Mockito - Anotaciones y Métodos Explicados

Anotaciones de Mockito

1. **@Mock**
   - Crea un mock de una clase o interfaz.
   - Ejemplo:
     ```java
     @Mock
     private MiRepositorio miRepositorio;
     ```

2. **@InjectMocks**
   - Inyecta los mocks en la clase bajo prueba.
   - Ejemplo:
     ```java
     @InjectMocks
     private MiServicio miServicio;
     ```

3. **@Captor**
   - Permite capturar los argumentos pasados a un método mockeado.
   - Ejemplo:
     ```java
     ```

```
@Captor
private ArgumentCaptor<String> captor;
```

4. **@BeforeEach + MockitoAnnotations.openMocks(this)**
   - Inicializa los mocks antes de cada prueba.
   - Ejemplo:
   ```java
   @BeforeEach
   void setUp() {
       MockitoAnnotations.openMocks(this);
   }
   ```

Métodos de Mockito

1. **when(...).thenReturn(...)**
   - Define el comportamiento de un mock.
   - Ejemplo:
   ```java
   when(miRepositorio.obtenerDato()).thenReturn("dato simulado");
   ```

2. **verify(mock).metodo()**
   - Verifica si un método fue llamado.

- Ejemplo:

  ```java
  verify(miRepositorio, times(1)).obtenerDato();
  ```

3. **doThrow(...).when(mock).metodo()**
   - Hace que un método del mock lance una excepción.
   - Ejemplo:

     ```java
     doThrow(new RuntimeException()).when(miRepositorio).guardar(null);
     ```

4. **doNothing().when(mock).metodo()**
   - Hace que un método mock no haga nada.
   - Ejemplo:

     ```java
     doNothing().when(miRepositorio).eliminar(1);
     ```

5. **doReturn(...).when(mock).metodo()**
   - Similar a `when().thenReturn()`, pero para evitar restricciones en métodos `void`.
   - Ejemplo:

     ```java
     doReturn("valor esperado").when(miRepositorio).obtenerDato();
     ```

6. **verify(mock, never()).metodo()**

- Verifica que un método **nunca fue llamado**.
- Ejemplo:

```java
verify(miRepositorio, never()).guardar(null);
```

7. **verify(mock, atLeast(n)).metodo()**
   - Verifica que un método fue llamado al menos `n` veces.
   - Ejemplo:

```java
verify(miRepositorio, atLeast(2)).obtenerDato();
```

8. **verify(mock, atMost(n)).metodo()**
   - Verifica que un método fue llamado como máximo `n` veces.
   - Ejemplo:

```java
verify(miRepositorio, atMost(3)).obtenerDato();
```

9. **reset(mock)**
   - Restablece un mock a su estado original.
   - Ejemplo:

```java
reset(miRepositorio);
```

10. **spy(obj)**
    - Crea un "espía" que permite usar el comportamiento real de una clase.
    - Ejemplo:
      ```java
      List<String> lista = new ArrayList<>();
      List<String> spyLista = spy(lista);
      ```

11. **verifyNoMoreInteractions(mock)**
    - Verifica que **no haya más interacciones** con un mock después de ciertas verificaciones.
    - Ejemplo:
      ```java
      verify(miRepositorio).obtenerDato();
      verifyNoMoreInteractions(miRepositorio);
      ```

12. **verifyZeroInteractions(mock)**
    - Verifica que un mock **no haya sido usado en absoluto**.
    - Ejemplo:
      ```java
      verifyZeroInteractions(miRepositorio);
      ```

Métodos Adicionales en Mockito

1. **ArgumentCaptor**

   - Captura los argumentos pasados a un método.

   - Ejemplo:

     ```java

     ArgumentCaptor<String> captor = ArgumentCaptor.forClass(String.class);

     verify(miRepositorio).guardar(captor.capture());

     assertEquals("dato esperado", captor.getValue());

     ```

2. **timeout(ms)**

   - Verifica que un método se ejecuta dentro de un tiempo determinado.

   - Ejemplo:

     ```java

     verify(miRepositorio, timeout(1000)).obtenerDato();

     ```

3. **inOrder(mock1, mock2, ...)**

   - Verifica que los mocks fueron llamados en orden.

   - Ejemplo:

     ```java

     InOrder inOrder = inOrder(mock1, mock2);

     inOrder.verify(mock1).accion1();

     inOrder.verify(mock2).accion2();

     ```
