



# *Módulo 4: Proceso y herramientas de pruebas*

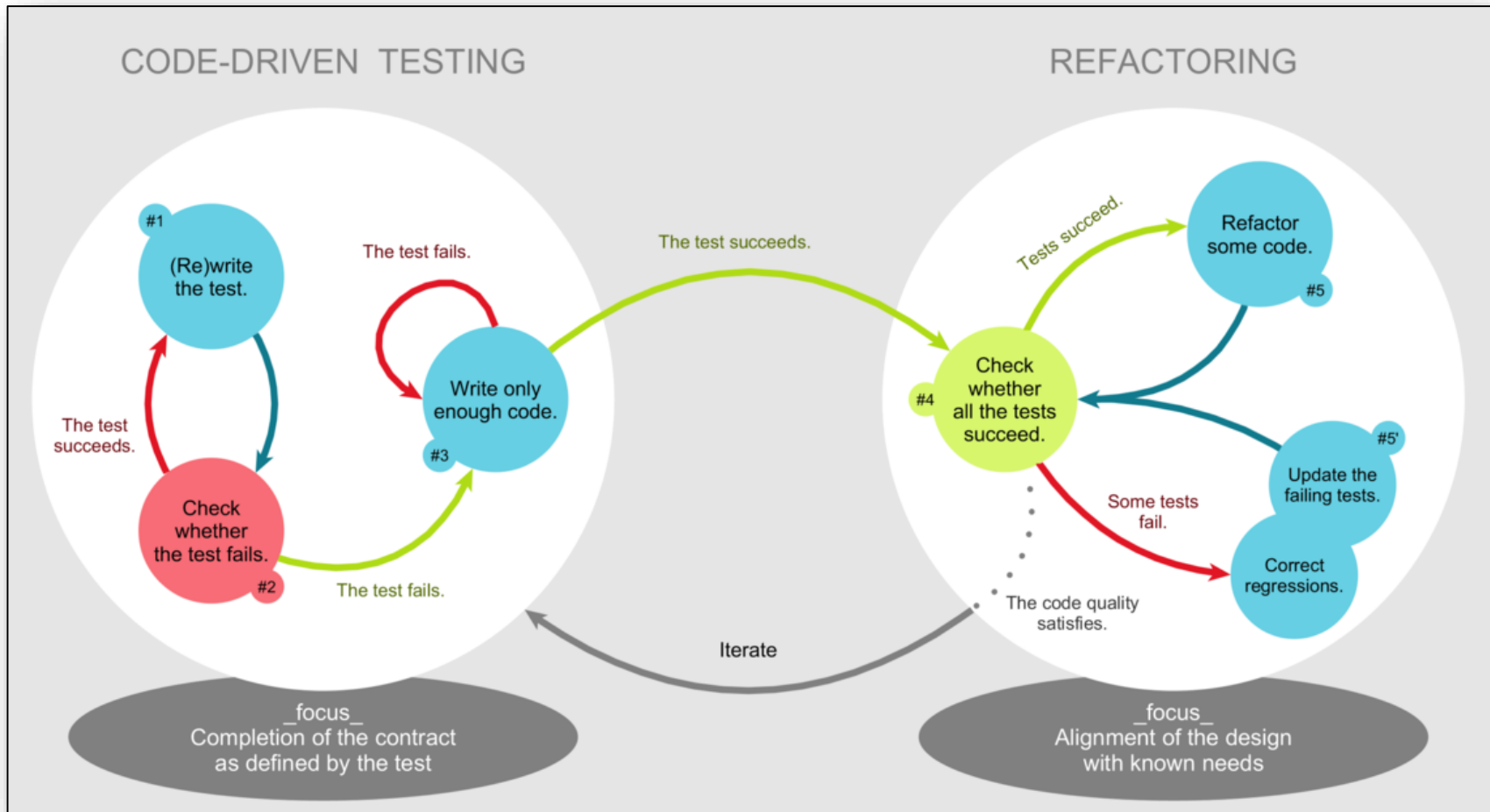
Pruebas Unitarias  
de las clases con ***xUnit***

## ¿Qué es xUnit?

---

- Una serie de librerías para realizar pruebas unitarias automatizadas de las clases
- Nace en 1998 en Smalltalk
  - Se populariza con TDD (Test Driven Development)
  - La “X” se reemplaza con el nombre del lenguaje (SUnit=Smalltalk, JUnit=Java, etc.)
- Ver: <https://en.wikipedia.org/wiki/XUnit>

# Test Driven Development



By Xarawn - Own work, CC BY-SA 4.0, <https://commons.wikimedia.org/w/index.php?curid=44782343>

