

JA1017 – Intermediate Java Programming

Course Synopsis

Duration:	Five (5) days.
Audience:	Technical Users, Applications Programmers, and Systems Programmers.
Prerequisites:	Introduction to Java course or equivalent experience with basic Java programming.
Delivery Method:	Instructor led, Hands-on workshops

Brief Description

This intermediate level Java course provides programmers, having a fundamental understanding of the basics of Java, with additional details regarding some of the more advanced capabilities provided by the Java Programming Language and its associated standard classes and packages. Topics include JUnit, reflection and JavaBeans, Java 5 type safety enhancements, the Collections Framework, Java Database Connectivity (JDBC), multithreading, inner classes, and networking. There is also an optional section that provides a review of Java basics available in case the students require a Java refresher.

For the most part, this course is independent of the Java development environment that is being used. Although the only requirement is access to the Java Development Kit, the course is best taught using a more formal Integrated Development Environment (IDE) such as Eclipse or RAD.

Course Objectives What You'll Learn

Upon successful completion of this course, the student will be able to:

- Design and code test cases using JUnit and take advantage of the JUnit Wizards.
- Use reflection and introspection to control the publishing and discovery of the properties, events, and methods of Java classes
- Use the enhanced capabilities initially provided with Java 5 to write more type-safe code
- Use the classes and interfaces that comprise the Collections Framework
- Process databases using JDBC
- Create, control, and synchronize threads
- Create and use inner classes and nested classes
- Describe and use the classes in the java.net package

Topics Covered

- | | |
|---|---|
| <p>I. REVIEW OF JAVA BASICS</p> <ul style="list-style-type: none"> • OO Concepts • Classes and Interfaces • IO Package <p>II. UNIT TESTING WITH JUNIT</p> <ul style="list-style-type: none"> • Unit Testing Objectives • Developers Often Avoid Unit Testing • How to Make Unit Testing Succeed • Thorough and Automated Unit Testing • Automated Testing Advantages • Best Practices • JUnit Overview • The JUnit Framework | <ul style="list-style-type: none"> • Assert Methods Available with JUnit • JUnit Assert Methods with Additional Parameter • Project Structure Supporting Test Cases • JUnit Test Case Wizard • What's In the Test Case? • Run Unimplemented Test Case • Things to Test • Constructing a JUnit Test • Sample Method to be Tested • Test Case Execution • Test Fixtures • Constructor Testing • Testing with Exception Handling • Test Suites • Best Practices |
|---|---|

JA1017 – Intermediate Java Programming

Topics
Covered Continued

III. JAVABEANS, REFLECTION, AND INTROSPECTION

- JavaBean Requirements
- Determining Type of an Object
- Reflection Overview and Uses
- Reflection Issues
- Reflection Using Class Object
- Invoking Methods Using Reflection
- Methods Available on the Class Object
- Creating a New Instance
- Introspection
- Customizing BeanInfo
- Invoking Methods Using Introspection
- Using Reflection to Access Properties
- Review of equals Method

IV. TYPE SAFETY AND JAVA 5 ENHANCEMENTS

- Annotations
 - Standard Annotations
 - User Defined Annotations
 - Reflection Annotation Information
- The Enum Data Type
- Generics
- Autoboxing
- Methods Having Variable Parameters
- Assertions

V. COLLECTIONS FRAMEWORK

- What is the Collections Framework
- Simple Arrays and Arrays of Objects
- Legacy Container Classes
- Collections Framework Overview
- Collections Interfaces
 - Lists
 - Sets
 - Queues
- Map Interfaces
- Interface Implementations
 - Lists
 - Sets
 - Queues
 - Maps
- Iterators

VI. JDBC

- What is JDBC
- JDBC Drivers
- Accessing the Database
 - Loading Driver
 - Connecting to Data Source
 - Creating Statements
 - Executing Statements
- Processing Result Sets
 - Cursor Positioning
 - Column Retrieval
 - Updating Result Sets
- Connection Pooling
- Processing Errors and Warnings
- Using Prepared Statements
- Using Stored Procedures
- Metadata
 - Result Set Metadata
 - Database Metadata
- Transaction Processing
- Isolation Levels
- SQL Batches

VII. THREADS

- Definition of Thread
- Creating Threads
- Naming Threads
- Data Sharing Among Threads
 - Local Data
 - Instance Data
 - Class Data
 - volatile Keyword
- Thread States
- Thread Priority
 - Minimum, Maximum, Normal Priorities
 - Preemptive Thread Priority
 - setPriority Method
 - getPriority Method
- Piping Data between Threads
- Coordination and Controlling Threads
- Synchronizing Threads
 - Why Synchronization is Needed
 - Producer/Consumer Example
 - synchronized Keyword
 - Thread Synchronization Methods

JA1017 – Intermediate Java ProgrammingTopics
Covered Continued**VIII. INNER AND NESTED CLASSES**

- What are inner classes
 - Benefits of Inner Classes
 - Terminology
 - Restrictions
 - Syntax
- Types of inner classes
 - Member inner class
 - Local inner class
 - Anonymous inner class
 - Nested top level class

IX. NETWORKING

- Overview of java.net package
- Format of URL
- URL class
- URLConnection class
- InetAddress class
- Definitions
 - Client
 - Server
 - Port
 - Socket
- Socket class
 - Writing client side applications
 - Read from Socket
 - Write to Socket
- ServerSocket class
 - Writing server side applications
 - Daemon servers
 - Thread servers