JAVA BEST PRACTICES AND DESIGN PATTERNS

Course number : 104

Overview

Solve real-world software development problems, and deliver responsive applications that are fast and reliable. In this training course, you learn how to leverage Java best practices, avoid pitfalls, perform industry-standard software development techniques, use design patterns to implement proven solutions to reoccurring problems, and apply idioms and patterns to improve your Java code

What you'll learn

- Employ best practices to build reliable and scalable Java applications
- Effectively apply test-driven development to enhance program maintainability
- Solve architectural problems with proven design patterns
- Employ advanced Java APIs for multi-threaded programming

Who should attend

Pre-requis

- Knowledge at the level of:
  - Course 103, Java Programming Introduction
- Three to six months of Java programming experience
- You should be able to:
  - Understand Java classes, the inheritance model, polymorphism, and encapsulation
  - Use fundamental standard edition Java APIs
  - Apply object-oriented analysis and design, including defining classes and creating objects

Outline

Effective Programming in Java

- Clarifying the goals of best practices
- Identifying the key characteristics of high-quality software
- Organizing classes, packages and subsystems into layers
- Designing to the principles of SOLID

Exploiting a testing framework

- Composing and maintaining JUnit tests
- Taking advantage of advanced JUnit features
- Testing in the presence of exceptions

Monitoring software health using logging libraries

- Configuring logging with log4j and SLF4J
- Minimizing the impact of logging on performance

Creating matchers and mock objects

- Writing custom Ham crest matchers
- Testing with fake objects and mocks

Employing common design patterns

- Observer
- Iterator
- Template method
- Strategy
- State
- Singleton
- Data Accessor Object
- Data Transfer Object
- Composite
- Service Locator
- Proxy
- Factory

Refactoring legacy code

- Identifying reasons to change software
- Clarifying the mechanics of change
- Writing tests for legacy classes and methods

Improving type safety with generics and enum types

- Creating generic classes and methods
- Navigating generic class hierarchies
- Implementing enum types for fixed sets of constants

Adding metadata by writing annotations

- Leveraging the built-in and custom annotations
- Annotating with meta-annotations

Modifying runtime behavior with reflection

- Retrieving class and method data dynamically
- Flagging methods with naming conventions
- Adding information to code with annotations
- Assessing disadvantages of reflection

Measuring and improving performance

- Assessing response time
- Conducting load and stress tests

Schedule

Location	Dates	Status
Montreal	Dec 01, 2017 - Dec 07, 2017 08:00 AM - 05:00 PM	Available Register Now >>

Tuition

IN CLASSROOM OR ONLINE PRIVATE TEAM TRAINING

STANDARD	**$2990**	**Contact Us »**

IN CLASSROOM OR ONLINE PRIVATE TEAM TRAINING

GOVERNMENT $2990

FAQ

Certification

Enterprise architecture

Cyber Security

Networking & Wireless

B sis

IT S ement

Data Center

Cloud Computing

Project Management

Java Programming