JUnit Testing Exercises

1)Setting up JUnit

Code:

Inside pom.xml, under the dependencies added the following depencies:

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.hamcrest</groupId>

<artifactId>hamcrest</artifactId>

<version>2.2</version>

<scope>test</scope>

</dependency>

Inside src/main/java created Calculator class:

**package**com.example.JUnitDemo;

**public** **class**Calculator {

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

**return** a+ b;

}

}

Inside src/test/java created CalculatorTest as JUnit Test case:

**package**com.example.JUnitDemo;

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

**import** org.junit.Test;

**public** **class**CalculatorTest {

@Test

**public** **void**test() {

Calculator calc=**new** Calculator();

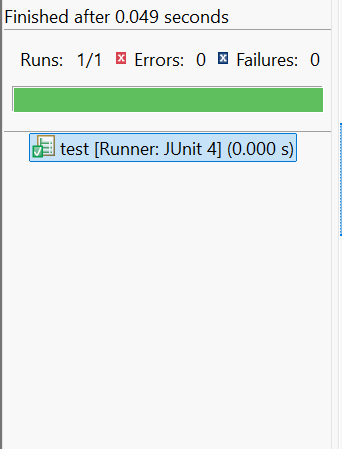
*assertEquals*(5,calc.add(2, 3));

}

}

After this,saved the file and right click it then Run As->JUnit Test.

Output:



3) Assertions in JUnit

Code:

Inside src/test/java create a new file of JUnit Test case:

AssertionsTest.java

**package** com.example.JUnitDemo;

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

**import** org.junit.Test;

**public** **class** AssertionsTest {

@Test

**public** **void** testAssertions() {

// Assert equals

*assertEquals*(5, 2 + 3);

// Assert true

*assertTrue*(5 > 3);

// Assert false

*assertFalse*(5 < 3);

// Assert null

*assertNull*(**null**);

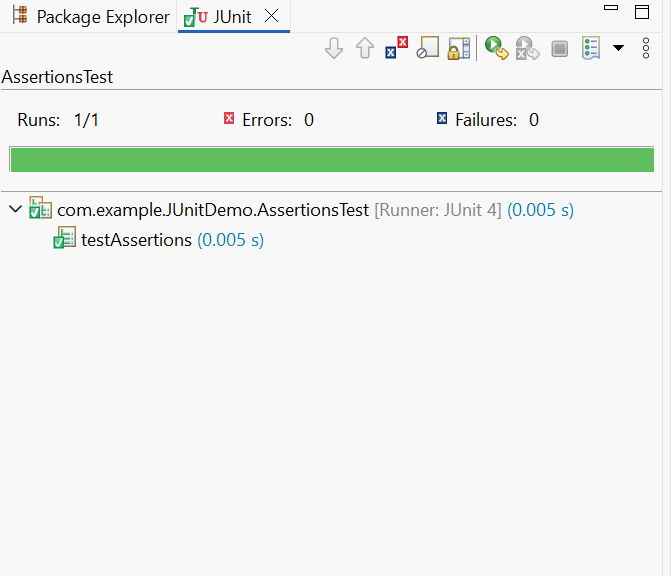
// Assert not null

*assertNotNull*(**new**Object());

}

}

pom.xml must have the dependencies as previously mentioned.  
  
Then right click Run As->JUnit Test.

Output:

4) Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in JUnit.

Code:

Inside src/main/java created Calculator1 class

**package** com.example.JUnitDemo;

**public** **class** Calculator1 {

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

**return** a+b;

}

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

**return** a-b;

}

}

Inside src/test/java created JUnit Test case

**package** com.example.JUnitDemo;

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

**import** org.junit.After;

**import** org.junit.Before;

**import** org.junit.Test;

**public** **class** Calculator1Test {

**private** Calculator1 calculator;

@Before

**public** **void** setUp() {

calculator= **new** Calculator1(); // Arrange

System.***out***.println("Setup called");

}

@After

**public** **void** tearDown() {

System.***out***.println("Teardown called");

}

@Test

**public** **void** testAddition() {

// Act

**int** result= calculator.add(4, 5);

// Assert

*assertEquals*(9, result);

}

@Test

**public** **void** testSubtraction() {

// Act

**int** result= calculator.subtract(10, 6);

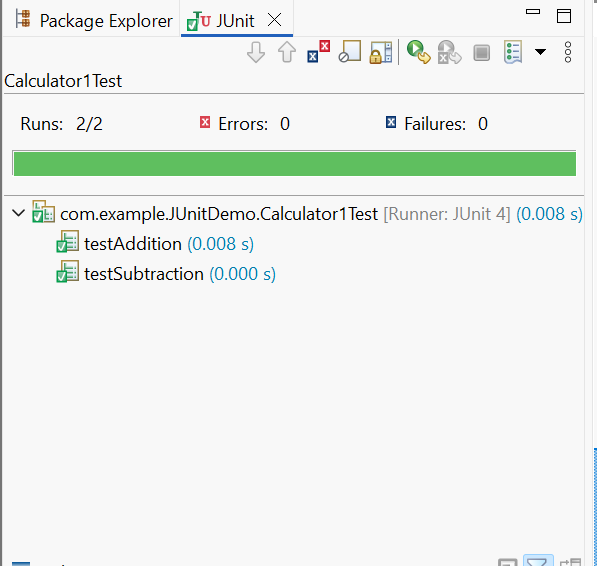
// Assert

*assertEquals*(4, result);

}

}

Output:



Mockito Hands-On Exercises

1) Mocking and Stubbing

Code:

Created a new project with Maven, in the pom.xml added the following dependencies:

<dependency>

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

<artifactId>junit-jupiter</artifactId>

<version>5.10.0</version>

<scope>test</scope>

</dependency>

<!-- Mockito -->

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-core</artifactId>

<version>5.11.0</version>

<scope>test</scope>

</dependency>

Right clicked the project, Maven->Update.

