Advanced Java Programming

Duration: 4 days Price: € 2650

Open Schedule

Audience Course Advanced Java Programming

The course Advanced Java Programming is intended for experienced Java developers who want to gain more in depth knowledge of Java.

Prerequisites Course Advanced Java Programming

Knowledge of the Java language and syntax and basic experience in **Java** programming is required to participate in this course.

Realization Training Course Advanced Java Programming

The theory is treated on the basis of presentations and is interspersed with exercises. Demos are used to clarify the theory. The course material is in English. The course times are from 9.30 up and to 16.30.

Certification Course Advanced Java Programming

Participants receive an official certificate Advanced Java Programming after successful completion of the course.

Content Course Advanced Java Programming

In the course Advanced Java Programming a series of advanced aspects of Java are discussed. The course covers the topics that are asked for on the Oracle Certified Java Professional or OCP exam and is a good preparation to pass this exam.

Advanced Classes

In the first place attention is paid to aspects of Advanced Class Design such as the implementation of inheritance and composition, the use of polymorphism, interfaces, inner and anonymous classes and the singleton pattern.

Concurrency

Next multithreaded applications are discussed and the synchronization between threads when accessing shared data. During the treatment of the concurrency package advanced synchronization mechanisms such as cyclic barriers and countdown latches are discussed.

Lambda's

Also the functional language constructs introduced in recent Java versions are discussed with lambdas and functional interfaces.

Generics

Next generics are on the course program with which classes and methods can be parameterized, strong typing is imposed and the chance of runtime errors is limited. Generics are used a lot in the Collection Framework and the most important container classes in this Framework are discussed.

Stream API

Next attention is paid to the **Stream API** that enables transformations on data collections to be performed by a combination of successive simpler methods like map and reduce.

Exceptions

The various possibilities for dealing with errors and exceptions are also on the program and attention is paid to file I/O and new I/O when accessing files and directories.

JDBC

Database access with Java Database Connectivity (JDBC) is treated and queries, prepared statements and transactions are part of that.

Reflection

Finally, if time permits, reflection is optionally on the course program, with which compiled Java classes can be analyzed by means of software, and optionally various aspects of enhancing the Java performance are discussed.

Modules Course Advanced Java Programming

Module 1 : Advanced Class Design	Module 2 : Multiple Threads	Module 3 : Concurrency
Encapsulation and Inheritance	Java Thread Model	Concurrency Package
Implementing Composition	Extending Thread Class	Task Scheduling Framework
Polymorphism	Implementing Runnable	Executor Interface
Singleton Patterns	Daemon Threads	ExecutorService
Abstract Classes	Thread Alive States	Callables and Futures
Final Classes	Sleeping and Yielding Control	Synchronizers
Inner Classes	Using join and interrupt	Semaphores and Exchanger
Static Inner Classes	Synchronized Statement	CountdownLatch and CyclicBarrier
Anonymous Inner Classes	Locking and Statics	Concurrent Collections
Autonomous Classes	Deadlock	Lock Interface
Enumerated Types	Condition Synchronization	Reentrant Locks
Implementing hashCode and equals	Using wait and notify	Atomic Variables
Module 4 : Lambda's	**Module 5 : Generics**	**Module 6 : Streams**
Passing Functionality	Type Erasure and Raw Types	What are Streams?
Lambda Expressions	Generics and Subtyping	Lazy Evaluation and Parallelization
Lambda Variable Access	Bounded Type Parameters	forEach, Map and Filter
Lambda Scoping Rules	Wildcards	findFirst and findAny
Functional Interfaces	Generics in Collections	toArray and collect
Predicate Interface	ArrayList and LinkedList	Optional Class
Consumer Interface	TreeSet and Hash Set	Limiting Stream Size
Supplier Interface	HashMap and TreeMap	allMatch and anyMatch
Function Interface	Comparable and Comparator	Number Specialized Streams
UnaryOperator Interface	Collections Streams and Filters	Parallel and Infinite Streams
BinaryOperator Interface	Iteration using forEach	collect Method
Method References	Filtering using Lambda's	Grouping with Collectors class
User Defined Functional Interfaces	Stream Pipeline	Using flatMap Method
Module 7 : Exception Handling	**Module 8 : Java IO and NIO**	**Module 9 : Database Access**
Errors and Exceptions	Standard I/O Streams	JDBC Architecture
Checked and Unchecked Exceptions	Reading and Writing Files	JDBC Drivers and URL's
Exception Hierarchy	Buffered Streams	Database Connections
Multiple Catch Clauses	Data Conversion Streams	Executing Statements
finally Clause	Serialization	Retrieving Results
try with Resources	Object Streams	Handling Errors
Auto Closeable Resources	NIO and Asynchronous I/O Processing	Prepared Statements
Common Exceptions	IO Channels	Database Metadata
Throwing Exceptions	Stream API with NIO.2	Transactions
User Defined Exceptions	Using Path Class	Commit and Rollback
Chained Exceptions and Stack Traces	Directory Traversing	Rowset Interfaces
Assertions	PathMatcher class	Using RowsetProvider
Module 10 : Localization	**Optional Module 11 : Reflection**	**Optional Module 12 : Performance**
LocalDate Class	What is Reflection?	Influences on Performance
LocalTime and LocalDateTime	Reflection Classes	JIT Compilation and Hotspot JVM
Instant and Period	Class Loading	Garbage Collection
Duration and TemporalUnit	The Class Class	String Types
Defining Properties	Creating Objects	Buffered and New I/O
Reading Property Files	Reflection Methods in Class	Synchronization and Concurrency
Creating Resource Bundles	Field Class	Primitives versus Wrappers
Formatting Date and Times	Constructor Class	Collections
Locale Class	Method Class	Exception Handling
Localizing Dates	AccessibleObject Class	Serialization
Localizing Numbers	Dynamic Proxies	Native methods
Localizing Currencies	Invocation Handler	Lazy Loading and Object Reuse