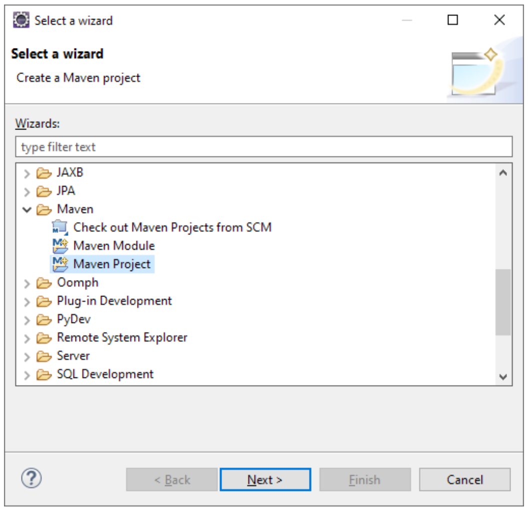
Εργαστήριο 02-UnitTest-Debugging

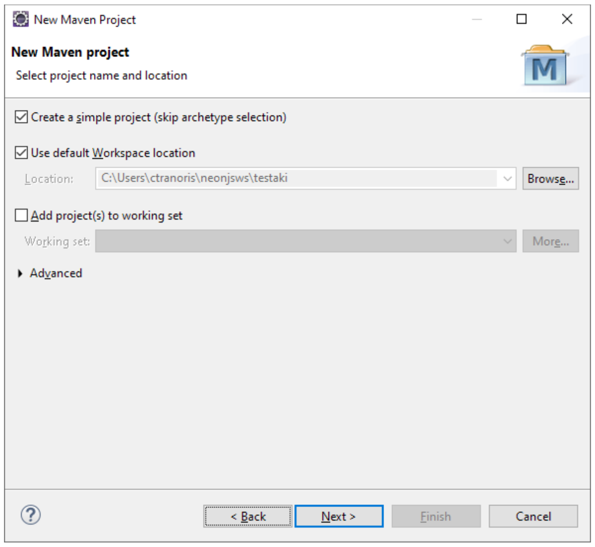
ΑΣΚΗΣΗ

Γράψιμο ενός τεστ JUnit 5 με Maven και Eclipse

Σε αυτή την άσκηση μαθαίνετε να γράφετε ένα τεστ JUnit5 χρησιμοποιώντας το Maven και το Eclipse IDE.

5.1. Project creation



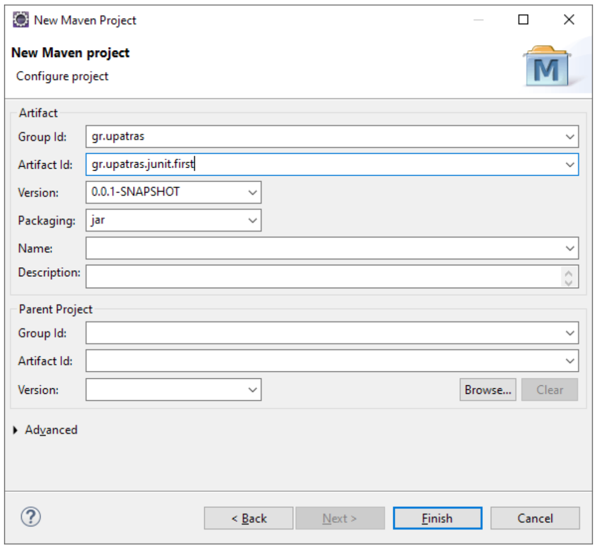


☑ Create a simple project (skip archetype selection)

☑ Use default Workspace location

Location: C:\Users\ctranoris\neonjsws\testaki

☐ Add project(s) to working set


New Maven Project

New Maven project

Configure project

Artifact

Group Id: gr.upatras

Artifact Id: gr.upatras.junit.first

Version: 0.0.1-SNAPSHOT

Packaging: jar

Name:

Description:

Parent Project

Group Id:

Artifact Id:

Version:

Create a new Maven project with the following settings:

