BASIC JAVA PROGRAMMING FOR DEVELOPERS NEW TO OO

Course Code: 101020

Hands-on Java Workshop: Gain the skills required to create efficient, scalable high-performance applications using Java.

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.

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.

NOTE: Developers new or newer to programming should consider the TT2000 Getting Started with Programming, OO and Java Basics as an alternative.

What You'll Learn

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

Who Needs to 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.
- **Full-Stack Developers:** Professionals who have experience with front-end technologies and want to deepen their knowledge of back-end development using Java.
- **Game Developers:** Developers who are interested in creating games for various platforms, including desktop, mobile, and web, using Java as their primary programming language.

Prerequisites

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

BASIC JAVA PROGRAMMING FOR DEVELOPERS NEW TO OO

Course Code: 101020

CLASSROOM LIVE	$3,504 CAD	5 Day

Classroom Live Outline

Session: Java: A First Look

Lesson: The Java Platform

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

Lesson: Using the JDK

- Setting Up Environment
- Locating Class Files
- Compiling Package Classes
- Source and Class Files
- Java Applications
- Setting Up Environment
- Locating Class Files
- Compiling Package Classes
- Source and Class Files
- Java Applications

Lesson: The Eclipse Paradigm

- Workbench and Workspace
- Views
- Editors

- Perspectives
- Projects
- Workbench and Workspace
- Views
- Editors
- Perspectives
- Projects

Session: Getting Started with Java

Lesson: Writing a Simple Class

- Classes in Java
- Class Modifiers and Types
- Class Instance Variables
- Primitives vs. Object References
- Creating Objects
- Classes in Java
- Class Modifiers and Types
- Class Instance Variables
- Primitives vs. Object References
- Creating Objects

Lesson: Adding Methods to the Class

- Passing Parameters Into Methods
- Returning a Value From a Method
- Overloaded Methods
- Constructors
- Optimizing Constructor Usage
- Passing Parameters Into Methods
- Returning a Value From a Method
- Overloaded Methods
- Constructors
- Optimizing Constructor Usage

Session: OO Concepts

Lesson: Object-Oriented Programming

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

Lesson: Inheritance, Abstraction, and Polymorphism

- Encapsulation

- Inheritance
- Method Overriding
- Polymorphism
- Encapsulation
- Inheritance
- Method Overriding
- Polymorphism

Session: Essential Java Programming

Lesson: Language Statements

- Operators
- Comparison and Logical Operators
- Looping
- Continue and Break Statements
- The switch Statement
- The for-each() Loop
- Operators
- Comparison and Logical Operators
- Looping
- Continue and Break Statements
- The switch Statement
- The for-each() Loop

Lesson: Using Strings

- 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
- 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

Lesson: Specializing in a Subclass

- Extending a Class
- Casting
- The Object Class
- Default Constructor
- Implicit Constructor Chaining
- Extending a Class
- Casting
- The Object Class

- Default Constructor
- Implicit Constructor Chaining

Lesson: Fields and Variables

- Instance vs. Local Variables: Usage Differences
- Data Types
- Default Values
- Block Scoping Rules
- Final and Static Fields
- Static Methods
- Instance vs. Local Variables: Usage Differences
- Data Types
- Default Values
- Block Scoping Rules
- Final and Static Fields
- Static Methods

Lesson: Using Arrays

- Arrays
- Accessing the Array
- Multidimensional Arrays
- Copying Arrays
- Variable Arguments
- Arrays
- Accessing the Array
- Multidimensional Arrays
- Copying Arrays
- Variable Arguments

Lesson: Local-Variable Type Inference

- Type inference
- Infering Types of Local Variables
- The var Reserved Type name
- Benefits of Using var
- Backward Compatibilty
- Type inference
- Infering Types of Local Variables
- The var Reserved Type name
- Benefits of Using var
- Backward Compactibilty

Lesson: Java Packages and Visibility

- Class Location of Packages
- The Package Keyword
- Importing Classes
- Executing Programs
- Visibility in the Modular System

- Java Naming Conventions
- Class Location of Packages
- The Package Keyword
- Importing Classes
- Executing Programs
- Visibility in the Modular System
- Java Naming Conventions

Session: Object Oriented Development

Lesson: Inheritance and Polymorphism

- Polymorphism: The Subclasses
- Upcasting vs. Downcasting
- Calling Superclass Methods From Subclass
- The final Keyword
- Polymorphism: The Subclasses
- Upcasting vs. Downcasting
- Calling Superclass Methods From Subclass
- The final Keyword

Lesson: Interfaces and Abstract Classes

- Separating Capability from Implementation
- Abstract Classes
- Implementing an Interface
- Abstract Classes vs. Interfaces
- Separating Capability from Implementation
- Abstract Classes
- Implementing an Interface
- Abstract Classes vs. Interfaces

Session: Exception Handling

Lesson: Introduction to Exception Handling

- Exception Architecture
- Throwing Exceptions
- Checked vs. Unchecked Exceptions
- Exception Architecture
- Throwing Exceptions
- Checked vs. Unchecked Exceptions

