import org.junit.jupiter.api.Test;

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

public class SampleTest {

@Test

public void testAddition() {

assertEquals(2, 1 + 1);

}

}

import org.junit.jupiter.api.Test;

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

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());

}

}

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

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

public class AAAPatternTest {

private int value;

@BeforeEach

public void setUp() {

// Arrange: Initialize or set up resources before each test

value = 10;

}

@AfterEach

public void tearDown() {

// Teardown: Clean up resources after each test

value = 0;

}

@Test

public void testAddition() {

// Arrange

int addend = 5;

// Act

int result = value + addend;

// Assert

assertEquals(15, result);

}

@Test

public void testSubtraction() {

// Arrange

int subtrahend = 3;

// Act

int result = value - subtrahend;

// Assert

assertEquals(7, result);

}

}

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

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

import org.mockito.Mockito;

// ExternalApi interface to be mocked

interface ExternalApi {

String getData();

}

// Service class that depends on ExternalApi

class MyService {

private final ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

public class MyServiceTest {

@Test

public void testExternalApi() {

// Create a mock object for the external API

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

// Stub the methods to return predefined values

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

// Use the mock in the service

MyService service = new MyService(mockApi);

// Assert the result

String result = service.fetchData();

assertEquals("Mock Data", result);

}

}

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

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

import org.mockito.Mockito;

// ExternalApi interface to be mocked

interface ExternalApi {

String getData();

}

// Service class that depends on ExternalApi

class MyService {

private final ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

public class MyServiceTest {

@Test

public void testVerifyInteraction() {

// Create a mock object

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

// Use the mock in the service

MyService service = new MyService(mockApi);

// Call the method

service.fetchData();

// Verify the interaction

verify(mockApi).getData();

}

}

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");

}

}