- Group: `gr.upatras`
- Artifact: `gr.upatras.junit.first`
- Version: `0.0.1-SNAPSHOT`
- Packaging: jar

5.2. Configure Maven dependencies for JUnit 5

5.2.1. Steps required to configure Maven to use JUnit5

Για να χρησιμοποιήσετε το JUnit5 σε ένα έργο Maven, πρέπει:

• Ρυθμίστε τις παραμέτρους για χρήση Java 11 ή νεότερης έκδοσης, καθώς αυτό απαιτείται από το JUnit5

• Ρυθμίστε τις παραμέτρους του maven-surefire-plugin και του maven-failsafe-plugin ώστε να βρίσκονται στην έκδοση 2.22.2, ώστε να μπορούν να εκτελούν το JUnit5

• Προσθέστε εξαρτήσεις στο JUnit5 API και στη μηχανή για τον κωδικό δοκιμής σας

5.2.2. Configure Maven

Therefore, you need to adjust your pom file, similar to the following:

```
    <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>gr.upatras</groupId>
  <artifactId>gr.upatras.junit.first</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <maven.compiler.source>11</maven.compiler.source>
        <maven.compiler.target>11</maven.compiler.target>
    </properties>

    <!--1 -->
    <build>
        <plugins>
            <plugin>
                <artifactId>maven-surefire-plugin</artifactId>
                <version>2.22.2</version>
            </plugin>
            <plugin>
                <artifactId>maven-failsafe-plugin</artifactId>
                <version>2.22.2</version>
            </plugin>
        </plugins>
    </build>

    <!--2 -->
    <dependencies>
        <!-- https://mvnrepository.com/artifact/org.hamcrest/hamcrest-library -->
        <dependency>
            <groupId>org.junit.jupiter</groupId>
            <artifactId>junit-jupiter-api</artifactId>
            <version>5.7.2</version>
            <scope>test</scope>
        </dependency>
        <dependency>
            <groupId>org.junit.jupiter</groupId>
            <artifactId>junit-jupiter-engine</artifactId>
            <version>5.7.2</version>
            <scope>test</scope>
        </dependency>

    </dependencies>

</project>
```

Once you have done this, you can start using JUnit5 in your Maven project for writing unit tests.

5.2.3. Update Maven settings (in case you are using the Eclipse IDE)

Right-click your pom file, select **Maven Update Project** and select your project. This triggers an update of your project settings and dependencies.

5.3. Package creation

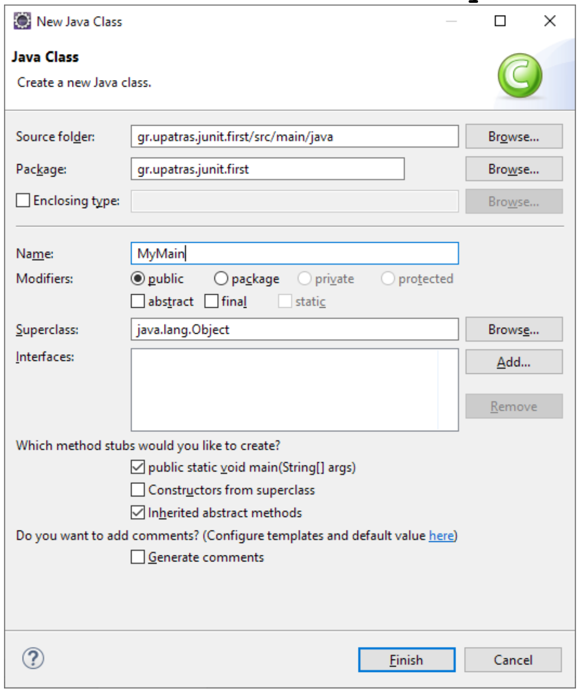
Create a package named *gr.upatras.junit.first* in the *src/main/java* and *src/test/java* folder.

