
PROYECTO 3

TRIMESTRE

**ABEL GÁMEZ KMIEC &
SERGIO JIMÉNEZ RAMÍREZ**

ÍNDICE

- CAPTURAS DE LAS PRUEBAS UNITARIAS
- CAPTURAS DE ALGUNOS MÉTODOS DE REFACTORIZACIÓN

PRUEBAS UNITARIAS

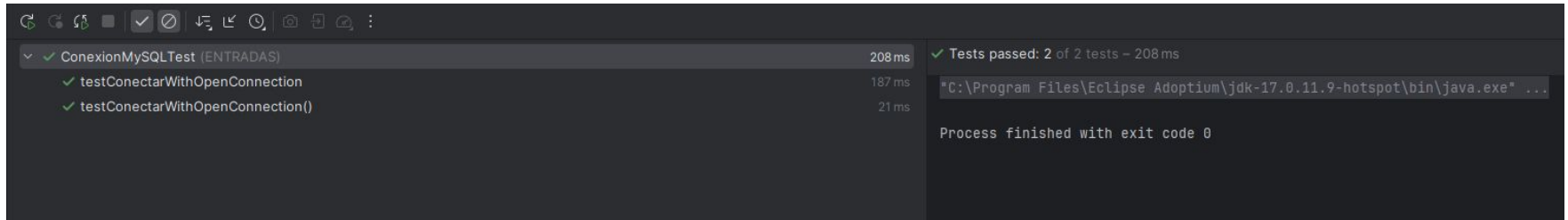
Las pruebas unitarias han sido realizadas en IntelliJ

Esta prueba unitaria está referenciada a la conexión de la base de datos con el proyecto. Lo que hace esta prueba es verificar que la conexión con la base de datos existe.

```
1 package ENTRADAS;
2
3 import org.junit.jupiter.api.Test;
4
5 import java.sql.Connection;
6 import java.sql.SQLException;
7
8 import static org.junit.jupiter.api.Assertions.assertFalse;
9 import static org.junit.jupiter.api.Assertions.assertNotNull;
10
11 public class ConexionMySQLTest {
12
13     @org.junit.Test
14     @Test
15     public void testConectarWithOpenConnection() throws SQLException {
16
17         // Arrange
18         ConexionMySQL conexionMySQL = new ConexionMySQL(usuario: "root", pass: "", bd: "sri_lanka");
19         Connection connection = null;
20
21
22         // Act
23
24         try {
25             conexionMySQL.conectar();
26             connection = conexionMySQL.getConnection();
27             // Assert
28             assertNotNull(connection);
29             assertFalse(connection.isClosed());
30         } finally {
31             if (connection != null) {
32                 connection.close();
33             }
34         }
35     }
36 }
37 }
```

RESULTADO DE LA PRUEBA UNITARIA

TEST DE LA CONEXIÓN A LA BASE DE DATOS



The screenshot displays the Eclipse IDE's test results window. On the left, a tree view shows the test class 'ConexionMySQLTest (ENTRADAS)' with a total duration of 208 ms. It contains two sub-items, both marked with green checkmarks: 'testConectarWithOpenConnection' (187 ms) and 'testConectarWithOpenConnection()' (21 ms). On the right, a summary bar indicates 'Tests passed: 2 of 2 tests - 208 ms'. Below this, a text area shows the command path for the Java runtime: 'C:\Program Files\Eclipse Adoptium\jdk-17.0.11.9-hotspot\bin\java.exe' followed by an ellipsis. At the bottom, it states 'Process finished with exit code 0'.

Test Case	Duration
ConexionMySQLTest (ENTRADAS)	208 ms
testConectarWithOpenConnection	187 ms
testConectarWithOpenConnection()	21 ms

Tests passed: 2 of 2 tests - 208 ms

C:\Program Files\Eclipse Adoptium\jdk-17.0.11.9-hotspot\bin\java.exe ...

