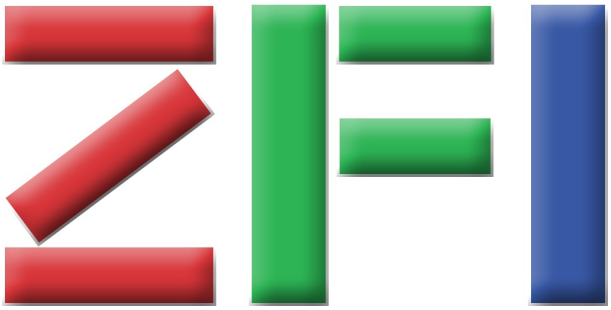
Fernkurs-Dokumentation



Zentrum für Informatik ZFI AG

Fernkurs: Oracle Certified Professional, Java SE 8 Programmer II, 1Z0-809 (FCP8)

http://www.zfi.ch/FCP8

Weitere Infos finden Sie unter www.zfi.ch oder via Adresse:

Zentrum für Informatik ZFI AG
Zentralsekretariat
Rütistrasse 28
CH-8952 Zürich-Schlieren
Telefon: 044 732 40 00
Telefax: 041 530 31 68

Zürich, Basel, Bern, ZÃ¼rich, Schweiz

Titel	**Fernkurs: Oracle Certified Professional, Java SE 8 Programmer II, 1Z0-809**
Untertitel	**Der Vorbereitungslehrgang auf die offizielle Oracle Certified Professional, Java SE 8 Programmer II 1Z0-809 Prüfung als Fernkurs.**
Einleitung	**Die Oracle Certified Professional (OCP), Java SE 8 Programmer II Zertifizierung richtet sich an Software-Entwickler, welche ihre Kenntnisse der Java Programmiersprache attestiert haben möchten. Das Bestehen der Prüfung Java Programmer II (1Z0-809) bestätigt, dass der Programmierer den Syntax und die Struktur der Java-Programmiersprache versteht und mit Java Applikationen entwickeln kann, die auf Server- oder Desktop-Systemen mit Java SE 8 laufen.** **Mit der Variante Fernkurs erarbeiten Sie sich die theoretischen Grundlagen selbständig und festigen Ihr Wissen anhand von vielen Übungen. Die wichtigsten Fakten werden in einem Factsheet zusammengefasst mitgeliefert und sind ein wichtiger Teil der Prüfungsvorbereitung.** **Eine begleitende grössere Programmierübung rundet den Fernkurs ab. Bei Fragen können Sie den ZFI Kursleiter per EMail kontaktieren und Ihre Fragen werden umgehend und kompetent beantwortet.** **Als Entwicklungsumgebung setzen wir die Eclipse Java IDE ein. Sie können aber auch die Oracle Netbeans IDE oder Intellij verwenden.** **Die erfolgreiche Absolvierung der OCA Prüfung 1Z0-808 (siehe FCA8) wird für diese Prüfung 1Z0-809, Oracle Certified Professional, Java Programmer II, vorausgesetzt.** **Nach der Anmeldung für diesen Fernkurs erhalten Sie von uns das Begleitbuch per Post zugestellt. Zugleich werden Sie von uns per EMail kontaktiert und Sie erhalten von uns den Link auf die elektronischen Unterlagen (Dokument, Übungen und Beispiele) für den Download. In einer ersten Phase arbeiten Sie die Unterlagen inkl. den Übungen durch. Die Übungen enthalten Musterlösungen. Eine Übungsreihe gilt als Repetition, die Lösung senden Sie per EMail an uns zur Kontrolle. Sie können uns jederzeit über den vereinbarten Mail-Account kontaktieren, wir helfen Ihnen gerne weiter. In einer 2. Phase erfolgt das Training auf die Prüfung selbst indem Sie viele echte Prüfungsfragen beantworten. Zu diesen Fragen gibt es wiederum Lösungen. Planen Sie für diese Vorbereitung mehrere Wochen ein, dann sind Sie für die Prüfung bereit und optimal vorbereitet.** **Die aktuellen Java SE 7/8 Programmer Zertifizierungen von Oracle:** **Java SE 7 Programmer I (1Z0-803)** **Java SE 7 Programmer II (1Z0-804)** **Upgrade to Java SE 7 Programmer (1Z0-805)** **Java SE 8 Programmer I (1Z0-808)** **Java SE 8 Programmer II (1Z0-809)**
Ihr Nutzen	**Dieser ZFI-Lehrgang richtet sich an Personen, welche die Sprache Java systematisch lernen möchten, um anschliessend Java-Applikationen in**

