Basic Java Programming for Developers New to OO (C, COBOL, etc.) - TT2120

Fast-track your Java and OO skills with hands-on coding, expert instruction, and a structured approach to object-oriented development.

Duration: 5 Days
Skill Level: Introductory
Available Format: Instructor-Led Online; Instructor-Led, Onsite In Person ; Blended; On Public Schedule

Shifting from procedural programming to object-oriented development is a critical step for developers looking to expand their skill set. This expert-led, hands-on course is designed for experienced developers who have worked with non-object-oriented languages and want to learn how to structure programs using Java's modern object-oriented features. You will learn the fundamentals of classes, objects, and inheritance, while also exploring key Java capabilities like records, pattern matching, and functional programming with lambda expressions. Through hands-on exercises, you will practice writing Java applications, working with collections and streams, handling exceptions, and managing dependencies. Whether you are transitioning from C, procedural Python, or another language, this course will give you the knowledge and confidence to write well-structured, scalable Java applications.

What You'll Learn

Overview

Object-oriented programming (OOP) is the foundation of modern software development, making code more structured, reusable, and maintainable. If you are an experienced developer in a procedural language like C, SQL, or procedural Python, **Basic Java Programming for Developers New to OO** will help you transition to the object-oriented mindset. This expert-led, hands-on course focuses on teaching you how Java organizes code using classes, objects, and methods, allowing you to build scalable applications with real-world best practices. You will learn not just how to write Java code but also

how to structure programs effectively, improving readability, maintainability, and long-term flexibility in your development projects.

Over five days, you will gain practical experience in designing and building Java applications while learning essential object-oriented concepts. You will start by writing simple Java programs and understanding how the Java Virtual Machine (JVM) executes your code. From there, you will explore how to create and manage objects, structure data using Java's collection framework, and apply key programming principles like inheritance and polymorphism to make your applications more dynamic and adaptable. Through guided exercises, you will see firsthand how Java's modern features—such as records, pattern matching, and lambda expressions—streamline development and reduce boilerplate code.

With 50% hands-on coding, this course ensures you are not just learning theory but actively applying it through structured labs and real-world examples. You will practice debugging, exception handling, and modular application design, gaining confidence in troubleshooting and optimizing your programs. By the end of the course, you will have the skills to write clean, efficient, and scalable Java applications, preparing you to work on object-oriented projects and collaborate more effectively in modern software development environments.

Objectives

Working in an interactive learning environment, led by our expert facilitator, you will learn to:

- **Master Object-Oriented Thinking**: Transition from procedural programming to object-oriented design, learning how to create reusable, scalable, and maintainable code structures.
- **Write Java Applications with Confidence**: Build practical skills in coding Java applications from scratch, leveraging modern tools and best practices for effective development.
- **Work Efficiently with Data**: Use Java's powerful data structures, such as collections and streams, to organize, process, and manipulate data efficiently.
- **Solve Real-World Problems**: Apply exception handling, inheritance, and polymorphism to create robust solutions for common programming challenges.
- **Leverage Modern Java Features**: Explore and utilize cutting-edge Java 21 features like records, sealed classes, and pattern matching to streamline development.
- **Develop Hands-On Coding Expertise**: Gain practical experience through engaging coding exercises, ensuring you leave with skills you can immediately apply on the job.

Specific Java 17 features that are covered in the course include:

- Switch Expressions
- Text blocks
- Pattern matching for instanceof
- Introduce records as carrier of immutable data

Specific Java 21 features that are covered in the course include:

- Sequenced Collections
- Pattern matching in Switch statements
- Record Patterns

If your team requires different topics, additional skills or a custom approach, our team will collaborate with you to adjust the course to focus on your specific learning objectives and goals.

Audience

This course is designed for developers with experience in procedural programming or scripting languages, such as C, Python, or shell scripting, who want to transition to object-oriented programming. It's ideal for roles like software engineers, application developers, or technical professionals seeking to build scalable, maintainable applications using Java. Whether you're a seasoned programmer looking to expand your skillset or a technical specialist aiming to adopt modern programming practices, this course provides the foundation you need to excel in object-oriented development.

Pre-Requisites

This course is designed for **experienced developers** who have worked with non-object-oriented programming languages and want to learn Java as their first object-oriented language. It is ideal for **C developers, SQL specialists, procedural Python programmers, and engineers working with legacy systems** who need to transition to Java's modern programming model.

