Java is one of the most important programming languages used today, and its correct usage is a critical part of many systems being written now. The basics of the language are relatively easy, but the challenge lies in learning how to use it well. This is especially true regarding the Object-Oriented nature of Java, which for many developers is a new way of approaching system design and construction.

In addition to teaching you everything you need to become productive in basic Java programming, this course draws on our extensive experience to provide a solid understanding of the core OO and Java concepts and practices needed to create well designed Java programs. It covers all the key OO capabilities and how to use them in Java, including material on creating well designed Java classes, using encapsulation, building more complex functionality with composition, and using inheritance to share functionality and create specialized types. It also covers more advanced concepts such as using interfaces, working with the Java Collections Framework, and accessing databases with JDBC. This course is suitable for environments using either Java 5 or Java 6. The material includes coverage of all the new Java capabilities, such as Generic Collection classes, and clearly points out which capabilities are supported in which Java release. All labs are doable in any of the supported Java environments. This course is available for most IDEs, such as Eclipse and IBM RAD.

**Audience:** Programmers moving into object-oriented programming using Java.

**Prerequisites:** Previous programming experience in C, C++, or C#®, and some knowledge of basic object-oriented concepts are recommended.

**Number of Days:** 5 days

---

1. **A Simple Java Class and Running a Java Program**
   - A Simple Application Class
   - The HelloWorld Program Broken Down
   - Compiling HelloWorld

2. **Java Overview**
   - Language and Platform Features
   - What is Java?
   - Java is Modern and Object-Oriented
   - Java is Portable and Safe
   - Java has Multiple Platforms
   - History of Java’s Naming and Versioning
   - Program Life Cycle
   - Java Source and Java Bytecode
   - Life Cycle of a Java Program
   - Java Programs Insulated from Environment
   - Java is Dynamic – The Runtime Process

3. **Class and Object Basics**
   - The Java SE Software Development Kit (JDK)
   - The Java API
   - Downloading and Installing the JDK
   - Class and Object Overview
   - What is Object-Oriented Programming?
   - What is an Object?
   - Important Characteristics of Objects
   - About Object-Oriented Programming (OOP)
   - What’s a Type?
   - Types, Instances, and Property Values
   - Classes and Objects
   - Classes, References, and Instantiation
   - The Class in Java
   - Class Definition
   - A Class Definition is a Blueprint
   - Creating and Referencing Objects
   - More About Identifiers
Methods and Data in a Class
Behavior and Methods
Invoking Methods
Storing Data in Objects
About Instance Variables
Data Access and Return Values in Methods
Accessing Data (Another Way)
Pretty Pictures
More About Variables
About Java Primitive Data Types
Numeric Literals
Strings
Primitive Types are Value Types
Arithmetic Operations
Primitive Type Conversion and Casting

4. More on Classes and Objects
Working with Methods and Data
Working Within Methods
Local Variable
The this Variable and Instance Data
Pretty Pictures
Overloading Methods
Calling Overloaded Methods
The toString() Method
Encapsulation and Access Control
Encapsulation: Black Boxes
Encapsulation
Key Advantages of Encapsulation
Program Correctness
Access Control
Access for Data Members and Methods
Private Access
Public Access
Constructors
Using Constructors
Explicit Constructor Call
Static or Class Members
Static Members
Accessing Static Members
Static Methods
Accessing Data in Static Methods
Final Variables
Odds and Ends
Scopes and Blocks
Assignment

Comparison
Null Objects
Wrapper classes
Reference Types as Method Parameters
Final Method Parameters

5. Flow of Control
Branching Statements
Program Execution Sequence in Java
The Comparison Operators
The Logical Operators
If Statement
If-else Statement
Switch Statement
Iteration Statements
While Statement
Do-while Statement
For Statement
Break Statement
Continue Statement

6. Strings and Arrays
String and StringBuffer/StringBuilder
Using Strings
Changing Strings
Classes StringBuffer and StringBuilder
Using StringBuffer and StringBuilder
Regular Expressions
Arrays
Creating Arrays and Accessing Elements
Array length
Arrays of Class Types
Array of Strings
args Array
Iterating over Arrays

