PROG 2700 **ASSIGNMENT 4 (PARTS A & B) – HTML DOM MANIPULATION**
Client-Side Programming

Part A: Enhancing the DOM Tutorial

https://www.linkedin.com/learning/javascript-enhancing-the-dom

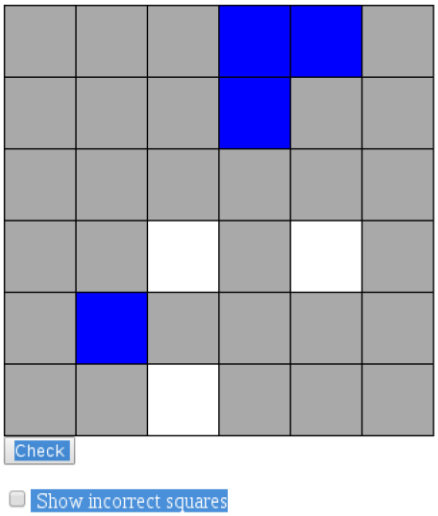
Complete the 2-hour JavaScript: Enhancing the DOM Tutorial by Ray Villalobos on 'LinkedIn Learning'.

Part B: Map, Filter, and Reduce (3-in-a-Row Game)

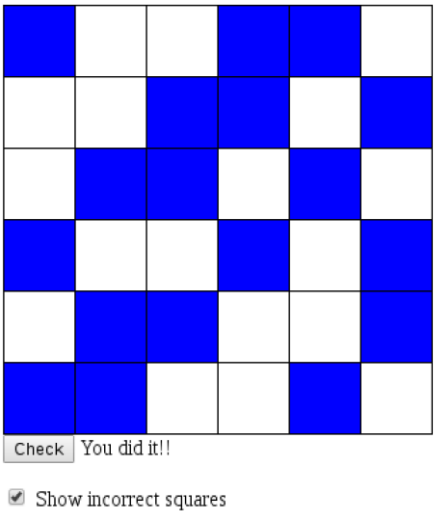
Summary

1. Examine the application at https://www.brainbashers.com/show3inarow.asp and play it a few times to understand how it works. Your job will be to recreate a portion of this application with your own implementation of *pure JavaScript* (no frameworks or libraries) and working with the Document Object Model (DOM). Starting puzzle data will be retrieved remotely via an available API.

Example Puzzle at beginning



Completed Example Puzzle



2. You will simulate the 3-in-a-row puzzle by writing it in pure JavaScript.
    a. The data that you will work with is pre-defined JSON data from a remote location. https://threeinarowpuzzle.herokuapp.com/sample . This JSON data will serve as the underlying data structure which will represent your puzzle.
    b. The display grid should be an HTML table. **However, the table must be generated only using JavaScript and without using the document.write() function to output the table tags.** (ie. You'll need to create elements/nodes and attach them to the DOM).

c. You will add unobtrusive JavaScript events to certain squares in the puzzle so that repeatedly left-clicking on the square will cycle through and change its state to one of three distinct possibilities:
    i. Empty (State 0)
    ii. State 1
    iii. State 2
d. Squares that are set to a color (or image if you wish) at the beginning of the puzzle should not be changeable as stated in the JSON data.
e. At any time during the playing of the puzzle the end user should be able to click a "Check Puzzle" button that displays one of the following status outputs
    i. "So far so good" (all colored squares are correct but the puzzle is incomplete)
    ii. "Something is wrong" (one or more of the colored squares is incorrectly assigned)
    iii. "You did it!!" (all squares are correct and the puzzle has been completely filled in)
f. A checkbox can be checked at any time which will cause the puzzle to display any incorrect squares. Unchecking the box will remove the indication of any incorrect squares.
g. All JavaScript code will be unobtrusive.
h. You will also **add one Innovative Feature** to your version of the puzzle which will add meaningful value to the playing of the game. When you have decided your feature, or if you are struggling deciding what to add, discuss with your instructor for approval.

Notes

- If solving this puzzle is not your thing, you can refer to the solution.png for the solution.
- The remote API data is test data for a 6x6 puzzle. You will however, be given a different API URL which will randomly send you a puzzle of varying sizes. **Your solution will need to accommodate the different puzzle sizes**. (ie, don't write your solution to simply handle a 6x6 puzzle size.)

General Requirements (37 points)

REQ-001 RETRIEVAL OF THE JSON STARTING DATA FOR THE PUZZLE (6 PTS.)

