EECS 1012: LAB 09 Code Breaker (Nov 18-22, 2019)

Note 1. This lab can count as a bonus point if you have already received 7 perfect grades in previous labs.
Note 2. This lab requires more reading/preparation compared to previous labs.
Note 3. While we encourage you to go to labs this week, get help from TAs and your peers towards learning the details, and submit your work by end of your lab time, we will also give you the opportunity to submit it until Dec 3.

A. REMINDERS

1) You **must** attend your **assigned** lab session.
2) You must arrive on time: anyone later than 15 minutes may not be admitted to the lab.
3) You must complete the pre-lab mini quiz posted on Moodle in the first 15 minutes of your lab time.
4) If you have already achieved 7 perfect labs and do lab09 properly, you will gain up to 1% bonus.
5) TAs are in the lab to help you. They will also verify and mark your work at the end. Signal a TA for help if you stuck on any of the steps below. Yet, note that TAs would need to help other students too.

B. IMPORTANT PRE-LAB WORKS YOU NEED TO DO BEFORE GOING TO THE LAB

1) Download this lab files and read them carefully to the end.
2) If you are not familiar with the Code Breaker board game, visit https://en.wikipedia.org/wiki/Mastermind_(board_game). Note that the one we make in this lab has 5 code pegs (not 4).
3) Revisit Slides 8 and 9 of the lecture notes on jQuery. We highly encourage you—prior to go to labs this week—to do two sets of exercises on jQuery selectors and events available in w3schools, starting here: https://www.w3schools.com/jquery/exercise_jq.asp?filename=exercise_jq_selectors1
4) In Slides 7, 12, and 14 of the lecture notes, we introduce the `stringify` (and `parse`) methods for converting a JavaScript *object* to a *string* (and vice versa). You may want to see further examples here:
   - https://www.w3schools.com/js/js_json_stringify.asp
   - https://www.w3schools.com/js/js_json_parse.asp
5) Read the wiki page of Node.js: https://en.wikipedia.org/wiki/Node.js

C. GOALS/OUTCOMES FOR LAB

- To be exposed to server-side code within which you acquaint yourself with jQuery, JSON, node.js, and express, all of which are JavaScript based libraries and notations

D. TASKS

1) TASK 1: client side of Code Breaker
2) TASK 2: server-side of the Code Breaker

E. SUBMISSIONS

1) Manual verification by a TA

   You will need to have one of the TAs of your lab to verify your work before submission. The TA will look at your various files in their progression. The TA may ask you to make minor modifications to the lab to demonstrate your knowledge of the materials. The TA will mark your name off a list; You are required to sign the list to show your attendance and that you have been verified.

2) Moodle submission

   You will see an assignment submission link on moodle. Create a **folder** named "**Lab09**" and copy all your lab materials inside. This folder should be compressed (or tar.gz) and the compressed file submitted. You are encouraged to submit it before the end time of your lab. This way you can get help from your Tas and peers in the lab. However, we will also give you another opportunity to submit it by December 3, 2019.

F. STARTING POINT

Task 1: Use html, CSS, and JS to design the client side of the code-breaker. We have provided you with a starter code, but you can design with any style you wish. If you like to use our starter code, you should:

1) Open `code_breakerV0.html` and save it as `code_breaker.html`. You should read the comments in your `code_breaker.html`, and make changes such that your html file becomes similar to `code_breakerV1.html`, eventually. Note that your html file instead of having 150 lines is going to have 28.
2) Also, open `code_breaker_clientV0.js` and save it as `code_breaker_client.js`. In the `createGameBoard` function of `code_breaker.js`, you should read the comments and write the code for all 15 lines specified by "`//...`". These lines dynamically build the html tags that you just deleted from your html file: you are creating those tags via your js file at run time (dynamically).

If you make the above changes properly, your code_breaker should work fine, that means you can start playing the game by opening your `code_breaker.html` in Firefox. We make sure that our `code_breaker_server` is up on `indigo.eecs.yorku.ca` until December 3, 2019. In next part, you can complete your own `server_side` code such that you be able to run it on your computer any time.

Show your code-breaker code to your TA.

Task 2: In this task, you complete the **server-side**. What do you need to do in this task? The following 4 steps:

i) Revisit details of the requests that are sent from the client-side: one is in `initGameBoard()` and the other is in `processAttempt()` function.
ii) Also, revisit Slides 7 and 12. Note that when the request is responded by the server, a callback function is called automatically. We anmed the callback function, *`response`*. The details of the response function were described in Slides 14-16 and can be found in the starter code we provide you with.
iii) Copy `code_breaker_serverV0.js` to a new file named `code_breaker_server.js` and follow the comments to complete the code. In particular, there are ten /*T0DO…*/ comments that you should replace with your code. More detailed instructions can be found in the file.
iv) Finally, follow the instructions in `readme.txt` to run the server locally on your machine and play the game on your device any time.

G. AFTER-LAB WORKS (THIS PART WILL NOT BE GRADED)

In order to review what you have learned in this lab as well as expanding your skills further, we recommend the following questions and extra practices:

1) This project is a great source of learning. Read all comments carefully and make sure you have a clear understanding of the code.
2) Add a button "Hint", such that every time the user clicks on it, the server reveals one of the five pegs, in any order you wish. The server should not reveal more than 3 pegs. In order to implement this component, you need to add a new action to your requests. So far, you had two actions: "generateCode" and "evaluate". Let's call the third one, "hintMe".

Please feel free to discuss any of these questions in the course forum or see the TAs and/or Instructors for help.