Java Developer SkillJourney Series

Getting Started with Programming, OO and Java Basics for Non-Developers (TT2000)

Learn to Think Like a Programmer: Jumpstart your Java coding skills in this engaging, skill-focused programming course

Course Snapshot

- **Course: Getting Started with Programming, OO and Java Basics for Non-Developers (TT2000)**
- **Duration**: 5 days
- **Audience & Skill Level:** This technical course is designed for anyone new to coding and eager to learn how to program using Java. While it's tailored for beginner-level students, please note that it is technical in nature. Students should have an IT background and have plans to become a software developer.
- **Hands-on:** This hands-on course combines engaging instructor-led presentations and practical demonstrations with extensive programming exercises, challenge labs, use case exploration and engaging group activities. Student machines are required.
- **Focus**: This course uses Java 21, which also covers the fundamental concepts and techniques in Java 11 and 17. This course is suited for Java 11, Java 17 and Java 21 skills development. Earlier versions available. Please inquire for options.
- **Format**: This course can be delivered for your team or organization **online live (virtual)**, **onsite in-person** or across our robust **blended learning platform (LXP).**
- **Public Schedule**: This course is currently available on our Public Open Enrollment Schedule.
- **Customizable**: This course agenda, topics, labs, hours and delivery modalities can be adjusted to target your specific training skills objectives, tools and learning goals. Please ask for details.

Overview

Getting Started with Programming, OO and Java Basics for Non-Developers is a fast-paced, five day course designed to provide you with a first look at how to code in Java to a very basic level. You'll gain light hands-on programming experience, while you begin your journey to develop a programmer's mindset.

Throughout the course you'll explore the intricacies of the application development cycle, program structure, and language syntax. You'll learn and practice core coding skills including fundamental OO concepts, vital programming constructs, string and character manipulation, dynamic memory allocation, standard input/output, and exception handling. You'll also explore the power of inheritance and polymorphism as you learn to define specialized object implementations. The course also covers processing command line arguments and environment variables, empowering you to set up your own development environment to create flexible, user-friendly programs. With a wealth of hands-on lab exercises, you'll practice and refine your newly acquired skills.

Becoming a modern software developer is like learning a new language; it requires study, practice, and dedication well beyond this course to apply your new skills effectively. While this five day program won't transform you into an experienced developer, it will lay a solid foundation in coding basics using Java, while teaching you to think like a programmer. Although this course is technical in nature, our instructors will guide you every step of the way, providing a supportive environment for you to explore, ask questions, and prepare for your next learning milestones.

NOTE: Although this course is geared for non-developers, it is helpful for attendees to have a somewhat technical background and to be comfortable working with computers, having the ultimate goal of becoming a Java software developer. This course uses Java 21, which also covers the fundamental concepts and techniques in Java 11 and 17. This course is suited for Java 11, Java 17 and Java 21 skills development. Earlier versions are available. Please inquire for options.

Learning Objectives

With our supportive instructors, you'll enjoy a safe environment to explore, ask questions, and grow, leaving you confident and ready to continue your exciting learning journey. Working in a hands-on learning environment, guided by our expert

team, you'll explore:

- The basic programming constructs that all programming languages share
- Fundamental programming concepts, such as variables, data types, loops, and conditional statements.
- Object-oriented programming principles, including encapsulation, inheritance, and polymorphism.
- How to handle problems that might occur during the execution of a Java application.
- How to use Java libraries and APIs for common tasks, such as file I/O, data manipulation, and networking.
- Implementing exception handling and debugging techniques to ensure robust and reliable code.
- Utilize industry-standard tools and Integrated Development Environments (IDEs) to efficiently write, test, and deploy Java applications.
- Best practices for code organization, documentation, and version control to enhance collaboration and maintainability.

Audience

This course is designed for anyone new to coding and eager to learn how to program using Java. While it's tailored for beginner-level students, please note that it is technical in nature. **If you're transitioning from a non-technical role to coding for the first time, feel free to reach out to us for additional guidance or pre-training suggestions that can better prepare you for this course.** Our goal is to make your learning experience exciting, challenging, and valuable, without being overwhelmed.

Attendees might include:

- Technically minded attendees who want or who want to begin the process of becoming an OO application developer
- Technical team members from non-development roles, re-skilling to move into software and application development roles within an organization
- Recent college graduates looking to apply their college experience to programming skills in a professional environment, or perhaps needing to learn the best practices and standards for programming within their new organization
- Technical managers tasked with overseeing programming teams, or development projects, where basic coding knowledge and exposure will be useful in project oversight or communications needs

Pre-Requisites

- Basic computer literacy: Familiarity with computer operating systems, file management, and general navigation to ensure a smooth learning experience.
- Foundational knowledge of IT concepts: Understanding of essential IT terminologies and concepts, such as computer networks, software applications, and data storage.
- Analytical thinking: Ability to analyze problems and think critically to develop logical solutions, fostering a programmer's mindset.
- Attention to detail: A keen eye for detail, ensuring the ability to spot errors and maintain code quality throughout the learning process.