5.4. Create a Java class

In the *src* folder, create the following class in the `gr.upatras.junit.first` package.

```
package gr.upatras.junit.first;

public class MyClass {
    // the following is just an example
  public int multiply(int x, int y) {
    if (x > 999) {
      throw new IllegalArgumentException("X should be less than 1000");
    }
    return x / y;
  }
}
```

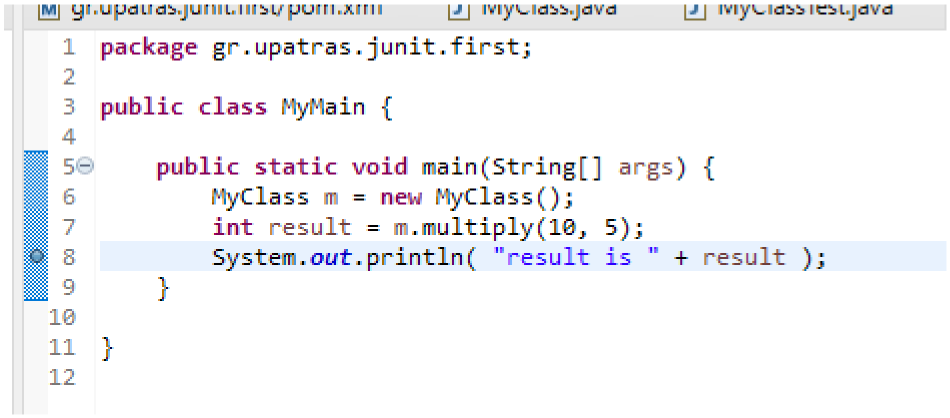
Create another class called MyMain



Source folder: gr.upatras.junit.first/src/main/java
Package: gr.upatras.junit.first
☐ Enclosing type:
Name: MyMain
Modifiers: [✓ public] package private protected
☐ abstract ☐ final ☐ static
Superclass: java.lang.Object
Interfaces:
Which method stubs would you like to create?
☑ public static void main(String[] args)
☐ Constructors from superclass
☑ Inherited abstract methods
Do you want to add comments? (Configure templates and default value here)
☐ Generate comments

Debugging

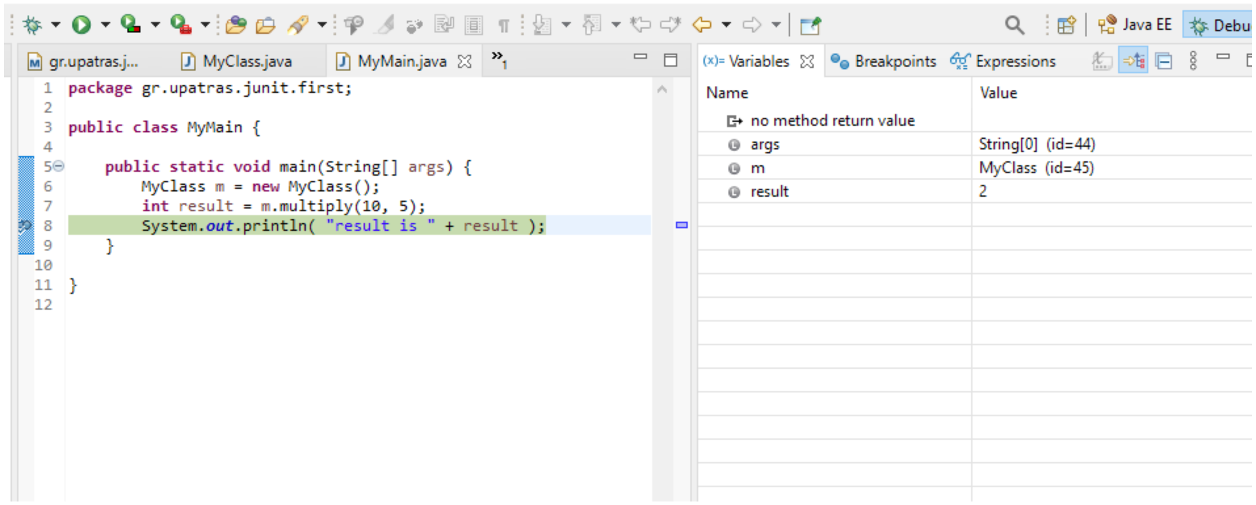
Insert break point



```java
package gr.upatras.junit.first;

public class MyMain {

    public static void main(String[] args) {
        MyClass m = new MyClass();
        int result = m.multiply(10, 5);
        System.out.println( "result is " + result );
    }

}
```

Debug As ->Java application

Eclipse will go in Debug mode perspective!



