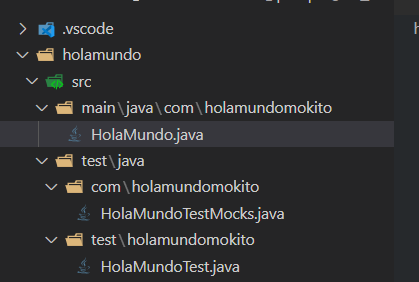
Pruebas con Mockito

Ejercicio de Referencia: Hola mundo

● Crea un método en tu clase Main que imprima "Hola Mundo" en la consola.

● Crea una clase test para testear este método.



Application.java cambiè a nombre con anticlik a **rename** HolaMundo,java

package com.holamundomokito;

public class HolaMundo {

public static *void* main(*String*[] *args*) {

System.out.println("Hola Mundo");

}

}

HolaMundotest (sin mokito)

package test.holamundomokito;

import static org.junit.jupiter.api.Assertions.assertEquals;

import java.io.ByteArrayOutputStream;

import java.io.PrintStream;

import org.junit.jupiter.api.AfterEach;

import org.junit.jupiter.api.BeforeEach;

import org.junit.jupiter.api.Test;

import com.holamundomokito.HolaMundo;

public class HolaMundoTest {

// Configuracion output

// Configuración de la captura de salida del sistema

private final *PrintStream* standarOut = System.out; // Guarda la salida estándar original.

// Inicializa un ByteArrayOutputStream para capturar la salida.

private final *ByteArrayOutputStream* ouutputStreamCaptor = new ByteArrayOutputStream();

@*BeforeEach* // Redirige la salida estándar al ByteArrayOutputStream antes de cada prueba.

public *void* setUp() {

System.setOut(new PrintStream(ouutputStreamCaptor));

}

@*AfterEach* // Restaura la salida estándar original después de cada prueba.

public *void* tearDown() {

System.setOut(standarOut);

}

@*Test*

*void* testMain() {// Ejecuta el método main de Ejercicio03.

HolaMundo.main(new *String*[1]);// Captura la salida del sistema.

*String* salida = ouutputStreamCaptor.toString();

// Compara el primer argumento (esperado) con el segundo argumento (salida) y lanza una excepción si no son iguales

assertEquals("Hola Mundo\r\n", salida);

}

}

ApplicationTest.java lo cambiè a HolaMundoMocks,java

package com.holamundomokito;

import org.junit.jupiter.api.Test;

import static org.mockito.Mockito.mock;

import static org.mockito.Mockito.verify;

import static org.junit.jupiter.api.Assertions.assertEquals;

import java.io.ByteArrayOutputStream;

import java.io.PrintStream;

import org.junit.jupiter.api.AfterEach;

import org.junit.jupiter.api.BeforeEach;

class HolaMundoTestMocks {

private final *ByteArrayOutputStream* outputStreamCaptor = new ByteArrayOutputStream();

private final *PrintStream* standarOut = System.out;

@*BeforeEach*

public *void* setUp() {

System.setOut(new PrintStream(outputStreamCaptor));

}

@*AfterEach*

public *void* tearDown() {

System.setOut(standarOut);

}

@*Test*

*void* testMainWithMocks() {

// Redirigir la salida estándar

System.setOut(new PrintStream(outputStreamCaptor));

// Llamar al método main

HolaMundo.main(new *String*[1]);

// Verificar la salida

assertEquals("Hola Mundo\r\n", outputStreamCaptor.toString());

}

@*Test*

*void* testMainWithMockito() {

// Creamos un mock de System.out

*PrintStream* mockedPrintStream = mock(PrintStream.class);

// Redirigimos System.out a nuestro mock

System.setOut(mockedPrintStream);

// Ejecutamos el método main

HolaMundo.main(new *String*[1]);

// Verificamos que se haya invocado println con "Hola Mundo"

verify(mockedPrintStream).println("Hola Mundo");

}

}

Ejercicio de Referencia: Imprimir nombre y edad

● Crea un método en tu clase Main que reciba por parámetro el nombre y la edad, y

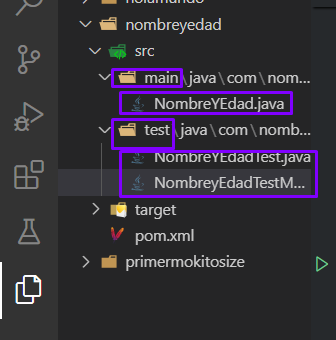
luego imprima el mensaje "Me llamo [nombre] y tengo [edad] años". No debes

hacer uso de la clase Scanner. En su lugar, simplemente crea el método

correspondiente y reemplaza [nombre] y [edad] con tus datos almacenados

previamente en una variable, para comparar con la salida del output.

