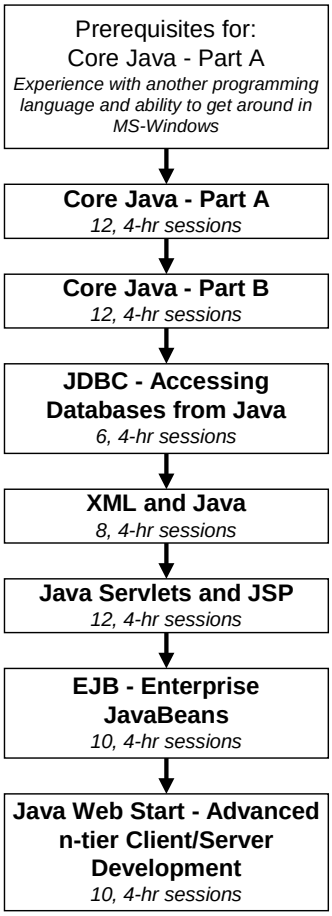
Enterprise Java Developer Training

The Java platform plays an important role in today's business environment. Our Enterprise Java Developer Training program covers the most important aspects of J2SE (Java 2 Platform, Standard Edition) and J2EE (Java 2 Platform, Enterprise Edition) technologies in an integrated series of courses (totaling 70, 4-hour sessions) designed to meet today's business needs. These comprehensive courses have been designed *by* developers *for* developers.

Recommended Course Path



Core Java – Part A

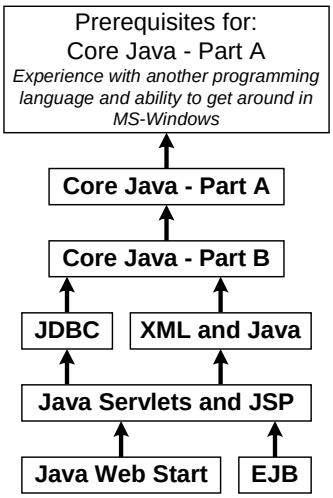
This course is an introduction to programming on the Java platform. Starting with an overview of Java's features and its roles in client/server development, this class gives programmers a good idea of what kinds of solutions are possible with Java. The class includes detailed coverage of:

- introduction to Object Oriented Technology including classes, objects, messages, inheritance, data encapsulation, and polymorphism
- introduction to UML (Unified Modeling Language) class and sequence diagrams
- *all of* the primitive data types, their ranges, and their peculiarities
- implicit and explicit type casting of primitive data types and object references
- *all of* the built-in operators and the built-in math functions
- *all of* the branching and looping structures
- Java's treatment of array objects, **`String`**'s, and **`StringBuffer`**'s
- Swing Graphical User Interface (GUI) components, asynchronous event handling, and the proper use of layout managers to arrange components in nested containers
- custom drawing using the advanced Java 2D API features (**`Graphics`**, **`Graphics2D`**, **`BufferedImage`**, **`java.awt.geom`** and **`java.awt.font`** packages)
- custom GUI component development and double-buffering techniques for drawing complex graphics
- I/O using byte and character streams and Java's exception mechanism

Along with in-class labs on various topics and quizzes, for a final project, students build a Swing application that loads files for viewing. Course is twelve, 4-hour sessions (48 hours): 75% lecture, 25% hands-on lab time. Additionally, students will need to spend time outside of the classroom reading and working on labs. As a prerequisite, students should have previous experience with another programming language and be familiar with how to get around in Microsoft Windows. Even students with some previous Java experience will benefit from the depth of the material in "Core Java – Part A".

Prerequisite Chart

(arrows from a course point to the prerequisites for that course)



Core Java – Part B

This course builds directly upon the material covered in "Core Java – Part A" going into more depth on some topics and covering many additional topics—most students will want to take both "Core Java – Part A" and "Core Java – Part B" back-to-back. Topics covered include:

- coverage of more Swing components, more event types, borders, tool tips, icons, and the safe use of background helper threads with the Swing libraries
- switching "look and feels" under Swing
- Swing menus and keyboard shortcuts (mnemonics and accelerators)
- coverage of additional layout managers (**`CardLayout`**, **`BoxLayout`**, and **`GridBagLayout`**)
- details of Java's class inheritance including all of the methods inherited from the base class **`Object`** by all classes including the correct use of **`equals()`**, **`hashCode()`**, and **`clone()`**.
- comprehensive coverage of all the member variable and method modifiers (**`public`**, **`protected`**, **`private`**, **`final`**, **`abstract`**, **`transient`**, **`native`**, **`volatile`**, **`synchronized`**, **`strictfp`**, and **`static`**).
- inner classes and interfaces – anonymous, nested static, nested non-static