Related Courses | Java Essentials Suite

The following is a subset of our related basic Java Foundation courses. Please visit www.triveratech.com for the full catalog and complete list.

- TT2000 Getting Started with Programming, OO and Java Basics for Non-Developers (5 days)
- TT2103 JumpStart to Java for OO Developers (such as C++, C#) (3 days)
- TT2104 Fast Track to Java Programming for Experience OO Developers (such as C++, C#) (4 days)
- TT2120 Basic Java Programming for Developers New to OO (such a C, Mainframe) (5 days)
- TT2135 Migrating from Java 11 to Java 17 (1 day)
- TT2136 Migrating from Java 11 to Java 21 (1 day)
- TT2010 Java Developer SkillJourney – Career Experience Program (for Core Java) (multi-week)
- TT2015 Java Full Stack Developer SkillJourney – Career Experience (for full stack / front and back end) (multi-week)

Next Steps / Follow-on Courses: We offer a wide variety of follow-on courses for next-level Java skills, Java for Web / Full Stack, Jakarta / Java EE, Spring, REST, Microservices, Unit Testing / TDD, Java secure coding, mobile development and more. Please see our **Java Developer Courses, Learning Paths & SkillJourneys** for options based on your specific role and goals.

Enhanced Learning Services: Please also ask about our robust Learning Experience Platform (LXP), Skills Assessment & Skills Prep Services, Skills Immersion Programs & Camps, Coaching and Mentoring Services and Extended Learning Support programs.

Setup Made Simple! All applicable course software, digital courseware files or course notes, lab environment, data sets and solutions, live coaching support channels and rich extended learning and post training resources are provided for you in our "easy access, no install required" online **Learning Experience Platform (LXP)**. We'll collaborate with you to ensure your team is set up and ready to go well in advance of the class. Please inquire about set up details and options for your specific course of interest.

Course Topics / 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. **Overview of Computer Programming**
- Explain what a program is
- Explain why there are different types of languages
- Explain what a compiler is
- Explain what an interpreter is

2. **Features of a Program**
- Understand what the entry and exit points of an application are
- Explain what variables are
- Explain what programming instructions are
- Explain what errors and exceptions are
- Understand what programming algorithms are

3. **Software Development Life Cycle**
- Explain the purpose of the software development life cycle
- Explain what each phase is for
- Explain the difference between the software development life cycle and a methodology

4. **Thinking in Objects**
- Understand the difference between a class and an object
- Deconstruct an object into attributes and operations
- Map an object to a class
- Define inheritance

5. **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
- Documentation and Code Reuse

6. **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

7. **Using the IntelliJ IDE**
- Introduce the IntelliJ IDE
- The Basics of the IntelliJ interface
- IntelliJ Projects and Modules
- Creating and running Java applications
- Tutorial: Working with your IDE IntelliJ 2023.2 (Community Edition) or Eclipse

8. **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

9. **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

10. **Object-Oriented Programming**
- Real-World Objects
- Classes and Objects
- Object Behavior
- Methods and Messages

11. **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

12. 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

13. 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

14. 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

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

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

17. 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

18. 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

19. 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

20. 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

21. Introduction to Exception Handling
- Introduce the Exception architecture
- Defining a try/catch blocks
- Checked vs Unchecked exceptions

22. Introduction to Generics and Collections
- Introduce Generics and Subtyping
- Explain Bounded Wildcard
- Generics Methods
- Provide an overview of the Collection API
- Review the different collection implementations (Set, List and Queue)
- Explore how generics are used with collections

For More Information

For more information about our training services (instructor-led, self-paced or blended), collaborative coaching services, robust Learning Experience Platform (LXP), Career Experiences, public course schedule, partner programs, courseware licensing options or to see our complete list of course offerings, solutions and special offers, please visit us at

www.triveratech.com, email Info@triveratech.com or call us toll free at **844-475-4559.** Our pricing and services are always satisfaction guaranteed.

TRIVERA TECHNOLOGIES • Collaborative IT Training, Coaching & Skills Development Solutions
www.triveratech.com • toll free +1-844-475-4559 • Info@triveratech.com • Twitter TriveraTech

ONSITE, ONLINE & BLENDED TRAINING SOLUTIONS • PUBLIC / OPEN ENROLLMENT COURSES
LEARNING EXPERIENCE PLATFORM (LXP) • COACHING / MENTORING • ASSESSMENTS • CONTENT LICENSING & DEVELOPMENT
LEARNING PLAN DEVELOPMENT • SKILLS IMMERSION PROGRAMS / RESKILLING / NEW HIRE / BOOT CAMPS
PARTNER & RESELLER PROGRAMS • CORPORATE TRAINING MANAGEMENT • VENDOR MANAGEMENT SERVICES

Trivera Technologies is a Woman-Owned Small-Business Firm