

# Pruebas de programas Java mediante JUnit

## ENTORNOS DE DESARROLLO

Marcos Fernández Sellers



# El framework JUnit

---

- JUnit es un *framework* para automatizar las pruebas de programas Java
- Escrito por Erich Gamma y Kent Beck
- Open Source, disponible en <http://www.junit.org>
- Adecuado para el Desarrollo dirigido por las pruebas (*Test-driven development*)

# El framework JUnit

---

- Consta de un conjunto de clases que el programador puede utilizar para construir sus casos de prueba y ejecutarlos automáticamente
- Los casos de prueba son realmente programas Java. Quedan archivados y pueden ser reejecutados tantas veces como sea necesario

# Un ejemplo sencillo

---

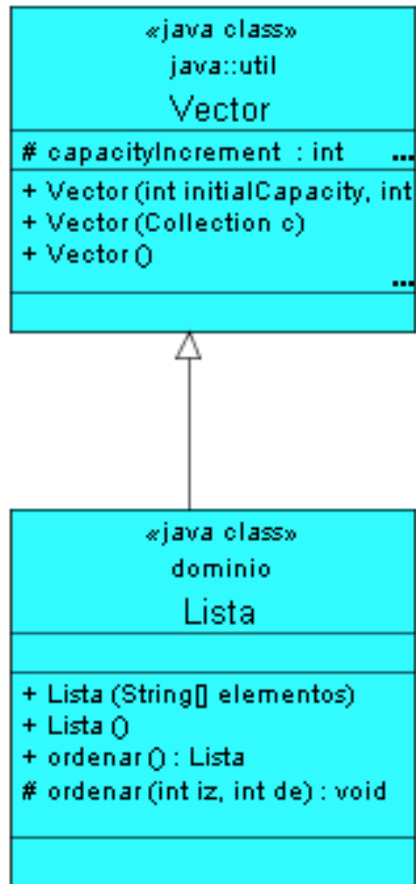
```
package dominio;  
import java.util.Vector;
```

```
public class Lista extends Vector {  
    public Lista() { ... }  
    public Lista(String[] elementos) {...}  
    public Lista ordenar() {...}  
    protected void ordenar(int iz, int de) {  
        ...  
    }  
}
```

← Representa una lista ordenable de forma creciente.

Se ordena llamando al método público *ordenar()*, que llama a su vez a *ordenar(0, size()-1)*

# Un ejemplo sencillo



Un posible caso de prueba es el siguiente:

```
String[] e3={"e", "d", "c", "b", "a"};
Lista revs=new Lista(e3);
Lista derecha=revs.ordenar();
```

...y el resultado esperado:

"a", "b", "c", "d", "e"

# Un ejemplo sencillo

---

```
String[] e3={"e", "d", "c", "b", "a"};  
Lista reves=new Lista(e3);  
Lista derecha=reves.ordenar();
```

Si *derecha* es igual al resultado esperado, entonces el caso de prueba ha sido superado:

```
{"a", "b", "c", "d", "e"}
```

# Un ejemplo sencillo

---

Construyamos manualmente un objeto *expected* y comparémoslo con el obtenido:

```
String[] e3={"e", "d", "c", "b", "a"};
Lista reves=new Lista(e3);
Lista derecha=reves.ordenar();
Lista expected={"a", "b", "c", "d", "e"};
if (derecha.equals(expected))
    ResultadoCorrecto();
else
    ResultadoIncorrecto();
```

# El framework JUnit (II)

---

El ejemplo anterior (*obtained* frente a *expected*) es una idea fundamental de Junit.

Ocurre que:

- JUnit nos va a permitir mantener de forma separada los casos de prueba
- JUnit permite ejecutarlos (y reejecutarlos) de forma automática
- Nos permite construir “árboles de casos de prueba” (*suites*)



# El framework JUnit (II)

---

Para el ejemplo anterior:

```
public void testOrdenarReves() {  
    String[] ex={"a", "b", "c", "d", "e"};  
    Lista expected=new Lista(ex);  
}
```

String **Construcción manual del objeto esperado**  
listaA

```
    this.assertEquals(expected, listaAlReves.ordenar());  
}
```

# El framework JUnit (II)

---

Para el ejemplo anterior:

```
public void testOrdenarReves() {  
    String[] ex={"a", "b", "c", "d", "e"};  
    Lista expected=new Lista(ex);
```

```
String[] e3={"e", "d", "c", "b", "a"};  
listaAlReves=new Lista(e3);
```

**Construcción manual del objeto obtenido haciendo uso de los métodos de la clase que estamos probando**

```
    this.a  
}
```

