ProTech protechtraining.com *"Charting the Course ... ... to Your Success!"*

Java Bootcamp

Course Summary

Description

This Java Bootcamp course serves as an introduction to the Java language and object oriented programming (OOP) in Java. The course provides students with the skills for analyzing, designing, developing, and troubleshooting Java applications. The participants learn the syntax and the constructs of the Java programming language, the concepts behind object-oriented programming (OOP) with Java, packaging, Java documentation, exception handling, Java libraries (I/O, utilities, networking, JDBC, etc.), concurrent programming with Java threads, and design patterns in Java.

Objectives

By the end of this course, students will be able to:

- Explain what Java is, what it is composed of, how it compares to other programming environments, what its advantages are, and how to install and configure the development environment.
- Read as well as write Java syntax, including declarations, assignments, operators, flow-control structures, generics, annotations, enumerations, naming conventions, etc.
- Understand object oriented programming principles, explain how OOP differs from structural programming, and discuss the advantages of OOP
- Utilize OOP in Java by designing and writing Java classes, encapsulating logic, reusing existing code through inheritance/polymorphism and composition, and modeling real-world relationships between objects
- Package and organize Java code into classes and libraries (JARs)
- Read, understand, and write Java code documentation (JavaDoc)
- Define and handle error conditions in Java through the use of exceptions
- Leverage Java libraries (I/O, Utilities, Collections, Networking, JDBC, etc.)
- Analyze and troubleshot complex Java programs
- Use best-practice design patterns when developing Java code

Topics

- About Java
- HelloWorld
- Data Types
- Operators
- Flow Control
- Object Oriented Programming
- Packaging
- JavaDoc
- java.lang library (Core Java)
- java.io library (I/O in Java)
- java.util library (Collections and Utilities)
- java.net library (Java Networking)
- java.sql library (JDBC)
- Java 5
- Design Patterns

Audience

Java Bootcamp is intended for individuals who wish to learn how to design, build, and debug Java applications. This includes software developers, quality assurance engineers, and other individuals with a programming background such as technical managers and system administrators.

Prerequisites

This course assumes students have a programming background, with prior exposure to other structured programming languages such as C or Python. As this course focuses on the advantages of object-oriented style in software engineering, students with recent experience developing and maintaining large software codebases will benefit from this course the most.

Duration

Five days

ProTech protechtraining.com *"Charting the Course ... ... to Your Success!"*

Java Bootcamp

Course Outline

I. About Java
- A. History of Java
- B. What is Java?
- C. Why Java?
- D. State of Java today

II. HelloWorld
- A. Installing and configuring Java Virtual Machine (lab)
- B. Implementing HelloWorld in Java (lab)
- C. Java class/file structure and naming conventions
- D. Java keywords and identifiers
- E. Compiling and running Java programs
- F. Comments in Java code
- G. The main() method
- H. Installing and configuring the Eclipse IDE

III. Data Types
- A. Declaring and assigning variables
- B. Primitive Java types
- C. Conversion between types
- D. Introduction to arrays and strings

IV. Operators
- A. Arithmetic operators - including shortcut operators
- B. Relational operators
- C. Logical-boolean operators
- D. Bitwise operators
- E. Assignment operators
- F. Additional operators
- G. Operator precedence

V. Flow Control
- A. Local variable storage: stack
- B. Branching statements: if-else and switch
- C. Loop statements: while and for
- D. Break and continue statements - including labeled
- E. Return statement
- F. **Lab:** Calculator

VI. Object Oriented Programming
- A. What is OOP?
- B. Why OOP?
- C. Class vs. Object
- D. OOP in Java: classes, fields, objects, methods
- E. Java memory model and garbage collection
- F. Static vs. instance data and methods
- G. Constructors - including constructor and method overloading
- H. Constants
- I. Encapsulation through access modifiers
  1. **Lab:** BankAccount
- J. Inheritance / in Java
  1. Types and subtypes
  2. **Lab:** School
- K. Interfaces and abstract classes / in Java
  1. **Lab:** OOP Calculator
- L. java.lang.Object: super class of them all
  1. Object Equality: equals() and hashCode() methods
  2. Converting objects to strings: toString() method
  3. **Lab:** Comparing students

VII. Packaging
- A. Reasons for packaging code
- B. Packages and sub-packages in Java
- C. Protecting package namespace
- D. Using packaged code
- E. Protecting packaged code
- F. Java CLASSPATH
- G. Java Archive (JAR)
- H. **Lab:** Packaging

VIII. JavaDoc
- A. Overview of JavaDoc
- B. Java API
- C. Defining and generating JavaDoc
- D. **Lab:** Documenting Java code

Java Bootcamp

Course Outline (cont'd)

IX. Exception Handling
- A. What are exceptions?
- B. Why exceptions?
- C. Built-in exceptions
- D. Exception life-cycle
- E. Handling exceptions
- F. Throwing exceptions
- G. Exception types: checked vs. unchecked
- H. Creating new exceptions
- I. Grouping and nesting exceptions
- J. **Lab:** Exceptions

X. java.lang library (Core Java)
- A. Primitive wrappers
- B. String and StringBuffer/StringBuilder
- C. java.lang.Math
- D. java.lang.System
- E. Multi-threaded programming in Java

XI. java.io library (I/O in Java)
- A. Managing files
- B. Byte and character streams
- C. Filtered streams
- D. Object serialization

XII. java.util library (Collections and Utilities)
- A. Collections Framework
  1. java.util.Collection
  2. java.util.Iterator
  3. java.util.List
  4. java.util.Set
  5. java.util.Queue
  6. java.util.Map
- B. PRNG in Java: java.util.Random
- C. String parsing and matching - including java.util.regex
- D. Date, Calendar, TimeZone

XIII. java.net library (Java Networking)
- A. java.net.InetAddress and java.net.NetworkInterface
- B. java.net.URL connections
- C. TCP sockets (java.net.Socket and java.net.ServerSocket)

XIV. java.sql library (JDBC)
- A. Overview of JDBC and its drivers
- B. JDBC API: connections, statements, result sets, metadata
- C. Using JDBC: updates, queries

XV. Java 5
- A. Generics
- B. Enhanced for[each] loop
- C. Auto boxing and unboxing
- D. Typesafe enums
- E. Varargs
- F. Static imports
- G. Annotations (metadata)

XVI. Design Patterns
- A. What are Design Patterns?
- B. Singleton, Factory Method, Abstract Factory
- C. Adapter, Composite, Decorator
- D. Chain of Responsibility, Observer / Publish-Subscribe, Strategy, Template
- E. Data Access Object (DAO)