AMBIENT-IT

+33 (0)1 48 10 09 46
formation@ambient-it.net
140 Avenue Jean Lolive 93500 Pantin - France

Updated 07/27/2023

Sign up

TDD Training

3 days (21 hours)

Presentation

Test-Driven Development (TDD) is a software development method in which each test is written iteratively before the software source code is written.

Objectives

- Mastering the Test Driven Development approach and implementation
- Integrate testing into the Java application development cycle
- Get to grips with the main testing and continuous integration tools
- Understand the advantages of Test Driven Development over other programming techniques
- Developing a simple application with TDD
- Explain and illustrate the principles of this approach
- Using TDD on a new project
- Apply specific TDD techniques to an existing project.
- Driving software design through test-driven development
- Understanding the TDD cycle;
- Design tests efficiently within an xUnit tool;
- Produce solid, reliable and adaptable code;
- Create code requiring a non-existent element with a mock tool;
- Understand the implications of testing on software design and architecture.
- Mastering test-driven development

Target audience

- Engineers, Software development project managers, Developers, Testers, Architects, Technical Leaders, Business managers, Functional analysts, Functional architectsSoftware architects, PCOs, Java/jee developers.

Prerequisites

- Knowledge of object programming with Java.
- Some knowledge of object programming, as well as basic software development experience.
- Practical object design
- Practical development experience with Java or C#

Program

The different types of test

- Unit testing
- Integration testing
- Functional testing
- Performance tests
- Non-regression test
- Automatic test.

Different lining techniques

- Dummy
- Stub
- Mock
- Fake
- Summary

Automated testing with the JUnit framework

- The need for a test framework. JUnit.
- Alternatives (TestNG) and complementary tooling.
- JUnit best practices.

Fundamentals and motivation of test-driven development

- Exception handling
- The test-driven development cycle;
- Best practices in unit test design;
- Develop by isolating yourself from external dependencies using mock objects;
- Fundamental principles and motivation for redesigning your code;
- Continuous compilation.

Workshop and case study

- Define satisfaction conditions
- Add acceptance criteria to user stories
- Scripting user acceptance tests

- Coding testers and unit tests
- Generating code from tests

Bringing code under test

- Identifying a point of change
- Find test points
- Breaking dependencies
- Create a seam
- Code modification and refactoring

The tools

- Open Source and commercial tools.
- Hardware test architecture.
- Study of a continuous integration tool.
- Study and choose a continuous integrator.
- Study of a test coverage tool.
- Study of a test management and communication tool between the project owner and the project manager.

Companies concerned

This course is aimed at both individuals and companies, large or small, wishing to train their teams in a new advanced computer technology, or to acquire specific business knowledge or modern methods.

Teaching methods

Practical course: 60% Practical, 40% Theory. Training material distributed in digital format to all participants.

Organization

The course alternates theoretical input from the trainer, supported by examples, with brainstorming sessions and group work.

Validation

At the end of the session, a multiple-choice questionnaire verifies the correct acquisition of skills.

Sanction

A certificate will be issued to each trainee who completes the course.