



Presentación Entorno



Manuel Rico Ruiz
David Guzman De la cuadra



Índice

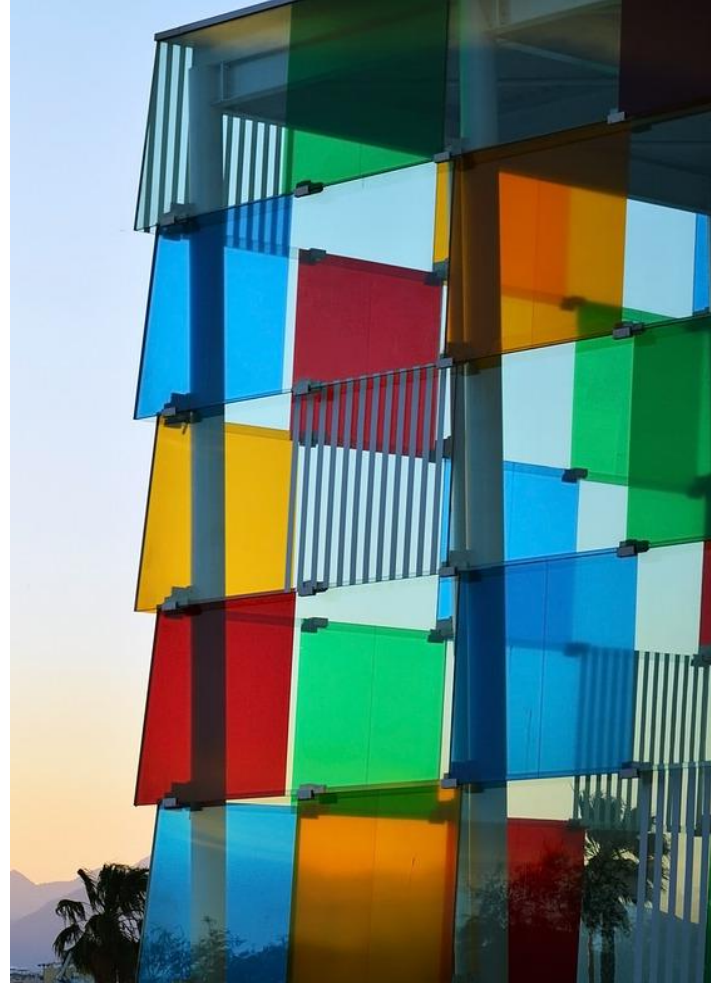
1. Refactorización ¿Qué es?
2. Pruebas unitarias¿Qué es?
 - a. Prueba Unitaria de Clase Crear Cuenta
 - b. Prueba Unitaria de Iniciar Sesión
3. Refactorización

Refactorización ¿Qué es?

- ❖ Una refactorización en un programa Java sirve para mejorar el código sin cambiar su funcionalidad.

Su objetivo principal es:

- Hacer el código más claro y fácil de entender
- Reducir la duplicación de código
- Mejorar el rendimiento en algunos casos
- Facilitar el mantenimiento y ampliación del programa

