

## 430: Essential Java (5 days)

### Course Description...

Students who attend Essential Java will leave the course armed with the skills they require to develop solid object-oriented applications written in Java, using sound coding techniques and best coding practices.

This is not an "exposure" class - we ensure that you will leave ready to program intermediate-level, Java applications using object-oriented programs in Java, using sound development techniques. This workshop has successfully provided COBOL, C and mainframe developers the skills needed to program Java.

### Who Should Attend...

Anyone interested in learning about software development with Java.

### Prerequisites...

Students should have working knowledge in a procedural language such as COBOL or C. Although not required, prior exposure to object oriented concepts will prove helpful.

### Learning Objectives...

After successfully completing this course, J2SE developers will be able to:

- Develop object-oriented applications in Java using best coding practices
- Compile and execute Java applications using Ant and/or a Java IDE
- Design with classes and interfaces taking advantage of inheritance and polymorphism
- Reuse and extend existing classes
- Implement classes with fields, constructors and methods
- Leverage interfaces for extensibility
- Employ arrays, references and the collections API to flexibly process lists
- Persist data using text files and Java serialization
- Throw and handle exceptions correctly
- Access databases using JDBC (Java Database Connectivity)
- Parse and create documents using the XML Document Object Model (DOM) APIs
- Create graphical user interfaces using Swing and the Java event delegation model
- Build simple Java web applications using Servlets and JSPs
- Discover and reduce the impact of bottlenecks in Java applications.
- Employ Java best practices to create maintainable programs.

### Hands-On Labs:

The hands-on labs are a key learning element of this course. Each lab reinforces the material presented in lecture allowing the student to gain confidence by successfully translating theory into practice. Taken as a whole, the labs identify the essential skills that will be used on the job in the development of Java applications.

**See next page for a detailed course outline...**



## Course Outline...

### 1. The Java run time environment

- How Java works
  - Why is Java popular both client-side and server-side?
  - How is Java platform-independent
  - Compiling to Byte code
  - The Java Virtual Machine
  - Where is Java used?
- Where Java programs run
  - Java desktop programs
  - Instructor demo: Demonstrate the class project application.
  - Java web applications
  - Java enterprise components
  - The Java microedition
- Java basic syntax
  - Where applications start
  - Primitive data types
  - Statements and blocks
  - Conditionals
  - Loops
  - Expressions and operators
  - Comments
  - Printing to console
- Java building tools
  - Organization of Java programs
  - Class path
  - Sun's software development toolkit
  - Integrated Development environments
  - Ant
  - Ex 1: Write and build your first Java program.

### 2. Working with classes and objects

- Classes and objects
  - Procedural programming
  - Functions and subroutines
  - Data integrity
  - Fields and methods
- Using pre-written classes
  - Reading the Javadoc
  - Declare variables
  - Create an object
  - Constructors and factory methods
  - Call methods
  - Use return values
  - Chained calls
- Java String
  - Creating a String
  - A String is an Object
  - Comparing strings
  - Useful string methods
  - Ex 2: Using Strings. Students will write programs to apply substring, concatenate strings, uppercase them, etc.
- java.util.Date and Calendar
  - The Date class
  - Package



- Importing a package
- Java is an internationalized language
- Using GregorianCalendar
- DateFormat
- Java.text
- Ex 3: Using Date, Calendar and DateFormat

### 3. Writing Java classes

- Class and object design
  - Identify objects
  - Identify object behavior
  - Incorporate relationships
  - Reusability of code
  - Aggregation
  - Inheritance
  - Polymorphism
  - Class hierarchies
  - Pen & Paper exercise: Examine requirements and design the business objects for application
- Writing a class
  - Representing a class
  - Think about typical usage
  - Representing behavior
  - Incorporating state
  - Constructors
  - Getters and setters
  - Overloaded methods
  - Placing a class in a package
  - private, protected, public
  - Ex 4: Write the Exhibit and Ticket classes.
- Inheritance
  - Inheritance for code reuse
  - Java syntax for inheritance
  - Over-riding methods
  - Abstract methods
  - Interfaces
  - Ex 5: Write the GeneralAdmissionTicket class. (LunchHour ticket is the bonus)