	ihrem beruflichen Umfeld zu entwickeln und zu warten. Lernziel ist selbstverständlich das Bestehen der von Oracle durchgeführten Prüfung zum Oracle Certified Professional Java Programmer II (1Z0_809).
Vorausetzungen	
Teilnehmerkreis	**Dieser ZFI-Lehrgang richtet sich an Programmierer, welche bereits erste Schritte in Java gemacht haben oder bereits in einer anderen Programmiersprache entwickelt haben und die Grundlagen der objektorientierten Software-Entwicklung bereits kennen.**
Unterlagen	**- Begleitbuch** **- Tutorials** **- Intranet Site**
Folgekurse	
Inhalt	**Oracle Certified Professional, Java SE 8 Programmer II** **- Java Class Design** **- Implement encapsulation** **- Implement inheritance including visibility modifiers and composition** **- Implement polymorphism** **- Override hashCode, equals, and toString methods from Object class** **- Create and use singleton classes and immutable classes** **- Develop code that uses static keyword on initialize blocks, variables, methods, and classes** **- Advanced Java Class Design** **- Develop code that uses abstract classes and methods** **- Develop code that uses final keyword** **- Create inner classes including static inner class, local class, nested class, and anonymous inner class** **- Use enumerated types including methods, and constructors in an enum type** **- Develop code that declares, implements and/or extends interfaces and use the @Override annotation.**

- Create and use Lambda expressions

- Generics and Collections

- Create and use a generic class

- Create and use ArrayList, TreeSet, TreeMap, and ArrayDeque objects

- Use java.util.Comparator and java.lang.Comparable interfaces

- Collections Streams and Filters

- Iterate using forEach methods of Streams and List

- Describe Stream interface and Stream pipeline

- Filter a collection by using lambda expressions

- Use method references with Streams

- Lambda Built-in Functional Interfaces

- Use the built-in interfaces included in the java.util.function package such as Predicate, Consumer, Function, and Supplier

- Develop code that uses primitive versions of functional interfaces

- Develop code that uses binary versions of functional interfaces

- Develop code that uses the UnaryOperator interface

- Java Stream API

- Develop code to extract data from an object using peek() and map() methods including primitive versions of the map() method

- Search for data by using search methods of the Stream classes including findFirst, findAny, anyMatch, allMatch, noneMatch

- Develop code that uses the Optional class

- Develop code that uses Stream data methods and calculation methods

- Sort a collection using Stream API

- Save results to a collection using the collect method and group/partition data using the Collectors class

- Use flatMap() methods in the Stream API

- Exceptions and Assertions

- Use try-catch and throw statements

- Use catch, multi-catch, and finally clauses

- Use Autoclose resources with a try-with-resources statement

- Create custom exceptions and Auto-closeable resources

- Test invariants by using assertions

- Use Java SE 8 Date/Time API

- Create and manage date-based and time-based events including a combination of date and time into a single object using LocalDate, LocalTime, LocalDateTime, Instant, Period, and Duration

- Work with dates and times across timezones and manage changes resulting from daylight savings including Format date and times values

- Define and create and manage date-based and time-based events using Instant, Period, Duration, and TemporalUnit

- Java I/O Fundamentals

- Read and write data from the console

- Use BufferedReader, BufferedWriter, File, FileReader, FileWriter, FileInputStream, FileOutputStream, ObjectOutputStream, ObjectInputStream, and PrintWriter in the java.io package.

- Java File I/O (NIO.2)

- Use Path interface to operate on file and directory paths

- Use Files class to check, read, delete, copy, move, manage metadata of a file or directory

- Use Stream API with NIO.2

- Java File I/O (NIO.2)

- Use Path interface to operate on file and directory paths

- Use Files class to check, read, delete, copy, move, manage metadata of a file or directory

- Use Stream API with NIO.2

- Java Concurrency

- Create worker threads using Runnable, Callable and use an ExecutorService to concurrently execute tasks

- Identify potential threading problems among deadlock, starvation, livelock, and race conditions

- Use synchronized keyword and java.util.concurrent.atomic package to control the order of thread execution

- Use java.util.concurrent collections and classes including CyclicBarrier and CopyOnWriteArrayList

- Use parallel Fork/Join Framework

- Use parallel Streams including reduction, decomposition, merging processes, pipelines and performance.

- Building Database Applications with JDBC

- Describe the interfaces that make up the core of the JDBC API including the Driver, Connection, Statement, and ResultSet interfaces and their relationship to provider implementations

- Identify the components required to connect to a database using the DriverManager class including the JDBC URL

- Submit queries and read results from the database including creating statements, returning result sets, iterating through the results, and properly closing result sets, statements, and connections

- Localization

- Read and set the locale by using the Locale object

- Create and read a Properties file

- Build a resource bundle for each locale and load a resource bundle in an application

Beitrag	**Der Teilnehmerbeitrag versteht sich rein netto. Das ZFI ist (gemäss MwSt-Gesetz) nicht Mehrwertsteuerpflichtig und erhebt somit keine MwSt. Bei länger als einen Monat dauernden Lehrgängen ist die Zahlung des Teilnehmerbeitrages in mehreren Raten möglich (pro rata temporis).**