**Assisted Practice: 3.4 Nested and Repeated Tests**

This section will guide you to:

* Create a Maven project for a standalone application
* Configure JUnit 5 to be a part of the project
* Create a JUnit class to test Nested Tests
* Create a JUnit class to test Repeated Tests
* Run the JUnit classes

**Development Environment**

* Eclipse IDE for Enterprise Java Developers v2019-03 (4.11.0)
* Apache Tomcat Server v9.0
* JRE: OpenJDK Runtime Environment 11.0.2
* All other dependencies are handled within pom.xml

This lab has seven subsections, namely:

* + 1. Creating a Maven standalone project
    2. Configuring pom.xml to add JUnit5 dependencies
    3. Create a Java class Calculator for testing purpose
    4. Creating a JUnit class NestedCases
    5. Creating a JUnit class RepeatedTests
    6. Running the JUnit classes
    7. Pushing the code to your GitHub repositories

**Step 3.4.1:** Creating a Maven standalone project

* Open Eclipse
* Go the **File** menu. Choose **New->Maven Project**
* Check **Create a Simple Project** checkbox and click on **Next**
* In **Group Id** and **Artifact id** enter **UsingJUnit** and click on **Finish**
* This will create the project files in the Project Explorer

**Step 3.4.2:** Configuring pom.xml to add JUnit5 dependencies

* Expand **UsingJUnit** in the Project Explorer
* Double click on **pom.xml** and enter the following data:

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>UsingJUnit</groupId>

<artifactId>UsingJUnit</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-engine</artifactId>

<version>5.4.2</version>

</dependency>

<dependency>

<groupId>org.junit.platform</groupId>

<artifactId>junit-platform-launcher</artifactId>

<version>1.2.0</version>

</dependency>

</dependencies>

</project>

**Step 3.4.3:** Create a Java class Calculator for testing purposes.

* In the Project Explorer, expand the project **‘UsingJUnit’**
* Expand **src->main->java**. Right click on **java** . Choose **New->Other**
* From the list of Wizards, select **Class** and click on **Next**
* Enter the **Package** as com.ecommerce.tests and **Name** as **Calculator** and click on **Finish**
* Enter the following code

**package** com.ecommerce.tests;

**public** **class** Calculator

{

**public** int add(int a, int b) {

**return** a + b;

}

}

**Step 3.4.4:** Creating a JUnit class NestedCases

* In the Project Explorer, expand the project **UsingJUnit**
* Expand **src->main->java**. Right click on **java**. Choose **New->Other**
* From the list of Wizards, select **Class** and click on **Next**
* Enter the **Package** as com.ecommerce.tests and **Name** as NestedCases and click on **Finish**
* Enter the following code:

**package** com.ecommerce.tests;

**import** org.junit.jupiter.api.**\***;

**import** org.junit.jupiter.api.AfterAll;

**import** org.junit.jupiter.api.BeforeAll;

**import** org.junit.jupiter.api.Test;

**import** org.junit.platform.runner.JUnitPlatform;

**import** org.junit.runner.RunWith;

**@DisplayName("JUnit 5 Nested Example")**

**@RunWith(JUnitPlatform.class)**

**public** **class** NestedCases {

**@BeforeAll**

**static** void beforeAll() {

**System**.out.println("Before all test methods");

}

**@BeforeEach**

void beforeEach() {

**System**.out.println("Before each test method");

}

**@AfterEach**

void afterEach() {

**System**.out.println("After each test method");

}

**@AfterAll**

**static** void afterAll() {

**System**.out.println("After all test methods");

}

**@Nested**

**@DisplayName("Tests for the method A")**

**class** A {

**@BeforeEach**

void beforeEach() {

**System**.out.println("Before each test method of the A class");

}

**@AfterEach**

void afterEach() {

**System**.out.println("After each test method of the A class");

}

**@Test**

**@DisplayName("Example test for method A")**

void sampleTestForMethodA() {

**System**.out.println("Example test for method A");

}

**@Nested**

**@DisplayName("When X is true")**

**class** WhenX {

**@BeforeEach**

void beforeEach() {

**System**.out.println("Before each test method of the WhenX class");

}

**@AfterEach**

void afterEach() {

**System**.out.println("After each test method of the WhenX class");

}

**@Test**

**@DisplayName("Example test for method A when X is true")**

void sampleTestForMethodAWhenX() {

**System**.out.println("Example test for method A when X is true");

}

}

}

}

**Step 3.4.5:** Creating a JUnit class RepeatedTests

* In the Project Explorer, expand the project **UsingJUnit**
* Expand **src->main->java**. Right click on **java**. Choose **New->Other**
* From the list of Wizards, select **Class** and click on **Next**
* Enter the **Package** as com.ecommerce.tests and **Name** as RepeatedTests and click on **Finish**
* Enter the following code:

**package** com.ecommerce.tests;

**import** org.junit.jupiter.api.**\***;

**import** org.junit.jupiter.api.AfterAll;

**import** org.junit.jupiter.api.BeforeAll;

**import** org.junit.jupiter.api.Test;

**import** org.junit.platform.runner.JUnitPlatform;

**import** org.junit.runner.RunWith;

**import** **static** org.junit.jupiter.api.Assertions.assertEquals;

**@DisplayName("JUnit 5 Repeated Tests Example")**

**@RunWith(JUnitPlatform.class)**

**public** **class** RepeatedTests {

**@BeforeAll**

**public** **static** void init(){

**System**.out.println("Before All init() method called");

}

**@BeforeEach**

**public** void initEach(){

**System**.out.println("Before Each initEach() method called");

}

**@Test**

**@DisplayName("Add operation test")**

**@RepeatedTest(5)**

void addNumber(TestInfo testInfo) {

Calculator calculator = **new** Calculator();

Assertions.assertEquals(2, calculator.add(1, 1), "1 + 1 should equal 2");

**System**.out.println("===addNumber testcase executed===");

}

**@AfterEach**

**public** void cleanUpEach(){

**System**.out.println("After Each cleanUpEach() method called");

}

**@AfterAll**

**public** **static** void cleanUp(){

**System**.out.println("After All cleanUp() method called");

}

}

**Step 3.4.6:** Running the JUnit classes

* Right click on **NestedCases.java** in the Project Explorer
* Select **Run As->JUnit Test**
* You can see the JUnit output in the Eclipse JUnit tab
* Right click on **RepeatedTests.java** in the Project Explorer
* Select **Run As->JUnit Test**
* You can see the JUnit output in the Eclipse JUnit tab

**Step 3.4.7:** Pushing the code to your GitHub repositories

* Open your command prompt and navigate to the folder where you have created your files.

**cd <folder path>**

* Initialize your repository using the following command:

**git init**

* Add all the files to your git repository using the following command:

**git add .**

* Commit the changes using the following command:

**git commit . -m “Changes have been committed.”**

* Push the files to the folder you initially created using the following command:

**git push -u origin master**