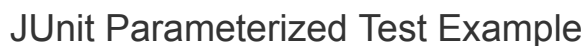
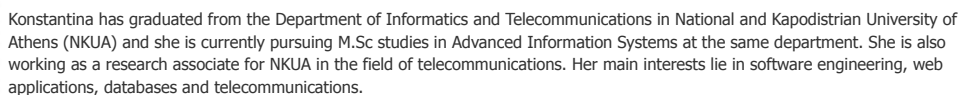




ABOUT KONSTANTINA DIMTSA



Posted by: Konstantina Dimtsa in iunit December 24th, 2013

1. Create the java class to be tested

Create a folder named

JUnitParameterized

. This is the folder where your classes will be located. Using a text editor, create a Java class named

Addition.java

. To make sure your file name is

Addition.java

, (not

Addition.java.txt

), first choose "Save as -> Save as type -> All files", then type in the file name

Addition.java

Want to be a JUnit Master ?

Subscribe to our newsletter and download the JUnit Programming Cookbook right now!

In order to help you master unit testing with JUnit, we have compiled a kick-ass guide with all the major JUnit features and use cases! Besides studying them online you may download the eBook in PDF format!

Email address:

Your email address

CARRER OPPORTUNITIES



Get the latest jobs to your ir

Enter your name

Enter your email

developer

Enter city, state or zip

Send me jobs!

Job Search



[Sign up](#)[Addition.java](#)

```

1 public class Addition {
2     public int addNumbers(int a, int b) {
3         int sum = a + b;
4         return sum;
5     }
6 }

```

2. Create JUnit test case

In the same directory (

JUnitParameterized

), use a text editor and create a java class named

JUnitAdditionTest.java

with the following code.

[JUnitAdditionTest.java](#)

```

01 import static org.junit.Assert.assertEquals;
02 import java.util.Arrays;
03 import java.util.Collection;
04
05 import org.junit.Test;
06 import org.junit.runner.RunWith;
07 import org.junit.runners.Parameterized;
08 import org.junit.runners.Parameterized.Parameters;
09
10 @RunWith(Parameterized.class)
11 public class JUnitAdditionTest {
12
13     private int expected;
14     private int first;
15     private int second;
16
17     public JUnitAdditionTest(int expectedResult, int firstNumber,
18                             int secondNumber) {
19         this.expected = expectedResult;
20         this.first = firstNumber;
21         this.second = secondNumber;
22     }
23
24     @Parameters
25     public static Collection<Integer[]> addedNumbers() {
26         return Arrays.asList(new Integer[][] { { 3, 1, 2 }, { 5, 2, 3 },
27                                                 { 7, 3, 4 }, { 9, 4, 5 }, });
28     }
29
30     @Test
31     public void sum() {
32         Addition add = new Addition();
33         System.out.println("Addition with parameters : " + first + " and "
34                             + second);
35         assertEquals(expected, add.AddNumbers(first, second));
36     }
37 }

```

A test class can be considered as parameterized test if the following requirements are fulfilled:

- **The class is annotated with**

@RunWith(Parameterized.class)

When a class is annotated with

@RunWith

, JUnit will invoke the class in which is annotated to run the tests, instead of using the runner built into JUnit.

Parameterized

is a runner inside JUnit that will run the same test case with different set of inputs.

- **The class has a single constructor that stores the test data.**
- **The class has a static method that generates and returns test data.**

The method that generates test data (in our case this method is

addedNumbers()

) must be annotated with

Job Search by

 ZipRecruiter



NEWSLETTER

182,134 insiders are already enjoying weekly updates and complimentary whitepapers!

Join them now to gain **EXCLUSIVE ACCESS** to the latest news in the Java world as well as insights about Android, Scala, Groovy and other related technologies.

Email address:

Your email address

☒ Receive Java & Developer job alerts in your Area from our partners over at ZipRecruiter

[Sign up](#)

JOIN US



With **1,240,600** unique visitors and **500** authors placed among related sites and Constantly being looked out for partner encourage you. So If you have unique and interesting content then you check out our **JCG** partners program. You be a **guest writer** for Java Code Geek your writing skills!

```
@Parameters
```

, and it must return a Collection of Arrays. Each array represents the data to be used in each test execution. The number of elements in each array must be the same with the number of parameters in constructor.

- **The class has a test.**

The test method is the method annotated with

```
@Test
```

annotation.

For further details regarding the

```
@Test
```

annotation, the

```
assertEquals
```

assertion (which are also mentioned in our code) and other JUnit Assertions and Annotations, you can have a look at JUnit using Assertions and Annotations Example.

