**Exercise 1: Mocking and Stubbing**

Scenario:

You need to test a service that depends on an external API. Use Mockito to mock the

external API and stub its methods.

Steps:

1. Create a mock object for the external API.

2. Stub the methods to return predefined values.

3. Write a test case that uses the mock object.

Solution Code:

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

public class MyServiceTest {

@Test

public void testExternalApi() {

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

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

MyService service = new MyService(mockApi);

String result = service.fetchData();

assertEquals("Mock Data", result);

}

}

**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

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

<modelVersion>4.0.0</modelVersion>

<groupId>week2</groupId>

<artifactId>junit</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<!-- JUnit 4 dependency -->

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.13.2</version>

<scope>test</scope>

</dependency>

<!-- Mockito Core dependency -->

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-core</artifactId>

<version>4.11.0</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

**Src/main/java/week2/ExternalApi.java**

**package** week2;

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

String getData();

}

**Src/main/java/week2/MyService.java**

**package** week2;

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

**private** ExternalApi externalApi;

**public** MyService(ExternalApi externalApi) {

**this**.externalApi = externalApi;

}

**public** String fetchData() {

**return** externalApi.getData();

}

}

**Src/test /java/week2/MyServiceTest.java**

**package** week2;

**import** org.junit.Test;

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

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

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

@Test

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

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

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

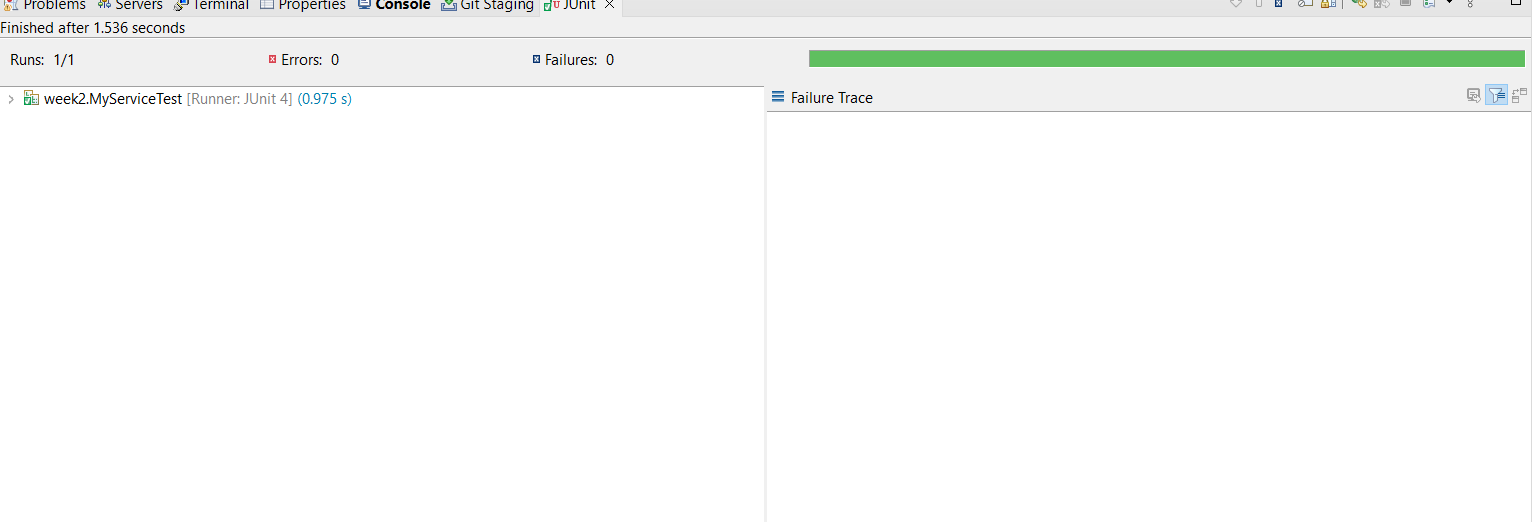
MyService service = **new** MyService(mockApi);

String result = service.fetchData();

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

}

}



**Exercise 2: Verifying Interactions**

Scenario:

You need to ensure that a method is called with specific arguments.

Steps:

1. Create a mock object.

2. Call the method with specific arguments.

3. Verify the interaction.

Solution Code:

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

public class MyServiceTest {

@Test

public void testVerifyInteraction() {

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

MyService service = new MyService(mockApi);

service.fetchData();

verify(mockApi).getData();

}

}

**Src/main/java/week2/ExternalApi.java**

**package** week2;

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

String getData();

}

**Src/main/java/week2/MyService.java**

**package** week2;

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

**private** ExternalApi externalApi;

**public** MyService(ExternalApi externalApi) {

**this**.externalApi = externalApi;

}

**public** String fetchData() {

**return** externalApi.getData();

}

}

**Src/test /java/week2/MyServiceTest.java**

**package** week2;

**import** org.junit.Test;

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

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

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

@Test

**public** **void** testFetchData() {

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

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

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

String result = service.fetchData();

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

}

@Test

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

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

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

service.fetchData();

*verify*(mockApi).getData();

}

}