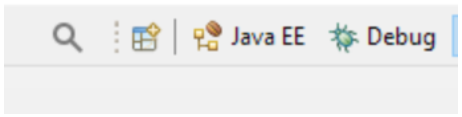
```java
package gr.upatras.junit.first;

public class MyMain {

    public static void main(String[] args) {
        MyClass m = new MyClass();
        int result = m.multiply(10, 5);
        System.out.println( "result is " + result );
    }

}
```

Name	Value
no method return value	
args	String[0] (id=44)
m	MyClass (id=45)
result	2

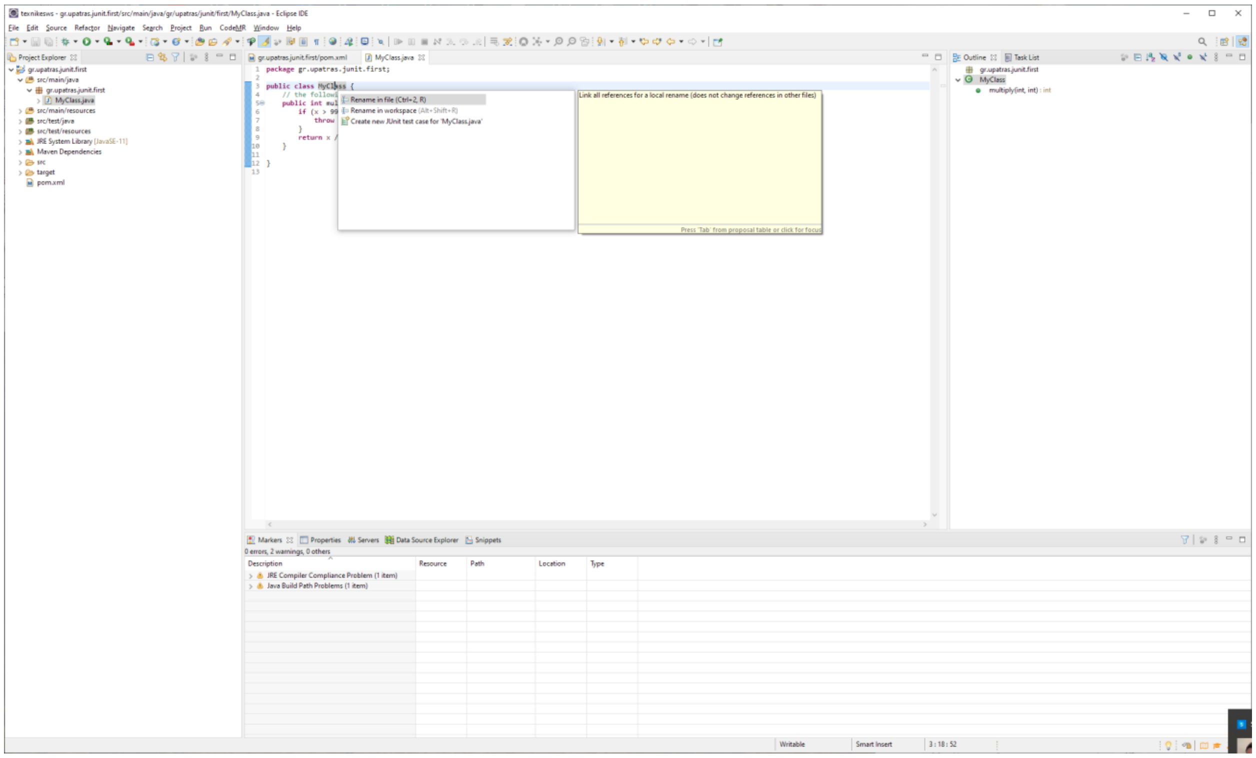
Inspect the result value

5.5. Create a JUnit test

Switch to java perspective! From the buttons on the right-top corner



Position the cursor on the `MyClass` in the Java editor and press Ctrl+1. Select that you want to create a new JUnit test from the list.



```
package gr.upatras.junit.first;

public class MyClass {
    // the followi
    public int mul
        if (x > 99
            throw
        }
        return x /
    }

}
```

Rename in file (Ctrl+2, R)
Rename in workspace (Alt+Shift+R)
Create new JUnit test case for 'MyClass.java'

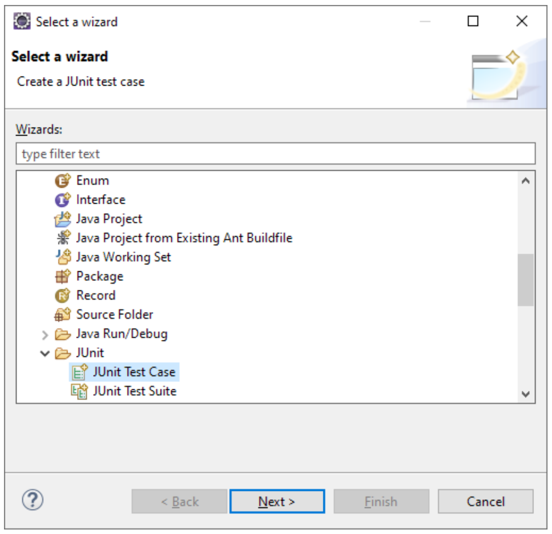
Link all references for a local rename (does not change references in other files)