Inside src/main/java created ExternalApi interface and MyService class

ExternalApi.java

**package** com.example.MockitoDemo;

**public** **interface** ExternalApi {

String getData();

}

MyService.java

**package** com.example.MockitoDemo;

**public** **class** MyService {

**private** ExternalApi api;

**public** MyService(ExternalApi api) {

**this**.api=api;

}

**public** String fetchData() {

**return** api.getData();

}

}

Inside src/test/java created MyServiceTest.java file

**Package** com.example.MockitoDemo;

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

**import** **static** org.mockito.Mockito.\*;

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

**public** **class** MyServiceTest {

@Test

**public** **void** testExternalApi() {

// Step 1: Mock

ExternalApi mockApi= *mock*(ExternalApi.**class**);

// Step 2: Stub

*when*(mockApi.getData()).then Return("Mock Data");

// Step 3: Use in service

MyService service= **new** MyService(mockApi);

String result= service.fetchData();

// Step 4: Assert

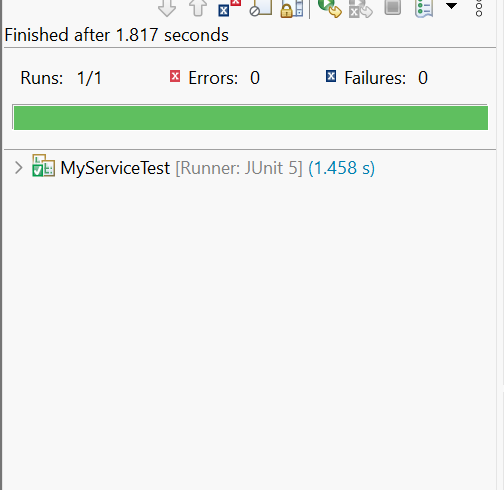
*assertEquals*("Mock Data", result);

}

}

Then run the MyServiceTest.java file as JUnit Test

Output:



2) Verifying Interactions

Code:

In pom.xml under dependencies added the following dependencies:

<dependency>

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

<artifactId>junit-jupiter</artifactId>

<version>5.10.0</version>

<scope>test</scope>

</dependency>

<!-- Mockito -->

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-core</artifactId>

<version>5.11.0</version>

<scope>test</scope>

</dependency>

Inside src/main/java created ExternalApi interface and MyService class

ExternalApi.java

**package** com.example.MockitoDemo1;

**public** **interface** ExternalApi {

String getData();

}

MyService.java

**package** com.example.MockitoDemo1;

**public** **class** MyService {

**private** ExternalApi api;

**public** MyService(ExternalApi api) {

**this**.api=api;

}

**public** String fetchData() {

**return** api.getData();

}

}

Inside src/test/java created MyServiceTest.java JUnit Test Case

**package** com.example.MockitoDemo1;

//import static org.junit.jupiter.api.Assertions.\*;

**import** **static** org.mockito.Mockito.\*;

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

**import** org.mockito.Mockito;

**public** **class** MyServiceTest {

@Test

**public** **void** testVerifyInteraction() {

// Step 1: Create mock object

ExternalApi mockApi= Mockito.*mock*(ExternalApi.**class**);

// Step 2: Inject mock into service

MyService service= **new** MyService(mockApi);

// Step 3: Call method on service

service.fetchData();

// Step 4: Verify the interaction

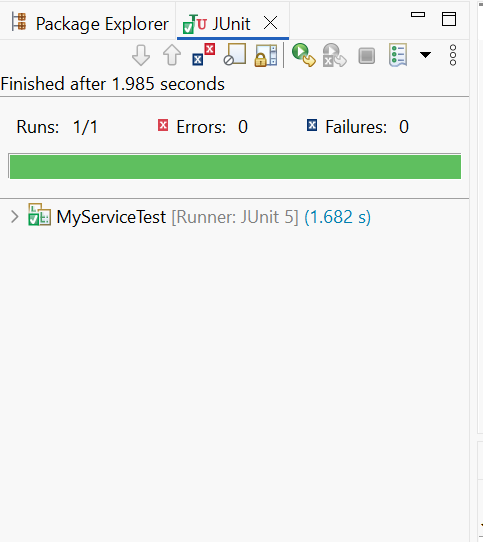
*verify*(mockApi).getData(); // This ensures getData() was called

}

}

Then run the MyServiceTest.java file as JUnit Test

Output:



Logging using SLF4J

1) Logging Error Messages and Warning Levels

Code:

Created a new Project with Maven. In pom.xml added the following dependencies under the dependencies section.

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

<version>1.7.30</version>

</dependency>

<!-- Logback Classic: SLF4J Implementation -->

<dependency>

<groupId>ch.qos.logback</groupId>

<artifactId>logback-classic</artifactId>

<version>1.2.3</version>

</dependency>

Inside src/main/java created LoggingExample class

**package** com.example.LoggingDemo;

**import** org.slf4j.Logger;

**import** org.slf4j.LoggerFactory;

**public** **class** LoggingExample {

**private** **static** **final** Logger ***logger***=LoggerFactory.*getLogger*(LoggingExample.**class**);

**public** **static** **void** main(String[] args) {

***logger***.error("This is an error message");

***logger***.warn("This is a warning message");

***logger***.info("This is an info message");

***logger***.debug("This is a debug message");

}

}

On running this file as Java application the following output is generated.

Output:

