**JavaScript "Rock, Paper, Scissors" Game**

Description:

In this assignment, you will be building a "Rock, Paper, Scissors" game using JavaScript. The HTML and CSS have already been provided. Your task is to create the **script.js** file to add functionality to the game. The game will consist of three buttons, each corresponding to the "rock", "paper", or "scissors" choice. When the player clicks on a button, the game will display the player's choice, the computer's random choice, and the result of the game (win, loss, or tie).

**Expected Result:** [**https://6476b6b97536584961733fae--fancy-choux-cafdc2.netlify.app/**](https://6476b6b97536584961733fae--fancy-choux-cafdc2.netlify.app/)

1. **Adding Buttons in HTML**: Open the HTML file and find the **<div class="buttons">** element. Inside this **div**, add three **button** elements. Each of these buttons will represent a choice in the game: "Rock", "Paper", or "Scissors".
2. **Adding onclick Events**: For each button, assign an **onclick** attribute. This attribute is used to call a JavaScript function when the button is clicked. For the "Rock" button, the **onclick** attribute will call the **playRock()** function. Similarly, the "Paper" and "Scissors" buttons will call the **playPaper()** and **playScissors()** functions, respectively. These functions will be defined in the JavaScript file.
3. **Creating JavaScript Variables**: Now, navigate to your **script.js** file. Declare an array named **choices** that contains the strings "rock", "paper", "scissors". This array will be used to generate the computer's choice in the game.
4. **Creating Game Statistics Object**: Create an object named **gameStats** with keys "wins", "losses", and "ties". Each of these keys should have a value initially set to 0. For example: **let gameStats = { wins: 0, losses: 0, ties: 0 };**.
5. **Creating Player Object**: Create a **player** object with a key called **choice**. This key will hold the player's current selection in the game. Set the initial value of **choice** to **null**. For example: **let player = { choice: null };**.
6. **Creating Game Control Functions**: Write three functions: **playRock()**, **playPaper()**, and **playScissors()**. These functions will be linked to the corresponding buttons in your HTML file. Inside each function, assign the appropriate string ("rock", "paper", or "scissors") to **player.choice** and call the **playGame()** function.
7. **Writing Main Game Function**: The **playGame()** function is the heart of your game. Here's what it needs to do:
   * Declare a boolean variable named **isAlive** and set it to **true**.
   * If **isAlive** is **true**, call a function **getComputerChoice()** that randomly selects a choice for the computer from your **choices** array.
   * Update the player's and the computer's choices on the webpage by grabbing the appropriate elements using their ids ("player-choice" and "computer-choice") and changing their **innerText**.
   * Compare the player's choice with the computer's choice to determine the game result. You'll do this with a separate function called **compareChoices()**.
8. **Writing Helper Functions**: Write the **getComputerChoice()** function to return a random choice from the **choices** array for the computer. In the **compareChoices()** function, use conditional statements (if-else) to determine the game result based on the player's and computer's choices.  
   **Hint: choices[Math.floor(Math.random() \* choices.length)]**
9. **Displaying Game Result**: Write a function **displayMessage()** to display a message to the player indicating the result of the game. This function should find the HTML element with id "message" and update its **innerText** to reflect the result.
10. **Updating Game Statistics**: Lastly, create a function **updateGameStats()** to update the game statistics each time a game round is completed. This function should find the HTML element with id "game-stats" and update its **innerText** with the current game statistics.

Take your time with each step and remember that frequent testing can help catch and correct errors early.

**Submission**

Once you have completed all the challenges, save your work, and submit your **Javascript\_Assignment.zip** folder.

**Due Date:** This Assignment is on Sunday 4th, June 2023 @ 11:59.