6.2 Code Coverage with JaCoCo

This section will guide you to:

* Perform code coverage using JaCoCo in Jenkins for a Maven application.

This guide has five subsections, namely:

6.2.1 Login to Jenkins

6.2.2 Adding JaCoCo in Jenkins

6.2.3 Adding JaCoCo in Maven project

6.2.4 Creating Jenkins job for Maven

6.2.5 Push code to GitHub repositories

**Step 6.2.1:** Login to Jenkins

* Open your browser and navigate to **localhost:8081**
* Provide your username and password and click on **Login.**

**Step 6.2.2:** Add JaCoCo plugin in Jenkins

* Navigate to **Manage Jenkins -> Plugin Manager -> Available**and find**JaCoCo Plugin.**
* Restart Jenkins.

**Step 6.2.3:** Adding JaCoCo in Maven project

* Add the below code in the **pom.xml file** of your Maven project:

**<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>com.mkyong</groupId>**

**<artifactId>MathOperations</artifactId>**

**<version>0.0.1-SNAPSHOT</version>**

**<name>Mathematical Operations</name>**

**<properties>**

**<jacoco.version>0.7.5.201505241946</jacoco.version>**

**<junit.version>4.12</junit.version>**

**</properties>**

**<dependencies>**

**<dependency>**

**<groupId>junit</groupId>**

**<artifactId>junit</artifactId>**

**<version>${junit.version}</version>**

**<scope>test</scope>**

**</dependency>**

**</dependencies>**

**<build>**

**<plugins>**

**<plugin>**

**<groupId>org.apache.maven.plugins</groupId>**

**<artifactId>maven-compiler-plugin</artifactId>**

**<version>3.6.1</version>**

**<configuration>**

**<skipMain>true</skipMain>**

**<skip>true</skip>**

**<source>1.8</source>**

**<target>1.8</target>**

**</configuration>**

**</plugin>**

**<plugin>**

**<groupId>org.jacoco</groupId>**

**<artifactId>jacoco-maven-plugin</artifactId>**

**<version>${jacoco.version}</version>**

**<executions>**

**<execution>**

**<id>prepare-agent</id>**

**<goals>**

**<goal>prepare-agent</goal>**

**</goals>**

**</execution>**

**<execution>**

**<id>report</id>**

**<phase>prepare-package</phase>**

**<goals>**

**<goal>report</goal>**

**</goals>**

**</execution>**

**<execution>**

**<id>post-unit-test</id>**

**<phase>test</phase>**

**<goals>**

**<goal>report</goal>**

**</goals>**

**<configuration>**

**<!-- Sets the path to the file which contains the execution data. -->**

**<dataFile>target/jacoco.exec</dataFile>**

**<!-- Sets the output directory for the code coverage report. -->**

**<outputDirectory>target/jacoco-ut</outputDirectory>**

**</configuration>**

**</execution>**

**</executions>**

**<configuration>**

**<systemPropertyVariables>**

**<jacoco-agent.destfile>target/jacoco.exec</jacoco-agent.destfile>**

**</systemPropertyVariables>**

**</configuration>**

**</plugin>**

**</plugins>**

**</build></project>**

* Create **Operation.java** and add the code given below:

**package math.operation;**

**public class ArithmeticOperations {**

**public Integer add(Integer a, Integer b)**

**{**

**return a+b;**

**}**

**}**

* Create **Test.java** and add the code given below:

**package math.operation;**

**import org.junit.Test;**

**import static org.junit.Assert.assertEquals;**

**public class ArithmeticOperationsTest {**

**@Test**

**public void testAdd()**

**{**

**ArithmeticOperations operations = new ArithmeticOperations();**

**Integer actual = operations.add(2, 6);**

**Integer expected = 8;**

**assertEquals(expected, actual);**

**}**

**}**

**Step 6.2.4:** Creating Jenkins job for Maven

* To create a new job in Jenkins, open the Jenkins dashboard with your Jenkins URL. For example, http://localhost:8081/.
* Click on **Create New Job**. Enter the item name, select **Maven Project** and click **OK**.
* Once you click **OK,**the page will be redirected to its project form. Here, you will need to enter the project information.
* Scroll to **Maven**option under **Source Code Management.**
* Enter the Git repository URL of the project to pull the code from GitHub or provide the location of **pom.xml** file in your local system.
* Build the job.

**Step 6.2.5:** Push 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**