Recibe una cálida:

Bienvenida!

Te estábamos esperando 😁







> Pruebas Unitarias en Java

Plan formativo: Desarrollo de Aplicaciones Full Stack Java Trainee V2.0





HOJA DE RUTA

¿Cuáles skill conforman el programa?









REPASO CLASE ANTERIOR



En la clase anterior trabajamos 📚:

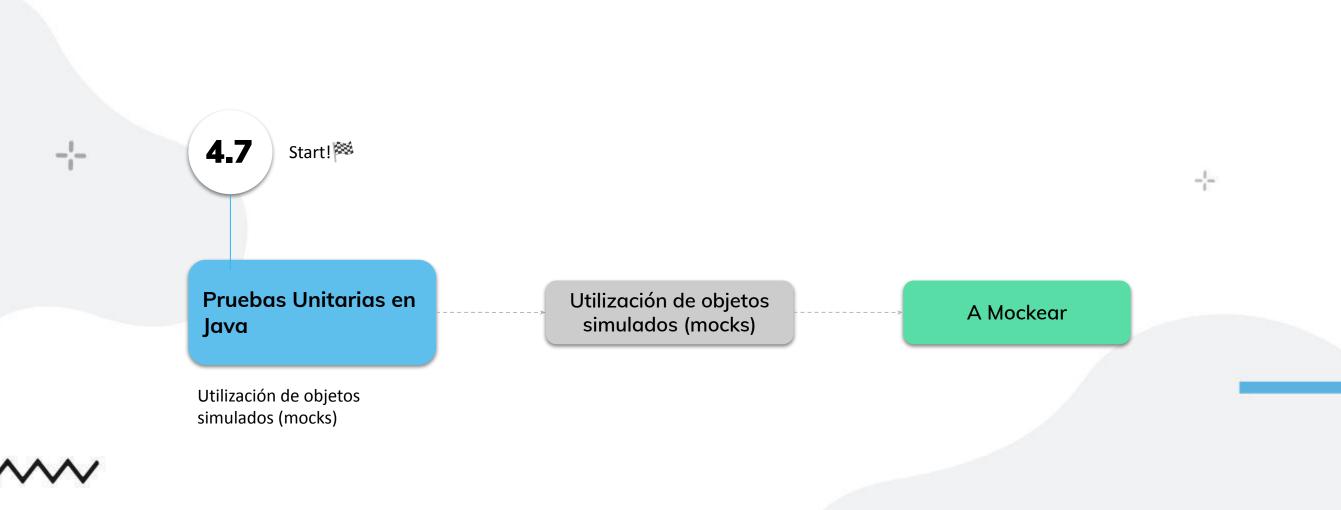
Concepto y utilización de objetos simulados en programación orientada a objetos







LEARNING PATHWAY





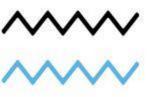


OBJETIVOS DE APRENDIZAJE

¿Qué aprenderemos?



Comprender la implementación de mocks en programación orientada a objetos





> Utilizando Mockito





Mockito

Dentro de **Spring Boot existe Mockito** que nos ayuda a simular pruebas teniendo en cuenta, por ejemplo, qué tipos de datos enviamos y qué tipos de datos esperamos como respuesta.

Para implementar mockito en un proyecto:

En primer lugar debe agregarse la dependencia.

Luego existen diferentes formas de habilitar el uso de anotaciones con las pruebas de Mockito:

- La primera opción que tenemos es anotar la prueba JUnit con un MockitoJUnitRunner
- Alternativamente, podemos habilitar las anotaciones de Mockito mediante programación invocando MockitoAnnotations.openMocks()

```
1  @RunWith(MockitoJUnitRunner.class)
2  public class MockitoTest {
3     ...
4  }
```

```
1  @Before
2  public void init() {
3     MockitoAnnotations.openMocks(this);
4 }
```







Para crear un objeto simulado (mock) en un método:

Podemos usar el método **mock()** de la clase **Mockito** para crear un objeto simulado de una clase o interfaz determinada. Esta es la forma más sencilla de simular un objeto.

```
1  @Test
2  public void test() {
    List<String> mockList = Mockito.mock(List.class);
4    Mockito.when(mockList.size()).thenReturn(5);
5    assertTrue(mockList.size() == 5);
6 }
```



><



Mockito

También podemos simular un objeto usando la anotación @Mock.