# El framework JUnit (II)

---

Para el ejemplo anterior:

```
public void testOrdenarReves() {  
    String[] ex={"a", "b", "c", "d", "e"};  
    Lista expected=new Lista(ex);
```

```
    String[] e3={"e", "d", "c", "b", "a"};
```

```
    lista
```

**Comparación de ambos objetos haciendo uso de las funcionalidades suministradas por JUnit**

```
        this.assertEquals(expected, listaAlReves.ordenar())
```

```
    }
```

# Clases fundamentales: Assert

## Assert

#Assert() : Assert

+assertTrue(message:String, in condition:boolean)

+assertTrue(in condition:boolean)

+assertFalse(message:String, in condition:boolean)

+assertFalse(in condition:boolean)

+fail(message:String)

+fail()

+assertEquals(message:String, expected:Object, actual:Object)

+assertEquals(expected:Object, actual:Object)

+assertEquals(message:String, expected:String, actual:String)

+assertEquals(expected:String, actual:String)

+assertEquals(message:String, in expected:double, in actual:double, in delta:double)

+assertEquals(in expected:double, in actual:double, in delta:double)

+assertEquals(message:String, in expected:float, in actual:float, in delta:float)

+assertEquals(in expected:float, in actual:float, in delta:float)

+assertEquals(message:String, in expected:long, in actual:long)

+assertEquals(in expected:long, in actual:long)

+assertEquals(message:String, in expected:boolean, in actual:boolean)

+assertEquals(in expected:boolean, in actual:boolean)

+assertEquals(message:String, in expected:byte, in actual:byte)

+assertEquals(in expected:byte, in actual:byte)

+assertEquals(message:String, in expected:char, in actual:char)

+assertEquals(in expected:char, in actual:char)

+assertEquals(message:String, expected:short, actual:short)

+assertEquals(expected:short, actual:short)

+assertEquals(message:String, in expected:int, in actual:int)

+assertEquals(in expected:int, in actual:int)

+assertNotNull(object:Object)

+assertNotNull(message:String, object:Object)

+assertNull(object:Object)

+assertNull(message:String, object:Object)

+assertSame(message:String, expected:Object, actual:Object)

+assertSame(expected:Object, actual:Object)

+assertNotSame(message:String, expected:Object, actual:Object)

+assertNotSame(expected:Object, actual:Object)

-failSame(message:String)

-failNotSame(message:String, expected:Object, actual:Object)

-failNotEquals(message:String, expected:Object, actual:Object)

~format(message:String, expected:Object, actual:Object) : String

# Anotaciones

---

Junit está basado en anotaciones para añadir metadatos al código

- `@Test` → Indica que el siguiente método es un Test
- `@Before` → Ejecuta el siguiente método antes de cada test
- `@After` → Ejecuta el siguiente método después de cada test

# Anotaciones

---

Junit está basado en anotaciones para añadir metadatos al código

- `@BeforeClass` → Ejecuta el siguiente método al inicio de la clase test
- `@AfterClass` → Ejecuta el siguiente método al final de la clase test

# Creación de Clases Test

---

Botón derecho sobre la clase Java → Tools →  
Create/Update Tests

NOTA: JUnit 5 no funciona para proyectos Java ANT. Los ejemplos realizados en clase se elaborarán con JUnit 4

# Creación de Clases Test

Create/Update Tests

Class to Test: ejemplopruebas.Calculadora

Class Name:

Location:

Framework:

☐ Integration Tests

The existing test class will be updated.

Code Generation

Method Access Levels	Generated Code
<input checked="" type="checkbox"/> Public	<input checked="" type="checkbox"/> Test Initializer
<input checked="" type="checkbox"/> Protected	<input checked="" type="checkbox"/> Test Finalizer
<input checked="" type="checkbox"/> Package Private	<input checked="" type="checkbox"/> Test Class Initializer
	<input checked="" type="checkbox"/> Test Class Finalizer
	<input checked="" type="checkbox"/> Default Method Bodies

