



## INTRODUCTION TO JAVA WIRELESS PROGRAMMING

**Course Duration:** 5 days.

**Course Overview:** This five-day course provides an introduction to Java programming with a focus on coding for wireless devices. Students first learn the Java language and development process, including OO concepts and techniques, working with the Java 2 Standard Edition, or J2SE. Then they learn how to create wireless applications with the Java 2 Micro Edition, or J2ME, and the Mobile Information Device Profile, or MIDP. (The MIDP is the “profile” defined within the J2ME for small mobile devices such as cellphones and PDAs.)

The first module introduces the Java software architecture, including the Java Virtual Machine, Java Runtime Environment, Core API, and the Developer’s Kit. Students learn to configure a Java development and runtime environment and to use the J2SE SDK command-line tools, and learn the basic software development process for Java.

The second module covers the fundamentals of the Java language, focusing on its grammar, data types, and procedural aspects such as flow control, including exception handling. By the end of the module students are building simple, practical Java classes and applications.

The third module covers Java as an object-oriented language. The module includes an optional primer on object-oriented methodology and concepts, and then looks at Java as an object-oriented implementation language, including classes, construction, visibility, inheritance, polymorphism, interfaces, abstract classes, and type identification. Lab work moves from analysis and design of a case study to implementation as a Java package of several classes including an application class that implements a command-line interface.

Study of wireless programming begins with a top-down tour of the J2ME architecture, focusing on wireless programming via the Connected, Limited Device Configuration, or CLDC, and the MIDP. Students learn the simple Core API of the CLDC – primarily by contrast to the Java 2 Standard Edition Core API – and then move into the individual packages of the MIDP.

The final MIDP module begins with three chapters on user-interface design, which in MIDP is dramatically different from standard Java. Students learn the high- and low-level UI frameworks, and how to use commands and events. Students then study the MIDP Record Management System for limited persistent storage on the device, and then work on mobile networking.

This course focuses on the application of concepts through substantial hands-on exercises: instructor-led demonstrations and individually-performed labs. A moderately complex MIDP application – a “MIDlet” – is developed over the course as a case study in all the course topics and skills.

## LEARNING OBJECTIVES

- Learn to program effectively in the Java language.
- Understand the Java software architecture, and the design decisions which make Java software portable, efficient, secure and robust.
- Learn how to configure a simple Java development environment.
- Know the grammar, data types and flow control constructs of the Java language.
- Understand Java as a purely object-oriented language.
- Understand the Java 2 Micro Edition architecture, and the stacking of virtual machine, configuration, and profile to address different types of “micro” devices.
- Understand the limitations of mobile devices and see how they drive the design of the Connected, Limited Device Configuration, and the supporting “K” Virtual Machine.
- Identify the key differences between the JVM and the KVM, including build process and code security implementations.
- Understand the design and mission of the Mobile Information Device Profile, and see how programming for mobile devices is fundamentally different from J2SE programming.
- Code cleanly to the CLDC using the KVM.
- Build a simple, functioning MIDlet.
- Understand the framework for packaging and deploying MIDlets to devices.
- Build user interfaces for mobile devices, including text presentation, input controls, 2D graphics, and multi-screen navigation.
- Handle pointer and keypad input in a MIDlet.
- Implement menus and commands.
- Save and re-load information from one MIDlet run to the next using the MIDP Record Management System.
- Make network connections from the mobile device.
- Retrieve remote information via HTTP, including dynamic information based on user requests.

**Prerequisites:** The course is intended for students with some (presumably non-Java) coding experience. Students without previous programming experience are unlikely to finish the course successfully in the five-day timeline.

# COURSE OUTLINE

## Module 1: The Java Software Architecture

### The Java Virtual Machine

- Overview of Architecture
- Java Virtual Machine
- Java Runtime Environment
- The Core API
- Java Developer's Kit
- Portability and Efficiency
- Security
- Robustness – Language Features

### Building Java Software

- Packages
- Tools
- JARs
- Security Model

## Module 2: The Java Language

### Fundamentals

- Source File Format
- Intro to Classes
- Fields and Methods
- Code Grammar and Expressions
- Identifiers
- Operators
- System Class

### Data Types

- Primitive Types
- Type Conversion
- Object References
- Comparing and Assigning References
- Garbage Collection
- Strings
- Arrays

### Flow Control

- Call and return
- Conditional Constructs
- Looping Constructs
- Exceptions

## Module 3: Object-Oriented Java Programming

### Object-Oriented Analysis and Design

- Complex Systems
- Abstraction
- Classes
- Responsibilities and Collaborators
- Relationships
- Visibility
- Simple UML
- Polymorphism

### Encapsulation in Java

- Java Classes
- Member Visibility
- Static Members
- Constructors and Finalizers
- Overloading Methods
- Collection Classes

### Inheritance and Polymorphism in Java

- Extending Classes
- Superclass Reference
- Overriding Methods and Polymorphism
- Defining and Implementing Interfaces
- Abstract Classes
- Type Identification

### Inner Classes (Optional)

- Motivation
- Named Inner Classes
- Outer Object Reference
- Anonymous Inner Classes

## Module 4. The Java 2 Micro Edition

## **The J2ME Architecture**

- Micro Devices
- Common Limitations
- The Need for J2ME
- The J2ME Software Layer Stack
- Virtual Machine
- Configurations
- Profiles
- The CLDC and the KVM
- Tools and Development Process
- Code Security

## **The Connected, Limited Device Configuration**

- Classification of CLDC Target Devices
- The Core API
- Limitations of Java Language Support in CLDC
- Differences between CLDC and J2SE Packages
- The java.lang Package
- The java.util Package
- CLDC Collections API
- The java.io Package
- CLDC Streams Model
- Using the KVM

## **The Mobile Information Device Profile**

- MIDP Target Devices
- Relationship of MIDP to CLDC
- MIDP Support
- MIDlets
- MIDlet Lifecycle
- Building, Packaging and Deploying MIDlets
- Application Descriptors
- The Application Manager
- Summary of MIDP Packages
- MIDP in Context

## **Module 5. MIDP Programming**

### **The High-Level User-Interface API**

- Presenting a User Interface in a MIDlet
- Organizing a UI by Screens
- The Displayable Hierarchy
- Forms and Items
- Layout Control, or the Lack Thereof
- The TextField Class
- The DateField Class
- The ChoiceGroup Class
- Other Item Classes
- Alerts
- Tickers

### **The Low-Level User-Interface API**

- The Canvas Class
- 2D Graphics
- Fonts
- Drawing Text
- Repainting

### **Event Handling**

- MIDP Event Architecture
- High-Level Event Handling
- Commands
- Item State Changes
- Low-Level Event Handling
- Keypad Input
- Pointer Input
- The Model/View/Controller Pattern
- MVC in Application Design
- Model Events

## **The Record Management System (Optional)**

- The Challenge of Wireless Persistence
- Remote Storage via MIDP Networking
- Local Storage via the Record Management System
- Scope of Record Management
- Defining a Record
- Writing Information
- Committing Changes
- Reading Information
- Using Streams for Record I/O
- Implementing Object Persistence

## **Networking**

- The CLDC Streams Model
- The CLDC Networking Package
- The MIDP Implementation
- Supported Protocols
- Creating a Connection
- Making an HTTP Request
- Building a CGI String
- Reading Information
- Controlling Download Size
- Using HTTP POST

## **Learning Resources**