



Core Java+ J2EE+Struts+Hibernate+Spring

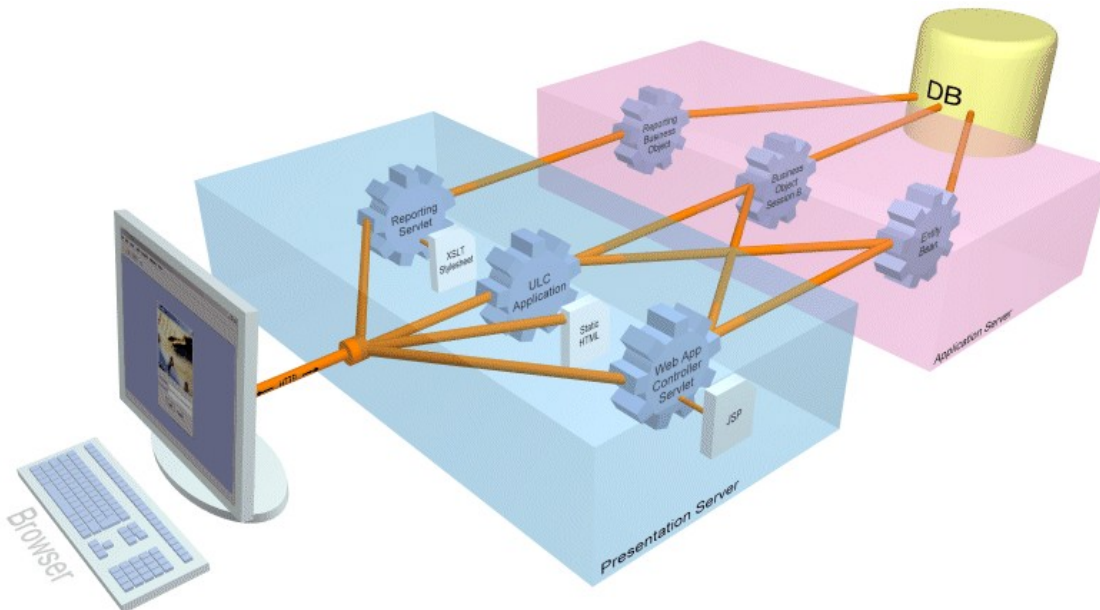
Training
**How much new information
can you fit in your brain?**



MIRACLE
Corporate Solutions Pvt. Ltd.

Java technology is a portfolio of products that are based on the power of networks and the idea that the same software should run on many different kinds of systems and devices

Java technology readily harnesses the power of the network because it is both a programming language and a selection of specialized platforms. As such, it standardizes the development and deployment of the kind of secure, portable, reliable, and scalable applications required by the networked economy. Because the Internet and World Wide Web play a major role in new business development, consistent and widely supported standards are critical to growth and success.





A- Core Java

The Core Java segment deals with the basics of Java. It is designed keeping in mind the basics of Java Programming Language that will help new students to understand the Java language, its syntax, its simplicity and what makes it a language of choice among so many languages available in the marketplace. It also talks about the Architecture of Java and the uniqueness of Java that makes it a platform along with programming language of choice.

Audience

The course is designed for Students to make them familiar with the Java Programming Language and the techniques used in industry to implement a project

Prerequisites

Students should have some basic-level knowledge of programming in any programming language, preferably C or C++, or other OOP languages.

Approach

Extensive practical exercises take students through all major aspects of the design and development of Java programs

Objectives

After completing this course, the student should be able to:

- Apply OOP principles
- Design OO applications using abstraction, encapsulation, modularity, and hierarchy
- Create applications
- Use Java Exceptions to guarantee graceful error recovery of application
- Do input/output using Java
- Use JDBC to access data base tables

Topics

A-1. Introduction to Features of Java

- 1.1 Architecture of Java
- 1.2 Garbage Collection
- 1.3 Difference between JDK, JRE, JVM and JIT
- 1.4 Installing JDK
- 1.5 Writing first program in Java and compiling and running it using Console

2. Introduction to Eclipse IDE.

- 2.1 Data Types in Java
- 2.2 Java Integer Types
- 2.3 Variable Declaration and Initialization
- 2.4 Integer Literals
- 2.5 Char Literals
- 2.6 Floating Point Literals
- 2.7 Comments in Java



- 2.8 Casting
- 2.9 Arithmetic, Bitwise, Boolean Operators and Shift Operators
- 2.10 Java Keywords and reserved words

- 3. The Java Control Statements** (if – else and switch – case statements)
 - 3.1 **Iteration Statements** (loops and labeled break and labeled continue)
 - 3.2 **Debugging** a program using Eclipse IDE

