
Entorno de Desarrollo

Pedro Pérez Fernández
Ainhoa Plata Del Río

Índice

1. Pruebas Unitarias
2. CestaTest
3. GetModeloTest
4. LoginTest
5. OpcionesVentaTest
6. VentaTest
7. Refactorización

Pruebas Unitarias

Hemos creado 5 pruebas unitarias.

1. CestaTest
2. GetModeloTest
3. LoginTest
4. OpcionesVentaTest
5. VentaTest

CestaTest

Esta clase CestaTest es un conjunto de pruebas unitarias hechas con JUnit 5 para verificar que la clase Cesta funciona correctamente.

```
1  import org.junit.jupiter.api.BeforeEach;
2  import org.junit.jupiter.api.Test;
3  import static org.junit.jupiter.api.Assertions.*;
4  import PixelDB.Cesta;
5
6  class CestaTest {
7
8      private Cesta cesta;
9
10     @BeforeEach
11     void setup() {
12         cesta = new Cesta();
13     }
14
15     @Test
16     void testAddItem() {
17         Cesta.Objeto obj = new Cesta.Objeto("Modelo1", "Opcion1", "Rojo", 100.0);
18         cesta.addItem(obj);
19         assertEquals(1, cesta.getConteoObjetos());
20         assertFalse(cesta.isEmpty());
21     }
22
23     @Test
24     void testRemoveItem() {
25         Cesta.Objeto obj = new Cesta.Objeto("Modelo2", "Opcion2", "Azul", 200.0);
26         cesta.addItem(obj);
27         cesta.removeItem(0);
28         assertTrue(cesta.isEmpty());
29     }
30
31     @Test
32     void testGetTotalPrecio() {
33         cesta.addItem(new Cesta.Objeto("M1", "Op1", "Negro", 50.0));
34         cesta.addItem(new Cesta.Objeto("M2", "Op2", "Blanco", 75.5));
35         assertEquals(125.5, cesta.getTotalPrecio(), 0.01);
36     }
37
38     @Test
39     void testClear() {
40         cesta.addItem(new Cesta.Objeto("M3", "Op3", "Verde", 30.0));
41         cesta.clear();
42         assertTrue(cesta.isEmpty());
43     }
44
45     @Test
46     void testIsEmptyInitially() {
47         assertTrue(cesta.isEmpty());
48     }
49 }
```

GetModeloTest

Esta prueba unitaria verifica que el método `getModeloFromProductId(int id)` de la clase `Principal` devuelve `null` cuando se le pasa un ID no válido.

```
1  import PixelDB.Principal;
2  import org.junit.jupiter.api.*;
3  import static org.junit.jupiter.api.Assertions.*;
4
5  ✓ public class GetModeloTest {
6
7      private Principal principal;
8
9      @BeforeEach
10     void setUp() {
11         principal = new Principal();
12     }
13
14     @Test
15     @DisplayName("getModeloFromProductId retorna null para ID inválido")
16     void testGetModeloFromProductIdInvalido() {
17         assertNull(principal.getModeloFromProductId(-1));
18         assertNull(principal.getModeloFromProductId(0));
19     }
20 }
```

LoginTest

La clase LoginTest contiene pruebas unitarias para verificar el comportamiento del método `mostrarLoginDialog()` de la clase `Principal`, que aparentemente gestiona el inicio de sesión (login) en la aplicación.

```
1  import PixelDB.Principal;
2  import org.junit.jupiter.api.*;
3  import static org.junit.jupiter.api.Assertions.*;
4
5
6  public class LoginTest {
7      private Principal principal;
8
9
10     @BeforeEach
11     void setup() {
12         principal = new Principal();
13     }
14
15
16     @Test
17     @DisplayName("Test login exitoso")
18     void testLoginSuccessful() {
19         boolean result = principal.mostrarLoginDialog();
20         assertTrue(result);
21     }
22
23
24     @Test
25     @DisplayName("Test login fallido")
26     void testLoginFailed() {
27         boolean result = principal.mostrarLoginDialog();
28         assertFalse(result);
29     }
30 }
```

OpcionesVentaTest

La clase OpcionesVentaTest contiene una única prueba unitaria para verificar que el método mostrarOpcionesVenta() de la clase Principal no lanza ninguna excepción al ejecutarse.

```
1  import PixelDB.Principal;
2  import org.junit.jupiter.api.DisplayName;
3  import org.junit.jupiter.api.Test;
4  import static org.junit.jupiter.api.Assertions.*;
5
6  public class OpcionesVentaTest {
7
8      Principal principal = new Principal();
9      @Test
10     @DisplayName("Test mostrarOpcionesVenta no lanza excepción")
11     void testMostrarOpcionesVenta() {
12         assertDoesNotThrow(() -> principal.mostrarOpcionesVenta());
13     }
14
15 }
```

