Javascript Project 1

Summary

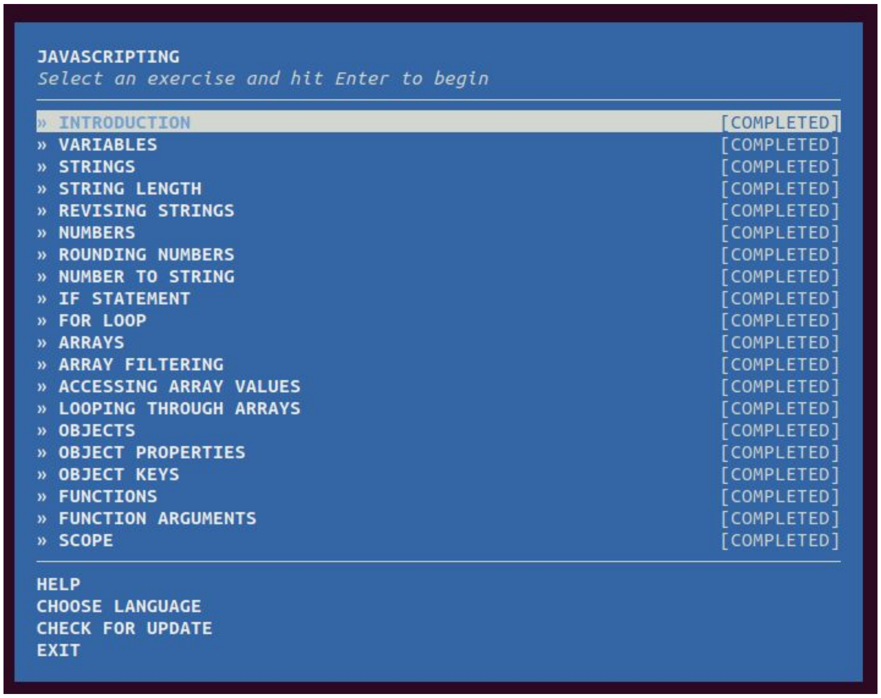
This is your first project in Javascript.
Make sure you submit each milestone to your weekly mentor.
Milestone 1 should be submitted separately from the other milestones (see instructions)
Milestone 2 onwards needs to be uploaded to a personal git repo through github classrooms (send the repo link to your mentor, and let them know when you finished a milestone, so they could check your project code)

Teachers note: when using bootstrap, or any other library, it is highly recommended to read their documentation before trying to implement something alone.

Milestone 1 - Gain Extra Skills

Features

- Complete this: [workshopper/javascripting: Learn JavaScript by adventuring around in the terminal.]
- Need to install before Node.js (LTS version) [Node.js]
- Show to your mentor the completed exercises list after finishing



```
JAVASCRIPTING
Select an exercise and hit Enter to begin

» INTRODUCTION                                              [COMPLETED]
» VARIABLES                                                 [COMPLETED]
» STRINGS                                                   [COMPLETED]
» STRING LENGTH                                             [COMPLETED]
» REVISING STRINGS                                          [COMPLETED]
» NUMBERS                                                   [COMPLETED]
» ROUNDING NUMBERS                                          [COMPLETED]
» NUMBER TO STRING                                          [COMPLETED]
» IF STATEMENT                                              [COMPLETED]
» FOR LOOP                                                  [COMPLETED]
» ARRAYS                                                    [COMPLETED]
» ARRAY FILTERING                                           [COMPLETED]
» ACCESSING ARRAY VALUES                                    [COMPLETED]
» LOOPING THROUGH ARRAYS                                    [COMPLETED]
» OBJECTS                                                   [COMPLETED]
» OBJECT PROPERTIES                                         [COMPLETED]
» OBJECT KEYS                                               [COMPLETED]
» FUNCTIONS                                                 [COMPLETED]
» FUNCTION ARGUMENTS                                        [COMPLETED]
» SCOPE                                                     [COMPLETED]

HELP
CHOOSE LANGUAGE
CHECK FOR UPDATE
EXIT
```

Milestone 2 - Fibonacci

Features

- Create a simple website that present the following text: “The Fibonacci of X is Y”, where X & Y are numbers declared in your JS code (should present numbers instead)
- X and Y should be declared as JS variables. Both X and Y should be added to the HTML with JavaScript (meaning, do not write the value of X and Y directly in your HTML code, use Javascript to do that)
- What is a Fibonacci number

Milestone 3

Features

- Instead of hardcoding Y (the result of the Fibonacci of X), calculate it with a for loop
- The calculation should be wrapped in a function, that gets X as an argument, and returns Y
- After the function, you should call the function, and assign the returned value in you HTML to present to the user

Milestone 4

Features

- Create an input (with number type) element and a button next to that calculate fibonacci.
- Follow this figma design (it is based on bootstrap, so use bootstrap): ITC Fibonacci Project Design (For this milestone, only the first row of screens is relevant)
- Add a click event listener to the button, that executes a function that takes the number value in the created input, calculates it’s Fibonacci value and presents it to the user
- Recommended googling: ‘HTML Input element’ and ‘get value in javascript of an input element’

Milestone 4.1 - Geekout

Features

- Implement the Fibonacci function with recursion

- We know you can google it and copy the code, try doing it by yourself (go over the lectures/youtube if needed)

Milestone 5

Features

- Run the following local server: ITC-fibonacci-server (read the readme!)
- Create a function that calls this server on this address: http://localhost:5050/fibonacci/:number, where number is a parameter passed to the server to be calculated
- The response is the calculated fibonacci, present it to the user.
- Calling the server should replace your implementation of calculating fibonacci

Milestone 6

Features

- Present a loader to the user when a call is made to the server (indicating the server is calculating)
- Present an error to the user if the input number is more than 50, and do not send a server request
- Try passing the number 42 to the server. The server will send back an error, present this error to the user. (read fetch() docs to see how to identify if the server sent an error)
- Follow the second row of screens in the figma design

Milestone 7

Features

- Create a function that calls the server with this url: http://localhost:5050/getFibonacciResults
- Call this function when the screen loads. You will get a list of fibonacci calculations that you previously submitted to the server
- Present the list to the user under the calculator
- The list should be updated every time the user makes a new calculation (suggestion: create a function the takes the data from the server response, and creates the html list to present to the user, and call this function after the user makes a new calculation)
- Follow the third row of screens in the figma design

Milestone 7.1 - Geekout

Features

- Transform all you functions with promises in them to async/await

Milestone 8

Features

- Add a checkbox, under the calculator with “Save Calculation” text
- If it is checked, calculate the fibonacci through the server (so it will save it to be presented in the list)
- If it is not checked, calculate the fibonacci locally in your function (won’t sent a request to the server)
- Follow the fourth row of screens in the figma design

Milestone 8.1 - Geekout

Features

- Add a select box with sort by date asc or desc / number asc or desc
- After the user is checking one of the items in the list, rearrange the list to match the sorting preferences
- Follow the fifth row of screens in the figma design