Recommended Skills Before Attending:

- Experience **writing and troubleshooting code** in a procedural or scripting language.
- Familiarity with **basic programming concepts**, such as loops, functions, and conditionals.
- Ability to **work with an IDE or command-line tools** to compile and run programs.

Agenda

Please note that this list of topics is based on our standard course offering, evolved from typical industry uses and trends. We'll work with you to tune this course and level of coverage to target the skills you need most. Topics, agenda and labs may adjust during live delivery based on audience skill-level, needs and participation.

1. The Java Platform

- Introduce the Java Platform
- Explore the Java Standard Edition
- Discuss the lifecycle of a Java Program
- Explain the responsibilities of the JVM
- Executing Java programs
- Garbage Collection

2. Using the JDK

- Explain the JDK's file structure
- Use the command line compiler to compile a Java class
- Use the command line Java interpreter to run a Java application class
- Documentation and Code Reuse

3. Using the IntelliJ IDE

- Introduce the IntelliJ IDE
- The Basics of the IntelliJ interface
- IntelliJ Projects and Modules
- Creating and running Java applications

4. Writing a Simple Class

- Write a Java class that does not explicitly extend another class
- Define instance variables for a Java class
- Create object instances
- Primitives vs Object References
- Implement a main method to create an instance of the defined class
- Java keywords and reserved words

5. Adding Methods to the Class

- Write a class with accessor methods to read and write instance variables
- Write a constructor to initialize an instance with data
- Write a constructor that calls other constructors of the class to benefit from code reuse

- Use the this keyword to distinguish local variables from instance variables
- Introducing annotations
- Deprecating classes and methods

6. Object-Oriented Programming

- Real-World Objects
- Classes and Objects
- Object Behavior
- Methods and Messages

7. Language Statements

- Arithmetic operators
- Operators to increment and decrement numbers
- Comparison operators
- Logical operators
- Return type of comparison and logical operators
- Use for loops
- Switch Expressions
- Switch Expressions and yield

8. Using Strings and Text Blocks

- Create an instance of the String class
- Test if two strings are equal
- Perform a case-insensitive equality test
- Contrast String, StringBuffer, and StringBuilder
- Compact Strings
- Text Blocks
- Unicode support

9. Fields and Variables

- Discuss Block Scoping Rules
- Distinguish between instance variables and method variables within a method
- Explain the difference between the terms field and variable
- List the default values for instance variables
- Final and Static fields and methods

10. Specializing in a Subclass

- Constructing a class that extends another class
- Implementing equals and toString

- Writing constructors that pass initialization data to parent constructor
- Using instanceof to verify type of an object reference
- Overriding subclass methods
- Pattern matching for instanceof
- Safely casting references to a more refined type

11. Using Arrays

- Declaring an array reference
- Allocating an array
- Initializing the entries in an array
- Writing methods with a variable number of arguments

12. Records

- Data objects in Java
- Introduce records as carrier of immutable data
- Defining records
- The Canonical constructor
- Compact constructors

13. Java Packages and Visibility

- Use the package keyword to define a class within a specific package
- Discuss levels of accessibility/visibility
- Using the import keyword to declare references to classes in a specific package
- Using the standard type naming conventions
- Introduce the Java Modular System
- Visibility in the Java Modular System

14. Utility Classes

- Introduce the wrapper classes
- Explain Autoboxing and Unboxing
- Converting String representations of primitive numbers into their primitive types
- Defining Enumerations
- Using static imports
- Introduce the Date/Time API
- LocalDate / LocalDateTime etc.
- Apply text formatting
- Using System.out.printf

15. Inheritance and Polymorphism

- Write a subclass with a method that overrides a method in the superclass
- Group objects by their common supertype
- Utilize polymorphism
- Cast a supertype reference to a valid subtype reference
- Use the final keyword on methods and classes to prevent overriding

16. Interfaces and Abstract Classes

- Define supertype contracts using abstract classes
- Implement concrete classes based on abstract classes
- Define supertype contracts using interfaces
- Implement concrete classes based on interfaces
- Explain advantage of interfaces over abstract classes
- Explain advantage of abstract classes over interfaces

17. Sealed Classes

- Introduce sealed classes
- The sealed and permits modifier
- Sealed interfaces
- Sealed classes and pattern matching

