

Mockito

Sang Shin
Passion.com
“Code with Passion!”



Topics

- What is & Why Mockito?
- How to use mocking?
- Mockito APIs

What is & Why Mockito?

What is & Why Mockito?

- Mockito is a mocking framework
 - > It lets you write tests with clean & simple API
 - > Mockito tests are very readable and they produce clean verification errors
- Use Mockito to get "smart" fake implementations of classes or interfaces out of your reach
- "We decided during the main conference that we should use JUnit 4 and Mockito because we think they are the future of TDD and mocking in Java"
 - > - Dan North, the originator of BDD

How to use Mocking?

Classical 3A Test

```
@Test  
public void test() throws Exception {  
    // Arrange  
  
    // Act  
  
    // Assert  
  
}
```

Classical 3A Test without Dependency

```
@Test
public void test() throws Exception {
    // Arrange
    Testee testee = new Testee();

    // Act
    actualResult = testee.doSomething();

    // Assert
    assertTrue(expectedResult, actualResult);
}
```

3A Test with a Dependency (using Mockito)

@Test

public void test() throws Exception {

// Arrange, prepare behaviour

Testee testee = new Testee();

Dependency aMock = mock(Dependency.class); // Or use @Mock

when(aMock.someMethod()).thenReturn(someObject);

// Act

actualResult = testee.doSomething();

// Assert – verify result and mock object behavior

assertTrue(expectedResult, actualResult);

verify(aMock).someMethod(someObject);

}

1. Create a mock

2. Train a mock

3. Verify a mock

How to Use Mockito?

Mocking API

- For creating Mock object
 - > `@Mock` or `mock(Dependency.class)`
 - > `@Spy`
- For injecting Mock object to the target class
 - > `@InjectMocks`
- For initializing Mock objects
 - > `@RunWith(MockitoJUnitRunner.class)` or
 - > `MockitoAnnotations.initMocks(this);`
- For training Mocks
 - > `when(methodCall).thenReturn(returnValue);`
 - > `when(methodCall).thenThrow(Throwable)`

Example

```
@RunWith(MockitoJUnitRunner.class) // Not needed because MockitoAnnotations.initMocks(this) below
public class PersonTest {
```

```
    @Mock
    private Address address;
```

```
    @InjectMocks
    private Person person;
```

```
    @Before
    public void setup() throws Exception {
        person = new Person();
        MockitoAnnotations.initMocks(this);
    }
```

```
    @Test public void
    should_return_complete_info_including_address() throws Exception {
        when(address.retrieveAddressInfo()).thenReturn("USA Cambridge");
        assertThat(person.retrievePersonInfo(), is("Sang Shin USA Cambridge"));
        verify(address, times(1)).retrieveAddressInfo();
    }
```

```
}
```

Code with Passion!
JPassion.com