Press 'Tab' from proposal table or click for focus

0 errors, 2 warnings, 0 others

JRE Compiler Compliance Problem (1 item)
Java Build Path Problems (1 item)

Alternatively you can right-click on your new class in the _Project Explorer_ or *Package Explorer* view and select **New Other Java JUnit Test Case**.



Select a wizard

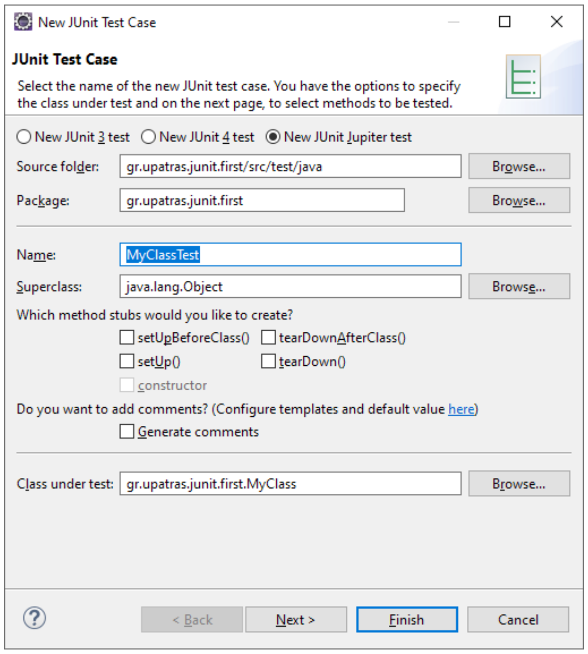
Select a wizard

Create a JUnit test case

Wizards:

type filter text

In the following wizard ensure that the *New JUnit Jupiter test* flag is selected. The source folder should select the *test* directory.


New JUnit Test Case

JUnit Test Case

Select the name of the new JUnit test case. You have the options to specify the class under test and on the next page, to select methods to be tested.

( ) New JUnit 3 test ( ) New JUnit 4 test (●) New JUnit Jupiter test

Source folder: gr.upatras.junit.first/src/test/java

Package: gr.upatras.junit.first

Name: MyClassTest

Superclass: java.lang.Object

Which method stubs would you like to create?

