



El futuro digital
es de todos

MinTIC

JUnit

Documentación y Configuración
JUNIT - Parte 3

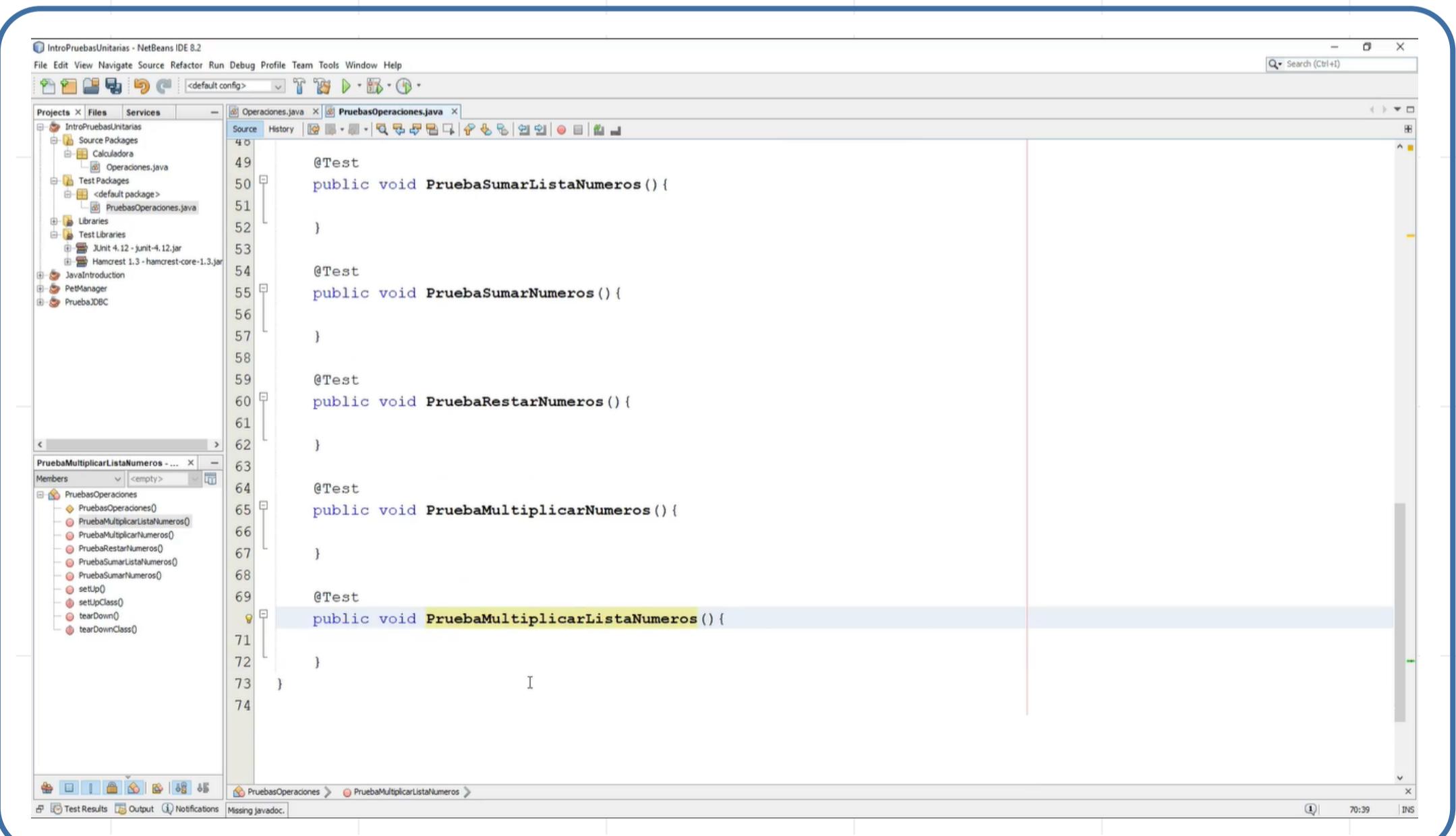


Universidad de Caldas

Hola:

En esta ocasión se concluirá el ejemplo de implementación de pruebas unitarias en JUNIT con base en las operaciones matemáticas que ya se desarrollaron. A continuación, se mostrará el desarrollo de las pruebas unitarias, cómo se verifica que un método haga lo que debe hacer y cómo se controlan y verifican las excepciones.

Video de explicación de JUNIT y todos sus componentes de manera práctica en pruebas de diferentes tipos.



