Exercise 2: Verifying Interactions

ExternalApi.java

package org.example;

public interface ExternalApi {

String getData();

void sendData(String payload);

void deleteItem(String id);

}

MyService.java

package org.example;

public class MyService {

private final ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

public void updateData(String payload) {

api.sendData(payload);

}

public void removeItem(String id) {

api.deleteItem(id);

}

}

MyServiceTest

package com.example;

import org.example.ExternalApi;

import org.example.MyService;

import org.junit.jupiter.api.Test;

import org.mockito.InOrder;

import static org.mockito.Mockito.\*;

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

public class MyServiceTest {

@Test

public void testFetchDataWithMock() {

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 testVerifySendDataCalled() {

ExternalApi mockApi = mock(ExternalApi.class);

MyService service = new MyService(mockApi);

service.updateData("payload");

verify(mockApi).sendData("payload");

}

@Test

public void testArgumentMatching() {

ExternalApi mockApi = mock(ExternalApi.class);

MyService service = new MyService(mockApi);

service.updateData("data\_123");

verify(mockApi).sendData(argThat(arg -> arg.startsWith("data")));

}

@Test

public void testHandleVoidDeleteMethod() {

ExternalApi mockApi = mock(ExternalApi.class);

MyService service = new MyService(mockApi);

doNothing().when(mockApi).deleteItem("abc123");

service.removeItem("abc123");

verify(mockApi).deleteItem("abc123");

}

@Test

public void testMultipleReturns() {

ExternalApi mockApi = mock(ExternalApi.class);

when(mockApi.getData())

.thenReturn("First")

.thenReturn("Second")

.thenReturn("Third");

MyService service = new MyService(mockApi);

assertEquals("First", service.fetchData());

assertEquals("Second", service.fetchData());

assertEquals("Third", service.fetchData());

}

@Test

public void testInteractionOrder() {

ExternalApi mockApi = mock(ExternalApi.class);

MyService service = new MyService(mockApi);

service.updateData("A");

service.removeItem("B");

InOrder inOrder = inOrder(mockApi);

inOrder.verify(mockApi).sendData("A");

inOrder.verify(mockApi).deleteItem("B");

}

@Test

public void testVoidThrowsException() {

ExternalApi mockApi = mock(ExternalApi.class);

doThrow(new RuntimeException("Delete failed"))

.when(mockApi)

.deleteItem("x1");

MyService service = new MyService(mockApi);

assertThrows(RuntimeException.class, () -> service.removeItem("x1"));

}}

Output:  
