

## Java Programming

(5 days or 10 nights Instructor-Led Course)

### Description

This hands-on course introduces programmers to Sun's Java™ technology and Java programming techniques. The Java platform provides an object-oriented, portable and robust framework for application development.

### What you will accomplish

You will learn how to:

- Write, compile and execute Java programs
- Build robust applications using Java's object-oriented features
- Create robust applications using Java class libraries
- Develop platform-independent GUIs
- Read and write data using Java streams
- Retrieve data from a relational database with JDBC

### Who Should Attend

The course is designed to leverage the participants' existing programming skills and to highlight the new and extended features of the Java programming framework as compared to other common languages. Comprehensive lab exercises using the Java Development Kit provide hands on practice crucial to developing competence and confidence with the new skills being learned.

### Prerequisites

Basic programming skills in a structured language. Knowledge and experience with Object-Oriented Design (OOD) is helpful, but not required.

### Certification Preparation

Sun Certified Java Programmer (SCJP 1.4 Certification)

### Following course

Java for Web Application Development

### Course material:

- Complete Java 2 Certification Study Guide
- Core Java(TM) 2, Volume II--Advanced Features (7th Edition)
- Borland JBuilder 2005 Foundation , Eclipse Development Environment

## Java Programming Course Overview Getting Started

- Describe the key features of Java technology
- Write, compile, and run a simple Java technology application
- Describe the function of the Java Virtual Machine (JVM)  
NOTE: The terms "Java Virtual Machine" and "JVM" mean a Virtual Machine for the Java platform.
- Define garbage collection
- List the three tasks performed by the Java platform that handle code security

## Identifiers, Keywords, and Types

- Use comments in a source program
- Distinguish between valid and invalid identifiers
- Recognize Java technology keywords
- List the eight primitive types
- Define literal values for numeric and textual types
- Define the terms primitive variable and reference variable
- Declare variables of class type
- Construct an object using new
- Describe default initialization
- Describe the significance of a reference variable
- State the consequence of assigning variables of class type

## Expressions and Flow Control

- Distinguish between instance and local variables
- Describe how to initialize instance variables

## Object-Oriented Programming

- Define modeling concepts: abstraction, encapsulation, and packages
- Discuss why you can reuse Java technology application code
- Define class, member, attribute, method, constructor, and package
- Use the access modifiers private and public as appropriate for the guidelines of encapsulation
- Invoke a method on a particular object
- Use the Java technology application programming interface (API) online documentation

## Identifiers, Keywords, and Types

- Use comments in a source program
- Distinguish between valid and invalid identifiers
- Recognize Java technology keywords
- List the eight primitive types
- Define literal values for numeric and textual types
- Define the terms primitive variable and reference variable
- Declare variables of class type
- Construct an object using new
- Describe default initialization
- Describe the significance of a reference variable
- State the consequence of assigning variables of class type**

## Arrays

- Declare and create arrays of primitive, class, or array types
- Explain why elements of an array are initialized

- Identify and correct a Possible reference before assignment compiler error
- Recognize, describe, and use Java software operators
- Distinguish between legal and illegal assignments of primitive types
- Identify Boolean expressions and their requirements in control constructs
- Recognize assignment compatibility and required casts in fundamental types
- Use if, switch, for, while, and do constructions and the labeled forms of break and continue as flow control structures in a program
- Explain how to initialize the elements of an array
- Determine the number of elements in an array
- Create a multidimensional array
- Write code to copy array values from one array to another

### Class Design

- Define inheritance, polymorphism, overloading, overriding, and virtual method invocation
- Use the access modifiers protected and the default (package-friendly)
- Describe the concepts of constructor and method overloading
- Describe the complete object construction and initialization operation

### Exceptions and Assertions

- Define exceptions
- Use try, catch, and finally statements
- Describe exception categories
- Identify common exceptions
- Develop programs to handle your own exceptions
- Use assertions
- Distinguish appropriate and inappropriate uses of assertions
- Enable assertions at runtime

### Advanced Class Features

- Create static variables, methods, and initializers
- Create final classes, methods, and variables
- Create and use enumerated types
- Use the static import statement
- Create abstract classes and methods
- Create and use an interface

### Text-Based Applications

- Write a program that uses command-line arguments and system properties
- Write a program that reads from standard input
- Describe the C-type formatted input and output
- Write a program that can create, read, and write files
- Describe the basic hierarchy of collections in Java 2 Software Development Kit (Java 2 SDK)
- Write a program to iterate over a collection
- Write a program that uses generic collections

### Building Java GUIs

- Describe the Abstract Windowing Toolkit (AWT) package and its components
- Define the terms containers, components, and layout managers, and describe how they work together to build a GUI
- Use layout managers
- Use the Flow Layout, Border Layout, and Grid Layout managers to achieve a desired dynamic layout
- Add components to a container
- Use the Frame and Panel containers appropriately
- Describe how complex layouts with nested containers work

### GUI-Based Applications

- Identify the key AWT components and the events that they trigger
- Describe how to construct a menu bar, menu, and menu items in a Java GUI
- Understand how to change the color and font of a component

### Threads

- Define a thread
- Create separate threads in a Java technology program, controlling the code and data that are used by that thread
- Control the execution of a thread and write platform-independent code with threads

### GUI Event Handling

- Define events and event handling
- Write code to handle events that occur in a GUI
- Describe the concept of adapter classes, including how and when to use them
- Determine the user action that originated the event from the event object details
- Identify the appropriate listener interface for a variety of event types
- Create the appropriate event handler methods for a variety of event types
- Understand the use of inner classes and anonymous classes in event handling

### Introduction to Databases and JDBC

- Purpose of JDBC
- JDBC Software Architecture
- JDBC and SQL
- The Java.Sql Package
- SQL Basic Statements: Select, Insert, Update, and Delete
- SQL Exceptions
- Advanced SQL Statements: Stored Procedures, Prepared Statements, and MetaData
- JDBC Drivers and Driver Types
- JDBC-ODBC Bridge
- Native API Drivers
- Date and Timestamp

### Advanced I/O Streams

- Describe the main features of the java.io package
- Construct node and processing streams, and use them appropriately
- Distinguish readers and writers from streams, and select appropriately between them

- Describe the difficulties that might arise when multiple threads share data
- Use wait and notify to communicate between threads
- Use synchronized to protect data from corruption

### Networking

- Develop code to set up the network connection
- Understand the Transmission Control Protocol/Internet Protocol (TCP/IP)
- Use ServerSocket and Socket classes for implementation of TCP/IP clients and servers