- 4. Introduction to Classes & Objects**
 - 4.1 **Constructors, Object Instantiation**
 - 4.2 **Arrays in Java** – declaration, definition and types of Arrays
 - 4.3 **Abstract classes and interfaces**

- 5. Introduction to Object Oriented Programming vs. Conventional Programming**
 - 5.1 Method Overloading & Overriding
 - 5.2 == vs. equals()
 - 5.3 The **this**, **this()**, **super** and **super()**, Constructor Chaining

- 6. Modifiers in Java** – private, protected, static, public, final and default
 - 6.1 **Packages and Access specifiers** and their scope

- 7. Immutable and Mutable Classes**
 - 7.1 Difference between String and String Buffer classes

- 8. Comparator and Comparable classes**
 - 8.1 **Shallow Copy vs. Deep Copy**

- 9. Exception Handling** and writing your own exceptions
 - 9.1 The **try**, **catch**, **throws**, **throw** and **finally**
 - 9.2 Re - throwing an exception
 - 9.3 Checked and Unchecked Exceptions

- 10. Multithreading**
 - 10.1 Difference between Thread and Process
 - 10.2 The Thread class and Runnable interface.
 - 10.3 Daemon and User Threads
 - 10.4 Creating your own threads
 - 10.5 The synchronized method and block
 - 10.6 Deadlocks
 - 10.7 Avoiding dead locks

- 11. The Collection Framework**
 - 11.1 Introduction to Map, Set and List.
 - 11.2 The Map, Set and List interfaces in Java.
 - 11.3 Implementation of collection interfaces by different classes in Java.



- 11.4 The Vector, ArrayList, HashMap, Hashtable, HashSet, SortedSet, and TreeSet.
- 11.5 Synchronizing unsynchronized collections

12. Java Database Connectivity API (JDBC)

- 12.1 Design of JDBC
- 12.2 Typical uses of JDBC
- 12.3 Types of JDBC Drivers
- 12.4 Getting a JDBC driver and loading it through java.sql.Driver Manager class
- 12.5 The Connection, Statement, and ResultSet interfaces
- 12.6 Accessing a database using JDBC and using SQL queries.
- 12.7 The Prepared Statement and Callable Statement
- 12.8 Scrollable and Updatable ResultSets

13. Java I/O package

- 13.1 Introduction to Files and Streams
- 13.2 DataInputStream and DataOutputStream
- 13.3 FileInputStream and FileOutputStream
- 13.4 Reader and Writer
- 13.5 Why use Readers and Writers
- 13.6 Object serialization using ObjectInputStream and ObjectOutputStream
- 13.7 using transient keyword

B- Java Server Side (Advance Java / J2EE)

This course presents the recommended Java “Model View Controller” Design pattern to implement the use of Servlets, JavaServer Pages and Java Beans, using JDBC to access RDBMS databases. It also reviews the Server architecture; explaining how applications are deployed in a server; and discussing Developers “runtime” customizations of Servers and Applications within the Server, using Server.xml and web.xml files. Server Security is overviewed and demoed.

Extensive workshops (65%) are used to supplement lecture (35%). At the completion of this course, the students will have all the tools necessary to write Java Server Side applications that access databases.

This course explores Java's database connectivity package, JDBC. Topics include JDBC elements, the steps used to access a database with JDBC, data retrieval issues, and some of the advanced features that will gain more support with future releases of the JDBC API. Participants will use the JDBC interface to provide a call-level API for SQL-based database applications. This hands-on course is composed of comprehensive lectures, practical project illustrations, and independent programming sessions. The session's exercises include a realistic database application using modular Java architecture where the participant produces specific reports based on data in the database.

This course teaches Servlets (Ver 2.4) programming as well as JSPs (Ver 2.0) programming. Students will learn the basics of creating both Servlets and JSPs including their interaction; how to dispatch Servlets/JSPs from within other Servlets/JSP



It teaches the recommend use of Java Beans with JDBC to manage data in the sever environment and provides explicate coding examples on how to handle the management of Java Bean scope (page, request, session and application) from Servlets to JSPs

Optionally, time permitting, overviews and examples of: Networking Distributed Objects (including RMI/ remote procedural calls), as well as Java Collections.

Audience

This course is intended for programmers who have at least six months experience in Java and who want to learn about server side programming, accessing DBs .