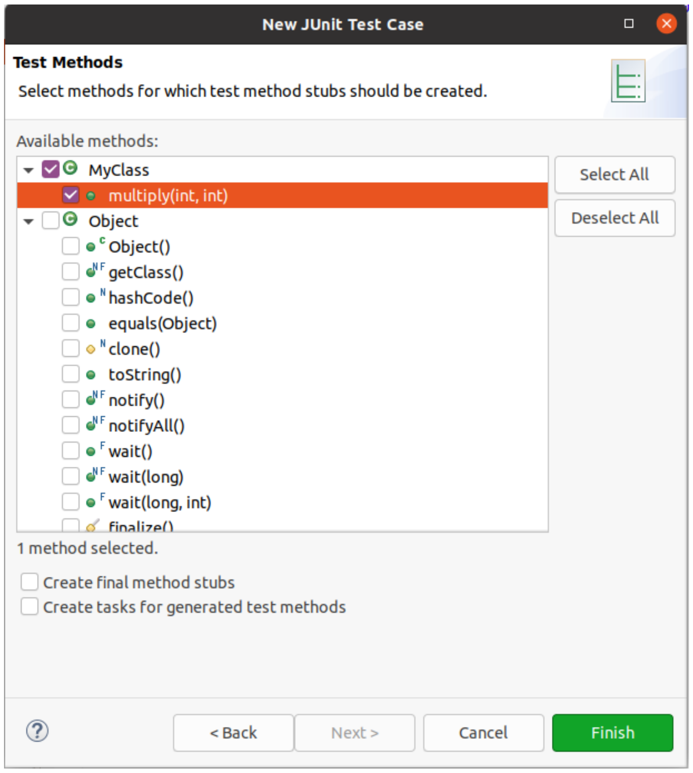
☐ setUpBeforeClass() ☐ tearDownAfterClass()
☐ setUp() ☐ tearDown()
☐ constructor

Do you want to add comments? (Configure templates and default value here)

☐ Generate comments

Class under test: gr.upatras.junit.first.MyClass

Press the **Next** button and select the methods that you want to test.


New JUnit Test Case

Test Methods

Select methods for which test method stubs should be created.

Available methods:

- ☑ MyClass
  - ☑ multiply(int, int)
- ☐ Object
  - ☐ Object()
  - ☐ getClass()
  - ☐ hashCode()
  - ☐ equals(Object)
  - ☐ clone()
  - ☐ toString()
  - ☐ notify()
  - ☐ notifyAll()
  - ☐ wait()
  - ☐ wait(long)
  - ☐ wait(long, int)
  - ☐ finalize()

1 method selected.

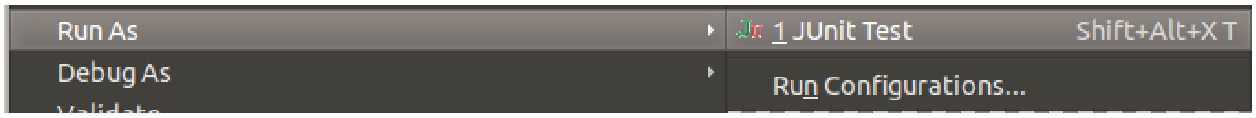
☐ Create final method stubs
☐ Create tasks for generated test methods

Create a test with the following code.

