1. JUnit\_Basic Testing Exercises

**Pom.xml:**

<?xml version="1.0" encoding="UTF-8"?>  
<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>org.example</groupId>  
 <artifactId>junitdemo</artifactId>  
 <version>1.0-SNAPSHOT</version>  
 <properties>  
 <maven.compiler.source>20</maven.compiler.source>  
 <maven.compiler.target>20</maven.compiler.target>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>junit</groupId>  
 <artifactId>junit</artifactId>  
 <version>4.13.2</version>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
</project>

**Exercise 1: Setting Up JUnit**

**CODE:**

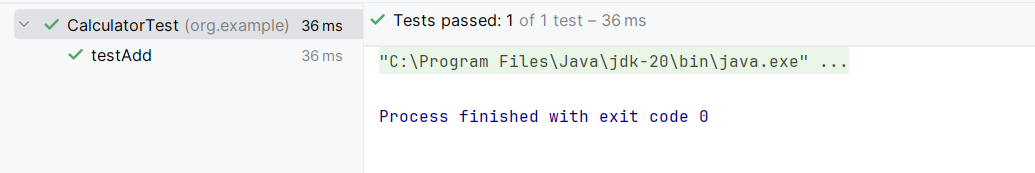
**Calculator.java**

package org.example;  
public class Calculator {  
 public int add(int a, int b) {  
 return a + b;  
 }  
}

**CalculatorTest.java**

package org.example;  
import org.junit.Test;  
import static org.junit.Assert.\*;  
public class CalculatorTest {  
 @Test  
 public void testAdd() {  
 Calculator calc = new Calculator();  
 int result = calc.add(2, 3);  
 assertEquals(5, result);  
 }  
}

**OUTPUT:**

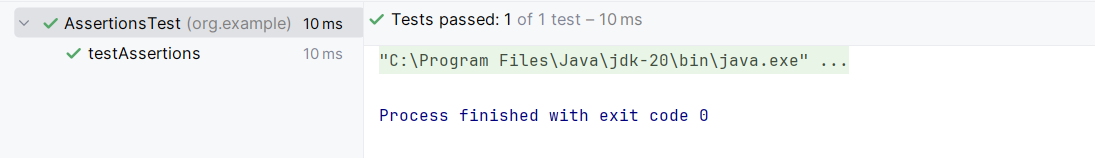


**Exercise 3: Assertions in Junit**

**CODE:**

package org.example;  
import org.junit.Test;  
import static org.junit.Assert.\*;  
 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());  
 }  
 }

**OUTPUT:**

****

**Exercise 4: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in Junit**

**CODE:**

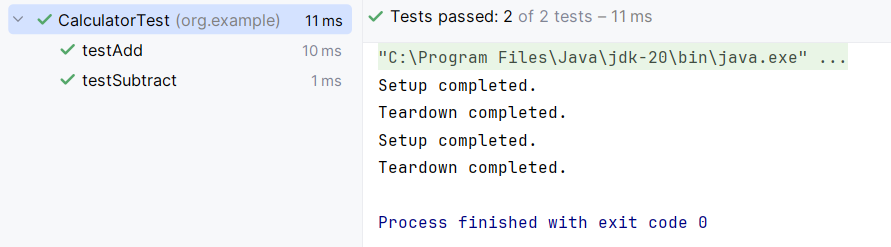
**Calculator.java:**

package org.example;  
public class Calculator {  
 public int add(int a, int b) {  
 return a + b;  
 }  
 public int subtract(int a, int b) {  
 return a - b;  
 }  
}

**CalculatorTest.java:**

package org.example;  
import org.junit.Test;  
import static org.junit.Assert.\*;  
import org.junit.After;  
import org.junit.Before;  
 public class CalculatorTest {  
 private Calculator calc;  
 @Before  
 public void setUp() {  
 calc = new Calculator();  
 System.out.println("Setup completed.");  
 }  
 @After  
 public void tearDown() {  
 calc = null;  
 System.out.println("Teardown completed.");  
 }  
 @Test  
 public void testAdd() {  
 int a = 10;  
 int b = 5;  
 int result = calc.add(a, b);  
 assertEquals(15, result);  
 }  
 @Test  
 public void testSubtract() {  
 int a = 10;  
 int b = 3;  
 int result = calc.subtract(a, b);  
 assertEquals(7, result);  
 }  
 }

**OUTPUT:**

****

1. Mockito exercises

**pom.xml:**

<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>org.example</groupId>  
 <artifactId>junitdemo</artifactId>  
 <version>1.0-SNAPSHOT</version>  
 <properties>  
 <maven.compiler.source>20</maven.compiler.source>  
 <maven.compiler.target>20</maven.compiler.target>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>junit</groupId>  
 <artifactId>junit</artifactId>  
 <version>4.13.2</version>  
 <scope>test</scope>  
 </dependency>  
 <dependency>  
 <groupId>org.mockito</groupId>  
 <artifactId>mockito-core</artifactId>  
 <version>3.12.4</version> <!-- This version is safe with JUnit 4 -->  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
</project>

**ExternalApi:**

package org.example;  
public interface ExternalApi {  
 String getData();  
}

**MyService:**

package org.example;  
public class MyService {  
 private final ExternalApi api;  
 public MyService(ExternalApi api) {  
 this.api = api;  
 }  
 public String fetchData() {  
 return api.getData();  
 }  
}

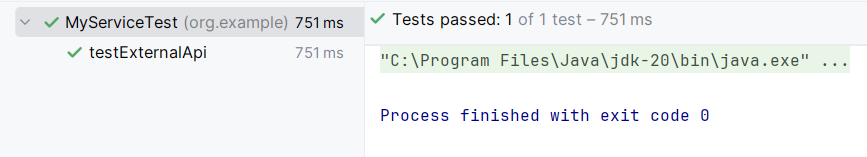
**Exercise 1: Mocking and Stubbing**

**CODE:**

**MyServiceTest:**

package org.example;  
import org.junit.Test;   
import static org.junit.Assert.\*;   
import static org.mockito.Mockito.\*;  
public class MyServiceTest {  
 @Test  
 public void testExternalApi() {  
 // Arrange  
 ExternalApi mockApi = mock(ExternalApi.class);  
 // Stub method  
 when(mockApi.getData()).thenReturn("Mock Data");  
 // Act  
 MyService service = new MyService(mockApi);  
 String result = service.fetchData();  
 // Assert  
 assertEquals("Mock Data", result);  
 }  
}

**OUTPUT:**



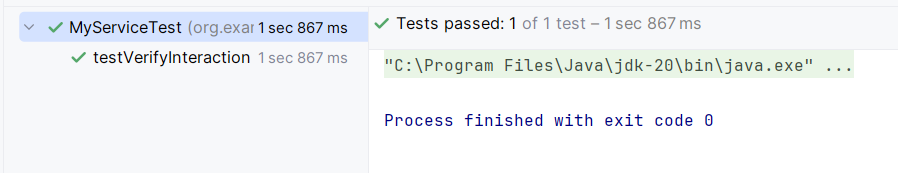
**Exercise 2: Verifying Interactions**

**CODE:**

**MyServiceTest:**

package org.example;  
import org.junit.Test;  
import static org.mockito.Mockito.\*;  
public class MyServiceTest {  
 @Test  
 public void testVerifyInteraction() {  
 ExternalApi mockApi = mock(ExternalApi.class);  
 MyService service = new MyService(mockApi);  
 service.fetchData();  
 verify(mockApi).getData();  
 }  
}

**OUTPUT:**

****