



El futuro digital
es de todos

MinTIC

mockito JUnit

Ejemplo de Mockito con JUNIT

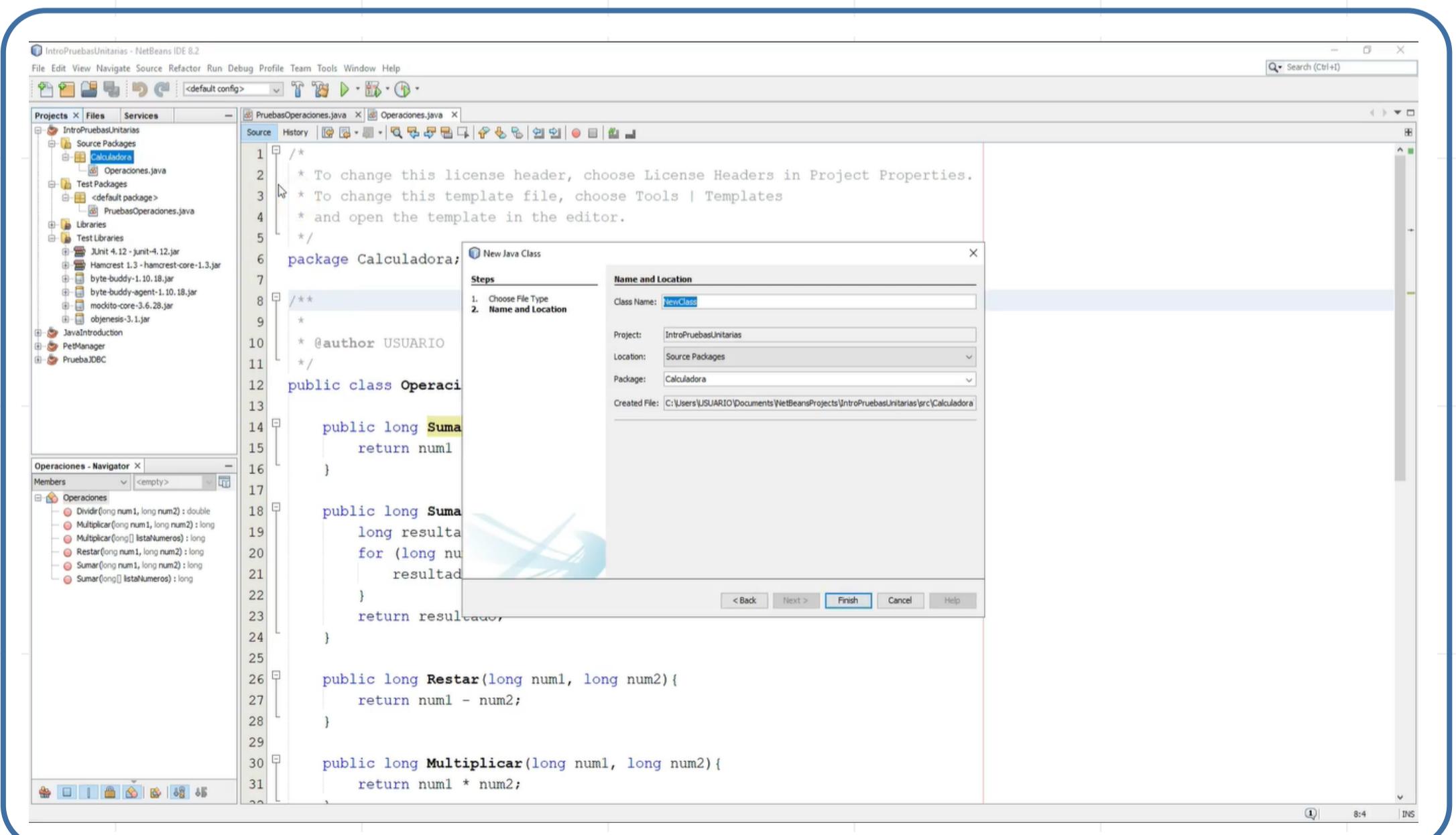


Universidad de Caldas

Hola:

Para continuar con el concepto de **calidad de software y pruebas unitarias** con la ayuda de las librerías Mockito y JUNIT se continuará incluyendo pruebas con base en las operaciones matemáticas que ya se desarrollaron. Para ello se construirá un nuevo método que requiere la inclusión de una instancia de una clase externa a la de operaciones y se creará un objeto Mock que simule su comportamiento para evitar depender directamente de la disponibilidad de dicha clase, además, se incluirá la reutilización de código de un método en otro.

Video de explicación de creación de objetos Mock y su funcionamiento con la integración a JUNIT.



