JUnit Testing Exercises

Exercise 1: Setting Up JUnit

1,2

Code

<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 https://maven.apache.org/xsd/maven-4.0.0.xsd">

 <modelVersion>4.0.0</modelVersion>

 <groupId>com.example</groupId>

 <artifactId>junit-demo</artifactId>

 <version>0.0.1-SNAPSHOT</version>

 <dependencies>

   <dependency>

     <groupId>junit</groupId>

     <artifactId>junit</artifactId>

     <version>4.13.2</version>

     <scope>test</scope>

   </dependency>

 </dependencies>

</project>

3

Code

package com.example;

public class Calculator {

   public int add(int a, int b) {

       return a + b;

   }

}

package com.example;

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

*@Test*

   public void testAdd() {

       Calculator calc = new Calculator();

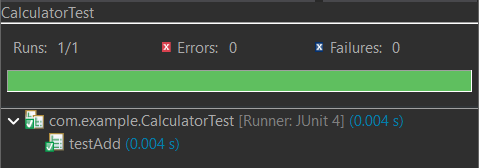
       int result = calc.add(2, 3);

*assertEquals*(5, result);

   }

}

Output



Exercise 2: Writing Basic JUnit Tests

1

Code

package com.example;

public class Calculator {

   public int add(int a, int b) {

       return a + b;

   }

   public int subtract(int a, int b) {

       return a - b;

   }

}

2

Code

package com.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);

   }

*@Test*

   public void testSubtract() {

       Calculator calc = new Calculator();

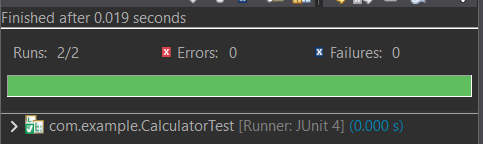
       int result = calc.subtract(5, 3);

*assertEquals*(2, result);

   }

}

Output



Exercise 3: Assertions in Junit

Code

package com.example;

import org.junit.Test;

import static org.junit.Assert.\*;

public class AssertionsTest {

*@Test*

   public void testAssertions() {

*assertEquals*(5, 2 + 3);

*assertTrue*(5 > 3);

*assertFalse*(5 < 3);

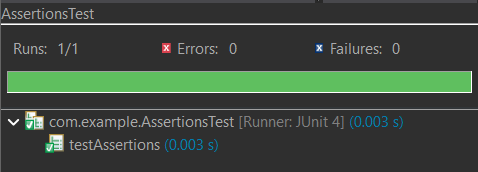
*assertNull*(null);

*assertNotNull*(new Object());

   }

}

Output



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

Code

package com.example;

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

public class TestCalculator {

   private Calculator calc;

*@BeforeEach*

   public void setUp() {

       calc = new Calculator();

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

   }

*@AfterEach*

   public void tearDown() {

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

   }

*@Test*

   public void testAdd() {

       int result = calc.add(10, 20);

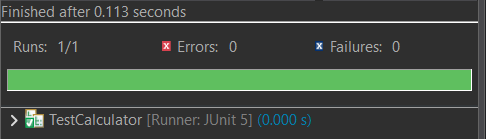
       Assertions.*assertEquals*(30, result);

   }

}

Output





Mockito Hands-On Exercises

Exercise 1: Mocking and Stubbing

Code

package com.example;

public interface ExternalApi {

   String getData();

}

package com.example;

public class MyService {

   private final ExternalApi api;

   public MyService(ExternalApi api) {

       this.api = api;

   }

   public String fetchData() {

       return api.getData();

   }

}

package com.example;

import org.junit.jupiter.api.Test;

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

import static org.mockito.Mockito.\*;

public class MyServiceTest {

*@Test*

   public void testExternalApi() {

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

*when*(mockApi.getData()).thenReturn("Mock Data");

       MyService service = new MyService(mockApi);

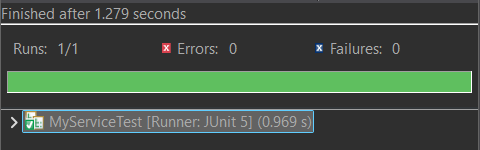
       String result = service.fetchData();

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

   }

}

Output



Exercise 2: Verifying Interactions

Code

package com.example;

import org.junit.jupiter.api.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 interaction

*verify*(mockApi).getData();

   }

}