The screenshot shows the NetBeans IDE interface with a blue rounded rectangle highlighting the central workspace. In the top menu bar, the path 'File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help' is visible. Below the menu is a toolbar with various icons. The main workspace contains two tabs: 'Operaciones.java' and 'PruebasOperaciones.java'. The 'Operaciones.java' tab shows a class definition with several methods. The 'PruebasOperaciones.java' tab shows a class definition with several test methods annotated with '@Test'. The bottom left corner of the workspace has a red vertical highlight. On the left side of the interface, there is a 'Projects' panel showing a project named 'IntroPruebasUnitarias' with packages like 'Calculadora' and 'Test Packages'. There is also a 'Libraries' panel listing 'JUnit 4.12 - junit-4.12.jar' and 'Hamcrest 1.3 - hamcrest-core-1.3.jar'. The bottom of the screen features a toolbar with icons for 'Test Results', 'Output', 'Notifications', and a status bar showing 'Missing Javadoc.', '70:39', and 'INS'.

```
40
41
42
43
44
45
46
47
48
49     @Test
50     public void PruebaSumarListaNumeros() {
51
52     }
53
54     @Test
55     public void PruebaSumarNumeros() {
56
57     }
58
59     @Test
60     public void PruebaRestarNumeros() {
61
62     }
63
64     @Test
65     public void PruebaMultiplicarNumeros() {
66
67     }
68
69     @Test
70     public void PruebaMultiplicarListaNumeros() {
71
72     }
73
74 }
```

IntroPruebasUnitarias - NetBeans IDE 8.2

```

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
<default config> T D G S P
Projects Files Services
Operaciones.java PruebasOperaciones.java
Source History < > Search (Ctrl+F)
35     public void setup() {
36         System.out.println("Inicio de método");
37     }
38
39     @After
40     public void tearDown() {
41         System.out.println("Fin de método");
42     }
43
44     // TODO add test methods here.
45     // The methods must be annotated with annotation @Test. For example:
46     //
47     // @Test
48     // public void hello() {}
49
50     @Test
51     public void PruebaSumarListaNumeros() {
52         long[] listaNumeros = new long[]{1, 2, 3, 4, 5};
53         Operaciones operaciones = new Operaciones();
54         assertEquals(15, operaciones.Sumar(listaNumeros));
55     }
56
57     /**
58      * @Test
59      */
60     public void PruebaSumarNumeros() {
61
62     }
63
64     @Test
65     public void PruebaRestarNumeros() {
66
67     }

```

PruebaSumarNumeros - Navigator

Members <empty>

- PruebasOperaciones
 - PruebaOperaciones()
 - PruebaDividirNumeros()
 - PruebaMultiplicarListaNumeros()
 - PruebaRestarNumeros()
 - PruebaSumarListaNumeros()
 - PruebaSumarNumeros()
 - setUp0
 - setUpClass()
 - tearDown0
 - tearDownClass()

PruebasOperaciones >

Test Results Output Notifications

PruebasOperaciones.java saved.

56:5 INS

IntroPruebasUnitarias - NetBeans IDE 8.2

```

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
<default config> T D G S P
Projects Files Services
Operaciones.java PruebasOperaciones.java
Source History < > Search (Ctrl+F)
43
44     // TODO add test methods here.
45     // The methods must be annotated with annotation @Test. For example:
46     //
47     // @Test
48     // public void hello() {}
49
50     @Test
51     public void PruebaSumarListaNumeros() {
52         long[] listaNumeros = new long[]{1, 2, 3, 4, 5};
53         Operaciones operaciones = new Operaciones();
54         assertEquals(15, operaciones.Sumar(listaNumeros));
55     }
56
57     /**
58      * @Test
59      */
60     public void PruebaSumarNumeros() {
61
62     }
63
64     @Test
65     public void PruebaRestarNumeros() {
66
67     }

```

PruebasOperaciones - Navigator

Members <empty>

- PruebasOperaciones
 - PruebaOperaciones()
 - PruebaSumarListaNumeros()
 - setUp0
 - setUpClass()
 - tearDown0
 - tearDownClass()

PruebasOperaciones >

Test Results X PruebasOperaciones X Calculadora.OperacionesTest
 Tests passed: 100,00 %
 The test passed. (0,055 s)

Método único antes de la clase

Inicio de método

Fin de método

Método al finalizar la clase

Output Notifications

80:8 INS

IntroPruebasUnitarias - NetBeans IDE 8.2

```

41     System.out.println("Fin de método");
42 }
43
44 // TODO add test methods here.
45 // The methods must be annotated with annotation @Test. For example:
46 //
47 // @Test
48 // public void hello() {}

49
50 @Test
51 public void PruebaSumarListaNumeros(){
52     long[] listaNumeros = new long[]{1, 2, 3, 4, 5, 6};
53     Operaciones operaciones = new Operaciones();
54     assertEquals(15, operaciones.Sumar(listaNumeros));
55 }
56 /*
57 @Test
58 public void PruebaSumarNumeros() {
59
}

```

PruebaSumarListaNumeros - Navigator

Members

- PruebasOperaciones
- PruebaSumarListaNumeros()
- setUp()
- setUpClass()
- tearDown()
- tearDownClass()

Test Results

Calculadora.OperacionesTest x PruebasOperaciones x

Tests passed: 0,00 %

No test passed, 1 test failed. (0,064 s)

PruebasOperaciones Failed

Método único antes de la clase
Inicio de método
Fin de método
Método al finalizar la clase

Output Notifications Finished building IntroPruebasUnitarias (test).

