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

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.

What You'll Learn

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.
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.

Objectives

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

- Understand the fundamentals of the Java platform, its lifecycle, and the responsibilities of the Java Virtual Machine (JVM), enabling you to create efficient and reliable Java applications.
- Gain proficiency in using the JDK, including navigating its file structure, utilizing the command-line compiler, and executing Java applications, ensuring a smooth development process.
- Master the IDE, including its interface, project management, and module creation, to enhance productivity, collaboration, and overall development workflow.
- Develop solid skills in writing Java classes, defining instance variables, creating object instances, and implementing main methods, forming a strong foundation in Java programming.
- Acquire expertise in adding methods to Java classes, writing constructors, and leveraging the 'this' keyword, allowing you to create more sophisticated and customizable Java applications.
- Comprehend and apply core object-oriented programming concepts, such as encapsulation, inheritance, and polymorphism, to create modular, maintainable, and reusable code.
- Enhance your knowledge of Java language statements, including arithmetic, comparison, and logical operators, as well as loops and switch expressions, to develop more complex and efficient Java applications.
- Learn to effectively handle exceptions, create custom exception classes, and use try/catch blocks to ensure the robustness and reliability of your Java applications, minimizing potential runtime issues.
- Gain proficiency in working with collections in Java, which includes learning about the different collection implementations (Set, List, and Queue), using iterators, and sorting collections. This - will enable you to manage data effectively in your Java programs.

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

Audience

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

Agenda

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 - 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, 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
- 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

Related Courses

TT2100	Core Java Programming Developer's Workshop
TT2000	Getting Started with Programming, OO & Java Basics for Non-Developers

Attend a Course

Please feel free to Register Online or call 844-475-4559 toll free to connect with our Registrar for assistance. If you ever need additional date options, please contact us for scheduling.