# Assertions (afirmaciones)

---

- Es una función que verifica el comportamiento (o el estado) de la unidad bajo prueba
- Expresa una condición lógica que se espera que sea verdadera
  - Si falla, se marca error y se aborta la ejecución de la prueba actual

- Ejemplo:

`assertEquals(0, miLista.ValorActual())`

---

# Ejemplo

JUnit con Eclipse

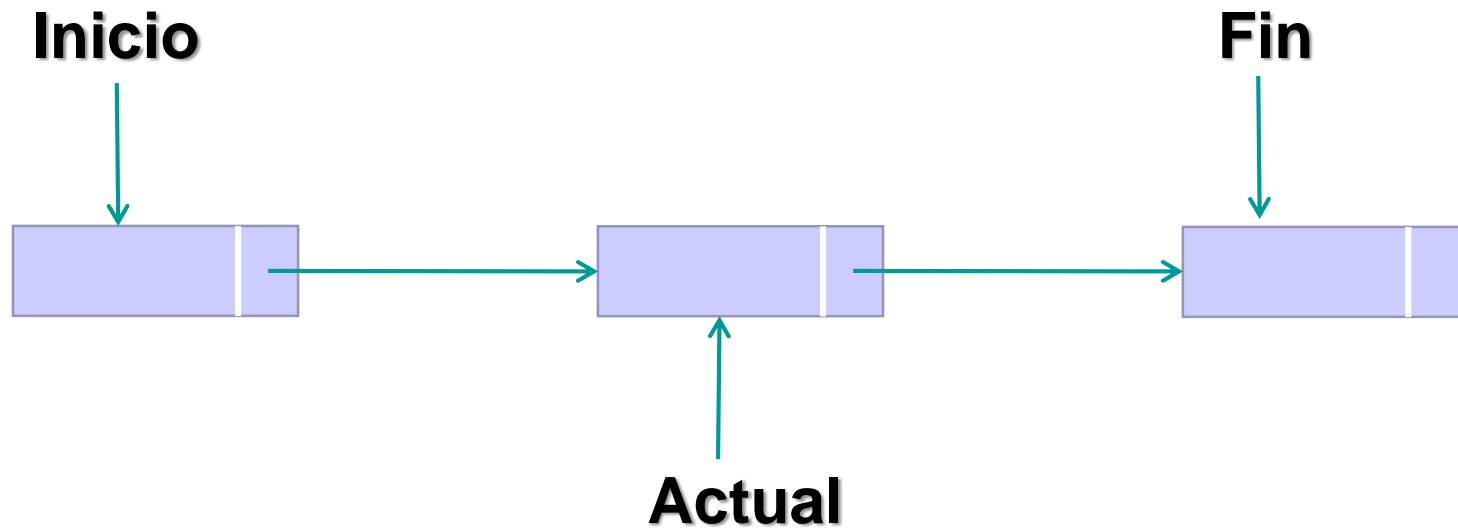
# Requerimiento

---

- Desarrollar una clase que permita almacenar datos reales en una lista encadenada
- Sus métodos deben permitir:
  - Agregar nuevos nodos
  - Recorrer la lista
  - Calcular el promedio

# Diseño: esquema

---



# Diseño: atributos y métodos

Atributos	
Declaración	Descripción
<b>inicio:</b> apuntador a Nodo	Apunta al primer nodo de la lista, si lista vacía ← NULO
<b>fin:</b> apuntador a Nodo	Apunta al último nodo de la lista, si lista vacía ← NULO
<b>actual:</b> apuntador a Nodo	Apunta al nodo en que se está trabajando, si lista vacía ← NULO

Métodos	
Declaración	Descripción
void AgregaFin(dato: real)	Agrega al final de la lista el valor contenido en "dato". "actual" apunta al valor insertado.
void AgregaInicio(dato: real)	Agrega al inicio de la lista el valor contenido en "dato". "actual" apunta al valor insertado.
void IrInicio()	Mueve "actual" al primer nodo de la lista. Si la lista está vacía... actual ← NULO
booleano Siguiente()	Mueve "actual" al siguiente valor en la lista, si puede. Regresa TRUE si lo pudo mover, FALSE si no pudo.
real ValorActual()	Regresa el valor al que apunta "actual". Si "actual" es NULO, regresa 0 (cero).
booleano Vacio()	Regresa TRUE si la lista está vacía, si no regresa FALSE
real Promedio()	Regresa el promedio de los datos almacenados en la lista. Si la lista está vacía regresa 0 (cero).



