**Setting Up Junit**

**CalculatorTest.java**

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

*@*Test

public void testAddition() {

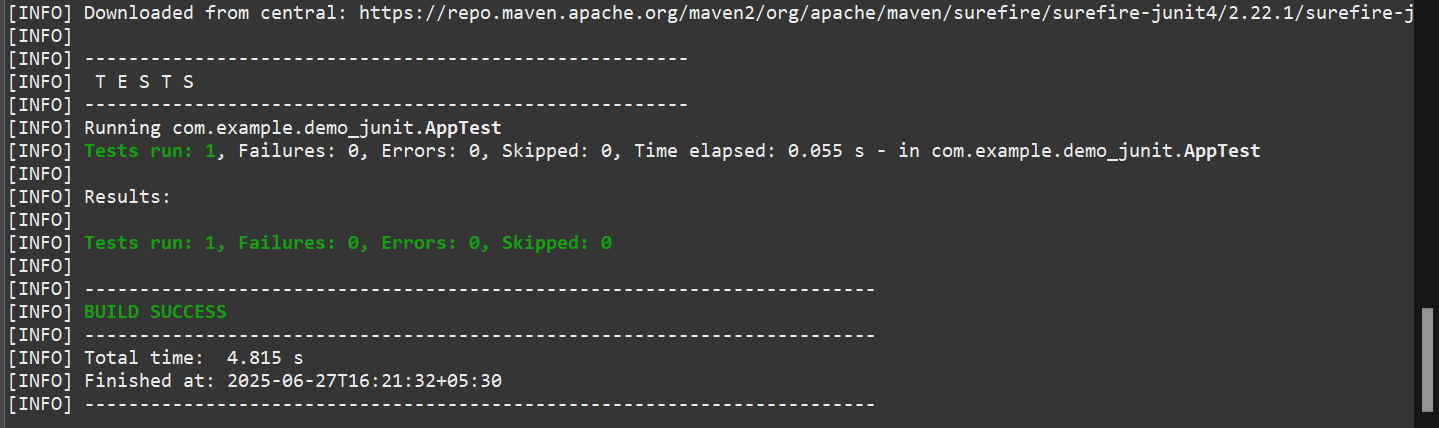
int result = 2 + 3;

assertEquals(5, result);

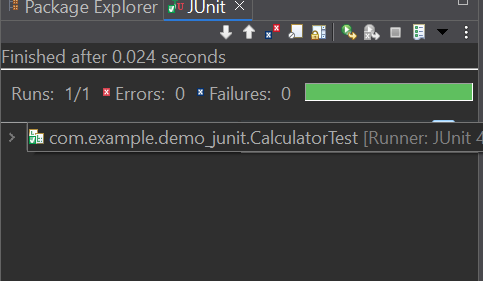
}

}

**Test Output:**



**JUnit Test:**



2. **Assertions in JUnit**

package com.example.demo\_junit;

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);

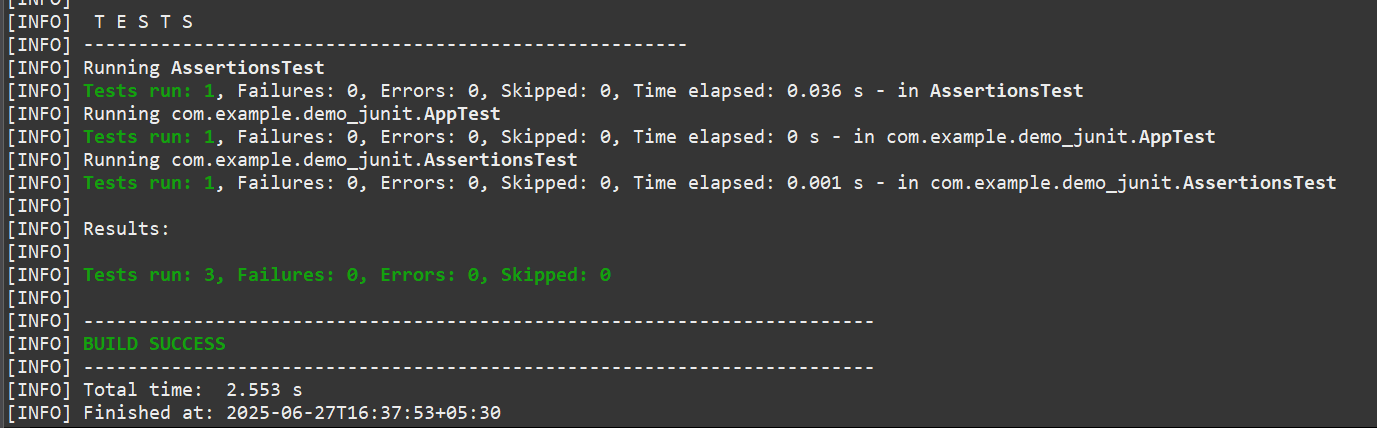
*assertNull*(null);