IntroPruebasUnitarias - NetBeans IDE 8.2

```

47 // TODO add test methods here.
48 // The methods must be annotated with annotation @Test. For example:
49 //
50 // @Test
51 // public void hello() {}
52
53 @Test
54 public void PruebaSumarListaNumeros() {
55     long[] listaNumeros = new long[]{1, 2, 3, 4, 5, 6};
56     assertEquals(21, operaciones.Sumar(listaNumeros));
57 }
58
59 @Test
60 public void PruebaSumarNumeros() {
61     long num1 = 3;
62     long num2 = 5;
63     assertEquals(8, operaciones.Sumar(num1, num2));
64 }
65

```

PruebaSumarNumeros - Navigator

Members

- PruebasOperaciones
- PruebaSumarListaNumeros()
- PruebaSumarNumeros()
- setUp()
- setUpClass()
- tearDown()
- tearDownClass()
- operaciones : Operaciones

Test Results

Calculadora.OperacionesTest x PruebasOperaciones x

Tests passed: 100,00 %

Both tests passed. (0,057 s)

Inicio de método
Fin de método
Inicio de método
Fin de método
Método al finalizar la clase

Output Notifications

Introducción a las Pruebas Unitarias con JUnit y Hamcrest en NetBeans

NetBeans IDE 8.2

Project: IntroPruebasUnitarias

Source Packages: Operaciones

Test Packages: PruebasOperaciones

Libraries: JUnit 4.12 - junit-4.12.jar, Hamcrest 1.3 - hamcrest-core-1.3.jar

Java Introduction, PetManager, PruebaJDBC

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Operaciones.java (selected)

```

47     // TODO add test methods here.
48     // The methods must be annotated with annotation @Test. For example:
49     //
50     // @Test
51     // public void hello() {}
52     @Test
53     public void PruebaSumarListaNumeros() {
54         long[] listaNumeros = new long[]{1, 2, 3, 4, 5, 6};
55         assertEquals(21, operaciones.Sumar(listaNumeros));
56     }
57
58     @Test
59     public void PruebaSumarNumeros() {
60         long num1 = 3;
61         long num2 = 5;
62         assertEquals(6, operaciones.Sumar(num1, num2));
63     }
64     /*
65     @Test

```

PruebasOperaciones - Navigator

- Members:
 - PruebasOperaciones
 - PruebaSumarListaNumeros()
 - PruebaSumarNumeros()
 - setUp()
 - setUpClass()
 - tearDown()
 - tearDownClass()
 - operaciones : Operaciones

Test Results

Calculadora.OperacionesTest x PruebasOperaciones

Tests passed: 50.00 %

1 test passed, 1 test failed, (0.07s)

- PruebasOperaciones Failed
 - PruebaSumarNumeros Failed: expected:<6> but was:<8>
 - junit.framework.AssertionFailedError at PruebasOperaciones.PruebaSumarNumeros

Inicio de método
Fin de método
Inicio de método
Fin de método
Método al finalizar la clase

Output Notifications

62:23 INS

Introducción a las Pruebas Unitarias con JUnit y Hamcrest en NetBeans

NetBeans IDE 8.2

Project: IntroPruebasUnitarias

Source Packages: Operaciones

Test Packages: PruebasOperaciones

Libraries: JUnit 4.12 - junit-4.12.jar, Hamcrest 1.3 - hamcrest-core-1.3.jar

Java Introduction, PetManager, PruebaJDBC

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Operaciones.java (selected)

```

80     public void PruebaMultiplicarListaNumeros() {
81         long[] listaNumeros = new long[]{1, 2, 3, 4, 5, 6};
82         assertEquals(720, operaciones.Multiplicar(listaNumeros));
83     }
84
85     @Test
86     public void PruebaDividirNumeros() {
87         long num1 = 10;
88         long num2 = 5;
89         assertEquals(2, operaciones.Dividir(num1, num2));
90     }
91
92     @Test
93     public void PruebaDividirNumerosException() {
94         long num1 = 10;
95         long num2 = 0;
96         assertEquals(ArithmetricException.class, operaciones.Dividir(num1, num2));
97     }
98 }