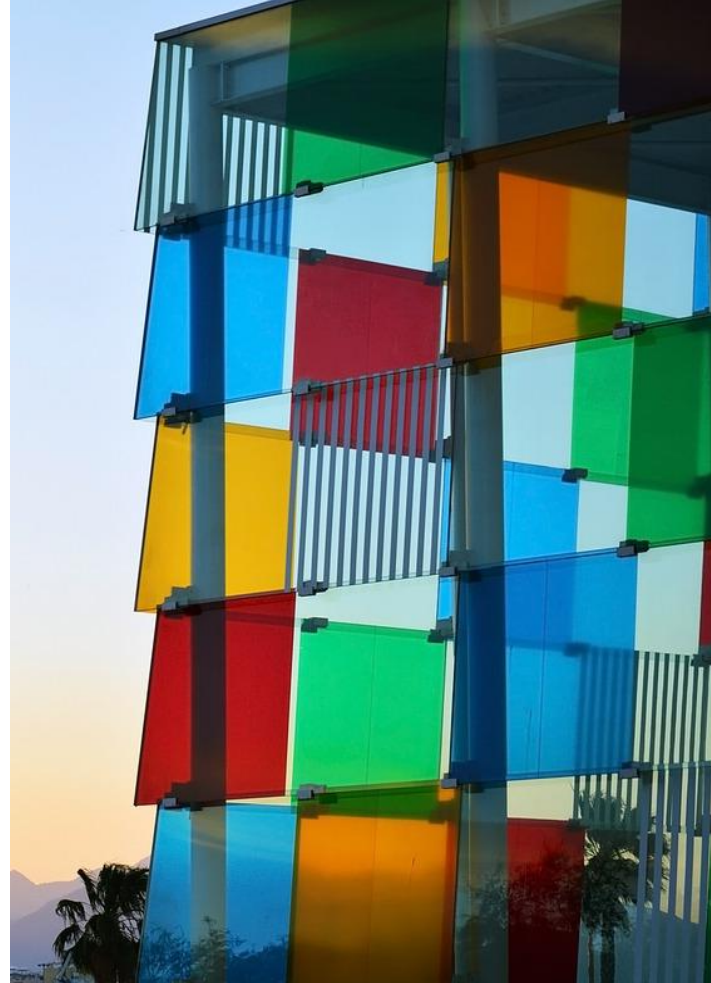


Pruebas unitarias

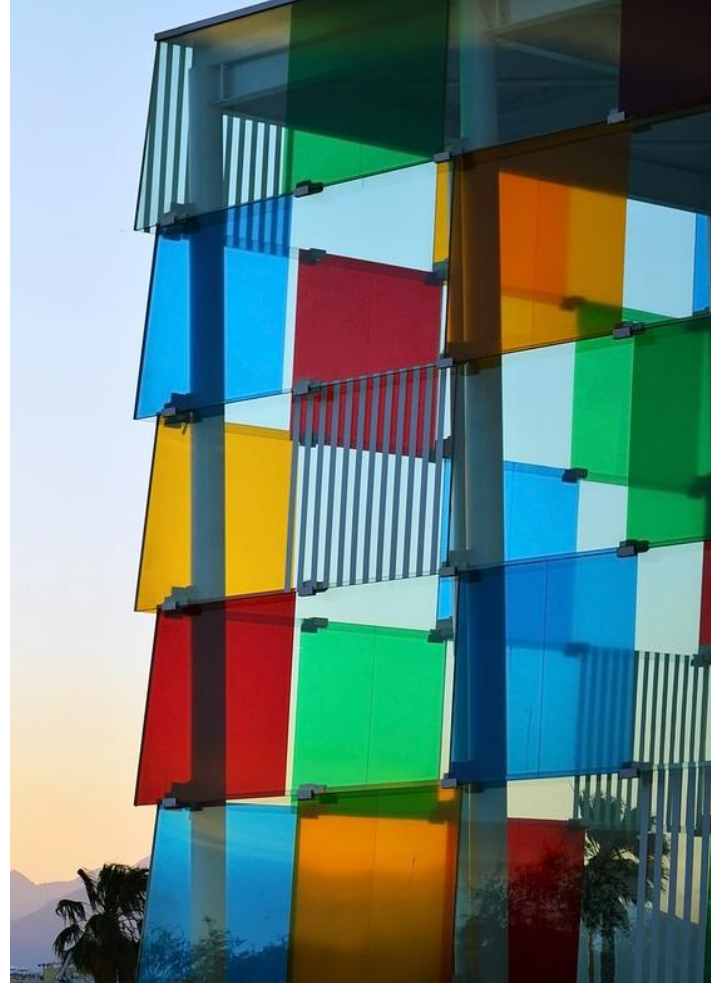
¿Qué es?