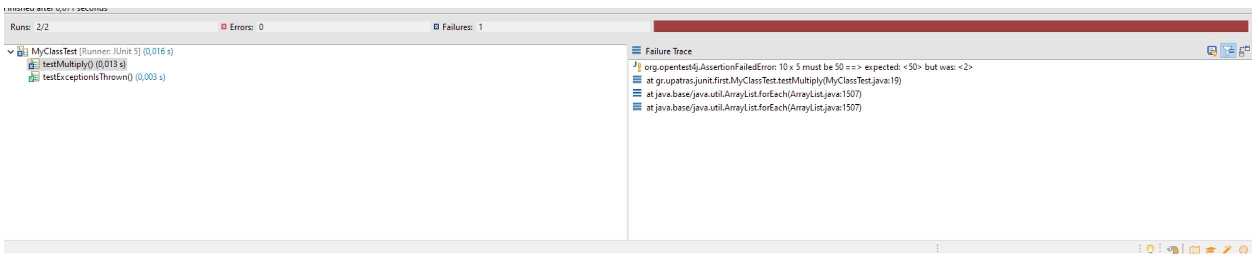
```
package gr.upatras.junit.first;

import static org.junit.jupiter.api.Assertions.assertAll;
import static org.junit.jupiter.api.Assertions.assertEquals;
import static org.junit.jupiter.api.Assertions.assertThrows;

import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

class MyClassTest {

    @Test
    void testExceptionIsThrown() {
        MyClass tester = new MyClass();
        assertThrows(IllegalArgumentException.class, () ->
tester.multiply(1000, 5));
    }

    @Test
    void testMultiply() {
        MyClass tester = new MyClass();
        assertEquals(50, tester.multiply(10, 5), "10 x 5 must be 50");
    }
}
```

5.6. Run your test in Eclipse

Right-click on your new test class and select **Run-As JUnit Test**.



The result of the tests are displayed in the JUnit view. In our example one test should be successful and one test should show an error. This error is indicated by a red bar.


Runs: 2/2 Errors: 0 Failures: 1

MyClassTest [Runner: JUnit 5] (0,016 s)
testMultiply() (0,013 s)
testExceptionIsThrown() (0,003 s)

Failure Trace

```
org.opentest4j.AssertionFailedError: 10 x 5 must be 50 ==> expected: <50> but was: <2>
at gr.upatras.junit.first.MyClassTest.testMultiply(MyClassTest.java:19)
at java.base/java.util.ArrayList.forEach(ArrayList.java:1507)
at java.base/java.util.ArrayList.forEach(ArrayList.java:1507)
```

You discovered a bug in the tested code!

5.7. Fix the bug and re-run your tests

The test is failing, because our multiplier class is currently not working correctly. It does a division instead of multiplication. Fix the bug and re-run the test to get a green bar.

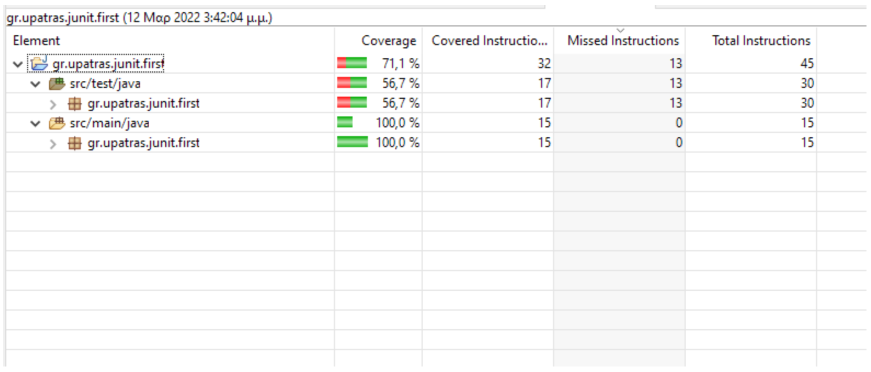
5.8. Review

After a few minutes you should have created a new project, a new class and a new unit test. Congratulations! If you feel like it, lets improve the tests a bit and write one grouped test.