*assertNotNull*(new Object());

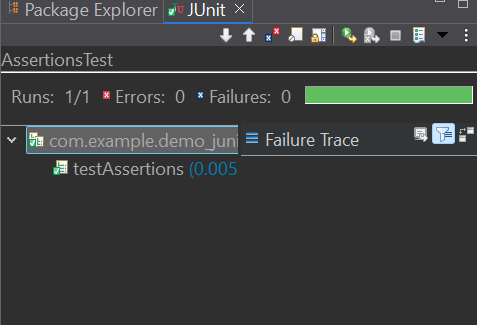
}

}

**Test Output:**



**Junit Test**



**3. AAA Pattern:**

**Calculator.java:**

package com.example.demo\_junit;

public class Calculator {

public int add(int a, int b) {

return a + b;

}

public int multiply(int a, int b) {

return a \* b;

}

}

**CalculatorTest.java:**

package com.example.demo\_junit;

import org.junit.Before;

import org.junit.After;

import org.junit.Test;

import static org.junit.Assert.\*;

public class CalculatorTest {

private Calculator calculator;

*@Before*

public void setUp() {

calculator = new Calculator(); // Arrange

System.***out***.println("Setup complete.");

}

// Teardown method – runs after each test

*@After*

public void tearDown() {

calculator = null;

System.***out***.println("Teardown complete.");

}

*@Test*

public void testAddition() {

// Arrange done in setUp()

// Act

int result = calculator.add(2, 3);

// Assert

*assertEquals*(5, result);

}

*@Test*

public void testMultiplication() {

// Act

int result = calculator.multiply(4, 5);