IntroPruebasUnitarias - NetBeans IDE 8.2

```

public long Multiplicar(long num1, long num2){
    return num1 * num2;
}

public long Multiplicar(long[] listaNumeros){
    long resultado = 1;
    for (long num : listaNumeros) {
        resultado *= num;
    }
    return resultado;
}

public double Dividir(long num1, long num2){
    try{
        return num1 / num2;
    }catch(ArithmeticsException e){
        throw e;
    }
}

public double Promedio(){
    long[] numeros = new GeneradorDeNumeros().ListaNumeros;
    double promedio;
    long sumatoria = 0;
    for (long numero : numeros) {
        sumatoria += numero;
    }
    promedio = this.Dividir(sumatoria, numeros.length);
    return promedio;
}

```

IntroPruebasUnitarias - NetBeans IDE 8.2

```

long num2 = 5;
assertEquals(2, operaciones.Dividir(num1, num2), 0.000000001);
}

@Test (expected = ArithmeticsException.class)
public void PruebaDividirNumerosException() {
    long num1 = 10;
    long num2 = 0;
    assertEquals(ArithmeticsException.class, operaciones.Dividir(num1, num2));
}

@Test
public void PruebaPromedioNumerosVacios() {
}

@Test
public void PruebaPromedioConNumeros() {
}

```

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

- Title Bar:** IntroPruebasUnitarias - NetBeans IDE 8.2
- Menu Bar:** File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
- Toolbar:** Standard NetBeans toolbar with icons for file operations.
- Project Explorer:** Shows the project structure under "IntroPruebasUnitarias".
- Source Editor:** Displays the `PruebasOperaciones.java` file with the following code:

```
97 @Test (expected = ArithmeticException.class)
98 public void PruebaDividirNumerosException() {
99     long num1 = 10;
100    long num2 = 0;
101    assertEquals(ArithmeticException.class, operaciones.Dividir(num1, num2));
102 }
103
104 @Test
105 public void PruebaPromedioNumerosVacios() {
106     Mockito.when(this.generador.ListaNumeros).thenReturn(new long[] {});
107 }
108
109 @Test
110 public void PruebaPromedioConNumeros() {
111 }
112
113 }
114 }
```

- Members View:** Shows the members of the `PruebasOperaciones` class.
- Bottom Status Bar:** Shows the status bar with 107:73 and INS.

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

- Title Bar:** IntroPruebasUnitarias - NetBeans IDE 8.2
- Menu Bar:** File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
- Toolbar:** Standard NetBeans toolbar with icons for file operations.
- Project Explorer:** Shows the project structure under "IntroPruebasUnitarias".
- Source Editor:** Displays the `PruebasOperaciones.java` file with the following code:

```
97 @Test (expected = ArithmeticException.class)
98 public void PruebaDividirNumerosException() {
99     long num1 = 10;
100    long num2 = 0;
101    assertEquals(ArithmeticException.class, operaciones.Dividir(num1, num2));
102 }
103
104 @Test(expected = ArithmeticException.class)
105 public void PruebaPromedioNumerosVacios() {
106     Mockito.when(this.generador.ListaNumeros).thenReturn(new long[] {});
107     assertEquals(ArithmeticException.class, this.operaciones.Promedio(this.generador.ListaNumeros));
108 }
109
110 @Test
111 public void PruebaPromedioConNumeros() {
112 }
113
114 }
```

- Members View:** Shows the members of the `PruebasOperaciones` class.
- Bottom Status Bar:** Shows the status bar with 108:95 and INS.

The screenshot shows the NetBeans IDE interface with the project "IntroPruebasUnitarias" open. The code editor displays the class `GeneradorDeNumeros` with its constructor implementation:

```
1  /*
2  * To change this license header, choose License Headers in Project Properties.
3  * To change this template file, choose Tools | Templates
4  * and open the template in the editor.
5  */
6 package Calculadora;
7
8 /**
9 *
10 * @author USUARIO
11 */
12 public class GeneradorDeNumeros {
13     private long[] listaNumeros;
14
15     public long[] ListaNumeros() {
16         return this.listaNumeros;
17     }
18 }
```

The Navigator pane on the left shows the members of the `GeneradorDeNumeros` class, including the constructor `ListaNumeros()` and the field `listaNumeros`. The status bar at the bottom indicates "Missing javadoc."