# Diseño: Casos de Prueba

---

- Lista vacía
- Lista con 1 dato
- Lista con 2 datos
- Lista con varios datos

# Diseño: Casos de Prueba (1/2)

Acción	Resultado esperado
Crear una lista nueva y vacía	<ul style="list-style-type: none"> <li>•Vacio() debe regresar true</li> <li>•ValorActual() debe regresar 0 (cero)</li> </ul>
Ejecutar IrInicio()	<ul style="list-style-type: none"> <li>•Vacio() debe regresar true</li> <li>•ValorActual() debe regresar 0 (cero)</li> </ul>
Ejecutar Siguiente()	<ul style="list-style-type: none"> <li>•Debe regresar falso</li> <li>•Vacio() debe regresar true</li> <li>•ValorActual() debe regresar 0 (cero)</li> </ul>
Ejecutar AgregaFin(8)	<ul style="list-style-type: none"> <li>•Promedio() debe regresar 0 (cero)</li> <li>•Vacio() debe regresar falso</li> <li>•ValorActual() debe regresar 8.0</li> </ul>
Ejecutar IrInicio()	<ul style="list-style-type: none"> <li>•Vacio() debe regresar falso</li> <li>•ValorActual() debe regresar 8.0</li> </ul>
Ejecutar Siguiente()	<ul style="list-style-type: none"> <li>•Debe regresar falso</li> <li>•Vacio() debe regresar falso</li> <li>•ValorActual() debe regresar 8.0</li> </ul>
Ejecutar AgregaInicio(7)	<ul style="list-style-type: none"> <li>•Promedio() debe regresar 8.0</li> <li>•Vacio() debe regresar falso</li> <li>•ValorActual() debe regresar 7.0</li> </ul>

## Diseño: Casos de Prueba (2/2)

Acción	Resultado esperado
Ejecutar Siguiente()	<ul style="list-style-type: none"><li>• Debe regresar true</li><li>• Vacio() debe regresar falso</li><li>• ValorActual() debe regresar 8.0</li></ul>
Ejecutar IrInicio()	<ul style="list-style-type: none"><li>• Vacio() debe regresar falso</li><li>• ValorActual() debe regresar 7.0</li><li>• Promedio() debe regresar 7.5</li></ul>
Ejecutar: <ul style="list-style-type: none"><li>• AgregaInicio(9)</li><li>• AgregaFin(1)</li><li>• AgregaFin(3)</li></ul>	<ul style="list-style-type: none"><li>• Vacio() debe regresar falso</li><li>• ValorActual() debe regresar 3.0</li><li>• Promedio() debe regresar 5.6</li><li>• La lista debe contener 9, 7, 8, 1, 3</li></ul>

---

# Código

---

# Demo Eclipse

Java - Demo Clase CPS/test/mx/itesm/demo/cps/ListaTest.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer Type Hierarchy JUnit

