

Java: Behind the Wheel

An Introduction to Java for C++ Programmers

Course Length:

4 days

Audience:

This course is intended for C++ programmers who want to learn the Java programming language and have no prior knowledge of Java. Students should have a good working knowledge of C++ and object-oriented programming concepts, especially inheritance, polymorphism and abstraction.

Objectives:

- ❑ Understand the importance of Java and how it is being used today in software development.
- ❑ Be able to use the Java language and tools using Java 2 (JDK 1.2).
- ❑ Understand the differences between Java and C++.
- ❑ Become familiar with how arrays are implemented in Java.
- ❑ Be able to use and create Java interfaces.
- ❑ Understand how exception handling works in Java, including how to create user-defined exceptions.
- ❑ Understand the difference between AWT and Swing.
- ❑ Be able to write a program using the AWT and Swing API's.
- ❑ Understand how the delegation model works as an event handling mechanism for GUI programming.
- ❑ Become familiar with how to create and start a thread in Java.
- ❑ Understand the input and output features of Java and how to take advantage of the java.io package.
- ❑ Learn how to connect two computers using sockets and TCP/IP.
- ❑ Understand Java applets and how to write and display one in a web page.

Remarks

This class is an excellent introduction to the Java programming language and is the foundation for any future Java training. The students will write many Java applications in the labs, including a complete chat program that is developed over the last few days of the course. The chat program involves a GUI, event handling, multithreading, stream I/O, and TCP/IP sockets.

Java for Programmers

Module Descriptions

Module 1: An Introduction to Java

Java Overview: What Java is and why it is so popular today.

The Lifecycle of Other Programs: A look at how programs are developed in other languages.

The Lifecycle of a Java Program: A look at how Java programs are developed.

Performance Issues: Translating vs. Interpreting.

A Simple Java Program: Writing a class in Java with main().

The Java Development Kit: The tools used to create a Java program.

Writing a Java Program: The details of creating and running a Java program.

Module 2: The Java Programming Language

Java Keywords: The Java language.

Identifiers: Names used to identify the various parts of a program, like names of classes, methods and fields.

The Built-in Data Types: The way Java stores data.

Literals:

Constants:

The String Class: A useful class in Java.

Declaring Variables: Allocating memory for data.

Arithmetic Operators: The syntax of the Java operators and their order of operation.

Comparison Operators

Boolean Expressions: Logic and the Boolean operators.

The if Statement: The basic tool for making decisions.

The if/else Statement: Extending an if statement.

The switch Statement: Another decision maker.

The while Loop: Repeating code.

The do/while Loop: A variation of the while loop.

The for Loop: Useful for repeating a specific number of tasks.

Module 3: OOAD (Object Oriented Analysis & Design)

OOP and Procedural Languages: Today's common programming languages.

Writing a Program Procedurally: An overview of how procedural programs are designed.

Writing a Program Using Objects: An overview of OOP.

Classes and Object: The fundamental components of an object-oriented program.

An Introduction to OOAD: Object Oriented Analysis and Design.

UML: The Unified Modeling Language.

Inheritance: Creating a new class from an existing class.

Module 4: Classes and Objects

Classes and Objects: An object is an instance of a class.

Writing a Class in Java: Determining fields and methods.

Instantiating Objects: The "new" keyword.

Using Objects: The dot operator.

Understanding References: Understanding the difference between a reference and an object.

Garbage Collection

The == operator

The "this" Reference

Methods: The signature of a method.

Invoking Methods: Using the dot operator.

Passing References by Value: Understanding call-by-value.

Method Overloading

Constructors: A special type of method that allows an object to be initialized when it is instantiated.

Access Specifiers

Encapsulation: Hiding the fields of a class.

Static Fields and Methods: Understanding the concept of static.

Instance and Static initializers

Packages: Java's well-defined namespace technique.

Module 5: Inheritance and Polymorphism

Inheritance: Creating new classes from existing classes.

The "is a" Relationship: Determining when inheritance is a good design.

The extends Keyword: Implementing inheritance in Java.

What Gets Inherited: Understanding what a child inherits from its parent.

Single Inheritance: A child can only have one parent.

Method Overriding: A child class overriding a behavior of the parent class.

The super Keyword:

The Object Class: The finalize() and toString() methods

Constructors: Using this() and super()

Polymorphism

Virtual methods

final Methods and Classes

Abstraction

Module 6: Arrays

Arrays: Contiguous memory for storing data.

Array References

Array Objects: Instantiating arrays.

Using Arrays: Indexes and the length attribute.

Arrays of References

Copying Arrays: System.arraycopy()

Multidimensional Arrays

Module 7: Interfaces

Interfaces: Creating an interface in Java.

Implementing an Interface: A class implements each method of an interface.

Constants in Interfaces: Interfaces can contain static final attributes.

Extending Interfaces: An interface can be subclassed by another interface.

Interfaces and Polymorphism: Using interface references.

Module 8: Exception Handling

Exceptions: The hierarchy of exception classes.

The Throwable Class: The parent of all the exception and error classes.

Catching Exceptions: Using try/catch blocks.

Declaring Exceptions: The handle or declare rule.

Throwing Exceptions: The “throws” keyword.

The finally Statement: Always executes after a try block.

User-defined Exceptions: Creating your own exception classes.

Module 9: GUI Programming

Swing vs. AWT: Understanding the difference options available.

Containers and Components: The relationship between components and containers.

The java.awt.Frame Class: Represents a standard window.

Layout Managers: Flow, border and grid layout.

Event Handling: The delegation model.

The GUI Events: The event classes and listener interfaces.

The Event Adapters

Components: The various Swing and AWT components.

Module 10: Threads

Processes vs. Threads: Understanding what a thread is.

Thread Scheduling: The lifecycle of a thread.

Creating a Thread: The Thread class and the Runnable interface.

Synchronization: Making your Java classes thread-safe.

Module 11: Input and Output

The File Class: Represents a file on a hard drive.

The java.io Package: An overview of the input/output classes.

Streams vs. Readers and Writers: Binary streams vs. character streams.

Low-level Streams: Connecting to the source of data.

High-level Streams: Stream filters and buffers.

Serialization: Java's object serialization.

Low-level Readers and Writers: Character streams.

High-level Readers and Writers

Module 12: Sockets

Sockets: An overview of sockets.

The Server: Listening for requests from clients.

The Client: Connecting to the server.

Socket Streams: Communicating between the client and server.

Appendix A: Applets

An Overview of Applets: A Java program that runs in a web browser.

The Applet Class: The parent class of all applets.

The Methods of the Applet Class: `init()`, `start()`, `stop()`, `destroy()` and `paint()`.

Embedding an Applet in a Webpage: The `<applet>` tag in HTML.

The Graphics Class: Used for drawing in the applet.

Parameters: Allows the HTML to pass data to the applet.

Java for Programmers

Weekly Schedule

The following is a tentative schedule for the pacing of the course. The actual flow of the course may vary.

Day One

Module 1: An Introduction to Java

Module 2: The Java Programming Language

Module 3: Java vs. C++

Module 4: Java Classes and Objects

Day Two

Module 5: Inheritance and Polymorphism

Module 6: Collections

Module 7: Interfaces

Module 8: Exception Handling

Day Three

Module 9: GUI Programming

Appendix A: Applets

Day Four

Module 10: Threads

Module 11: Input and Output

Module 12: Sockets

