Unit Testing with JUnit

Duration: 3 Days **Course Code: JUNIT**

Overview:

English - Please note this course is only available in English.
Nederlands - Let op: deze training is alleen in het Engels beschikbaar.

Français - Veuillez noter que ce cours est uniquement disponible en anglais. **Unit Testing with JUnit Course Overview**
JUnit is the most popular and widely used Java unit testing framework.
This hands-on course,delivered using JUnit v5,comprises sessions dealing with unit testing,setup,annotations,assertions and assumptions,conditional tests,parameterised tests,stubbing and mocking,Mockito,DAO and Servlet testing,testing multi-threaded code,and TDD.
Exercises and examples are used throughout the course to give practical hands-on experience with the techniques covered.

Target Audience:

Who will the Course Benefit?

Objectives:

- Course Objectives
- This course aims to provide the delegate with the knowledge to be able to design and code good quality,isolated unit tests with JUnit that exploit all the features of the JUnit framework.

Prerequisites:

Delegates attending this course should be familiar with basic programming concepts and have some experience coding with Java at least. Ideally delegates will be competent Java developers who have a good understanding of OO principles and are able to build simple applications that exploit APIs including Collections,IO,and JDBC. This knowledge can be obtained by attendance on the pre-requisite Java Programming 1 / Java Developer course.

JUnit 5 exploits modern Java features including lambda expressions. Some experience or familiarity with such features,therefore,is desirable. These skills can be obtained by attendance on the Java Programming 2 / Java Advanced Developer course.

Content:

Unit Testing with JUnit Training Course Course Contents - DAY 1

Course Introduction

- Administration and Course Materials
- Course Structure and Agenda
- Delegate and Trainer Introductions

Session 1: UNIT TESTING

- What is unit testing
- The purpose of unit testing
- Terminology
- The structure of a test
- Assertions
- Code coverage
- Guidelines for writing good unit tests
- Other types of testing

Session 2: GETTING STARTED WITH JUNIT

- What is JUnit
- The main features of JUnit
- JUnit in IntelliJ
- JUnit with Maven/Gradle
- A simple unit test
- Alternatives to JUnit

Session 3: ANNOTATIONS

- @Test
- @BeforeEach
- @AfterEach
- @BeforeAll
- @AfterAll
- @TestMethodOrder
- @TestInstance
- @DisplayName
- @Disabled
- @Timeout
- @ExtendWith

Session 4: ASSERTIONS ; ASSUMPTIONS

- assertEquals
- assertTrue/False
- assertNull
- assertThrows
- assertArrayEquals
- assertIterableEquals
- assertLinesMatch
- assertSame
- assertInstanceOf
- assertTimeout
- assertNot*
- assertAll
- fail
- assumeTrue/False
- assumingThat Unit Testing with JUnit Training Course Course Contents - DAY 2

Session 5: CONDITIONAL TESTS

- OS conditions
- CRE conditions
- System property conditions
- Environment variable conditions
- Custom conditions

Session 6: PARAMETERISED TESTS

- What is a parameterised test
- The junit-jupiter-params artefact
- @ParameterizedTest
- Consuming arguments
- Argument sources
- Argument conversion
- Argument aggregation
- Customising display names

Session 7: STUBS,MOCKS,; PROXIES

- Dependencies
- Test doubles/fake objects
- Stubs
- Mocks
- Proxies

Session 8: MOCKITO

- What is Mockito
- The mockito-core and mockito-junit-jupiter artefacts
- The Mockito JUnit extension
- Creating a mock/spy
- Setting expectations (when,then)
- Argument matchers
- Verifying behaviour
- Spying
- Annotations
- Mocking static methods Unit Testing with JUnit Training Course Course Contents - DAY 3

Session 9: IO,DAO,; SERVLET TESTING

- Testing methods that use IO streams
- Testing DAOs by rolling back transactions
- Moving business logic out of the Servlet to simplify testing

Session 10: TESTING MULTI-THREADED CODE

- Sharing data among threads (a review)
- The difficulty in testing concurrent applications
- Testing the code with one thread
- Testing the code with many threads
- @RepeatedTest
- The problem with non-deterministic tests
- 3rd-party libraries
- Best practices

Session 11: TEST-DRIVEN DEVELOPMENT (TDD)

- What is TDD
- The pros and cons of TDD
- The three rules
- The red green refactor lifecycle

Further Information:

For More information, or to book your course, please call us on 0800/84.009

info@globalknowledge.be

www.globalknowledge.com/en-be/