## Standard Kurs Java Advanced

Dieser Java Kurs bietet als Java Vertiefungskurs den professionellen Einsteig in die Java Basiskonzepte wie Multithreading, I/O, Collections, Beans, Reflection, Generics, Lambda, Streams und vieles mehr.

### Information

Kurscode: JPF2Das Angebot Firmenkurs finden Sie hier.Das Angebot Online Kurs finden Sie hier.Infrastruktur: Bring your own Computer (Processor i7 oder vergleichbar, mindestens 8GB RAM), VMWare Workstation Player Version 12+.Alle Kursunterlagen werden elektronisch abgegeben, damit leisten wir einen Beitrag an die Umwelt.

### Information

Kurscode: JPF2Kursdauer: 4 TageKurspreis Firmenkurs Inhouse: sFr. 7499.- (3-5 Teilnehmer)sFr. 10499.- (6-8 Teilnehmer)sFr. 12499.- (9-12 Teilnehmer)Inhouse Firmenkurse werden vor Ort bei Ihnen durchgeführt. Je nach Ort und Distanz können zusätzliche Kosten für Übernachtung und Anfahrt anfallen.Das Angebot Standard Kurs finden Sie hier.Das Angebot Online Kurs finden Sie hier.Infrastruktur: Bring your own Computer (Processor i7 oder vergleichbar, mindestens 8GB RAM), VMWare Workstation Player Version 12+.Alle Kursunterlagen werden elektronisch abgegeben, damit leisten wir einen Beitrag an die Umwelt.

### Information

Kurscode: JPF2Das Angebot Firmenkurs finden Sie hier.Das Angebot Standard Kurs finden Sie hier.Infrastruktur: Bring your own Computer (Processor i7 oder vergleichbar, mindestens 8GB RAM), VMWare Workstation Player Version 12+.Alle Kursunterlagen werden elektronisch abgegeben, damit leisten wir einen Beitrag an die Umwelt.

### Einleitung

Dieser Kurs behandelt die fortgeschrittenen Basiskonzepte der Programmiersprache Java. Zu den Kernthemen gehört das Arbeiten mit Collections, Streams, Threads, und Reflection.Weiter werden die Java Tiger Features wie Generics detailliert behandelt.Anhand des Factory Design Patterns wird das Zusammenspiel der verschiedenen Java API's transparent aufgezeigt.Sie lernen, wie man in Java parallele Tasks via Multithreading programmiert und synchronisiert.Der korrekte Einsatz des java.io Package (IO-Streams) mit den Modellen Byte- und Char-Streams wird aufgezeigt.Die Wahl der richtigen Collection (List, Map, Set) wird diskutiert und eingesetzt.Sie kennen die dynamischen Eigenschaften des Reflection API's und können diese einsetzen.Sie können Lambda Expressions und das Stream API in der Praxis einsetzen.Zahlreiche Beispiele und Übungen zu jedem Thema begleiten diesen Kurs.Dieser Kurs dient als Vorbereitung für die spätere Oracle Certified Java Programmer (OCJP) Prüfung.Daneben werden auch die Concurrency (java.util.concurrent) Klassen in das Arbeiten mit Threads einbezogen sowie das Package java.nio (New I/O) für die schnelle Verarbeitung von I/O Streams.Zentraler Bestandteil sind weiter die Programmierung mit Lambda Expressions und dem Stream API.In diesem Kurs setzen wir die Entwicklungsumgebung Eclipse oder auf Wunsch IntellJ ein.

### Ihr Nutzen

- Die Möglichkeiten des Reflection API's kennen und anwenden.
- Die richtige Collection-Klasse zu einer gegebenen Situation finden und anwenden.
- Maps effizient einsetzen können
- Dateien mit dem Java Stream Konzept öffnen und verarbeiten
- Parallele Tasks und Services in Java programmieren und synchronisieren können (Multithreading).
- Die wichtigsten Eigenschaften der Java Beans Konvention kennen
- Die neuen Java Tiger Features kennen und anwenden können.
- Lambda Expression und Stream API anwenden können.

### Verwandte Kurse

- Java SE Einführung (JEGL)
- Java SE für nicht OO Programmierer (JNOO)

**Voraussetzungen**

Gute Grundkenntnisse von Java analog den Kursen JEGL (Java Einführung) oder JNOO (Java für nicht OO Programmierer).

**Teilnehmerkreis**

Java Entwickler/Architekten

**Unterlagen**

- Tutorial
- Code Walks
- Internet / Intranet

**Folgekurse**

- Java Design Patterns (JDPP)
- Java Refactoring (JREF)

**Inhalt**

- Reflection API
- ClassLoader
- Factory Design Pattern
- Collection Klassen
  - Vector
  - Hashtable
  - Lists
  - Sets/Maps
- IO Streams
  - InputStream / OutputStream
  - Reader und Writer Klassen
  - File und RandomAccessFile
  - Interface Serializable
  - Interface Externalizable
  - java.nio New I/O
- Threads
  - Grundlagen
  - Klasse Thread
  - Interface Runnable
  - Synchronisierung
  - wait/notify
  - java.util.concurrent Package
  - Thread Pooling
    - Executor Service
    - Timed Tasks mit ScheduledThreadPoolExecutor
- Java Tiger Features
  - Das neue API
  - Autoboxing/-Unboxing
  - Generics - generische Typen
  - For-Each Loop
  - Varargs: Variable-Length Arguments
  - Enumerations
  - Static Import
  - Formatted I/O
  - Annotations
- Beans

  - Grundlagen
  - Bean Properties
  - Event Modell
- Lambda Expressions
  - Was sind Lambda Expressions
  - Syntax
  - Functional, Method, Constructor
  - Lambda Programmierung
  - Anonyme Klasse vs Lambda Ausdruck
  - Functional Interfaces
  - Method References
  - Constructor References
  - Variable Scope
  - Default Methods
  - Static Methods in Interfaces
  - Parameters of Lambda Expressions
  - Composition, Laziness
  - Parallelizing Operations
  - Lambda and Exceptions
- Stream API
  - Stream Operations (for Iteration)
  - Stream Creation
  - filter, map, flatMap Methods
  - Stream Extracting, Combining
  - Stateful Transformations
  - Primitive Type Streams
  - Parallel Streams
  - Collecting into Maps

## Kontakt

Simtech AG
Finkenweg 23
3110 Münsingen
Schweiz

## Impressum

Das Copyright für sämtliche Inhalte dieser Website liegt bei Simtech AG, Schweiz.
Beachten Sie auch unsere Hinweise zum Urheberrecht, Datenschutz und Haftungsauschluss.
Jeder Hinweis auf Fehler nehmen wir gerne entgegen.

## Copyright

2024 Simtech AG, All rights reserved, Powered by stack.ch written in Golang by Daniel Schmutz

## https://www.simtech-ag.ch/unboxing