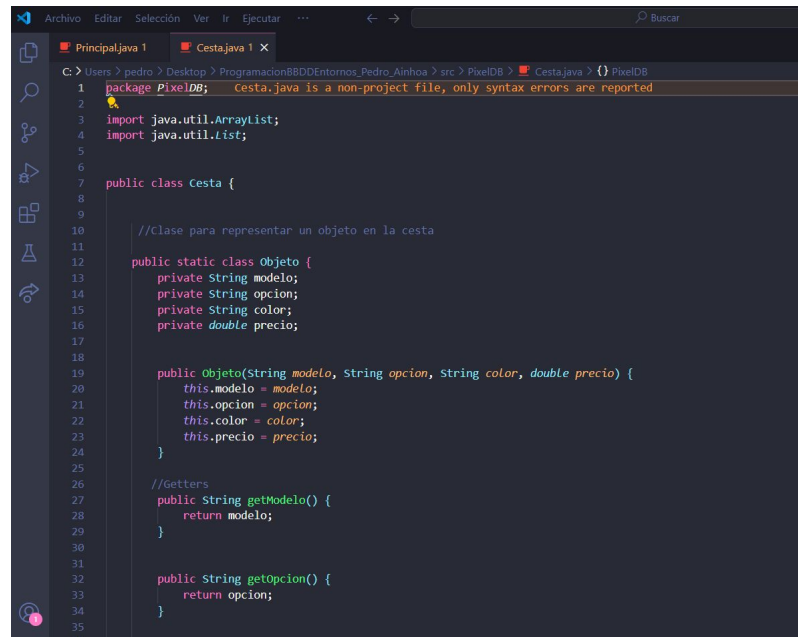
VentaTest

Este archivo VentaTest contiene tres pruebas unitarias que verifican que distintos métodos relacionados con el proceso de venta en la clase Principal no lanzan excepciones al ejecutarse.

```
1  import PixelDB.Principal;
2  import org.junit.jupiter.api.*;
3  import static org.junit.jupiter.api.Assertions.*;
4  import javax.swing.*;
5
6
7  public class VentaTest {
8      private Principal principal;
9
10
11      @BeforeEach
12      void setup() {
13          principal = new Principal();
14      }
15
16
17      @Test
18      @DisplayName("Test mostrar opciones venta")
19      void testMostrarOpcionesVenta() {
20          assertDoesNotThrow(() -> principal.mostrarOpcionesVenta());
21      }
22
23
24      @Test
25      @DisplayName("Test mostrar carrito")
26      void testMostrarCarrito() {
27          assertDoesNotThrow(() -> principal.mostrarCarrito());
28      }
29
30
31      @Test
32      @DisplayName("Test mostrar formulario datos")
33      void testMostrarFormularioDatos() {
34          assertDoesNotThrow(() ->
35              principal.mostrarFormularioDatos("8GB RAM/64GB ROM", "Negro"));
36      }
37  }
```


Refactorización

Hemos llevado a cabo una refactorización significativa del proyecto, en la que se ha creado una clase independiente llamada Cesta para gestionar de forma más clara y estructurada los productos seleccionados por el usuario. Esta mejora ha permitido separar responsabilidades, facilitando el mantenimiento del código. Además, se han optimizado diversos fragmentos de la clase Principal, reorganizando su estructura y simplificando métodos que antes eran extensos o poco legibles, logrando así un código más limpio, modular y fácil de testear.



```
1 package PixelDB;
2
3 import java.util.ArrayList;
4 import java.util.List;
5
6
7 public class Cesta {
8
9
10     //Clase para representar un objeto en la cesta
11
12     public static class Objeto {
13         private String modelo;
14         private String opcion;
15         private String color;
16         private double precio;
17
18
19         public Objeto(String modelo, String opcion, String color, double precio) {
20             this.modelo = modelo;
21             this.opcion = opcion;
22             this.color = color;
23             this.precio = precio;
24         }
25
26         //Getters
27         public String getModelo() {
28             return modelo;
29         }
30
31
32         public String getOpcion() {
33             return opcion;
34         }
35     }
36 }
```

