Basic Java Programming for Developers New to OO (TT2120)

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

Geared for experienced developers, Basic Java Programming for Developers New to OO, this hands-on, workshop-style course will provide you with an immersive learning experience that will expand your skillset and open doors to new opportunities within the ever-growing technology landscape. Mastering Java and its powerful capabilities will provide you with the competitive edge you need to stand out in today's fast-paced development world. Working in a hands-on learning environment led by our expert coach, you ll thoroughly explore the foundations of the Java platform, essential programming concepts, and advanced topics, ensuring you acquire a strong understanding of the language and its ecosystem. The object-oriented programming principles taught in this course promote code reusability and maintainability, enabling you to streamline development processes and reduce long-term costs. As you progress through the course, you will also gain familiarity with using an IDE, enhancing your development workflow and collaboration with other Java developers, enabling you to integrate seamlessly into new projects and teams. You ll also gain practical experience in applying the concepts and techniques learned, solidifying your newly acquired skills and facilitating their direct application in real-world scenarios. You ll exit this course empowered to create robust, scalable, and efficient Java-based applications that drive innovation and growth for your organization

About This Course

Geared for experienced developers, Basic Java Programming for Developers New to OO, this hands-on, workshop-style course will provide you with an immersive learning experience that will expand your skillset and open doors to new opportunities within the ever-growing technology landscape. Mastering Java and its powerful capabilities will provide you with the competitive edge you need to stand out in today's fast-paced development world. Working in a hands-on learning environment led by our expert coach, you ll thoroughly explore the foundations of the Java platform, essential programming concepts, and advanced topics, ensuring you acquire a strong understanding of the language and its ecosystem. The object-oriented programming principles taught in this course promote code reusability and maintainability, enabling you to streamline development processes and reduce long-term costs. As you progress through the course, you will also gain familiarity with using an IDE, enhancing your development workflow and collaboration with other Java developers, enabling you to integrate seamlessly into new projects and teams. You ll also gain practical experience in applying the concepts and techniques learned, solidifying your newly acquired skills and facilitating their direct application in real-world scenarios. You ll exit this course empowered to create robust, scalable, and efficient Java-based applications that drive innovation and growth for your organization

Who Should Attend

In order to be successful in this course you should have incoming hands-on experience with another programming language.

This course is not for non-developers or new developers. Possible roles that may attend this course include:

Software Developers: Professionals who have been working with other programming languages and want to expand their skillset by learning Java and its object-oriented features.

Web Developers: Those who work on web applications and want to enhance their back-end development capabilities with Java.

Mobile App Developers: Developers who wish to enter the world of Android app development, where Java is a widely used language for creating mobile applications.

Prerequisites & Entry Requirements

General Prerequisites:

In order to be successful in this course you should have incoming hands-on experience with another programming language. This course is not for non-developers or new developers.

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 core OO coding and Java development skills, 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 to:
Understand what OO programming is and what the advantages of OO are in today's world
Work with objects, classes, and OO implementations
Understand the basic concepts of OO such as encapsulation, inheritance, polymorphism, and abstraction
Understand not only the fundamentals of the Java language, but also its importance, uses, strengths and weaknesses
Understand the basics of the Java language and howit relates to OO programming and the Object Model
Learn to use Java exception handling
Understand and use classes, inheritance and polymorphism
Understand and use collections, generics, autoboxing, and enumerations
Become familiar with the concept of functional programming using Lambda Expressions
Process large amounts of data using the Stream API introduced in Java 8
Discover the new Date/Time API
Use the JDBC API for database access
Work with annotations
Take advantage of the Java tooling that is available with the programming environment being used in the class
Java 8 Features: Lambda Expressions, Method and Constructor references, The Streams API, Collectors, The Optional class

Detailed Course Outline

The Java Platform

- The Java Platform
- Lifecycle of a Java Program
- Responsibilities of JVM
- Documentation and Code Reuse

Using the JDK

- Explain the JDKs file structure
- Use the command line compiler to compile a Java class
- Use the command line Java interpreter to run a Java application class

The IntelliJ Paradigm

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

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

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

Object-Oriented Programming

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

Inheritance, Abstraction, and Polymorphism

- Encapsulation
- Inheritance
- Method Overriding
- Polymorphism

Essential Java Programming

- Essential Java Programming

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

Using Strings and Text Blocks

- Create an instance of the String class
- Test if two strings are equal
- Get the length of a string Parse a string for its token components
- Perform a case-insensitive equality test
- Build up a string using StringBuffer
- Contrast String, StringBuffer, and StringBuilder
- Compact Strings
- Text Blocks

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
- Pattern matching for instanceof
- Overriding subclass methods
- Safely casting references to a more refined type

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
- Local Variable type inference

Using Arrays

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

Records

- Data Objects in Java
- Introduce records as carrier of immutable data
- Defining records

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
- Visibility in the Java Modular System Correctly executing a Java application class
- The Java modular system
- Defining Modules

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

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
- Static, default and private methods in interfaces

Sealed classes

- Introduce Sealed classes
- The sealed and permits modifiers
- Sealed Interfaces Exception Handling

Introduction to Exception Handling

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

Exceptions

- Defining your own application exceptions
- Automatic closure of resources
- Suppressed exceptions
- Handling multiple exceptions in one catch
- Helpful Nullpointers
- Enhanced try-with-resources Java Developer's Toolbox

Developing applications

- Introduce the wrapper classes
- Explain Autoboxing and Unboxing
- Converting String representations of primitive numbers into their primitive types
- Defining Enumerations
- Using static imports
- Deprecating methods Advanced Java Programming

Introduction to Generics

- Generics and Subtyping
- Bounded Wildcards
- Generic Methods
- Legacy Calls To Generics
- When Generics Should Be Used

Lambda Expressions and Functional Interface

- Understanding the concept of functional programming
- Writing lambda expressions
- Understanding functional interfaces

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

Using Collections

- Collection Sorting
- Comparators
- Using the Right Collection
- Lambda expressions in Collections
- Bonus Topics: Time Permitting

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 takeWhile and dropWhile intermediate operations

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

, 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

¢ Documentation and Code Reuse

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

3. 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 the IDE (This course is also offered using Eclipse €œ please inquire for details and options)

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

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, 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

¢ Tutorial: Importing code Using Maven

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

25. Using Collections

¢ Collection Sorting

¢ Comparators

¢ Using the Right Collection

¢ Lambda expressions in Collections

¢ Sequenced Collections

Bonus Topics / Time Permitting

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

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

Additional Course Details

Nexus Humans Basic Java Programming for Developers New to OO (TT2120) 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 Basic Java Programming for Developers New to OO (TT2120) 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 Basic Java Programming for Developers New to OO (TT2120)?

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 Basic Java Programming for Developers New to OO (TT2120) 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 Basic Java Programming for Developers New to OO (TT2120) 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 Basic Java Programming for Developers New to OO (TT2120)?

Yes, we provide corporate training, dedicated training, and closed classes for Basic Java Programming for Developers New to OO (TT2120). 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 Basic Java Programming for Developers New to OO (TT2120)?

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 Basic Java Programming for Developers New to OO (TT2120) 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.