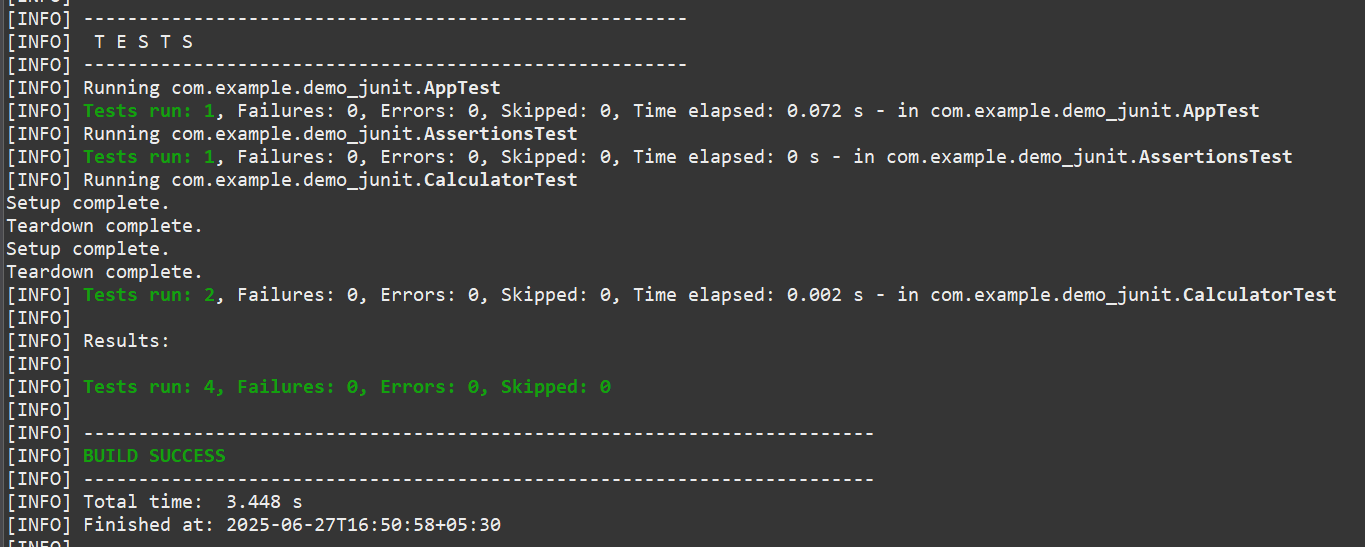
// Assert

*assertEquals*(20, result);

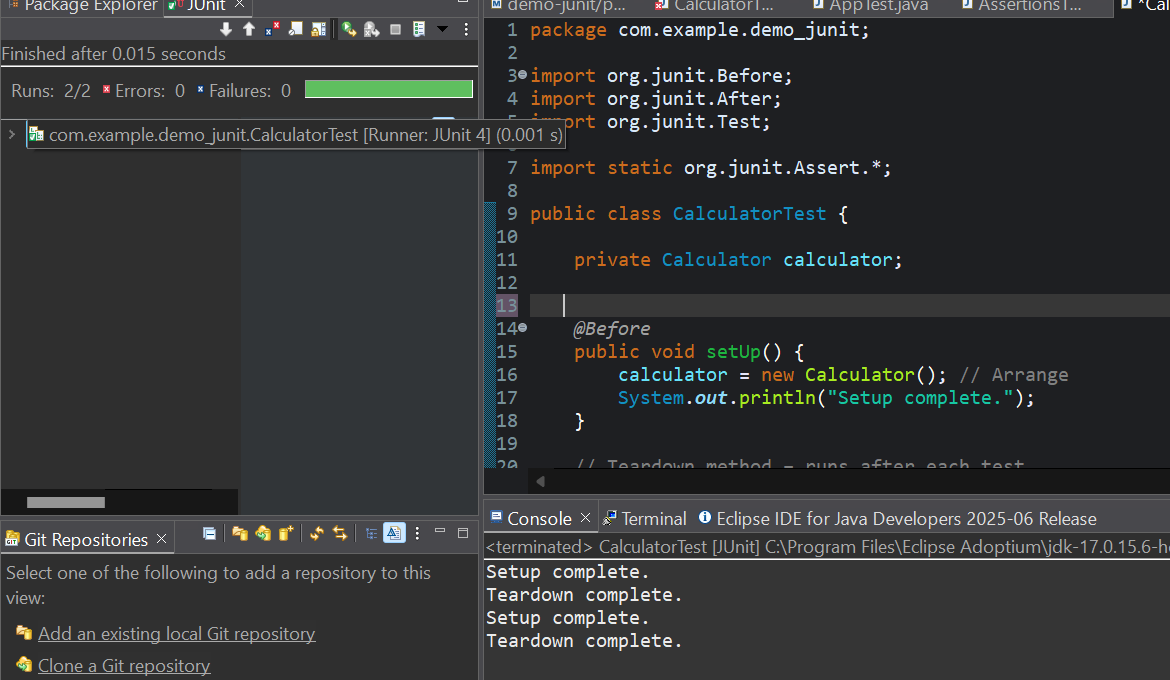
}

}

**Output:**

****

**Junit Output:**



**4. Mockito :**

**Mocking and Stubbing**

**ExternalApi.java**

package com.example.mockito;

public interface ExternalApi {

String getData();

