Lab4 – Unit Testing in JavaScript -Initiation to Jest

The aim of this tutorial is to understand how to run unit tests on JavaScript modules using **Jest**, a JavaScript-based unit testing framework.

1. What is Jest?

Jest is a JavaScript Testing framework built by Facebook, it is primarily designed for React (which is also built by Facebook) based apps.

Jest is designed to ensure the correctness and reliability of JavaScript codebases. It provides a user-friendly and feature-rich API that enables developers to write comprehensive tests with ease. Jest's popularity stems from its simplicity, efficiency, and adaptability to diverse JavaScript environments.

2. Why is Jest important?

Testing plays a crucial role in software development, ensuring that code functions as intended and meets the desired requirements. Jest simplifies the testing process, making it an essential tool for JavaScript developers.

Here are some key reasons why Jest is important:

- **Improved code quality:** Jest helps developers write high-quality code by identifying and preventing bugs early in the development process.
- **Reduced development time**: Jest's efficient test execution provides quick feedback, enabling developers to fix issues promptly, saving time and effort.
- **Increased confidence in code:** Comprehensive testing with Jest instills confidence in the code's reliability, reducing the risk of unexpected failures in production.
- **Enhanced maintainability:** Well-tested code is easier to understand and maintain, reducing the burden on future developers.

3. Installing and using Jest

Getting started with Jest is straightforward. Follow these steps to install and use Jest:

1. **Install Jest:** Install Jest using npm :

```
npm install --save-dev jest
```

2. **Create test files**: Create test files with the `.test.js` or `.test.jsx` extension alongside the corresponding JavaScript or React component files.
3. Write test cases: Use Jest's assertion library to write test cases that verify the expected behaviour of your code.
4. Run tests: Run the tests using the Jest CLI:

```
jest
```

Jest will execute the tests and provide detailed results, indicating any failures or unexpected behaviour.

To get started, it is highly recommended to consult the Jest tutorial provided on the link: *https://www.softwaretestinghelp.com/jest-testing-tutorial/*

Exercise 1:

Based on the tutorial example, write, and test the test cases from Exercises 1 and 2 of Lab3.

Exercise 2:

Write and test the test cases for each function below:

1. The first n elements of an array.

```
function first(array, n) {
 if (array == null || n<=0)
      return [];
 if (n == null)
      return array[0];
 return array.slice(0, n);
```

2. The last n elements of an array.

```
function last(array, n) {
 if (array == null)
      return [];
 if (n == null)
      return array[array.length - 1];
 return array.slice(Math.max(array.length - n, 0));
```

3. String = the concatenation of all the elements of an array of Strings.

```
myColor = ["Red", "Green", "White", "Black"];
console.log(myColor.toString()); console.log(myColor.join());
console.log(myColor.join(''));
```

4. Divide an array into several sub-arrays of predefined size.

```
function chunk(array, size){
 var chunkedArr = [];
 var index = 0;
 while (index < array.length) {
      chunkedArr.push(array.slice(index, size + index));
      index += size;
 }
 return chunkedArr;
}
```

- **Deadline:**

At the end of Lab session (no later than Saturday, November 18 at 23:59)
To: adil.chekati@univ-constantine2.dz

- **Link to be submitted:**

GitHub repository link.

Classroom code

Google Classroom

aoa5lne

Drive QRcode

SCAN ME!