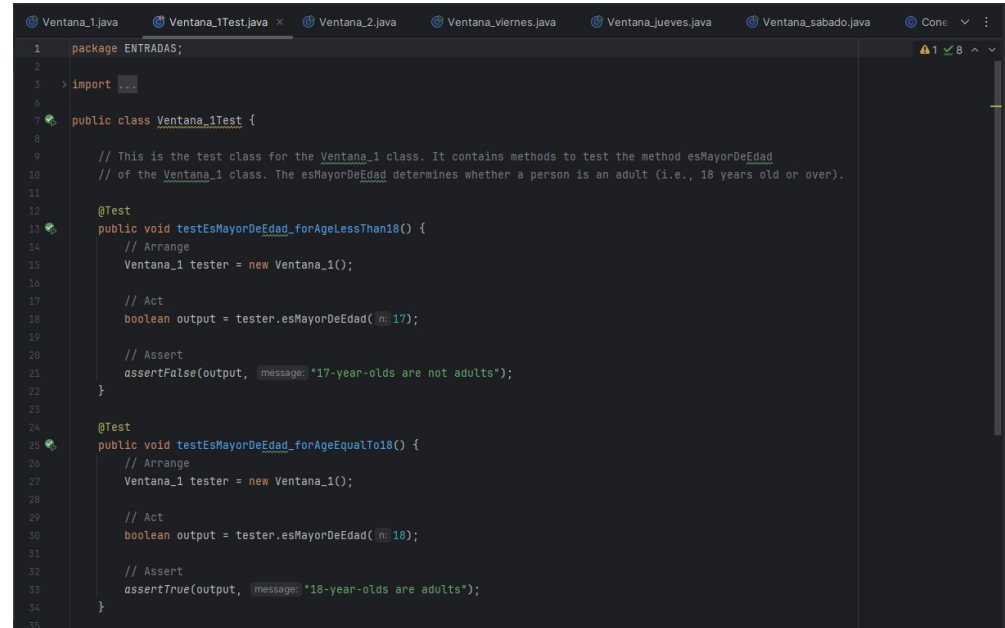
Process finished with exit code 0

Aquí podemos ver el resultado de los test de la conexión a la base de datos. Como podemos ver se realiza correctamente sin errores.

PRUEBAS UNITARIAS

Las pruebas unitarias han sido realizadas en IntelliJ

Esta prueba unitaria está referenciada a la ventana 1, lo que hacen estos test son comprobar que la edad sea mayor o igual que 18 para poder acceder a la venta de entradas.



```
1 package ENTRADAS;
2
3 > import
4
5 public class Ventana_1Test {
6
7     // This is the test class for the Ventana_1 class. It contains methods to test the method esMayorDeEdad
8     // of the Ventana_1 class. The esMayorDeEdad determines whether a person is an adult (i.e., 18 years old or over).
9
10    @Test
11    public void testEsMayorDeEdad_forAgeLessThan18() {
12        // Arrange
13        Ventana_1 tester = new Ventana_1();
14
15        // Act
16        boolean output = tester.esMayorDeEdad(17);
17
18        // Assert
19        assertFalse(output, "17-year-olds are not adults");
20    }
21
22    @Test
23    public void testEsMayorDeEdad_forAgeEqualTo18() {
24        // Arrange
25        Ventana_1 tester = new Ventana_1();
26
27        // Act
28        boolean output = tester.esMayorDeEdad(18);
29
30        // Assert
31        assertTrue(output, "18-year-olds are adults");
32    }
33
34 }
```

PRUEBAS UNITARIAS

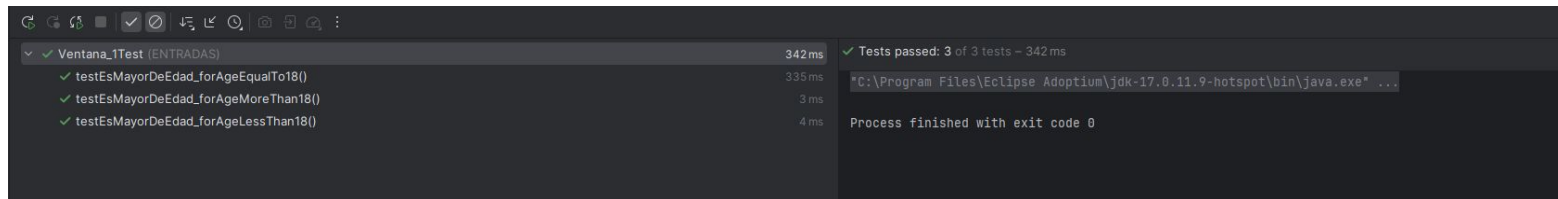
Las pruebas unitarias han sido realizadas en IntelliJ

Este es el último test de la prueba unitaria de la ventana 1.

```
35
36
37  @Test
38     public void testEsMayorDeEdad_forAgeMoreThan18() {
39         // Arrange
40         Ventana_1 tester = new Ventana_1();
41
42         // Act
43         boolean output = tester.esMayorDeEdad(n: 19);
44
45         // Assert
46         assertTrue(output, message: "19-year-olds are adults");
47     }
```

RESULTADO DE LA PRUEBA UNITARIA

TEST DE LA VENTANA 1



Aquí podemos ver el resultado de los test de la ventana 1. Como podemos ver se realiza correctamente sin errores.