Lesson: Exceptions

- Handling Multiple Exceptions
- Automatic Closure of Resources
- Creating Your Own Exceptions
- Handling Multiple Exceptions
- Automatic Closure of Resources
- Creating Your Own Exceptions

Session: Java Developer's Toolbox

Lesson: Utility Classes

- Wrapper Classes
- Autoboxing/Unboxing
- Enumeration Syntax
- Using Static imports
- Wrapper Classes
- Autoboxing/Unboxing
- Enumeration Syntax
- Using Static imports

Lesson: Java Date/Time

- The Date and Calendar classes
- Introduce the new Date/Time API
- LocalDate, LocalDateTime, etc.
- Formatting Dates
- Working with time zones
- Manipulate date/time values
- The Date and Calendar classes
- Introduce the new Date/Time API
- LocalDate, LocalDateTime, etc.
- Formatting Dates
- Working with time zones
- Manipulate date/time values

Session: Advanced Java Programming

Lesson: Introduction to Generics

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

Lesson: Lambda Expressions and Functional Interface

- Lambda Expression Syntax
- Functional Interfaces
- Type Inference in Java 8
- Method references
- Lambda Expression Syntax
- Functional Interfaces
- Type Inference in Java 8
- Method references

Session: Working with Collections

Lesson: Collections

- Characterizing Collections
- Collection Interface Hierarchy
- The Set, List and Queue Interfaces
- Map Interfaces
- Characterizing Collections
- Collection Interface Hierarchy
- The Set, List and Queue Interfaces
- Map Interfaces

Lesson: Using Collections

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

Session: Stream API

Lesson: Streams

- Processing Collections of data
- The Stream interface
- Reduction and Parallelism
- Filtering collection data
- Sorting Collection data
- Map collection data
- Find elements in Stream
- Numeric Streams
- Create infinite Streams
- Sources for using Streams
- Processing Collections of data
- The Stream interface
- Reduction and Parallelism
- Filtering collection data
- Sorting Collection data
- Map collection data
- Find elements in Stream
- Numeric Streams
- Create infinite Streams
- Sources for using Streams

Lesson: Collectors

- Creating Collections from a Stream
- Group elements in the Stream
- Multi-level grouping of elements
- Partitioning Streams
- Creating Collections from a Stream
- Group elements in the Stream
- Multi-level grouping of elements
- Partitioning Streams

Session: The Java Module System

Lesson: Introduction to the Module System

- Introduce Project Jigsaw
- Classpath and Encapsulation
- The JDK internal APIs
- Java 9 Platform modules
- Defining application modules
- Define module dependencies
- Implicit dependencies
- Implied Readability
- Exporting packages
- Introduce Project Jigsaw
- Classpath and Encapsulation
- The JDK internal APIs
- Java 9 Platform modules
- Defining application modules
- Define module dependencies
- Implicit dependencies
- Implied Readability
- Exporting packages

Additional Topics: Time Permitting

Lesson: Formatting Strings

- String.format
- System.out.printf
- The Formatter class
- Using the formatting syntax
- String.format
- System.out.printf
- The Formatter class
- Using the formatting syntax

Lesson: Introduction to Annotations

- Annotations Overview
- Working with Java Annotations
- Annotations Overview
- Working with Java Annotations

Lesson: Java 12 and beyond

- Provide an overview of changes since Java 11
- Introduce Preview Features
- Records (Java 14)
- Switch Expressions (Java 12, Java 13, Java 14)
- Text Blocks (Java 13, Java 14)
- Helpful NullPointerExceptions (Java 14)
- Pattern Matching for instanceof (Java 14)
- Provide an overview of changes since Java 11
- Introduce Preview Features
- Records (Java 14)
- Switch Expressions (Java 12, Java 13, Java 14)
- Text Blocks (Java 13, Java 14)
- Helpful NullPointerExceptions (Java 14)
- Pattern Matching for instanceof (Java 14

Classroom Live Labs

This hands-on course focuses on ‘learning by doing’, combining expert lecture, practical demonstrations and group discussions with plenty of machine-based real-world programming labs and exercises. Student machines are required.

BASIC JAVA PROGRAMMING FOR DEVELOPERS NEW TO OO

Course Code: 101020

VIRTUAL CLASSROOM LIVE	$3,504 CAD	5 Day

Virtual Classroom Live Outline

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

Virtual Classroom Live Labs

This course is rich with hands-on activities and examples, combining robust real-world hands-on labs with expert instruction, engaging activities and group discussions and review. You'll learn and practice new skills under the guidance of our expert instructor, who will prepare you to apply these in the job, role or project with confidence.

Feb 9 - 13, 2026 | 10:00 AM - 6:00 PM EST

Apr 6 - 10, 2026 | 10:00 AM - 6:00 PM EST

BASIC JAVA PROGRAMMING FOR DEVELOPERS NEW TO OO

Course Code: 101020

PRIVATE GROUP TRAINING	5 Day

Visit us at www.globalknowledge.com or call us at 1-866-716-6688.

Date created: 12/6/2025 3:00:27 AM