- full coverage **`Throwable`**, **`Error`**, **`Exception`** and **`RuntimeException`** and chaining of exceptions
- full coverage of **`try`**-**`catch`**-**`finally`** including using **`try`**-**`finally`** effectively to bullet-proof code
- Assertion facility
- Collections API—including Lists, Maps, Sets, sorting and searching, and issues related to using Collection objects safely in a multithreaded environment.
- Reflection API—including class introspection and dynamic runtime instantiation of objects and invocation of methods
- Object Serialization API
- TCP/IP networking and Sockets
- Java RMI (Remote Method Invocation)
- safe multithreaded programming in Java including detailed coverage of:
  - spawning of new threads
  - **`Thread`**, **`Runnable`**
  - self-running objects
  - thread prioritization and interruption
  - concurrent access to objects and variables
    - proper use of **`volatile`** and **`synchronized`** language features
    - locking at the object level and class level.
  - wait-notify mechanism—full coverage of this built-in, efficient inter-thread signaling mechanism
    - avoiding missed notifications
    - waiting for the full timeout

Along with in-class labs on various topics and quizzes, for a final project, students build a client/server application that supports multiple simultaneous Swing-based client connections. The server is multithreaded and the client/server communication uses object serialization over TCP/IP sockets. Course is twelve, 4-hour sessions (48 hours): 70% lecture, 30% hands-on lab time. As a prerequisite, students should have completed "Core Java – Part A" or have extensive Java experience covering all of the topics in Part A.

JDBC – Accessing Databases from Java

This course is an introduction to JDBC (Java Database Connectivity). The JDBC API is used to connect Java applications and servlets to any RDBMS (Relational Database Management System). This course starts with a refresher on how a relational database works and the syntax of SQL (Structured Query Language) including full coverage of **`SELECT`** including table joins, plus **`INSERT`**, **`UPDATE`**, **`DELETE`**, and **`CREATE TABLE`** syntax in SQL. The class emphasizes vendor independent techniques that can be used with any RDBMS and includes detailed coverage of:

- SQL Crash Course
- JDBC Overview
- 2 and 3-tier JDBC solutions
- JDBC Drivers – **`DriverManager`**
- Getting a **`Connection`**
- Using **`Statement`** for queries and updates
- **`ResultSet`** and data types
- **`ResultSetMetaData`** information
- **`SQLException`** and **`SQLWarning`**
- Persistence Layer - abstracting data storage *beyond* JDBC to an object view
- Converting rows to objects and objects to rows
- Transactions and Data Integrity
- Locking Rows
- Using **`finally`** to *safely* do transactions
- Generating Unique ID's
- Dates and Times
- Dealing with quotes in user-supplied data
- Using **`PreparedStatement`**
- Binary Data – **`byte[]`**
- Storing Serialized Objects

This class includes lecture, handout examples, in-class lab time, quizzes, and a graded final project. Course is six, 4-hour sessions (24 hours) comprised of 60% lecture, 40% hands-on lab time. As a prerequisite, students should have completed the "Core Java – Part A" and "Core Java – Part B" classes. Experience with relational databases is a plus, but not required.

XML and Java

This course is an introduction to XML (eXtensible Markup Language) using Java-based examples and labs in parsing and generating XML. The class includes detailed coverage of:

- XML Overview
- XML applications/uses
- XML Documents
  - constructs
  - elements and attributes
  - entities
  - processing instructions
  - well-formed documents
- Generating XML Documents
- JAXP – Java API for XML Parsing
- SAX (Simple API for XML) Parsing
- DOM (Document Object Model) Parsing
- DTD's (Document Type Definition)
- XML Schemas
- JDOM Parsing
- XML Transformations
  - XSL
  - Xalan from Apache
- Project: Design an XML document and then parse it into Java objects

This class includes lecture, handout examples, in-class lab time, quizzes, and a graded final project. Course is eight, 4-hour sessions (32 hours) comprised of 60% lecture, 40% hands-on lab time. As a prerequisite, students should have completed the "Core Java – Part A" and "Core Java – Part B" classes.