3. Run your test case from the command line

You can run your JUnit test outside Eclipse, by using the

```
org.junit.runner.JUnitCore
```

class. This class provides the

```
runClasses()
```

method which allows you to execute one or several test classes. The return type of

```
runClasses()
```

method is an object of the type

```
org.junit.runner.Result
```

. This object can be used to collect information about the tests. Also, in case there is a failed test, you can use the object

```
org.junit.runner.notification.Failure
```

which holds description of the failed tests.

The procedure below shows how to run your test outside Eclipse.

In the directory

```
JUnitParameterized
```

, use a text editor and create a new Java class named

```
JUnitAdditionTestRunner.java
```

with the following code.

[JUnitAdditionTestRunner.java](#)

```
01 import org.junit.runner.JUnitCore;
02 import org.junit.runner.Result;
03 import org.junit.runner.notification.Failure;
04
05 public class JUnitAdditionTestRunner {
06
07     public static void main(String[] args) {
08
09         Result result = JUnitCore.runClasses(JUnitAdditionTest.class);
10         for (Failure fail : result.getFailures()) {
11             System.out.println(fail.toString());
12         }
13         if (result.wasSuccessful()) {
14             System.out.println("All tests finished successfully...");
15         }
16     }
17 }
```

- Open command prompt and move down directories so as to find the directory where your java classes are located:

```
1 C:\Users\konstantina>cd JUnitParameterized
```

Attention: If your classes are located inside a package, for example

```
package com.javacodegeeks.core.junit
```

, you can have a look at *JUnit Ignore Test Example*, where we describe exactly what you should do in that case.

- When

```
JUnitParameterized
```

is your current directory, compile all the classes in the directory

Attention: To run your JUnit tests outside Eclipse properly you need to add the needed JUnit library jars to the classpath of your program. You can find those library jars [here](#).

```
1 C:\Users\konstantina\JUnitParameterized>javac -classpath "C:\Users\konstantina\Downloads\junit-4.11.jar";"C:\Users\konstantina\Downloads\hamcrest-core-1.3.jar"; Addition.java
JUnitAdditionTest.java JUnitAdditionTestRunner.java
```

- Now run the

```
JUnitAdditionTestRunner
```

```
1 C:\Users\konstantina\JUnitParameterized>java -classpath "C:\Users\konstantina\Downloads\junit-4.11.jar";"C:\Users\konstantina\Downloads\hamcrest-core-1.3.jar"; JUnitAdditionTestRunner
```

• Output:

```
Addition with parameters : 1 and 2
Addition with parameters : 2 and 3
Addition with parameters : 3 and 4
Addition with parameters : 4 and 5
All tests finished successfully...
```

As we see in the output, the test case is executed four times, which is the provided number of inputs in the method annotated with

```
@Parameters
```

annotation.

Download the source code

This was an example of parameterized test in JUnit testing framework.

Download the source code of this example : [JUnitParameterized.zip](#)

Do you want to know how to develop your skillset to become a **Java Rockstar**?

Subscribe to our newsletter to start Rocking [right now!](#)

To get you started we give you our best selling eBooks for **FREE!**

1. JPA Mini Book
2. JVM Troubleshooting Guide
3. JUnit Tutorial for Unit Testing
4. Java Annotations Tutorial
5. Java Interview Questions
6. Spring Interview Questions
7. Android UI Design

and many more

Email address:

☒ Receive Java & Developer job alerts in your Area from our partners over at ZipRecruiter

[Sign up](#)

KNOWLEDGE BASE

[Courses](#)[News](#)[Resources](#)[Tutorials](#)[Whitepapers](#)

THE CODE GEEKS NETWORK

[.NET Code Geeks](#)[Java Code Geeks](#)[System Code Geeks](#)[Web Code Geeks](#)

HALL OF FAME

[Android Alert Dialog Example](#)[Android OnClickListener Example](#)[How to convert Character to String and a String to Character Array in Java](#)[Java Inheritance example](#)[Java write to File Example](#)[java.io.FileNotFoundException – How to solve File Not Found Exception](#)[java.lang.arrayindexoutofboundsexception – How to handle Array Index Out Of Bounds Exception](#)[java.lang.NoClassDefFoundError – How to solve No Class Def Found Error](#)[JSON Example With Jersey + Jackson](#)[Spring JdbcTemplate Example](#)

ABOUT JAVA CODE GEEKS

JCGs (Java Code Geeks) is an independent online community focused on creating the ultimate Java to Java developers resource center; targeted at the technical and non-technical team lead (senior developer), project manager and junior developer. JCGs serve the Java, SOA, Agile and Telecom communities with daily news and domain experts, articles, tutorials, reviews, announcements, code snippets and source projects.

DISCLAIMER

All trademarks and registered trademarks appearing on Java Code Geeks are the property of their respective owners. Java is a trademark or registered trademark of Oracle Corporation in the United States and other countries. Examples Java Code Geeks is not connected to Oracle Corporation and is not sponsored by Oracle Corporation.