Generated Comments

☒ Javadoc Comments

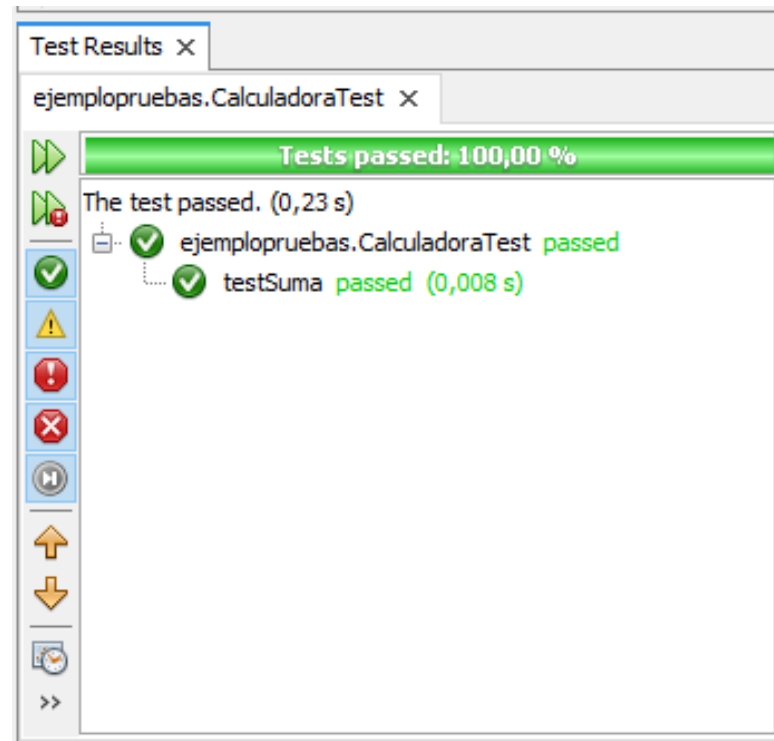
☒ Source Code Hint

OK Cancel Help



# Visualización de resultados

---



# Pruebas de excepciones (*fail*)

---

- Igual que es necesario comprobar cómo se comporta el programa en situaciones idóneas, es también importante probarlo en situaciones en que se producen errores.
- Es decir, que a veces el comportamiento correcto de nuestro programa consisten en se produzca un error

# Pruebas de excepciones (*fail*)

---

Podemos desear que *ordenar()* dé un error cuando la lista esté vacía:

```
public Lista ordenar() throws Exception {  
    if (size()==0)  
        throw new Exception("No se puede ordenar una lista vacía");  
    ordenar(0, size()-1);  
    return this;  
}
```

# Pruebas de excepciones (*fail*)

---

Modificamos los dos métodos *test*

```
public void testOrdenarNula2() throws Exception {  
    try  
    {  
        String[] ex=null;  
        Lista expected=new Lista(ex);  
        this.assertEquals(expected, listaNula2.ordenar());  
        fail("Debería haberse lanzado una excepción");  
    }  
    catch (Exception e)  
    {  
        // Capturamos la excepción para que el caso no falle  
    }  
}
```

# Clases de prueba abstractas

---

Se pueden posponer las pruebas hasta que se tengan especializaciones concretas de la clase abstracta

Pero también puede construirse una clase de Test abstracta

# Referencia JUnit

<https://junit.org>

 JUnit 5



The 5th major version of the programmer-friendly testing framework for Java and the JVM

[User Guide](#) [Javadoc](#) [Code & Issues](#) [Q & A](#) [Support JUnit](#)

### About

JUnit 5 is the next generation of JUnit. The goal is to create an up-to-date foundation for developer-side testing on the JVM. This includes focusing on Java 8 and above, as well as enabling many different styles of testing.

JUnit 5 is the result of JUnit Lambda and its [crowdfunding campaign on Indiegogo](#).

### Resources

You're invited to follow our ongoing work, review it, and give feedback. This short list of links will get you started:

- [User Guide](#)
- [Javadoc](#)
- [GitHub Repository](#) with all the code and issues

### Sponsoring

We ask you – our users – to [support us](#) so we can keep up the pace. We will continue our work on JUnit regardless of how many donations we receive. However, your support would enable us to do so with greater focus and not only on weekends or in our spare time. For example, we want to meet regularly and work colocated for a few days in order to get things done faster in face-to-face design and coding sessions. Your donations will help to make that a reality!

### Latest Release

[JUnit v5.8.2](#) [JUnit v5.8.2](#) [JUnit v5.8.2](#)

JUnit artifacts are deployed to Maven Central and can be downloaded using the above links. All files are signed using the keys listed in the [KEYS](#) file.

### Upcoming Events

There are no upcoming events at the moment.

[+ Add your talk](#)

### Thank You

In addition to our sponsors, we'd like to thank the following companies.

  
The JUnit team uses

  
The JUnit team relies on



# Conclusiones

---

- Marco de pruebas semiautomático
- Automatiza las pruebas de regresión
- Los casos de prueba documentan el propio código fuente
- Adecuado para Desarrollo dirigido por las pruebas (TDD, Test-Driven Development)