}

**MyService.java**

package com.example.mockito;

public class MyService {

private final ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

**MyServiceTest.java**

package com.example.mockito;

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() {

// Create mock

ExternalApi mockApi = mock(ExternalApi.class);

// Stub method

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

// Inject mock into service

MyService service = new MyService(mockApi);

String result = service.fetchData();

// Assert the result

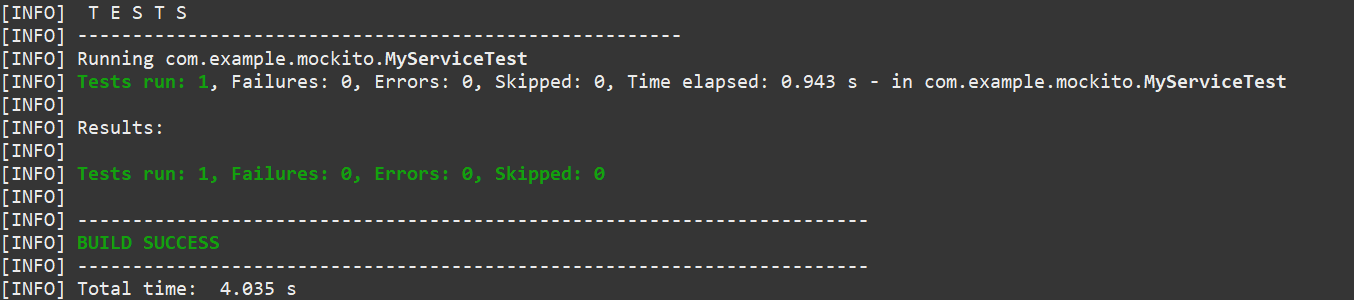
assertEquals("Mock Data", result);

}

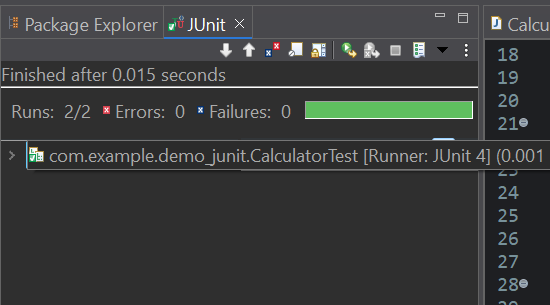
}

**Maven Tests:**

**MyService.java**



**Junit Test:**



**Verifying Interations**

**TestDataUploader.java**

package com.mockito;

import static org.mockito.Mockito.\*;

import org.junit.jupiter.api.Test;

import org.mockito.Mockito;

public class TestDataUploader {

*@Test*

public void testUploadCalledWithCorrectArgument() {

// Step 1: Create mock

NetworkClient mockClient = Mockito.*mock*(NetworkClient.class);

// Step 2: Call method with specific argument

DataUploader uploader = new DataUploader(mockClient);

uploader.performUpload();

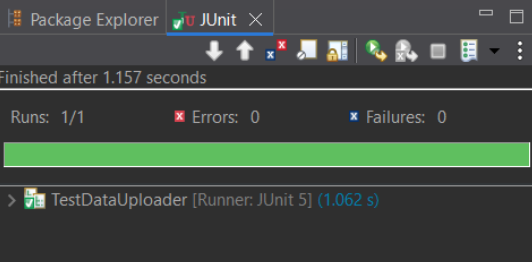
// Step 3: Verify interaction

*verify*(mockClient).upload("TestPayload");