### 4. Arrays, Collections and References

- References
  - Method calls revisited
  - What happens to primitive parameters
  - What happens to object parameters
  - Upcasting
  - Downcasting
  - Need for special syntax
- Built-in arrays
  - Array syntax
  - Creating an array
  - Initializing the array variables
  - Looping through an array
  - Fixed size restriction
- Java collections API
  - ArrayList
  - HashMap
  - Type-safe containers (generics)
  - Java 5 for loop



- Iteration syntax
    - Vector and HashTable
  - Ex 6: Write the SpecialExhibitTicket class. This contains a list of exhibits the holder is allowed to visit. Implement the canVisit(Exhibit e) method
5. Persisting data using files
- Working with Files
    - java.io package
    - FileReader, FileWriter
    - BufferedReader, PrintWriter
    - Getting list of files in a directory
    - URL
    - Files from classpath
    - Scanner
  - Java Exceptions
    - Handling exceptions
    - Throwing exceptions
    - Run-time vs compile-time
  - Ex 7: Write the ExhibitsDAO which will obtain list of all the current special exhibits at the museum.
  - Java serialization
    - Need to read/write objects
    - Problems with a relational approach
    - Reading and writing objects
    - Transient and serial version
6. [Optional] Persisting data using databases
- JDBC
    - Drivers
    - Statements
    - ResultSet
    - PreparedStatement
  - Ex 8: Write the TicketsDAO which will connect to the database and update it every time a ticket is sold. It also has a method to retrieve all the tickets sold. Because a ticket may have multiple exhibits, there will be multiple tables to retrieve from – this part will already be written for them.
7. [Optional] Persisting data using XML
- Reading and writing objects in XML
    - What is XML?
    - XML is hierarchical
    - Reading in XML documents using DOM
    - Extracting data from an XML document
    - Creating XML documents using DOM
    - Writing out an XML document
  - Ex 9: Rewrite the ExhibitsDAO to obtain list of exhibits from a XML file. Create a TicketXML class that will print out a XML representation of a single ticket.
8. [Optional] Building user interfaces with Swing
- Swing Graphical User Interface building
    - GUI components
    - Panels and Frames
    - Layout Manager
    - Menus and Menu bars
    - Dialog boxes
    - Displaying a graphical user interface



- Ex 9: Build a GUI to sell tickets. The GUI should allow users to select from a list of exhibits and print the total price for the ticket.
- Event handlers
  - ActionListeners
  - Adding listeners to components
  - Inner classes
  - Static and non-static inner classes
  - Other types of listeners
- Ex 10: Program in the user interaction for the GUI. When the user buys a ticket, the GUI shows the XML representation of the ticket, for printing.

## 9. [Optional] Building user interfaces with Servlets and JSPs

- Web applications
  - Advantages of Http
  - A stateless protocol
  - Get and Put
  - Html forms
  - How a webserver works
- Servlets and JSPs
  - Servlet engine
  - Writing a simple servlet
  - Writing a simple JSP
  - How to deploy a Servlet
  - War files
  - Obtaining information from HTML forms
  - Http Session
- Ex 11: Build the user interface in the form of a HTML form. The result should be the XML ticket that can be printed out.

## 10. [Optional] Java Performance Tuning

- How the garbage collector works
- Memory leaks in Java
- The problem with finalizers
- Setting heap size appropriately
- Measuring performance
- Instructor demo: measuring performance
- Profiling an application
- Instructor demo: profiling the project
- Minimizing object creation
- Reusing objects
- Optimizing the use of JDBC
- Optimizing the use of arrays
- Ex 12: Improve the performance of the class project by reusing Exhibit objects.

## 11. Best practices in Java programming

- Creating and destroying objects
- General naming conventions
- Use interfaces
- Exception handling
- Documentation conventions
- Where to go from here

**Please contact your ROI representative to discuss course tailoring!!!**