Una **refactorización** en un programa Java sirve para **mejorar el código sin cambiar su funcionalidad**. Su objetivo principal es:

- Hacer el código **más claro y fácil de entender**
- **Reducir la duplicación** de código
- **Mejorar el rendimiento** en algunos casos
- Facilitar el **mantenimiento y ampliación** del programa



Pruebas unitarias



Prueba Unitaria de Clase Crear Cuenta

Test 1:

Verifica que el título de la ventana es correcto

Confirma que el título de la ventana sea "IP Tracker" al inicializarse.

Test 2:

Verifica que la contraseña se puede escribir correctamente

Simula que el usuario escribe en el campo de contraseña y verifica que el texto se almacena correctamente.

Test 3:

Verifica que el botón "Atrás" cambia de ventana y cierra la actual

Simula la pulsación del botón "Atrás" y comprueba si la nueva ventana

Ventana_Identificarse se muestra y la actual se cierra

Test 4:

Verifica que el botón "Crear" ejecuta la lógica de base de datos

Simula rellenar los campos de usuario y contraseña, pulsa "Crear" y verifica si se muestra un JOptionPane de éxito.

```
package Testing;

import ...

public class VentanaCrearCuentaTest {

    @Test
    public void testVentanaNoEsNull() {
        VentanaCrearCuenta ventana = new VentanaCrearCuenta();
        assertNotNull(message: "La ventana no debe ser null", ventana);
    }

    @Test
    public void testTituloVentana() {
        VentanaCrearCuenta ventana = new VentanaCrearCuenta();
        assertEquals(message: "El título debe ser 'IP Tracker'", expected: "IP Tracker", ventana.getTitle());
    }

    @Test
    public void testVentanaTieneContentPane() {
        VentanaCrearCuenta ventana = new VentanaCrearCuenta();
        assertNotNull(message: "El contentPane no debe ser null", ventana.getContentPane());
    }

    @Test
    public void testVentanaEsInvisiblePorDefecto() {
        VentanaCrearCuenta ventana = new VentanaCrearCuenta();
        assertFalse(message: "La ventana no debe ser visible por defecto", ventana.isVisible());
    }
}
```

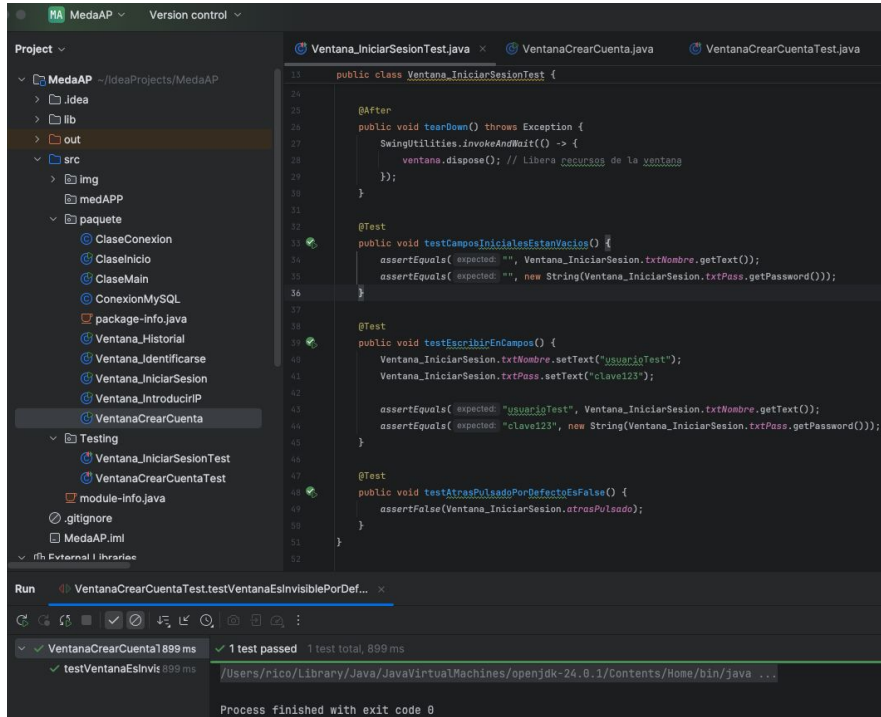
Run VentanaCrearCuentaTest.testVentanaEsInvisiblePorDefecto... x

