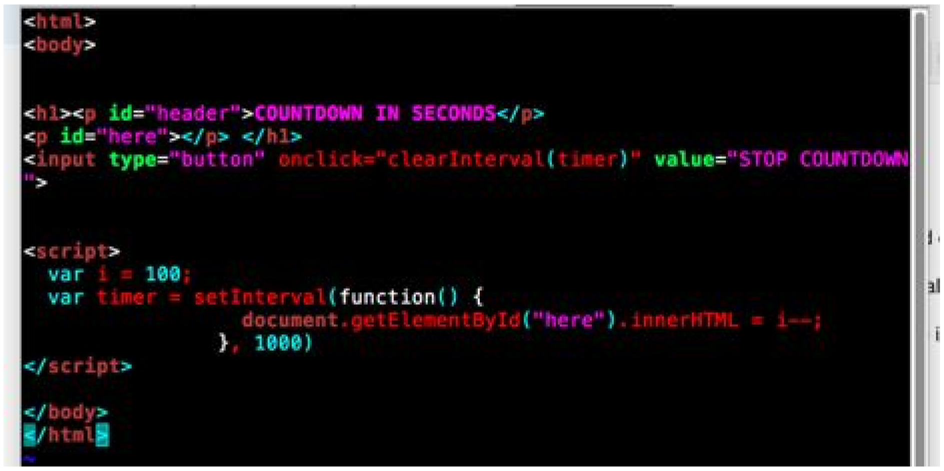
1. **The line you create can go over any previous paths. [4 points]**
2. **The line will bend left when left button is clicked. [4 points]**
3. **The line will bend right when right button is clicked. [4 points]**
4. **The line should stop if it touches any boundary. [3 points]**

Hints:

1. **Use HTML5 Canvas (see http://www.w3schools.com/graphics/canvas_intro.asp)**
2. **Make sure to use a timer (see example below) to update the canvas (so that the snake keeps moving). A Timer has two main functionalities that can be used in the project.**

a. **The setInterval(function, delay) schedules the "code" after every "delay" microseconds.**
b. **The clearInterval removes the timer**

Here is an example of timer code. This will countdown from 100 until you press stop!



```
<html>
<body>


<h1><p id="header">COUNTDOWN IN SECONDS</p>
<p id="here"></p> </h1>
<input type="button" onclick="clearInterval(timer)" value="STOP COUNTDOWN
">


<script>
  var i = 100;
  var timer = setInterval(function() {
                document.getElementById("here").innerHTML = i--;
              }, 1000)
</script>

</body>
</html>
```

__What to Submit:__

Make sure your solutions work on Chrome as TAs will use it to grade the assignment.

Submit via Canvas a **compressed file (.zip)** containing the following:

- ***lab.html***, ***calculator.js***, for **Task 1** and ***snake.html*** and ***snake.js*** for **Task 2**. **[Task 1+Task 2 = 30+15 = 45 Points]**
- README file explaining how to compile and run your program & a **Report** (.docx or .pdf) describing your solution approach and screenshots of every required output. **[5 points]**.