5.9. Test Coverage

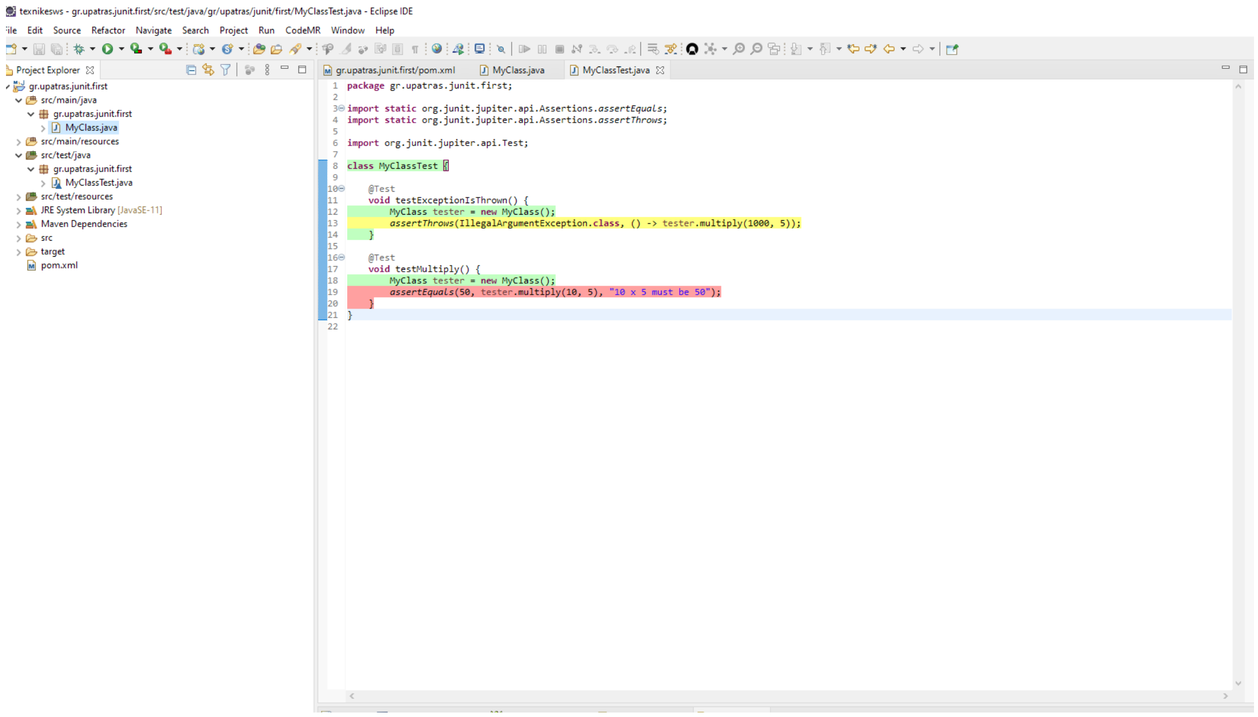
Right click on the project Name

Coverage As -> Junit test..



gr.upatras.junit.first (12 Μαρ 2022 3:42:04 μ.μ.)

Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
gr.upatras.junit.first	71,1 %	32	13	45
src/test/java	56,7 %	17	13	30
gr.upatras.junit.first	56,7 %	17	13	30
src/main/java	100,0 %	15	0	15
gr.upatras.junit.first	100,0 %	15	0	15



texnikesws - gr.upatras.junit.first/src/test/java/gr/upatras/junit/first/MyClassTest.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run CodeMR Window Help
Project Explorer
gr.upatras.junit.first/pom.xml
MyClass.java
MyClassTest.java

```
gr.upatras.junit.first
  src/main/java
    gr.upatras.junit.first
      MyClass.java
  src/main/resources
  src/test/java
    gr.upatras.junit.first
      MyClassTest.java
  src/test/resources
  JRE System Library [JavaSE-11]
  Maven Dependencies
  src
  target
  pom.xml
```

```java
package gr.upatras.junit.first;

import static org.junit.jupiter.api.Assertions.assertEquals;
import static org.junit.jupiter.api.Assertions.assertThrows;

import org.junit.jupiter.api.Test;

class MyClassTest {

    @Test
    void testExceptionIsThrown() {
        MyClass tester = new MyClass();
        assertThrows(IllegalArgumentException.class, () -> tester.multiply(1000, 5));
    }

    @Test
    void testMultiply() {
        MyClass tester = new MyClass();
        assertEquals(50, tester.multiply(10, 5), "10 x 5 must be 50");
    }
}
```