Approach

Jdk 1.5, Jakarta-Tomcat-5.x, APIs for Java, Servlets and JSPs and Access DBs. Students Will be able to set up and run a Web-Server on their own PC. Using Exadel Studio with Eclipse to build and debug Web Applications

Content

1. Introduction to Web Applications

- 1.1. The Web App Architecture
- 1.2. Why servlets over CGI,
- 1.3. The Servlet inheritance hierarchy
- 1.4. Servlet lifecycle
- 1.5. Difference between send redirect and forward
- 1.6. The ServletConfig and ServletContext
- 1.7. Writing your servlets
- 1.8. Installing and understanding Servlet Container
- 1.9. Starting and Stopping Servlet Container
- 1.10. Running your servlets
- 1.11. The Servlet's Single Thread Model
- 1.12. Session tracking using HttpSession, Cookies, Hidden Form Fields and URL rewriting
- 1.13. Understanding the limitations of the session tracking techniques
- 1.14. Debugging using Eclipse IDE, the web.xml files tags.
- 1.15. The FilterServlet API
- 1.16. Securing web applications
- 1.17. Authentication, Authorization, Roles, etc
- 1.18. Packaging web application and deploying it
- 1.19. Understanding the JDBC 2.0 API
- 1.20. Working with DataSource and InitialContexts
- 1.21. Configuring the JDBC driver with Web Server



- 1.22. Configuring the DataSource with Web Server
- 1.23. Configuring the DataSource with Web App
- 1.24. Running the Web App and accessing the database

2. The Server Status Codes

- 2.1 The Java Server Pages (JSP 2.0)
- 2.2 The JSP life – cycle
- 2.3 JSP directives,
- 2.4 Limitations of Servlets over JSP and vice-versa
- 2.5 Implicit objects
- 2.6 Attributes
- 2.7 Session tracking
- 2.8 JSP and JDBC
- 2.9 Servlet, JSP communication

3. The Java Beans

- 3.1 How to use Java Beans in JSP and servlets
- 3.2 The MVC architecture
- 3.3 Accessing Java Bean in a JSP page
- 3.4 The scope of a Java Bean – page, request, session, application
Applying MVC

4. The Enterprise Java Beans- Introduction,

- 4.1 EJB vs. Java Beans
- 4.2 Understanding the Home Interface, Remote Interface
- 4.3 The Session Bean – stateless and stateful
- 4.4 The Entity Bean – CMP vs. BMP,
- 4.5 The Java Message Bean
- 4.6 The deployment descriptor and the Weblogic deployment descriptor,
- 4.7 The EJB container and its functionality
- 4.8 The roles of various players in an Enterprise Application
- 4.9 Writing and deploying your EJBs, calling EJBs from JSP and Servlets and consoles.

Struts course outline

Module 1: Struts Architecture

- ☛ MVC and Model 2
- ☛ Command Pattern
- ☛ Jakarta Struts
- ☛ More XML, Less Java!
- ☛ Action Mappings
- ☛ JavaBeans in Struts
- ☛ Working with Forms
- ☛ Validation
- ☛ Relational Models
- ☛ Presentation Technology
- ☛ Tiles

Module 2: Action Mappings

- ☛ Command Pattern for Web Applications
- ☛ Action Servlets



- ☛ Action
- ☛ Action Mapping
- ☛ Struts Configuration
- ☛ Selecting a Forward
- ☛ Global Forwards
- ☛ Forwarding Actions
- ☛ Others Action Subtypes
- ☛ Declarative Exception Handling

Module 3: Forms

- ☛ Working with HTML Forms
- ☛ What Not To Do
- ☛ Action Forms
- ☛ Relationship to Input
- ☛ Relationship to Actions
- ☛ Relationship to the Model
- ☛ Relationship to Output
- ☛ Dyna Action Form and Map – Backed Forms
- ☛ Validation
- ☛ Coarse – Grained Form Beans

Module 4: Relational Data

- ☛ JDBC
- ☛ Drivers
- ☛ Driver Manager (JDBC 1.0)
- ☛ Data Source (JDBC 2.0)
- ☛ Connection
- ☛ Statement
- ☛ Result Set
- ☛ The Struts Data- Source Manager
- ☛ Multi – Tier Design
- ☛ Business Logic Beans
- ☛ Persistence Logic
- ☛ EJB

Module 5: Struts Tag Libraries

- ☛ Building View Components
- ☛ Struts Tag Library
- ☛ Attributes
- ☛ Building Forms
- ☛ <html : Forms>
- ☛ <html : test> et. al.
- ☛ Forms and Form Beans
- ☛ Scope and Duration of Forms Data
- ☛ Managing Hyperlinks
- ☛ Error Messages
- ☛ Logic Tags

Module 6: The JSP Standard Tag Library

- ☛ JSTL Overview
- ☛ JSP Expression Language
- ☛ Core Tags
- ☛ Formatting Tags
- ☛ XML Tags
- ☛ SQL Tags
- ☛ Mixing JSTL, EL, Scripts and Actions

