CSE 264 – Web Systems Programming, Spring 2015
Class Project – Step 2 (Final)

Due ……

- **Objective**
  The project will bring together a number of different topics we have covered in class and provide additional implementation experience.

- **Description**
  This step adds additional functionality to your Set game.

- **Instructions**

1. Open your project from Step 1 in NetBeans.
2. Right now your display page should be displaying a fixed set of 12 cards. Also, if the user clicks on a card a border is added to the image to indicate it was selected. Now you need to add the rest of the functionality.
3. If you haven't already done this, implement the following: If the user tries to click on an unselected card with 3 cards already selected, display an alert box with an error message and do not select the 4th card.
4. Create an empty function in your script: `function nullFcn(result) { }`
5. Add the following javascript function to your script:

```
function doAjaxCall(method, cmd, params, fcn) {
 $.ajax(
      "http://localhost:3000/setgameserver/" + cmd,
         {
           type: method,
           processData: true,
           data: params,
           dataType: "jsonp",
           success: function (result) {
                        fcn(result)
                    },
           error: function (jqXHR, textStatus, errorThrown) {
                      alert("Error: " + jqXHR.responseText);
                      alert("Error: " + textStatus);
                      alert("Error: " + errorThrown);
                  }
         }
 );
}
```

Where **method** is the http method to be used in the call,
cmd is the command/service we are calling on the node server,
params is the *JavaScript object* holding the parameters for the Ajax call,
fcn is the one parameter function to be called on successful completion.

This function will be called by the event handler for each of the buttons below. For example, the call for the Add Row button would look like this: doAjaxCall("GET", "addrow", { id: myid }, nullFcn);

6. In Step 1 you placed the four command buttons and a login on the page; now you'll need to add the following functionality to these buttons using the doAjaxCall function:

(a) Login Button:

   i. Make an Ajax call to the server to the **login** service and pass the entered login name as the **loginName** parameter. The server will login the user and return an integer id. Save this is in a variable. It should then call another function to do an Ajax call to retrieve the user name from the server and display it on the screen (Method = GET, Command = loginname, Parameters =id, Function = /* Save returned login name on screen */).

(b) Set Button:

i. If the 3 selected cards do not constitute a set, then display a message skip the rest of the steps.

ii. Method = GET, Command = submitset, Parameters =id and cardlist, Function = nullFcn

(c) Add Row:

i. Method = GET, Command = addrow, Parameters = id, Function = nullFcn

(d) Shuffle:

i. Method = GET, Command = shuffle, Parameters = id, Function = nullFcn

(e) End Game:

i. Method = GET, Command = endgame, Parameters = id, Function = nullFcn

7. Add the following <script> definition to your html page to load the socket.io client:

```
<script src="https://cdn.socket.io/socket.io-1.3.4.js"></script>
```

8. In your document ready function, call functions to:

(a) pre-load the images

(b) attach all the event handlers to the buttons

(c) execute the following code:

```
var socket = io.connect('localhost:3000');

socket.on('hand', function (cards) {
   loadGrid(cards);
});

socket.on('players', function (players) {
   loadStatus(players);
});
```

where loadGrid is the function you've already written to load the grid of cards and loadStatus takes an array of players (as defined in the server code) and displays their info in the player table on the right side of the page. Put a * in the row/col if the value is true (for row, shuffle, end) and a blank otherwise. This will set up the socket.io event handlers for your page to receive and display the card and player lists from the server.

9. In order to test your client you need to install nodejs on your computer (nodejs.org). Once this is done, go to the node installation directory and run: “npm install socket.io” from your shell. A later version of the server will do this automagically.

10. To make your life easier you should also install the nodejs plugin in NetBeans (http://plugins.netbeans.org/plugin/36653/nodejs).

11. Fire up the server, run your client and start debugging...