Week #3 :: { MIDTERM EXAM } - Part #1 (2.5h)

Assignment in ** WEEK THREE (3) LECTURE, LAB, AND HOMEWORK **

23
JAN

STATUS

MIDTERM EXAM - Part #1

STUDENT OBJECTS - ADDIO Challenge

PART #1: STUDENT OBJECTS - (ADDIO Challenge)

For this part of the Midterm Exam, you will use ADDIO to analyze the requirements for setting up a Student Application that requires an array of student objects. You will demonstrate your ability to use ADDIO so that the application adds a new student object to the array of students, then uses the DOM to output the student information using innerHTML.

THIS MIDTERM ONLY REQUIRES YOU TO COMPLETE THE ANALYSIS & DESIGN PHASE of ADDIO.

During this assignment, you will use ADDIO to demonstrate your understanding of this assignment. **THERE IS NO CODE PROVIDED OR AVAILABLE FOR THIS ASSIGNMENT (---only a file structure). Using the rubric & other instructions below**, you will analyze the requirements for this assignment, design a flowchart that demonstrates how this program would interact (with Student Objects), and write OUR OWN PSEUDO-CODE FROM SCRATCH for the basic functionality requirements.

Again, In this assignment you will ONLY analyze & design a solution for the STUDENT WEB APPLICATION assignment. Please turn in the following items for Week #3:

1. Create a Flow Chart Diagram that graphically demonstrates how this program functions (---based on requirements below)
2. Based on the rubrics & instructions below, write basic Pseud-ocode Logic (---NOT JavaScript Code) for the main functionality
3. HINT: Not sure how to get started? Watch the video below and use the criteria below to think through your Client requirements.
4. Submit a Zip File containing ALL of the above items
5. **NOTE: You are only required to complete the first two (2) phases of ADDIO *(Analyze & Design)* as listed above**

OBJECTIVES & OUTCOMES

Successful completion of this activity will show that you can:

- Use ADDIO to analyze & design the requirements, code, and draw a flowchart.
- Assess your skills and comprehension of the materials taught up to this point. (Objects, Arrays, Functions, Conditionals, Methods, Scoping, Debugging, DOM).
- Assess your ability to use all your new programming skills and apply it to a very practical assignment.
- Assess your ability to logically separate code into functions.
- Assess your ability to utilize JavaScript DOM manipulation functions effectively.
- Employ solid craftsmanship.

LEVEL OF EFFORT

This activity should take approximately 240m to complete. It will require:

- 0m Research
- 15m Prep & Delivery
- 225m Work

If you find that this activity takes you significantly less or more time than this estimate, please contact me for guidance.

READING & RESOURCES

REQUIRED: Use ADDIO Model to analyze code/requirements for this application. Pseudo-code & Flowchart required!

Mid-Term - Rubric (https://docs.google.com/spreadsheet/ccc?key=0ArlgIg6OoecWdDBUVzVWcTEzY3g2ckp0VnV6MlVtOEE&usp=sharing) (necessary)

This rubric outlines the points for the assignment. Make sure you check off each one as done before submitting your assignment.

IMPORTANT FILES: Assignment Files for Midterm (https://assethub.fso.fullsail.edu/assethub/goal6_assign_midterm_a64e99b4-1aec-46b8-903e-c4d0da5e1710_0d547381-be8d-4581-95af-055e3b97631e.zip) (necessary)

This link has all the files you need to get started with this programming assignment.

goal6 Mid Term Practical Demo (https://www.youtube.com/watch?v=f5B2Te0pEUs&index=2&list=PLhRlE5h84yWZKShdti88tQhC8eFTCFsu9) (necessary) Approx. Duration: 11:54m

This screencast reviews the requirements for the Mid Term Practical.

Getting Started:

- ONLY SUBMIT YOUR MIDTERM TO FSO using a zip file. Please "do not" submit your Midterm on GitHub.com. Your zip file on FSO should be named as follows: lastName_firstName_midterm.zip
- This midterm is OPEN… to a point. There will only be 2 restrictions.