18. Pattern Matching

- Pattern Matching in switch statements
- Pattern Matching and sealed classes
- Record Patterns

19. Introduction to Exception Handling

- Introduce the Exception architecture
- Defining a try/catch blocks
- Checked vs Unchecked exceptions

20. Exceptions

- Defining your own application exceptions
- Automatic closure of resources
- Suppressed exceptions
- Handling multiple exceptions in one catch
- Enhanced try-with-resources
- Helpful NullPointerException(s)

21. Building Java Applications

- Explain the steps involved in building applications
- Define the build process
- Introduce build scripts
- Explain the standard folder layout
- Resolving project dependencies

22. Introduction to Generics

- Generics and Subtyping
- Bounded Wildcards
- Generic Methods

23. Introducing Lambda Expressions and Functional Interfaces

- Understanding the concept of functional programming
- Understanding functional interfaces
- Lambda's and type inference

24. Collections

- Provide an overview of the Collection API
- Review the different collection implementations (Set, List and Queue)
- Explore how generics are used with collections
- Examine iterators for working with collections
- Sequenced Collections

25. Using Collections

- Collection Sorting
- Comparators
- Using the Right Collection
- Lambda expressions in Collections
- Sequenced Sets

26. Streams

- Understanding the problem with collections in Java
- Thinking of program solutions in a declarative way
- Use the Stream API to process collections of data
- Understand the difference between intermediate and terminal stream operations
- Filtering elements from a Stream
- Finding element(s) within a Stream
- Collecting the elements from a Stream into a List

27. Collectors

- Using different ways to collect the items from a Stream
- Grouping elements within a stream
- Gathering statistics about numeric property of elements in a stream

Bonus Topics: Time Permitting

These topics will be included in your course materials but may or may not be presented during the live class depending on the pace of the course and attendee skill level and participation.

28. Introduction to Annotations

- Discuss how annotations work in Java
- Understand what is required to work with Java's annotations
- Use annotations
- Other technologies that are using annotations

29. Java Data Access JDBC API

- Connecting to a database using JDBC
- Executing a statement against a database that returns a ResultSet
- Setting up and working with PreparedStatements
- Extracting multiple rows of data from a ResultSet
- Inserting, updating and deleting rows in a table

Follow On Courses

TT3503	Test Driven Development (TDD) and Unit Testing Essentials
TT3320	Core Spring Quick Start | Introduction Spring 6.x and Spring Boot
TT3335	Mastering Spring 5.x Developer Boot Camp
TTAI2300	Quick Start to Prompt Engineering for Software Developers
TTAI2305	Turbocharge Your Code! Generative AI Boot Camp for Developers
TT2211	Intermediate Java Programming | Next-Level Java Skills

Related Courses

TT2136	Migrating Java 11 to Java 21 | Java 21 New Features and Skills
TEST-01	Mastering Machine Learning Operations (MLOps) and AI Security Boot Camp
TT2100	Core Java Programming Developer's Workshop

TT2104	Fast Track to Core Java Programming for OO Experienced Developers
TT2000	Getting Started with Programming, OO & Java Basics for Non-Developers

Setup Made Simple! All of our AI for Business course software, digital course files or course notes, labs, data sets and solutions, live coaching support channels and rich extended learning and post training resources are provided for you in our easy access, single source, no install required online Learning Experience Platform (LXP), remote lab and content environment. Or we can provide a local installation (trial edition) to setup and use on your machine. Access periods and versions vary by course. Please inquire about set up details and options for your specific course of interest. Regardless of setup option, we will collaborate with you to ensure your team is set up and ready to go well in advance of the class.

Ways to Learn: At Trivera, we believe that Experience is Everything. Our customizable, hands-on courses are delivered live online, onsite, or in a blended format for maximum flexibility. We provide real-time expert-led training and coaching for all skill levels, from small groups to enterprise-wide programs, ensuring every learner gains the latest, most relevant job-ready skills they can apply with confidence. This course is also available for individuals or small groups on our extensive Public Schedule (see current dates below). We look forward to helping you take the next steps in your modern web developer learning journey.

For More Information

Please contact us or call 844-475-4559 toll free for more information about our training services (instructor-led, self-paced or blended), coaching and mentoring services, public course enrollment or questions, partner programs, courseware licensing options and more.