✓ VentanaCrearCuentaTest 899 ms ✓ 1 test passed 1 test total, 899 ms

✓ testVentanaEsInvisiblePorDefecto 899 ms

Process finished with exit code 0

Prueba Unitaria de Iniciar Sesión



Test 1: testCamposInicialesEstanVacios

Verifica que al iniciar la ventana de inicio de sesión, los campos de texto (nombre de usuario y contraseña) están vacíos por defecto.

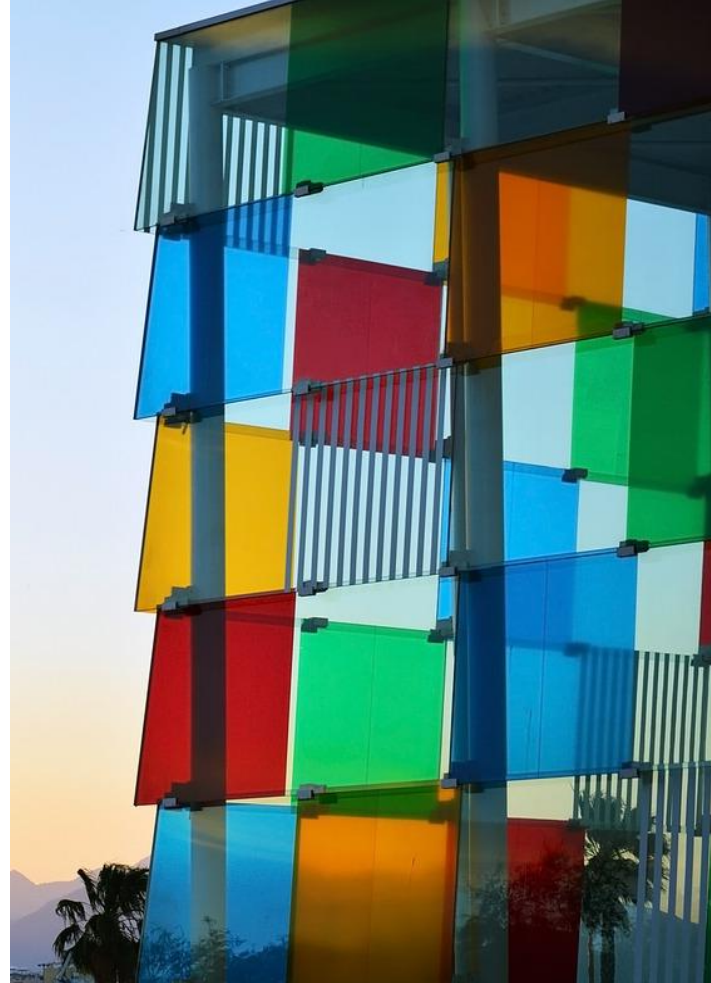
Test 2: testEscribirEnCampos

Confirma que los campos de texto aceptan la entrada del usuario correctamente y que se puede recuperar esa entrada.

Test 3: testAtrasPulsadoPorDefectoEsFalse

Comprueba que la variable booleana `atrasPulsado` está en `false` cuando se crea la ventana.

Refactorización



Refactorización de las clases

Hemos refactorizado tanto las variables como los nombres de las variables.

también hemos suprimido código que no servía para nada

Lo hemos hecho en java ya que creemos que se hace de una manera más fácil que en otros IDEs dedicado a java como podría ser eclipse