}

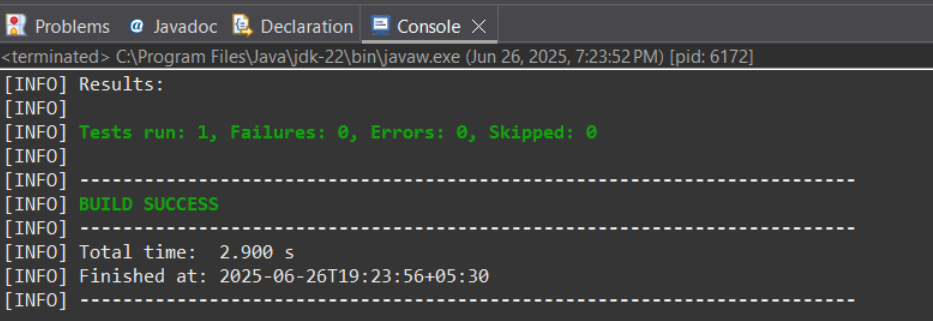
}

**JUnit Test:**



**Maven Test:**

**TestDataUploader.java**



**Logging Error Messages and Warning Levels**

**LoggingExample.java**

package com.example.mockito;

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");

}

}

**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>com.example</groupId>

<artifactId>mockito</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>mockito</name>

<!-- FIXME change it to the project's website -->

<url>http://www.example.com</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<maven.compiler.source>17</maven.compiler.source>

<maven.compiler.target>17</maven.compiler.target>

</properties>

<dependencies>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter</artifactId>

<version>5.10.0</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

<version>1.7.30</version>

</dependency>

<dependency>

<groupId>ch.qos.logback</groupId>

<artifactId>logback-classic</artifactId>

<version>1.2.3</version>

</dependency>

<!-- ✅ Mockito -->

<dependency>

<groupId>org.mockito</groupId>

<artifactId>mockito-core</artifactId>

<version>5.11.0</version>

<scope>test</scope>

</dependency> </dependencies>

<build>

<pluginManagement><!-- lock down plugins versions to avoid using Maven defaults (may be moved to parent pom) -->

<plugins>

<!-- clean lifecycle, see https://maven.apache.org/ref/current/maven-core/lifecycles.html#clean\_Lifecycle -->

<plugin>

<artifactId>maven-clean-plugin</artifactId>

<version>3.1.0</version>

</plugin>

<!-- default lifecycle, jar packaging: see https://maven.apache.org/ref/current/maven-core/default-bindings.html#Plugin\_bindings\_for\_jar\_packaging -->

<plugin>

<artifactId>maven-resources-plugin</artifactId>

<version>3.0.2</version>

</plugin>

<plugin>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.8.0</version>

</plugin>

<plugin>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.22.1</version>

</plugin>

<plugin>

<artifactId>maven-jar-plugin</artifactId>

<version>3.0.2</version>

</plugin>

<plugin>

<artifactId>maven-install-plugin</artifactId>

<version>2.5.2</version>

</plugin>

<plugin>

<artifactId>maven-deploy-plugin</artifactId>

<version>2.8.2</version>

</plugin>

<!-- site lifecycle, see https://maven.apache.org/ref/current/maven-core/lifecycles.html#site\_Lifecycle -->

<plugin>

<artifactId>maven-site-plugin</artifactId>

<version>3.7.1</version>

</plugin>

<plugin>

<artifactId>maven-project-info-reports-plugin</artifactId>

<version>3.0.0</version>

</plugin>

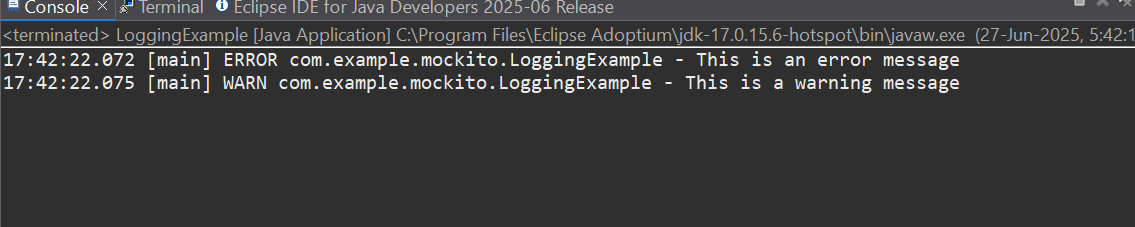
</plugins>

</pluginManagement>

</build>

</project>

**Output (Run as Java Application):**

****