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

Category: Software Development & Programming | **Vendor:** Trivera Tech

Duration: 40.00 hours (5 days) **32.5 CPD Hours** **Rating:** ⭐ 4.6 (5,878 reviews)

Course Information

Delivery Format: Instructor Led - Online

Course Overview

Getting Started with Programming, OO and Java Basics for Non-Developers is a skills-focused, hands-on coding course that teaches students the fundamentals of programming object oriented (OO) applications with Java to a basic level, using sound coding skills and best practices for OO development. This course is presented in a way that enables interested students to embrace the fundamentals of coding as well as an introduction to Java, in a gentle paced environment that focuses on coding basics.Students are introduced to the application development cycle, structure of programs, and specific language syntax. The course introduces important algorithmic constructs, string and character manipulation, dynamic memory allocation, standard I/O, and fundamental object-oriented programming concepts. The course explains the use of inheritance and polymorphism early on so the students can practice extensively in the hands-on labs. Structured programming techniques and error handling are emphasized. The course includes the processing of command line arguments and environment variables, so students will be able to write flexible, user-friendly programs. Students will leave this course armed with the required skills to begin their journey as a Java programmer using modern coding skills and technologies.

About This Course

Getting Started with Programming, OO and Java Basics for Non-Developers is a skills-focused, hands-on coding course that teaches students the fundamentals of programming object oriented (OO) applications with Java to a basic level, using sound coding skills and best practices for OO development. This course is presented in a way that enables interested students to embrace the fundamentals of coding as well as an introduction to Java, in a gentle paced environment that focuses on coding basics.Students are introduced to the application development cycle, structure of programs, and specific language syntax. The course introduces important algorithmic constructs, string and character manipulation, dynamic memory allocation, standard I/O, and fundamental object-oriented programming concepts. The course explains the use of inheritance and polymorphism early on so the students can practice extensively in the hands-on labs. Structured programming techniques and error handling are emphasized. The course includes the processing of command line arguments and environment variables, so students will be able to write flexible, user-friendly programs. Students will leave this course armed with the required skills to begin their journey as a Java programmer using modern coding skills and technologies.

Who Should Attend

This basic course is intended for anyone who is new to software development and wants, or needs, to gain an understanding of the fundamentals of coding and basics of Java and object-oriented programming concepts.

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

Prerequisites & Entry Requirements

General Prerequisites:

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.

Learning Outcomes

Upon successful completion of this course, participants will be able to:

This “skills-centric” course is about 50% hands-on lab and 50% lecture, designed to train attendees in basic coding with Java, coupling the most current, effective techniques with the soundest industry practices. Our engaging instructors and mentors are highly experienced practitioners who bring years of current 'on-the-job' experience into every classroom.
Working in a hands-on learning environment, guided by our expert team, attendees will learn:
The steps involved in the creation and deployment of a computer program
What OO programming is and what the advantages of OO are in today's world
To work with objects, classes, and OO implementations
The basic concepts of OO such as encapsulation, inheritance, polymorphism, and abstraction
The basic constructs that all programming languages share
The basic Java constructs supporting processing as well as the OO orientation
How to use Java exception handling
About and how to use classes, inheritance and polymorphism
About use collections, generics, autoboxing, and enumerations
How to take advantage of the Java tooling that is available with the programming environment being used in the class

Detailed Course Outline

Introduction to Computer Programming

- Introduction to Programming
- Programming Tools

Programming Fundamentals

- Thinking About Objects
- Program Basics
- Programming Constructs

Java: A First Look

- The Java Platform
- Using the JDK
- The Eclipse Paradigm
- Writing a Simple Class

OO Concepts

- Object-Oriented Programming
- Inheritance, Abstraction, and Polymorphism

Getting Started with Java

- Adding Methods to the Class
- Language Statements
- Using Strings
- Specializing in a Subclass

Essential Java Programming

- Fields and Variables
- Using Arrays
- Java Packages and Visibility

Advanced Java Programming

- Inheritance and Polymorphism
- Interfaces and Abstract Classes
- Exceptions

Java Developer's Toolbox

- Utility Classes
- Enumerations and Static Imports
- Formatting Strings

Collections and Generics

- Introduction to Generics
- Collections

, 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

Additional Course Details

Nexus Humans Getting Started with Programming, OO and Basic Java for Non-Developers (TT2000) training program is a workshop that presents an invigorating mix of sessions, lessons, and masterclasses meticulously crafted to propel your learning expedition forward.

This immersive bootcamp-style experience boasts interactive lectures, hands-on labs, and collaborative hackathons, all strategically designed to fortify fundamental concepts.

Guided by seasoned coaches, each session offers priceless insights and practical skills crucial for honing your expertise. Whether you're stepping into the realm of professional skills or a seasoned professional, this comprehensive course ensures you're equipped with the knowledge and prowess necessary for success.

While we feel this is the best course for the Getting Started with Programming, OO and Basic Java for Non-Developers (TT2000) course and one of our Top 10 we encourage you to read the course outline to make sure it is the right content for you.

Additionally, private sessions, closed classes or dedicated events are available both live online and at our training centres in Dublin and London, as well as at your offices anywhere in the UK, Ireland or across EMEA.

Frequently Asked Questions

Q: What delivery options are available for Getting Started with Programming, OO and Basic Java for Non-Developers (TT2000)?

We offer multiple delivery formats:

- Live Instructor-Led Classroom Online (Virtual/Live Online)
- Traditional Instructor-Led Classroom Training (ILT)
- On-site delivery at your offices anywhere in United Kingdom
- Private dedicated courses customized for your team

Q: How many CPD hours does this course provide?

The 5-day Getting Started with Programming, OO and Basic Java for Non-Developers (TT2000) course provides up to 32.5 CPD hours of structured learning. CPD certificates can be provided upon request.

Q: What is the duration of the Getting Started with Programming, OO and Basic Java for Non-Developers (TT2000) training?

The training takes place over 5 day(s), with each day lasting approximately 40.00 hours including breaks for lunch and refreshments.

Q: Do you provide corporate training for Getting Started with Programming, OO and Basic Java for Non-Developers (TT2000)?

Yes, we provide corporate training, dedicated training, and closed classes for Getting Started with Programming, OO and Basic Java for Non-Developers (TT2000). Training can take place anywhere in United Kingdom including London, Manchester, Birmingham, Edinburgh, or live online allowing teams from across United Kingdom or internationally to attend.

Q: What related terms do people search for?

Popular related searches include: Java

Q: Why choose Nexus Human for Getting Started with Programming, OO and Basic Java for Non-Developers (TT2000)?

Nexus Human is recognized as one of the leading training providers. Our trainers have won multiple awards including:

- Small Firms Best Trainer Award
- National Training Partner of the Year (Ireland) - Multiple Years
- Global Top 30 Instructor Awards (2012, 2019, 2021)
- Tech Excellence Award Nominations
- Learning Performance Institute (LPI) External Training Provider Sponsor 2024

Q: Are there any discount codes available?

Yes! Use discount code **PENPAL5** when booking your Getting Started with Programming, OO and Basic Java for Non-Developers (TT2000) training. Please note that only one discount code can be used per booking and cannot be combined with other special offers.

Nexus Human

Professional Training & Development

Email: info@nexushuman.com

Website: www.nexushuman.com

Phone: +353 1 889 8700 (Ireland) | +44 20 8142 8961 (UK)

© 2026 Nexus Human. All rights reserved. This brochure was generated on 13/11/2026.