REFACTORIZACIÓN

BOTÓN 500€ COMPRAR SIN REFACTORIZAR

Aquí podemos ver el botón que tiene como función seleccionar la entrada de 500€. En la siguiente diapositiva lo mostraré refactorizado.

```
 JButton btn_comprar_negras = new JButton("500€ COMPRAR\r\n");
 btn_comprar_negras.addActionListener(new ActionListener() {
     public void actionPerformed(ActionEvent e)
     {
         try
         {
             Precio=500;
             ConexionMySQL x= new ConexionMySQL("root", "", "sri_lanka");
             x.conectar();
             String sentencia="INSERT INTO entradas(Nombre,Apellido,Edad,Correo,Dia,TipoEntrada) VALUES ('"+Nombre+"','"+Apellido+"','"+Edad+"','"+Correo+"','"+Dia+"','"+Precio+"')";
             x.ejecutarInsertDeleteUpdate(sentencia);
             x.desconectar();
         }
         catch (SQLException ex)
         {
             ex.printStackTrace();
         }
     }

     Ventana_Pago venta= new Ventana_Pago();
     venta.setVisible(true);
     dispose();
 });
 });
```


REFACTORIZACIÓN

BOTÓN 500€ COMPRAR REFACTORIZADO

Aquí vemos el botón de 500€ comprar ya refactorizado, ahora se llama *crearBotonJueves* ya que ese botón corresponde a la entrada de 500€ del jueves.

```
private JButton crearBotonJueves() {  
    JButton btn_comprar_negras = new JButton("500€ COMPRAR\r\n");  
    return btn_comprar_negras;  
}
```

```
JButton btn_comprar_negras = crearBotonJueves();  
btn_comprar_negras.addActionListener(new ActionListener() {  
    public void actionPerformed(ActionEvent e)  
    {  
        try  
        {  
            Precio=500;  
            ConexionMySQL x= new ConexionMySQL("root", "", "sri_lanka");  
            x.conectar();  
            String sentencia="INSERT INTO entradas(Nombre,Apellido,Edad,Correo,Dia,TipoEntrada) VALUES ('"+Nombre+"','"+Apellido+"','"+Edad+"','"+Correo+"','"+Dia+"','"+Precio+"')";  
            x.ejecutarInsertDeleteUpdate(sentencia);  
            x.desconectar();  
        }  
        catch (SQLException ex)  
        {  
            ex.printStackTrace();  
        }  
        Ventana_Pago_venta= new Ventana_Pago();  
        venta.setVisible(true);  
        dispose();  
    }  
});
```

REFACTORIZACIÓN

BOTÓN 20€ COMPRAR SIN REFACTORIZAR

Aquí podemos ver el botón de los 20€ sin refactorizar en la siguiente diapositiva lo veremos ya refactorizado.

```
JButton btn_comprar_2consu_sabado = new JButton("20€ COMPRAR");
btn_comprar_2consu_sabado.addActionListener(new ActionListener()
{
    public void actionPerformed(ActionEvent e)
    {
        try
        {
            Precio=20;
            ConexionMySQL x= new ConexionMySQL("root", "", "sri_lanka");
            x.conectar();
            String sentencia="INSERT INTO entradas(Nombre,Apellido,Edad,Correo,Dia,TipoEntrada) VALUES ('"+Nombre+"','"+Apellido+"')";
            x.ejecutarInsertDeleteUpdate(sentencia);
            x.desconectar();
        }
        catch (SQLException ex)
        {
            ex.printStackTrace();
        }
    }
});

Ventana_Pago venta= new Ventana_Pago();
venta.setVisible(true);
venta.dispose();
}
```

REFACTORIZACIÓN

BOTÓN 20€ COMPRAR REFACTORIZADO

Aquí vemos el botón de 20€ comprar ya refactorizado, ahora se llama *boton20euros* ya que ese botón corresponde a la entrada de 20€ del sábado .

```
JButton btn_comprar_2consu_sabado = boton20euros();  
btn_comprar_2consu_sabado.addActionListener(new ActionListener()
```

```
private JButton boton20euros() {  
    JButton btn_comprar_2consu_sabado = new JButton("20€ COMPRAR");  
    return btn_comprar_2consu_sabado;  
}
```

REFACTORIZACIÓN

IMAGEN TOTEM SIN REFACTORIZAR

Aquí podemos ver el JLabel que tiene como función colocar la imagen del tótem en el lado izquierdo. En la siguiente diapositiva lo mostraré refactorizado.

```
JLabel lbl_totem_izq = new JLabel("");  
lbl_totem_izq.setIcon(new ImageIcon(Ventana_Principal.class.getResource("/IMAGENES/Totem.png")));  
lbl_totem_izq.setBounds(110, 150, 360, 371);  
contentPane.add(lbl_totem_izq);
```

REFACTORIZACIÓN

IMAGEN TOTEM REFACTORIZADO

Aquí vemos el JLabel de la imagen del tótem ya refactorizado, ahora se llama *imgTotem* ya que ese botón corresponde a la imagen del tótem del lado izquierdo.

```
JLabel lbl_totem_izq = imgTotem();  
lbl_totem_izq.setIcon(new ImageIcon(Ventana_Principal.class.getResource("/IMAGENES/Totem.png")));  
lbl_totem_izq.setBounds(110, 150, 360, 371);  
contentPane.add(lbl_totem_izq);
```

```
private JLabel imgTotem() {  
    JLabel lbl_totem_izq = new JLabel("");  
    return lbl_totem_izq;  
}
```