    1. You will work on your own to accomplish the programming tasks in this mid term.
    2. You can use the web as a reference only. You cannot use any technology to ask for help. (i.e. NO iChat, screen shares, texting, document sharing, etc…)

- **REQUIRED**: Use ADDIO Model to create a flowchart and analyze code/requirements by adding comments to your code.
- You may ask for help, guidance and direction from others (other students, Lab Instructor(s), or Course Director), but they may not help with coding, nor share code.
- You will need to create a script tag for js/main.js in your .html file.
- Use comments to add your name, etc.. to the top of ALL JavaScript files.
- You will need to wrap your code (in the js/main.js) in a self-executing function.
- Make sure you adhere to proper folder constructs, if applicable (css, images, js, etc).

Criteria:

To obtain full credit on the assignment your submission should match the functionality of the demonstration. The following criteria must be adhered to and you must satisfy all items on this assignment's rubric.

START WITH ANALYZING & DESIGN:

- Read through ALL Requirements (----see below)
- Use ADDIO Model to analyze requirements and write out pseudo-code for how this program "should" work,
- Use ADDIO Model to draw a flowchart to demonstrate what this web application should do (NO Omnigraffle files!)
- **REMINDER: You are only required to complete the first two (2) phases of ADDIO *(Analyze & Design)***

REQUIREMENTS:

- Open the js/main.js file.
- There will be no more then 4 global variables. 1 for the student array of objects, 1 for the button (the click event), and 2 other global variables you can use as you wish.
- Create an **array of student objects** containing 3 main properties (keys) and 3 sub properties. The 6 total properties are:
    - name
    - address (this should be an object with 3 sub properties, see below)
        1. street
        2. city
        3. state
    - GPA (this will be an array of GPA scores (grade point averages) for the student in this object)

  Example of 1 object:
  -- {name: 'jbond', address:{address: '3300 University Blvd', city: 'Winter Park', state: 'Fl'}, gpa: [2.5, 3.5, 4.0]};
- In the **array of student objects** above you will need to populate it with at least 2 objects of information (fully populate all key values). The GPA field is an array so you will need to populate the array with a minimum of 3 GPA scores (see the example above).
- Console.log ALL the information in ALL objects on 3 lines.
    1. name
    2. address
    3. GPA:
       Example:
        1. Name: jbond
        2. Address: 3300 University Winter Park Florida
        3. GPA: [2.5, 3.5, 4.0]
- Add a new object to the array of objects above:
    - Create a function call that passes arguments for a new object. The arguments being passed would be the same data items in the already existing objects (name, address, GPA).

      Example of a function call with arguments we are passing to the function:
        - addData('super man', '123 Test Dr', 'Orlando', 'Florida', [3.2, 4.0, 2.2]);
    - The addData function will add this new student information to the existing array of student objects, documented above.
    - The new object format will be similar to the already existing objects.
    - Console.log ALL the information in ALL objects on 3 lines. This should display with the new object that was just added directly above.
        1. name
        2. address
        3. GPA:
           Example:
            1. Name: jbond
            2. Address: 3300 University Winter Park Florida
            3. GPA: [2.5, 3.5, 4.0]
- Create an on-click event on the 'Next' button which, when clicked calls a function that does the following:
    - Use JavaScript's innerHTML property to display ALL object data (one student at a time) in the HTML (like the demonstration).
    - The student's information stored in the object, should display on 4 lines.
        1. name
        2. address
        3. GPA
        4. Average GPA

- The “Average GPA” directly above should be calculated in a separate function. The function should calculate the student’s Average GPA and return the results.
- When the last student object has been detected AND has already displayed, you will need to do the following:

    1. disable the click event
    2. change the “Next” button text to display “DONE!!!” in the button

- There should not be any duplicate code in the main.js file. If there is, you will need to refractor the code by creating a function for the duplicate code and then calling the function when it is needed.
- Using the **date method**, add a date to the end of each student object. When you refresh your browser , the browser should now display the date as well, for the student that is being displayed. Make sure the date displays in the console.logs data and the innerHTML.
- Make sure to put comments throughout your code (if applicable).

DELIVERABLES

You will submit ALL your ADDIO files for the Midterm in **one (1) zip** file as follows:

- Psuedocode (in JavaScript or Text/Notepad File)
- Flowchart
- Submit to FSO using the file format name lastName_firstName_midterm.zip.

Comments