Java Servlets and JSP

This course is an introduction to the use of Java Servlets and JSP's (Java Server Pages). These J2EE technologies dynamically generate web pages in response to user input in browsers. In many cases, Servlets and JSP's talk to databases on the back-end to gather information to respond to user input. This class explores both technologies and includes detailed coverage of:

- Web Architectures
- HTML Crash Course
- Tomcat from Apache
- Servlet Life Cycle
- **`HttpRequest`** and **`HttpResponse`**
- Form Processing
- Cookies
- WAR (Web ARchive) files
- Initialization Parameters for Deployment Flexibility
- Session Tracking
- Connecting to a Database via JDBC
- Object Oriented HTML Generation from Servlets
- **`PageProcessor`** Utility
- JSP Communication with JavaBeans
- Developing and Using Custom JSP Tags
- Generating Dynamic Images, Graphics, and Charts
- Multithreading Servlets:
  - the critical job of making servlets multithread-safe
  - background processing of long-running tasks
  - automatic page refreshing to check on the progress of background tasks
- Generating XML responses
- Generating Serialized Object responses for use with Swing clients
- Project: Shopping Cart for a DVD Store

This in-depth class includes lecture, handout examples, in-class lab time, and quizzes. For a graded final project, students build an e-commerce application that tracks user sessions, queries and updates a database, and dynamically generates different pages. Course is twelve, 4-hour sessions (48 hours) comprised of 65% lecture, 35% hands-on lab time. As a prerequisite, students should have completed the "Core Java – Part A", "Core Java – Part B", "JDBC", and "XML and Java" classes.

EJB – Enterprise JavaBeans

This course is an introduction to EJB (Enterprise JavaBeans). EJB technology is a vital component of the J2EE platform, playing a central role in the development of n-tier, scalable applications. The class includes detailed coverage of:

- EJB Technology and Component models
- Using EJB's
- Designing EJB's
- Bean-Container contract
- Developing EJB's
  - Deployment Descriptors
  - EAR (Enterprise ARchive) files
  - Entity Beans
    - CMP – Container Managed Persistence
    - BMP – Bean Managed persistence
    - CMP vs. BMP
  - Session Beans
    - Stateless
    - Stateful
  - Message Driven Beans (EJB 2.0)
- EJB Transactions
- Using with the rest of the J2EE platform
- Integrating with Servlet and Swing clients
- Project: Banking application using EJB's to talk to a database

This class includes lecture, handout examples, in-class lab time, quizzes, and a final project. Course is ten, 4-hour sessions (40 hours) comprised of 60% lecture, 40% hands-on lab time. As a prerequisite, students should have completed the "Core Java – Part A", "Core Java – Part B", "JDBC", "XML and Java", and "Servlets and JSP" classes.

Java Web Start – Advanced n-Tier Client/Server Development

This course is an introduction to using Java Web Start *and* its integration into advanced multi-tier architectures. Java Web Start is a technology used to deliver (and *automatically* keep up-to-date) client-side Java application code from a web page link. This extremely powerful technology allows for the development of rich, interactive, client-side graphical user interfaces with Swing, while simultaneously eliminating the task of keeping the client-side code in the field up-to-date. The class includes detailed coverage of:

- Java Web Start Architectures
- JNLP – Java Network Launching Protocol
  - JNLP file elements
  - dynamic configuration
- In-depth Swing GUI (Graphical User Interface) Development
- Working with Java Web Start's Security Layer to:
  - print using the Java 2D API
  - read and write files
  - paste from the system clipboard
- n-Tier Development with Java Web Start Clients:
  - advantages over a Servlet/JSP solution
  - communication via:
    - JDBC directly to databases
    - Java RMI (Remote Method Invocation) to a Java application middle tier
    - Serialized Objects to a Java Servlet middle tier
    - XML documents to a Java Servlet middle tier
    - incorporating SSL for secure communications
- Generation of Charts and Graphs
- Project: Use a Java Web Start client, a Java Servlet middle tier, and a relational database to view and edit employee records

This class includes lecture, handout examples, in-class lab time, quizzes, and a final project. Course is ten, 4-hour sessions (40 hours) comprised of 60% lecture, 40% hands-on lab time. As a prerequisite, students should have completed the "Core Java – Part A", "Core Java – Part B", "JDBC", "XML and Java", and "Servlets and JSP" classes.