● Crea una clase para testear esta actividad.



Application.java lo cambiè a nombre con anticlik a **rename** a NombreYEdad.java

package com.nombreyedad;

public class NombreYEdad {

public static *void* main(*String*[] *args*) {

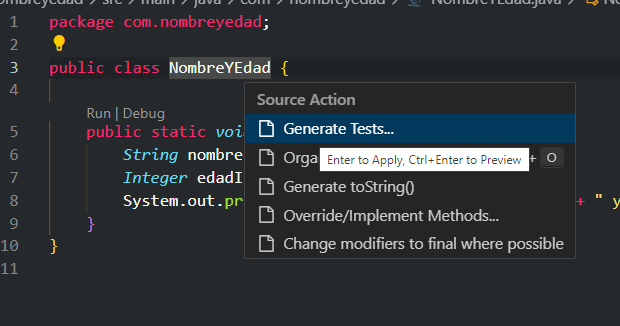
*String* nombreIngresado = "Pedro";

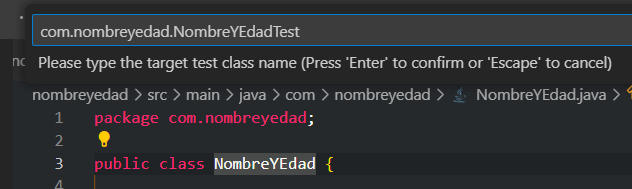
*Integer* edadIngresada = 70;

System.out.println("Me llamo " + nombreIngresado + " y tengo " + edadIngresada + " años");

}

}





NombreYTest (sin mokito)

package com.nombreyedad;

import static org.junit.jupiter.api.Assertions.assertEquals;

import java.io.ByteArrayOutputStream;

import java.io.PrintStream;

import org.junit.jupiter.api.AfterEach;

import org.junit.jupiter.api.BeforeEach;

import org.junit.jupiter.api.Test;

public class NombreYEdadTest {

private final *PrintStream* standarOut = System.out;

private final *ByteArrayOutputStream* outputStreamCaptor = new

ByteArrayOutputStream();

@*BeforeEach*

public *void* setUp() {

System.setOut(new PrintStream(outputStreamCaptor));

}

@*AfterEach*

public *void* tearDown() {

System.setOut(standarOut);

}

@*Test*

*void* testMain() {

NombreYEdad.main(new *String*[1]);

*String* salida = outputStreamCaptor.toString();

assertEquals("Me llamo Pedro y tengo 70 años\r\n", salida);

}

}

ApplicationTest.java lo cambiè a NombreYEdadMocks,java

package com.nombreyedad;

import org.junit.jupiter.api.Test;

import static org.mockito.Mockito.mock;

import static org.mockito.Mockito.verify;

import java.io.ByteArrayOutputStream;

import java.io.PrintStream;

import org.junit.jupiter.api.AfterEach;

import org.junit.jupiter.api.BeforeEach;

class NombreyEdadTestMocks {

private final *PrintStream* standarOut = System.out;

private final *ByteArrayOutputStream* outputStreamCaptor = new

ByteArrayOutputStream();

@*BeforeEach*

public *void* setUp() {

System.setOut(new PrintStream(outputStreamCaptor));

}

@*AfterEach*

public *void* tearDown() {

System.setOut(standarOut);

}

@*Test*

*void* testMainWithMocks() {

// Creamos un mock de PrintStream

*PrintStream* mockedPrintStream = mock(PrintStream.class);

// Redirigimos System.out a nuestro mock

System.setOut(mockedPrintStream);

// Llamamos al método main de la clase NombreyEdad

NombreYEdad.main(new *String*[0]);

// Verificamos que el método println haya sido invocado correctamente con el mensaje esperado

verify(mockedPrintStream).println("Me llamo Pedro y tengo 70 años");

}

}