Es útil cuando queremos usar el objeto simulado en varios lugares porque evitamos llamar al método mock() varias veces. El código se vuelve más legible y podemos especificar un nombre de objeto simulado que será útil en caso de errores.

```
1   @Mock
2   List<String> mockList;
3
4   @Test
5   public void test() {
      when(mockList.get(0)).thenReturn("Hola Mundo");
      assertEquals("Hola Mundo", mockList.get(0));
   }
```





Mockito

when().thenReturn()

Los objetos simulados (mocks) pueden devolver diferentes valores según los argumentos pasados a un método. La cadena de métodos when(....).thenReturn(....) se utiliza para especificar un valor de retorno para una llamada de método con parámetros predefinidos.

```
0Test
void testeandoConWhenThenReturn() {
    UsuarioDto esperado = UsuarioDto(0, "Juan");
    // aqui simularemos que queremos que el método devuelva
    Mockito.when(usuarioRepositorio.obtenerUsuario()).thenReturn(UsuarioDto(0, "Juan"));
    UsuarioDto resultado = servicio.obtenerUsuario(0);
    Assertions.assertEquals(esperado, resultado);
}
```



×



><

Mockito

Ejemplo de implementación



```
class CalculadoraTest {
   @Mock
   private Matematica matematica;
   public CalculadoraTest() {
       MockitoAnnotations.openMocks(this);
   @Test
   public void sumarRestar() {
       // Configuración del mock
       when(matematica.sumar(0,0)).thenReturn(100);
       when(matematica.restar(0,0)).thenReturn(4);
       // Creación del objeto bajo prueba
       Calculadora calculadora = new Calculadora(matematica);
       // Ejecución del método bajo prueba
       int actual = calculadora.sumarRestar("Sumar", 50, 50);
       assertEquals(10, actual);
       actual = calculadora.sumarRestar("Restar", 12, 8);
       assertEquals(0, actual);
```

Evaluación Integradora

¿Listos para un nuevo desafío? En esta clase comenzamos a construir nuestro....



Iremos completándolo progresivamente clase a clase.







LIVE CODING

Ejemplo en vivo

Creando mocks:

En este caso vamos a implementar algunos mocks para manipular objetos simulados en el ejercicio que venimos trabajando.

1. Escribir las anotaciones necesarias de las clases Test: CuentaMockitoTest, TransaccionMockitoTest y MonedaMockitoTest

Tiempo: 30 minutos







Ejercicio N° 1 A Mockear







Consigna: 🚣 Vamos a generar nuestras primeros objetos simulados

Ahora vamos a escribir la anotaciones y configuraciones necesarias en los métodos que creas pertinentes implementar en las clases Test generadas en el ejercicio anterior (de las clases de Polimorfismo)

Recuerda que puedes declarar los mocks a testear, pero aún no es necesario que pienses en su implementación. Vamos a hacer hincapié en las anotaciones y métodos a utilizar.



Tiempo : 30 minutos





¿Alguna consulta?



RESUMEN

¿Qué logramos en esta clase?



✓ Comprender el concepto e implementación de los objetos simulados (mocks)







#WorkingTime

Continuemos ejercitando



¡Antes de cerrar la clase! Te invitamos a: 👇 👇 🔷



- 1. Repasar nuevamente la grabación de esta clase
- 2. Revisar el material compartido en la plataforma de Moodle (lo que se vio en clase y algún ejercicio adicional)
 - a. Lectura Módulo 4, Lección 7: páginas 13 14
- Traer al próximo encuentro, todas tus dudas y consultas para verlas antes de iniciar nuevo tema.







Muchas Gracias!

Nos vemos en la próxima clase 🤎



M alkemy

>:

Momento:

Time-out!

⊘5 min.



