**Assisted Practice: 3.5 Dynamic 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 Dynamic Tests
* Run the JUnit class

**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 five subsections, namely:

* + 1. Creating a Maven standalone project
    2. Configuring pom.xml to add JUnit5 dependencies
    3. Creating a JUnit class DynamicTests
    4. Running the JUnit class
    5. Pushing the code to your GitHub repositories

**Step 3.5.1:** Creating a Maven standalone project

* Open Eclipse
* Go to 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 3.5.2:** 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 3.5.3:** Creating a JUnit class DynamicTests

* 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 DynamicTests and click on **Finish**
* Enter the following code:

**package** com.ecommerce.tests;

**import** java.util.Arrays;

**import** java.util.Collection;

**import** java.util.List;

**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.jupiter.api.condition.DisabledIf;

**import** org.junit.jupiter.api.condition.EnabledOnOs;

**import** org.junit.jupiter.api.condition.OS;

**import** org.junit.jupiter.api.function.Executable;

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

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

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

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

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

**@RunWith(JUnitPlatform.class)**

**public** **class** DynamicTests {

**@TestFactory**

**Collection**<DynamicTest> dynamicTests() {

**return** **Arrays**.asList(

dynamicTest("simple dynamic test", () -> assertTrue(**true**)),

dynamicTest("My Executable Class", **new** MyExecutable()),

dynamicTest("Exception Executable", () -> {**throw** **new** **Exception**("Exception Example");}),

dynamicTest("simple dynamic test-2", () -> assertTrue(**true**))

);

}

}

**class** MyExecutable **implements** Executable {

**@Override**

**public** void execute() **throws** **Throwable** {

**System**.out.println("Hello World!");

}

}

**Step 3.5.4:** Running the JUnit class

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

**Step 3.5.5:** 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**