Course : Best practices in Java programming

Practical course - 3d - 21h00 - Ref. BPJ
Price : 1940 CHF E.T.

★★★★⯪ **4,7 / 5**

This practical training course covers the most sensitive points of the Java language and the APIs of the Java SE platform. It also enables you to integrate the best practices of Java development and acquire the automatisms essential to the design of robust enterprise applications.

◎ Teaching objectives

At the end of the training, the participant will be able to:

- ✔ Assimilate good design practices
- ✔ Assimilate the different language paradigms: object, functional, declarative
- ✔ Mastering basic types
- ✔ Understanding how the JVM works

Intended audience
Java application developers, architects and project managers.

Prerequisites
Good knowledge of Java programming. Experience in Java application development required.

Practical details
Hands-on work
The points covered are put into practice through exercises in Eclipse. All exercises are validated by writing and running unit tests.

Course schedule

PARTICIPANTS
Java application developers, architects and project managers.

PREREQUISITES
Good knowledge of Java programming. Experience in Java application development required.

TRAINER QUALIFICATIONS
The experts leading the training are specialists in the covered subjects. They have been approved by our instructional teams for both their professional knowledge and their teaching ability, for each course they teach. They have at least five to ten years of experience in their field and hold (or have held) decision-making positions in companies.

ASSESSMENT TERMS
The trainer evaluates each participant's academic progress throughout the training using multiple choice, scenarios, hands-on work and more.
Participants also complete a placement test before and after the course to measure the skills they've developed.

1 Good design practices

- Layering and orthogonality.
- Managing consistency (strong consistency, long-term consistency).
- The KISS, DRY, POJO and SOC principles.
- Some essential patterns: factory, singleton, prototype, strategy, proxy.
- Asynchronous, non-blocking access to resources.
- Testing: unit tests, integration tests, use of stoppers.

Hands-on work
Applying the layer breakdown approach to a Java project, writing unit tests and integration tests.

2 Best practices in language use

- Best practices in object-oriented programming.
- Best practices in functional programming with lambda expressions.
- Best practices in declarative programming with annotations.
- Immutability.
- Genericity: advantages and limitations.
- Exception handling: business exceptions, technical exceptions, exception wrapping.

Hands-on work
Enhancement of the project started earlier using lambda expressions, annotations and generic types.

3 Best practices for using base types

- The Object class and its methods: equals, hashcode, toString, clone...
- The String class and its special features.
- Representing time with classes in the java.time package.
- Collections (List, Set, Queue), maps.
- The Optional class to limit the risk of NullPointerException.
- The Optional class.
- Stream interface.
- The CompletableFuture class for asynchronous programming.

Hands-on work
Application of Stream to the project started earlier, optimal use of Java base types.

4 Understanding how the JVM works

- Just in time (JIT) compiler.
- The principle of tiered compilation.
- Two memory zones: the heap and the stack.
- The various garbage collectors (G1, Shenandoah, ZGC).
- Compiling ahead of time (AOT) with GraalVM.

Hands-on work
Observation of application memory usage.

Dates and locations

REMOTE CLASS
2026 : 2 Mar., 17 June, 26 Oct.