7. Packages
Packages Overview
Dealing with Complexity
Packages
Package Statement
The Default Package
Import Statement
The Import Statement
Importing a Complete Package
Importing a Single Package Member
Using the Fully Qualified Name
Standard Imports
Resolving Naming Conflicts
8. Composition and Inheritance
   Composition
   Dealing with Complexity and Composition
   Composition
   Delegation
   Benefits/Problems of Composition
   About Object Relationships
   The USES Relationship
   Other Kinds of Relationships
   Inheritance
   Inheritance and Dealing with Complexity
   Inheritance Hierarchy
   The extends Keyword
   Inheriting from the Superclass
   Inheritance and Superclass Data Members
   A Subclass is a Superclass
   Accessing Superclass Members
   Constructors and Inheritance
   Final Classes
   Overriding and Polymorphism
   Changing Behavior with Method Overriding
   OO Concepts – Polymorphism
   Polymorphism
   Importance of Polymorphism
   The super keyword
   Access Control – Protected Access
   Class Object
   Methods of Class Object
   Automatic Storage Management
   Abstract Classes
   Abstract Methods
   Using Abstract Classes

9. Interfaces
   What if All You Have to Share is an Idea?
   Interface Types
   Interface Definitions
   The implements Keyword
   Interface Types – Revisited
   Extending Interfaces
   Implementing Extended Interfaces
   Interfaces are Abstract
   Data Members in Interfaces
   Implementing Multiple Interfaces

10. Exceptions
    Overview of Exceptions
    Exception Hierarchy
    Exception, Error, RuntimeException
    Handling Exceptions with Try and Catch
    Exceptions and Program Flow
    Variable Scope
    The throws Clause
    Throwing Exceptions with throw
    User-Defined Exceptions
    Multiple catch Blocks
    Finally Block
    Runtime Exceptions

11. JDBA Java Database Connectivity
    JDBC Overview
    Relational Database Description
    Web-Based Data Access Architecture
    What is JDBC?
    JDBC Characteristics
    JDBC Specification and Packages
    JDBC Architecture and API
    The Fundamental JDBC API
    The DriverManager Class
    What is a JDBC Driver?
    Drivers Must be Loaded
    Loading a Driver
    Naming Databases with URLs
    JDBC URL Syntax
    Database Connections
    Establishing a Database Connection
    Closing the Connection
    Using finally to Close the Connection
    Issuing Statement and Processing Data
    Creating Statements
12. **Collections**
   - Java Collections Framework Overview
   - java.util Collection Interfaces
   - Collection Interface
   - List and ArrayList
   - List Interface
   - ArrayList
   - The for-each Construct
   - Generics and Autoboxing
   - Generics and Type-Safe Collections
   - Autoboxing
   - Summarizing New Collections Features
   - Other Collection Types
     - Set Interface
     - Map Interface
     - HashMap
     - Creating and Using HashMap
     - Iterating Through a HashMap
     - Generic HashMaps – Java 5
     - Iterator
     - Processing Items with an Iterator
     - Iterator Interface

13. **Additional Language Features**
    - Assertions (1.4)
    - Assertions Defined
    - Assertion uses
    - Assertion Non-Uses
    - Assertion Syntax
    - Using Assertions to Check Flow of Control
    - Assuring Their Presence in Compiled Code
    - Enabling/Disabling Assertions at Runtime
    - What They Look Like at Runtime
    - Type-Safe Enums (5.0)
    - Enumerated Types Defined
    - Problems with int Enumerated Types
    - The enum Keyword
    - Switch on enum
    - For-Each with enum

14. **I/O Streams**
    - Readers and Writers
    - Overview of I/O Streams
    - Character of Streams
    - Class Reader
    - Class Writer
    - Common Reader Subclasses
    - Common Writer Subclasses
    - High-Level and Low-Level Streams
    - Using Readers and Writers
    - Path Separators
    - Filter Streams
    - Filter Stream Base Classes
    - InputStreams and OutputStreams
    - Java 1.0 Byte Stream Classes
    - Common InputStream Subclasses
    - Common OutputStream Subclasses
    - Converting Between Streams and readers/Writers
      - high-Level and Low-Level Streams
      - Converting Between Byte and Character Streams
      - Character Stream and Byte Stream Equivalents
    - New I/O (NIO) APIs
    - New I/O (NIO)
    - NIO Features