



Capstone Courseware, LLC

33 Boylston Street
Jamaica Plain, MA 02130

877-227-2477
capstonecourseware.com

106. Advanced Java Programming

Version 1.4

This course provides advanced training in developing software using the Java 2 Platform, Standard Edition, or J2SE. It is intended for students with solid experience in structured and object-oriented Java programming, including use of the Collections API and exception handling. The course software also includes an optional overlay of workspace and project files to support use of the Eclipse IDE in the classroom. (This requires that the instructor be experienced in use of Eclipse and able to walk students through basic tasks in the IDE.)

This revision of the course focuses on the 1.4.2 SDK and language, but highlights missing features and areas that are improved in the 5.0 JDK and language. For training entirely within the Java 5.0 environment, see version 5.0 of this course; to read more about different versions of Java and for help deciding on which version of this course to use, see "Java Versions and Terminology Demystified".)

The course is organized into four modules. The first covers several general-purpose topics such as writing multi-threaded applications, the Reflection API, and network programming using sockets. Then the course takes up the challenge of building multi-tier applications using the standard Java platform. Multi-tier development most frequently uses the Java 2 Platform, Enterprise Edition, or J2EE, and we have a wide range of courses in that area. However it is quite possible to build lightweight multi-tier applications using only J2SE, and for some purposes the J2EE is more trouble than it's worth.

So the course looks at each of the traditional tiers and the J2SE APIs most suited to developing them. The second module of the course covers the Java Foundation Classes, or JFC, for building graphical user interfaces (GUIs) for the presentation tier. The third module introduces Java Remote Method Invocation, or RMI, as a way of distributing Java objects over multiple processes, which may be spread over a network of machines; this provides a means of developing a separate business tier. Finally, the fourth module teaches the Java Database Connectivity, or JDBC, API, for the persistence tier. Students will complete the course with a good working knowledge of each of these technologies, prepared to build distributed Java applications and to pursue JFC, RMI, or JDBC in greater depth.

Prerequisites



106. Advanced Java Programming

Outline

- Solid Java programming experience is essential -- especially object-oriented use of the language. Language features and techniques that are integral to some lab exercises include interfaces and abstract classes, threading, generics and collections, and recursive methods. Course 103, "Java Programming," is excellent preparation.





Learning Objectives

- Write multi-threaded Java applications.
- Use the Reflection API for highly generic tasks, discovery, or code-generation.
- Communicate between processes using network sockets.
- Understand the basics of the JFC architecture.
- Build simple GUI applications using JFC.
- Build more complex GUIs using various JFC controls.
- Use the many hooks into the JFC architecture to easily customize rendering and editing within JFC controls.
- Understand the significance of the MVC decomposition in using JFC controls.
- Build GUI classes that make effective use of events as fired from model, view and controller elements of the GUI itself.
- Understand the role of Java RMI in distributed Java software development.
- Understand the relationship between RMI and various J2EE technologies such as JNDI, EJB, and CORBA.
- Understand the RMI framework and architecture, especially the role of generated stubs and skeletons.
- Implement simple RMI clients and servers.
- Connect to a database using JDBC and perform a simple query.
- Update relational data using JDBC to execute updates, inserts and deletes.
- Use prepared statements to produce reusable database queries and optimize execution time.
- Use callable statements to access database procedures.
- Use scrollable and updatable results sets for more robust solutions.
- Use commit, rollback, and savepoint to build transactional systems.
- Use batch processing for efficient handling of large datasets.

Timeline: 5 days.