```

PruebasOperacionesException - Navigator

- Members:
 - PruebasOperaciones
 - PruebaDividirNumeros()
 - PruebaDividirNumerosException()
 - PruebaMultiplicarListaNumeros()
 - PruebaMultiplicarNumeros()
 - PruebaRestarNumeros()
 - PruebaSumarListaNumeros()
 - PruebaSumarNumeros()
 - setUp()
 - setUpClass()
 - tearDown()
 - tearDownClass()
 - operaciones : Operaciones

Test Results

Calculadora.OperacionesTest x PruebasOperaciones

Tests passed: 50.00 %

1 test passed, 1 test failed, (0.07s)

- PruebasOperaciones Failed
 - PruebaDividirNumerosException Failed: expected:<ArithmetricException> but was:<java.lang.ArithmetricException>
 - junit.framework.AssertionFailedError at PruebasOperaciones.PruebaDividirNumerosException

Output Notifications

96:26 INS

The screenshot shows the NetBeans IDE interface with the following details:

- Project:** IntroPruebasUnitarias - NetBeans IDE 8.2
- Source Editor:** PruebasOperaciones.java


```

80 public void PruebaMultiplicarListaNumeros(){
81     long[] listaNumeros = new long[]{1, 2, 3, 4, 5, 6};
82     assertEquals(720, operaciones.Multiplicar(listaNumeros));
83 }
84
85 @Test
86 public void PruebaDividirNumeros(){
87     long num1 = 10;
88     long num2 = 5;
89     assertEquals(2, operaciones.Dividir(num1, num2));
90 }
91
92 @Test
93 public void PruebaDividirNumerosException(){
94     long num1 = 10;
95     long num2 = 0;
96     assertEquals(ArithmeticException.class, operaciones.Dividir(num1, num2));
97 }
98 }
```
- Test Results:** Shows 5 tests passed, 1 test failed, and 1 test caused an error. The failed test is PruebaOperaciones.
- Output:** Displays log messages: "Inicio de método", "Fin de método", "Inicio de método", "Fin de método", and "Método al finalizar la clase".

The screenshot shows the NetBeans IDE interface with the following details:

- Project:** IntroPruebasUnitarias - NetBeans IDE 8.2
- Source Editor:** PruebasOperaciones.java


```

80 public void PruebaMultiplicarListaNumeros(){
81     long[] listaNumeros = new long[]{1, 2, 3, 4, 5, 6};
82     assertEquals(720, operaciones.Multiplicar(listaNumeros));
83 }
84
85 @Test
86 public void PruebaDividirNumeros(){
87     long num1 = 10;
88     long num2 = 5;
89     assertEquals(2, operaciones.Dividir(num1, num2), 0.000000001);
90 }
91
92 @Test
93 public void PruebaDividirNumerosException(){
94     long num1 = 10;
95     long num2 = 0;
96     assertEquals(ArithmeticException.class, operaciones.Dividir(num1, num2));
97 }
98 }
```
- Test Results:** Shows 6 tests passed, 1 test causes an error. The error is PruebaOperaciones.
- Output:** Displays log messages: "Inicio de método", "Fin de método", "Inicio de método", "Fin de método", and "Método al finalizar la clase".

The screenshot shows the NetBeans IDE interface with the following details:

- Project Explorer:** Shows the project "IntroPruebasUnitarias" with Source Packages (Operaciones.java) and Test Packages (PruebasOperaciones.java).
- Code Editor:** Displays Java test code in PruebasOperaciones.java. A specific line of code is highlighted with a red rectangle: `@Test (expected = ArithmeticException.class)`.

```
public void PruebaMultiplicarListaNumeros(){
    long[] listaNumeros = new long[]{1, 2, 3, 4, 5, 6};
    assertEquals(720, operaciones.Multiplicar(listaNumeros));
}

@Test
public void PruebaDividirNumeros(){
    long num1 = 10;
    long num2 = 5;
    assertEquals(2, operaciones.Dividir(num1, num2), 0.000000001);
}

@Test (expected = ArithmeticException.class)
public void PruebaDividirNumerosException(){
    long num1 = 10;
    long num2 = 0;
    assertEquals(ArithmeticException.class, operaciones.Dividir(num1, num2));
}
```
- Test Results:** Shows the execution results for the tests:
 - Calculadora.OperacionesTest x PruebasOperaciones x
 - Tests passed: 100,00 %
 - All 7 tests passed. (0,076 s)
 - Inicio de método
 - Fin de método
 - Inicio de método
 - Fin de método
 - Método al finalizar la clase
- Status Bar:** Shows the time as 92:48 and status as INS.

Se ha aprendido de manera práctica cómo definir y ejecutar las pruebas unitarias del código Java. Es importante poner en práctica este conocimiento durante el desarrollo del reto de la semana, además, de profundizar en esta temática con ayuda del material adicional.



Universidad de Caldas