Automation Testing

Repeating Tests

A Day in the Life of an Automation Test Engineer

Marcelo completes his portion of default methods, and their different types for the project.

Now, Marcelo has to identify what Repeating Tests are, the lifecycle of the Repeating Tests, the Maven Dependency and its shape.

This lesson will help Marcelo to understand how this whole phase will help him in the project.

Learning Objectives

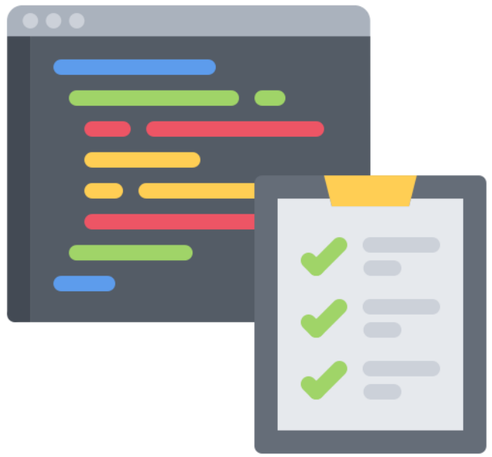
By the end of this lesson, you will be able to:

- Identify the need for Repeating Tests
- Illustrate the lifecycle of Repeating Tests
- Explain with an example how Maven Dependency works

Repeating Tests: Overview

Repeating Tests: Overview

Repeating Tests are tests that involve multiple measures of the same variable taken on the same or matched subjects either under different conditions or over two or more time periods.



Need of Repeating Tests

Need of Repeating Tests

- @RepeatingTests is used to signal that the annotated method is a test template method that should be repeated a specified number of times with a configurable display name.

- To repeat a test with different arguments, consider using @ParameterizedTests.

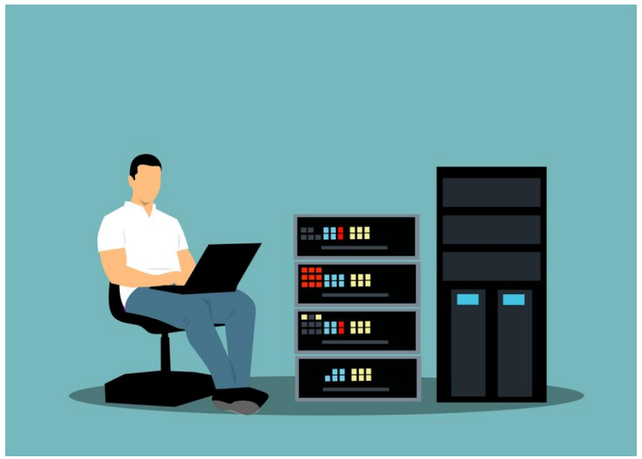
Need of Repeating Tests

Example of Repeating Tests:

- After four clicks on the payment link for a certain banking application, the environment "Exception: Page cannot be displayed" displays.
- This is a recurring and well-known problem for the customer. The user now understand that user cannot afford to have tests fail due to known environmental concerns, which the client would not accept as genuine flaws.
- Thus, in this example, the user could either construct the code logic within the test itself to handle the payment link being clicked four times, and have user test continue without failing, or the user could better take advantage of the repeated test capability enabled by JUnit 5.

Lifecycle Support for Repeating Tests

Lifecycle Support for Repeat Tests



- Each execution of the @RepeatingTests will behave like a regular @Test having full JUnit test life cycle support.
- Meaning that, during each execution, the @BeforeEach and @AfterEach methods will be called. To demonstrate this, the user should add appropriate methods in the test class.

Lifecycle Support for Repeat Tests

Example of lifecycle support for Repeating Tests:

```
@BeforeEach
void beforeEachTest()
 {
   System.out.println("Before Each Test");
 }

 @AfterEach
 void afterEachTest()
  {
    System.out.println("After Each Test");
    System.out.println("======================");
  }
```

Maven Dependency

Maven Dependency

- JUnit is the testing framework extensively used for Java projects, built in the maven project format for unit testing purposes.
- In Maven, the dependency provides the capability to manipulate artifacts. It can copy and/or unpack artifacts from local or remote repositories to a specified location.
- In Maven, a dependency is another archive JAR, ZIP, and so on that user's current project needs in order to compile, build, test, and/or run.

Example of Maven Dependency

```
<dependency>
   <groupId>org.junit.jupiter</groupId>
   <artifactId>junit-jupiter-engine</artifactId>
   <version>5.8.1</version>
   <scope>test</scope>
</dependency>
```

Key Takeaways

- Repeating Tests are used to signal that the annotated method is a test template method.
- To repeat a test with different arguments, consider using Parameterized Tests.
- Each execution of the @RepeatTests will behave like a regular @Test having full JUnit 5 test life cycle support.
- In Maven, the dependency provides the capability to manipulate artifacts.