The screenshot shows the NetBeans IDE interface with the project "IntroPruebasUnitarias" open. The code editor displays the class `PruebasOperaciones` with its constructor implementation:

```
18 import org.mockito.Mockito;
19 import org.mockito.junit.MockitoJUnitRunner;
20
21 /**
22 *
23 * @author USUARIO
24 */
25 @RunWith(MockitoJUnitRunner.class)
26 public class PruebasOperaciones {
27     @InjectMocks
28     Operaciones operaciones;
29
30     @Mock
31     GeneradorDeNumeros generador;
32
33     public PruebasOperaciones() {
34         operaciones = new Operaciones();
35     }
36 }
```

The Navigator pane on the left shows the members of the `PruebasOperaciones` class, including the constructor `PruebasOperaciones()` and the fields `operaciones` and `generador`. The status bar at the bottom indicates "PruebasOperaciones.java saved."

IntroPruebasUnitarias - NetBeans IDE 8.2

```

102     @Test (expected = ArithmeticException.class)
103     public void PruebaDividirNumerosException() {
104         long num1 = 10;
105         long num2 = 0;
106         assertEquals(ArithmeticException.class, operaciones.Dividir(num1, num2));
107     }
108
109     @Test (expected = ArithmeticException.class)
110     public void PruebaPromedioNumerosVacios() {
111         Mockito.when(this.generador.ListaNumeros()).thenReturn(new long[]{});
112         assertEquals(ArithmeticException.class, this.operaciones.Promedio(this.generador.ListaNumeros()));
113     }
114
115     @Test
116     public void PruebaPromedioConNumeros() {
117
118     }
119 }
```

Test Results x

All 9 tests passed. (0,475 s)

- PruebasOperaciones passed
 - PruebaSumarNumeros passed (0,387 s)
 - PruebaMultiplicarNumeros passed (0,0 s)
 - PruebaDividirNumeros passed (0,0 s)
 - PruebaRestarNumeros passed (0,013 s)
 - PruebaPromedioConNumeros passed (0,001 s)
 - PruebaMultiplicarListaNumeros passed (0,001 s)
 - PruebaSumarListaNumeros passed (0,001 s)
 - PruebaDividirNumerosException passed (0,001 s)

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

IntroPruebasUnitarias - NetBeans IDE 8.2

```

105     long num2 = 0;
106     assertEquals(ArithmeticException.class, operaciones.Dividir(num1, num2));
107 }
108
109     @Test (expected = ArithmeticException.class)
110     public void PruebaPromedioNumerosVacios() {
111         Mockito.when(this.generador.ListaNumeros()).thenReturn(new long[]{});
112         assertEquals(ArithmeticException.class, this.operaciones.Promedio(this.generador.ListaNumeros()));
113 }
114
115     @Test
116     public void PruebaPromedioConNumeros() {
117         Mockito.when(this.generador.ListaNumeros()).thenReturn(new long[]{1, 2, 3, 4, 5});
118         assertEquals(3.0, this.operaciones.Promedio(this.generador.ListaNumeros()), .01);
119 }
120 }
```

Test Results x

All 9 tests passed. (0,504 s)

- PruebasOperaciones passed
 - PruebaSumarNumeros passed (0,409 s)
 - PruebaMultiplicarNumeros passed (0,001 s)
 - PruebaDividirNumeros passed (0,0 s)
 - PruebaRestarNumeros passed (0,0 s)
 - PruebaPromedioNumerosVacios passed (0,014 s)
 - PruebaPromedioConNumeros passed (0,001 s)
 - PruebaMultiplicarListaNumeros passed (0,0 s)
 - PruebaSumarListaNumeros passed (0,001 s)
 - PruebaDividirNumerosException passed (0,0 s)

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

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

- Project Explorer:** Shows the project structure for "IntroPruebasUnitarias".
- Code Editor:** Displays the file "PruebasOperaciones.java" containing Java code for unit tests. A specific line of code is highlighted: `assertEquals(4.0, this.operaciones.Promedio(this.generador.ListaNumeros()), .01);`. The value `4.0` is highlighted with a red box.
- Test Results:** Shows the output of the test run. It indicates 8 tests passed and 1 test failed. The failed test is "PruebaPromedioConNumeros" with the message "Failed: expected:<4.0> but was:<3.0>". Other tests listed include PruebaSumarNumeros, PruebaMultiplicarNumeros, PruebaDividirNumeros, PruebaRestarNumeros, PruebaPromedioNumerosVacios, PruebaMultiplicarListaNumeros, PruebaDividirListaNumeros, and PruebaSumarListaNumeros.
- Output:** Shows the message "Finished building IntroPruebasUnitarias (test.)".

Una vez con el conocimiento para incluir pruebas unitarias en los diferentes proyectos de software que deban desarrollar, es necesario tener en cuenta que esto es un componente que ayuda a que la calidad del software sea de un nivel mayor.



Universidad de Caldas