Your starting JSON data for the puzzle will be retrieved from the Url https://threeinarowpuzzle.herokuapp.com/sample . This data will be used to inform the puzzle of its starting state. It also will serve as the data for the puzzle as it is played. (Ie, your JSON object should be updated as events on the puzzle are performed.

REQ-002 DRAWING AND DISPLAYING OF 3-IN-A-ROW TABLE WITH JAVASCRIPT ONLY (6 PTS.)

When the page loads, a grid based on the data structure defined in REQ-001 will have a similar display to the example images shown above. Feel free to use alternative colors if you wish. Styling should be implemented with basic CSS.

You must use the DOM and JavaScript with a mixture of CSS to build the above table. You are not permitted to use HTML tags to create the table and you are not permitted to use *document.write* to output the table tags. Research online how you might do that and apply what you discover to this requirement.

REQ-003 CHANGING OF SQUARE COLORS WITH MOUSE CLICKS (6 PTS.)

When the user left-clicks on a square in the grid, the color (or image) of the square should change to the next available color (in this case blue). If the square is left-clicked again, the color will change to the next option (in this case white). Another click will remove the square back to a neutral color (in this case grey).
Squares that are assigned a color when the puzzle loads should not be changeable with mouse clicks.

REQ-004 3-IN-A-ROW PUZZLE STATUS CHECKING (6 PTS.)

A button is provided which when clicked checks the current status of the puzzle as described above in the bulleted list. When the user clicks the button, there should be an appropriate message displayed to them which accurately describes the current status of the puzzle. If the puzzle is complete and the squares are correct, clicking the button will inform them that the puzzle is correct and complete.

REQ-005 ERROR DISPLAY CHECKBOX (6 PTS.)

A checkbox will be provided which when checked will mark any squares that have been incorrectly assigned. Unchecking the box will hide any error display.

REQ-006 ADDING AN INNOVATIVE FEATURE (7 PTS.)

A **significant and unique** feature will be added to the web site to give additional value to its role as a 3-in-a-row puzzle. This feature must provide **useful, value-adding** functionality in addition to the previous set of requirements. You might want to consider adding one of the additional bits of functionality that is available on the BrainBashers.com version of the puzzle. Check with the instructor to be sure that your choice is adequate.

Architecture Requirements (3 points)

REQ-007 UNOBTRUSIVE JAVASCRIPT. (3 PTS.)

The JavaScript will all be unobtrusive.

Instructions

1. **Don’t forget that a Code Review demonstration of your code is a necessary part of this assignment. You MUST complete the code explanation/code review part of the video submission checklist to get credit for the assignment. Part of the assessment will include your ability to speak about the code you wrote, even if it doesn’t completely work or do what you expect. You do not need audio or to speak during the rest of the video, but it is required for the code review section as indicated in the checklist.**

2. **Late submissions will be subject to the late penalties laid out in the course outline.**

Academic Integrity and Plagiarism

Code sharing by any means is considered plagiarism and is strictly forbidden under the NSCC Academic Integrity policy.

NSCC ACADEMIC INTEGRITY GUIDELINES
NSCC ACADEMIC INTEGRITY REPORTING POLICY

PROG2700: Assignment Four - Video Submission Checklist

Part A: LinkedIn Learning	Show the following, <u>if not already given credit for Part A</u>: ☐ in the Browser log into LinkedIn Learning and show completion of tasks
Part B: 3-in-a-Row Game	Show the following: ☐ the program running with the original 6 x 6 sample grid ☐ clicking on an “original” cell in the table does nothing ☐ clicking on an empty cell in the table cycles between 3 states ☐ activating the “Check Puzzle” button with an error ☐ showing the error location with the checkbox checked ☐ activating the “Check Puzzle” button with no errors ☐ activating the “Check Puzzle” button with the grid complete (refer to the solution.png if you need help solving the grid) ☐ indicating and demonstrating the usage of your innovative feature ☐ the program running with a random sized grid (refresh the page until you get something other than 6 x 6) ☐ basic usage of the game with a random sized grid (you do not need to completely solve it, though) ☐ in code, show the HTML has not been modified ☐ show that the JS code is not in global scope (e.g. IIFE usage) ☐ citations for any code samples used
Code Review: **<u>Mandatory</u>**	Show the following: ☐**explain the code in detail to <u>generate the original HTML table (i.e. Game Grid) from the API data in Assign4B</u> (i.e. querying the API, working with the returned data, & generating the HTML table via DOM) (needs audio)**