Module 1. Advanced Java

Chapter 1. Threads

- Java Thread Model
- Threads and ThreadGroups
- Creating and Running Threads
- Manipulating Thread State
- Creating Thread Classes
- Thread Synchronization
- Java 1.5: Concurrency Library
- wait and notify
- join and sleep

Chapter 2. Reflection

- Uses for Meta-Data
- The Reflection API
- The Class Class
- The java.lang.reflect Package
- Reading Type Information
- Navigating Inheritance Trees
- Dynamic Instantiation
- Dynamic Invocation
- Java 1.5: Annotations and Aspect-Oriented Programming

Chapter 3. Serialization

- The Challenge of Object Serialization
- Serialization API
- Serializable Interface
- ObjectInputStream and ObjectOutputStream
- The Serialization Engine
- Transient Fields
- readObject and writeObject
- Externalizable Interface

Chapter 4. Sockets

- The OSI Reference Model
- Network Protocols





106. Advanced Java Programming

Outline

- The Socket Class
- The ServerSocket Class
- Connecting Through URL Objects
- HTTP and Other TCP Servers
- Datagram Clients and Servers
- Non-Blocking Sockets





Module 2. Java Foundation Classes

Chapter 1. Introduction to JFC

- Abstract Windowing Toolkit Basics
- Simple Layout Management
- Simple Event Handling
- Lightweight Controls
- JFC Feature Set
- JFC Architecture and Relationship to AWT

Chapter 2. JFC Application Design

- Role of a JFrame
- Building a Frame-Based JFC Application
- Panes
- Using Dialogs

Chapter 3. JFC Components

- JFC Component Class Hierarchy
- JComponent Features
- Simple Control Types
- Text Components
- Menus
- Managing Look and Feel

Chapter 4. Architectural Patterns

- Observer Pattern
- Model-View-Controller Decomposition
- Strategy Pattern
- JList
- Factory Pattern
- JComboBox





Module 3. Remote Method Invocation

Chapter 1. RMI Architecture

- Motivation for RMI
- RMI and EJB
- RMI and CORBA
- Overview of RMI Architecture
- Stubs and Skeletons
- Remote Reference Layer
- RMI Transport
- Lifetime of a Remote Method Invocation
- RMI Registry
- Naming and URL Resolution
- RMI Up and Running

Chapter 2. RMI Implementation

- Interface Design
- The Remote Interface
- Implementation Classes
- The RemoteObject and RemoteServer Classes
- The UnicastRemoteObject Class
- Server Implementation
- Using the Registry
- Client Implementation
- Code Deployment

Chapter 3. Practical RMI

- Stub Source Code
- RMI Marshaling
- Skeleton Source Code
- Diagnostic Code Using RemoteServer
- Passing Objects
- The Factory Pattern
- Serialization vs. Remote Reference
- Designing for Latency
- The Value Object Pattern
- Exception Handling





Module 4. Java Database Connectivity

Chapter 1. Database and SQL Fundamentals

- Relational Databases and SQL
- Database, Schema, Tables, Columns and Rows
- SQL Versions and Vendor Implementations
- DDL -- Creating and Managing Database Objects
- DML -- Retrieving and Managing Data
- Sequences
- Stored Procedures
- Using SQL Terminals

Chapter 2. JDBC Fundamentals

- What is the JDBC API?
- JDBC Drivers
- Making a Connection
- Creating and Executing a Statement
- Retrieving Values from a ResultSet
- SQL and Java Datatypes
- SQL NULL Versus Java null
- Creating and Updating Tables
- Handling SQL Exceptions and Proper Cleanup
- Handling SQLWarning

Chapter 3. Advanced JDBC

- SQL Escape Syntax
- Using Prepared Statements
- Using Callable Statements
- Scrollable Result Sets
- Updatable Result Sets
- Transactions
- Commits, Rollbacks, and Savepoints
- Batch Processing
- Alternatives to JDBC

Chapter 4. Introduction to Row Sets

- Row Sets in GUI and J2EE programming





Advantages of RowSets
RowSet Specializations
Using CachedRowSets

Appendix A. Learning Resources

Appendix B. Course Schema

Appendix C. Methods and Types

System Requirements

Hardware Requirements (Minimum)

500 MHz, 256 meg RAM, 500 meg disk space.

Hardware Requirements (Recommended)

1.5 GHz, 512 meg RAM, 1 gig disk space.

Operating System

Tested on Windows XP Professional. Course software should be viable on all systems which support a J2SE 1.4 SDK.

Network and Security

Limited privileges required -- please see our standard security requirements at <http://capcourse.com/Guides/Security.html>.

Software Requirements

All free downloadable tools.