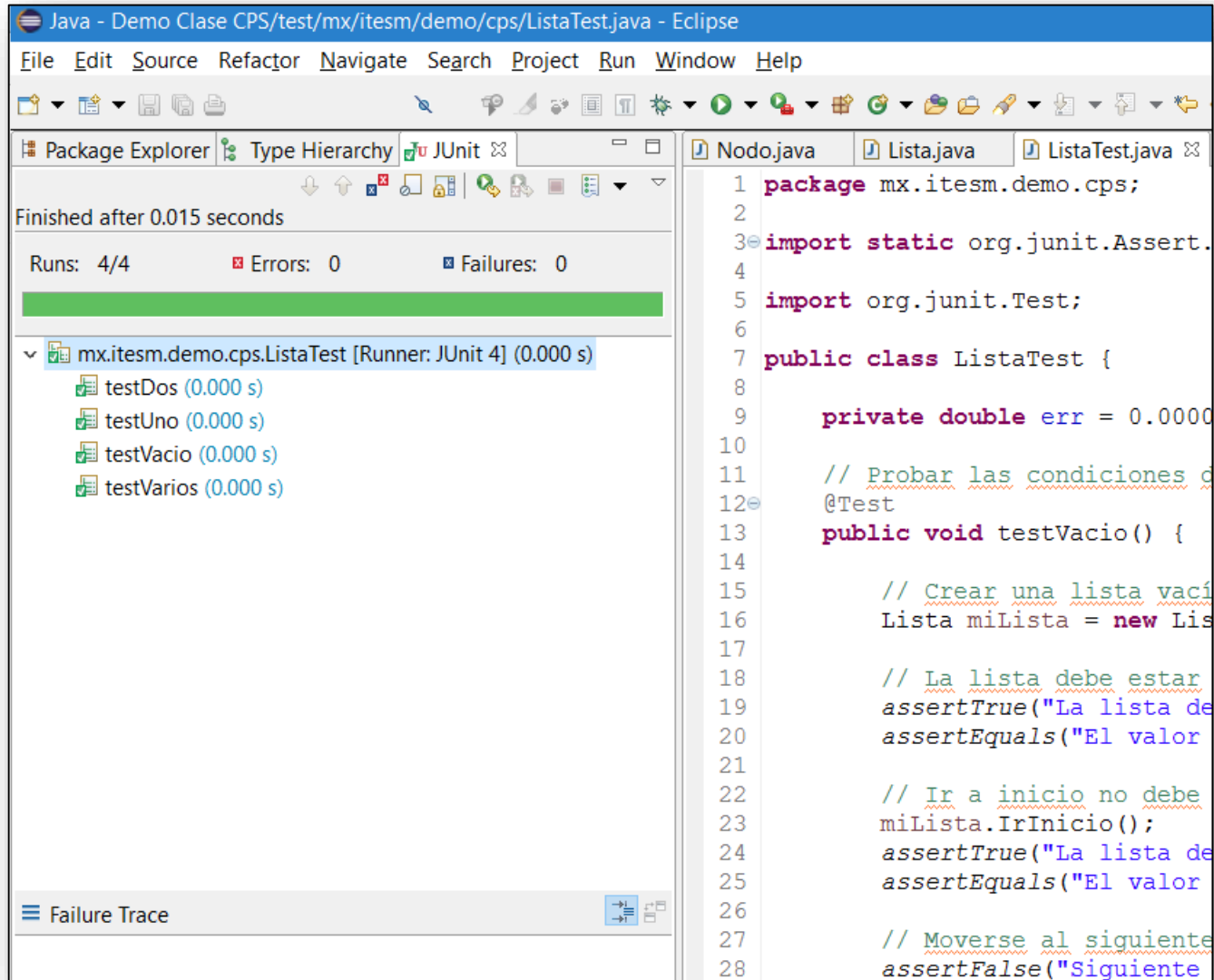
Demo 1  
Demo 1408  
Demo Clase CPS  
src  
mx.itesm.demo.cps  
Lista.java  
Nodo.java  
test  
mx.itesm.demo.cps  
ListaTest.java  
ListaTest  
err  
testDos() : void  
testUno() : void  
testVacio() : void  
testVarios() : void  
JRE System Library [JavaSE-1.8]  
JUnit 4  
PersistenceTutorial  
Prueba  
WorkbenchTutorial

Run (Ctrl+F11)  
Debug (F11)  
Run History  
Run As (Alt+Shift+X, T)  
Run Configurations...  
Debug History  
Debug As  
Debug Configurations...  
Toggle Breakpoint (Ctrl+Shift+B)  
Toggle Line Breakpoint  
Toggle Method Breakpoint  
Toggle Watchpoint  
Skip All Breakpoints (Ctrl+Alt+B)  
Remove All Breakpoints  
Add Java Exception Breakpoint...  
Add Class Load Breakpoint...  
All References...  
All Instances... (Ctrl+Shift+N)  
Instance Count...  
Watch  
Inspect (Ctrl+Shift+I)  
Display (Ctrl+Shift+D)  
Execute (Ctrl+U)  
Force Return (Alt+Shift+F)  
External Tools

```

es de una lista con vario
ser 7 5
1 JUnit Test
{
con 9, 7, 8, 1, 3
Lista();
);
p(7);
p(9);
);
);
estar vacia
ta no debia estar vacia",
lor actual debia ser 3",
medio debia ser 5.6", 5.
ner 9, 7, 8, 1, 3
ista.ValorActual();
iente()) {
*10 + miLista.ValorActual
112 assertEquals("El valor actual debia ser 9781

```



Java - Demo Clase CPS/test/mx/itesm/demo/cps/ListaTest.java - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer Type Hierarchy JUnit

Finished after 0.015 seconds

Runs: 4/4 Errors: 0 Failures: 0

mx.itesm.demo.cps.ListaTest [Runner: JUnit 4] (0.000 s)

- testDos (0.000 s)
- testUno (0.000 s)
- testVacio (0.000 s)
- testVarios (0.000 s)

Failure Trace

```
1 package mx.itesm.demo.cps;
2
3 import static org.junit.Assert.*;
4
5 import org.junit.Test;
6
7 public class ListaTest {
8
9     private double err = 0.0000
10
11     // Probar las condiciones d
12     @Test
13     public void testVacio() {
14
15         // Crear una lista vacía
16         Lista miLista = new Lis
17
18         // La lista debe estar
19         assertTrue("La lista de
20         assertEquals("El valor
21
22         // Ir a inicio no debe
23         miLista.IrInicio();
24         assertTrue("La lista de
25         assertEquals("El valor
26
27         // Moverse al siguiente
28         assertFalse("Siguiente
```