Module 7: Internationalization and Localization

- ☛ i18n in Java
- ☛ Locale
- ☛ Resource Bundle
- ☛ i18n in Actions
- ☛ i18n in JSTL
- ☛ i18n in Validation

Module 8: Input Validation

- ☛ Validation in Web Applications
- ☛ Validation in Struts
- ☛ The Struts Validator Plug – ins
- ☛ Validating Action Form Subtypes



- ☛Configuring Validation
- ☛Validators
- ☛Rules
- ☛Is <html: Form> Necessary?
- ☛Reporting Errors
- ☛Multi – Page Validation
- ☛Client – Side Validation
- ☛Limitation on the Client Side
- ☛Implementing a Validator
- ☛Implementing Action Form. Validate
- Module 9: Under the Hood**
- ☛Global Objects and Keys
- ☛Modules
- ☛Action Servlet, Request Processor, Exception Handler
- ☛Struts Configuration in Depth
- ☛The Org. apache. struts. config Package
- ☛Plugs – Ins
- ☛Logging with Commons and Log4J
- ☛Configuring Log4J
- ☛Logging in Web Applications
- ☛The Org. apache. struts. util Package
- ☛Common Bean Utils
- Module 10: Best Practices**
- ☛Cardinalities in Struts Design
- ☛Coarse – Grained Form Beans
- ☛Many Actions from One View
- ☛Multiple Forwards
- ☛Many Mappings to one Action
- ☛Chaining Actions
- ☛Dynamic Forwarding
- ☛Form Beans as Mediators
- ☛Using Reflection and Bean Utils
- ☛Reusing Validation Rules
- ☛Mapping – Based Validation
- ☛Graceful Validation
- Module 11: Tiles**
- ☛Consistent Look and Feel
- ☛Reusable Layouts and Content
- ☛The Tiles Framework
- ☛Instantiating Layouts
- ☛Body – Wrap Insertions
- ☛Tiles and Style sheets
- ☛Working with Tiles Attributes
- ☛The Tiles Context
- ☛Definitions
- ☛Aggregation and Inheritance
- ☛The Tiles Plug – In
- ☛Forwarding to Definitions
- ☛Performance Considerations

Hibernate Course Structure

1 Why Hibernate?

- Understanding object persistence
- Identity - Inheritance - Associations - Object/relational mapping
- Using direct JDBC
- Example application - Retrieving object graphs using JDBC - Persisting object graphs to a relational model - Deleting object graphs -
- Querying object graphs
- Persistence with Hibernate
- Simplicity and flexibility - Completeness - Performance
- Summary



2 Installing and building projects with Ant

- Getting a Hibernate distribution
- Installing Ant - Getting Ant - Extracting and installing Ant
- Setting up a database
- Getting MySQL - Testing MySQL - MySQL drivers
- Setting up a project
- Defining directories - Ant 101 - Running Ant

3 Hibernate basics

- Configuring Hibernate
- Basic configuration
- Creating mapping definitions
- IDs and generators - Properties - Many-to-one element - Proxies - Collections - Cascades - Fetching associated objects
- Building the SessionFactory
- Configuring the SessionFactory
- Persisting objects
- Retrieving objects
- The Session cache
- Advanced configuration
- Connection pools - Transactions - Cache providers
- Inheritance
- Table per class hierarchy - Table per subclass

4 Associations and components

- Associations
- Many-to-one relationships, in depth - The central configuration file - Defining sample data
- Building tables and SchemaExport
- Logging with log4j and Commons Logging - Running SchemaExport - Loading the Events - Refactoring - Finding Events - Cascades
- Components
- What's in a component? - Mapping a component - Why use a component?

5 Collections and custom types

- Persisting collections and arrays
- Using interfaces - Mapping persistent collections - Collection types - Lazy collections - Sorted collections - Bidirectional associations - Cascading collections
- Implementing custom types
- UserTypes - Implementing CompositeUserTypes

6 Querying persistent objects

- Using HQL
- session.find(?) - The Query interface - Outer joins and HQL - Show SQL
- Query substitutions - Query parser
- Querying objects with HQL
- The FROM clause - Joins - Selects - Using functions - HQL properties - Using expressions
- Criteria queries
- Stored procedures

7 Organizing with Spring and data access objects

- The ubiquitous DAO pattern
- Keeping the HQL together
- Analyzing the DAO
- Boilerplate code - Potential duplication - Detached objects only
- The Layer Supertype pattern
- Creating an AbstractDao
- The Spring Framework
- What's in a template? - Beans and their factories

