Assignment 2: Vanilla JavaScript Customizable Quiz

Course: Web-Based Systems
Number: SENG 513
Semester: Fall 2023
Due Date: Week of Feb 12th
Instructor: Steve Sutcliffe <steve dot sutcliffe at ucalgary dot ca>
Version: 1.0.0
Weight: 5%

Objective:

Develop a web application using vanilla JavaScript (ES6+), demonstrating a thorough understanding of JavaScript's core concepts, including OOP, generators, asynchronous programming, and the keyword `this`.

Group Size:

You can do this assignment individually or in groups of a maximum of 4. Your group members must be in the same tutorial as you to give the demonstration.

Part 1: Quiz Engine in OOP

1. Implement quiz logic using classes/constructors in order to add questions to a quiz, check the user's answers, update the score, and move to the next question after the user responds. You should consider using the following classes:
    1.1. Quiz: Manages the overall quiz operations like starting the quiz, keeping score, and determining the next question.
    1.2. Question: Represents a single quiz question, including the text, choices, and the correct answer.
    1.3. User: Represents a user taking the quiz and could include properties like username and score history.

Part 2: Dynamic Question Generator using Generators

1. Create a generator function to manage the flow of quiz questions (e.g., use `yield` to control presentation and user input to proceed to the next question).
2. Include the logic to change the sequence based on the previous answers (e.g., difficulty adjustment).

Part 3: Asynchronous Data Handling

1. Fetch a set of questions from Open Trivia DB (https://opentdb.com/)
2. Process and display questions one at a time in the UI

Part 4: Interactive UI and `this`

1. Create a UI with buttons for submitting answers and moving to the next question.
2. Demonstrate the use of `bind`, `call`, or `apply` to manipulate the context of `this` in callbacks.

Part 5: Demonstration

In your tutorial, each group (or individual if the assignment was done individually) will demonstrate the working application and explain their decisions and implementations.

1. Demonstration:
   1.1. Launch your application and show the functionality of answering trivia questions.
   1.2. Show the app fetching new questions and keeping score.

2. Explain:
   2.1. Identify and explain the use of OOP.
   2.2. Identify and explain the use of generators.
   2.3. Identify and explain the use of asynchronous data handling.
   2.4. Identify and explain the use of the keyword `bind`, `call`, or `apply` (whichever you used).

D2L/GitLab Submission

Please follow these guidelines for submitting your application to D2L:

1. Create and use the University of Calgary's Gitlab repository online; name the repository seng513-202401-assignment2.
2. Commit your work frequently. Only commits made before the deadline will be considered for grading.
3. Grant access to both TAs and your Instructor by adding them as members on your repository.
4. Once done, submit a single *.txt file to D2L containing a link to your repository with a list of your group members (each person will need to make a submission on D2L). The file should be called seng513-assignment2.txt.

Grading [30 Points]

Functionality of the Web App [7 Points]

[1 pt] Displays questions one at a time in the UI.
[1 pt] Receives and processes user input for each question.
[1 pt] Determines the correctness of answers.
[1 pt] Calculates and displays the total score after each question.
[1 pt] Maintains and displays a score history or progress.
[1 pt] Proceeds to the next question.
[1 pt] Clearly indicates the correct answer.

Asynchronous Question Loading [6 Points]

[3 pts] Correct implementation of asynchronous calls using `fetch` or XMLHttpRequest.
[3 pts] Application correctly processes and displays data fetched from API (i.e., Open Trivia DB).

Explanation and Understanding [12 Points]

[3 pts] Quality of explanation of `bind`, `call`, or `apply`.
[3 pts] Quality of explanation about asynchronous data handling.
[3 pts] Quality of explanation about generators.
[3 pts] Quality of explanation about OOP.

Achievement Points [Up to 5 Points]

Achievement points play a crucial role in securing full marks for this assignment. These points are designed to challenge your understanding and application of the course material, pushing you beyond the foundational requirements. To successfully earn these points, you will need to engage in independent research and demonstrate a higher level of mastery.

These tasks are deliberately more complex and may not have straightforward solutions. They are intended to stimulate critical thinking, creativity, and a deeper exploration of the subject matter. The answers we are looking for here go beyond the basics; we aim to see how you can apply your knowledge innovatively and effectively in less conventional scenarios.

Remember, the journey to these points is as valuable as the points themselves. This is an opportunity for you to excel and showcase the breadth and depth of your understanding.

Responsive and aesthetic UI.
Proper error handling.
Dynamic Question Difficulty Adjustment.

Bonus Points [Up to 3 points]

Uses all `bind`, `call`, and `apply` functions thoughtfully and demonstrates a high-level understanding of each.

Penalties

As part of our commitment to academic integrity and adherence to intellectual property laws, it is crucial that all submitted work strictly respects copyright laws and guidelines. This includes, but is not limited to, the use of copyrighted materials, code, images, or any other content without proper authorization or attribution.

Please be aware that any instance of copyright infringement will be taken very seriously. The following penalties apply:

Immediate Deduction: If any part of your submitted work is found to violate copyright rules, a significant deduction will be applied to your assignment grade. This deduction can range from a partial to a full loss of marks, depending on the severity of the infringement.

Retroactive Application: Importantly, this penalty can be applied at any point during the course, including during the final grade calculation. If it is discovered post-submission – even after initial grading – that your assignment contains copyrighted material used inappropriately, a retroactive deduction will be applied to your grade.

Severe Cases: In severe cases or in instances of repeated infringement, further disciplinary action may be taken in accordance with university policies, which could include failure in the course or referral to a disciplinary committee.

Responsibility and Vigilance: You must ensure that all materials used in your assignments are properly licensed or fall under fair use provisions and are correctly attributed. Please seek guidance before submission if you are unsure about using any material.

Honesty and Integrity: We encourage honesty and integrity in all academic endeavours. This policy is in place to uphold these values and to ensure a fair and legitimate educational environment for all students.

Additional Penalty: Use of !important [-5]

The use of `!important` is generally discouraged in the industry and demonstrates a lack of knowledge in structuring quality CSS. Using `!important` in this assignment will result in the deduction of marks.

Late Assignments

Assignments must be demonstrated to the TAs during your scheduled tutorial sessions. If you need to submit an assignment late, you must arrange this directly with your TA, subject to their availability. Late assignments will be graded on a simplified scale: they will receive either a passing grade of D+ (50%) or a failing grade of F (0).

In cases of special circumstances, such as illness or personal emergencies, accommodations may be made regarding submission deadlines. These will be considered on a case-by-case basis. Students facing such circumstances are encouraged to contact me as soon as possible to discuss potential adjustments.