Course duration	140 Hr

ITShare Company
01122339966-01122339955
Mostafa@iteshare.com

Java Full Stack Content

Java SE 7 Programming

Course Objectives

- Create Java technology applications with the latest JDK 7 Technology and the NetBeans Integrated Development Environment (IDE).
- Enhance object-oriented thinking skills using design patterns and best practices.
- Identify good practices in the use of the language to create robust Java applications.
- Manipulate files, directories and file systems.
- Write database applications using standard SQL queries through JDBC.
- Create high-performance, multi-threaded applications.
- Create classes that subclass other classes, extend abstract classes and program with interfaces.
- Properly use exceptions and the Collections framework.
- Develop applications that manipulate files, directories and file systems. **Course**

Outline

Java Platform Overview

- Introductions
- Course Schedule
- Java Overview
- Java Platforms
- OpenJDK
- Licensing

- Java in Server Environments
- The Java Community Process Java Syntax and Class Review
- Simple Java classes
- Java fields, constructors and methods
- Model objects using Java classes
- Package and import statements

Encapsulation and Polymorphism

- Encapsulation in Java class design
- Model business problems with Java classes
- Immutability
- Subclassing
- Overloading methods
- Variable argument methods Java Class Design
- Access modifiers: private, protected and public
- Method overriding
- Constructor overloading
- The instanceof operator
- Virtual method invocation
- Polymorphism
- Casting object references
- Overriding Object methods

Advanced Class Design

- Abstract classes and type generalization
- The static and final modifiers
- Field modifier best practices
- The Singleton design pattern
- Designing abstract classes
- Nested classes
- Enumerated types

Inheritance with Java Interfaces

- Java Interfaces
- Types of Inheritance
- Object composition and method delegation
- Implementing multiple interfaces
- The DAO design pattern Generics and Collections
- Generic classes and type parameters
- Type inference (diamond)
- Collections and generics
- List, set and Map
- Stack and Deque

String processing

- String manipulation with StringBuilder and StringBuffer
- Essential String methods
- Text parsing in Java
- Input processing with Scanner
- Text output and formatting
- Regular expressions with the Pattern and Matcher classes

Exceptions and Assertions

- Exceptions categories
- Standard Java Exception classes
- Creating your own Exception classes
- Using try-catch and the finally clause
- Using try-with-resources and the AutoCloseable interface
- The multi-catch feature
- Best practices using exceptions
- Assertions

I/O Fundamentals

- I/O using Java
- Reading the console input stream
- Writing to the console

- Using I/O Streams
- Chaining I/O Streams
- Channel I/O
- Reading and writing objects using Serialization

File I/O with NIO 2

- The Path interface
- The Files class
- Directory and File operations
- Managing file system attributes
- Reading, writing, and creating files
- Watching for file system changes

Threading

- Operating system task scheduling
- Recognizing multithreaded environments
- Creating multi-threaded solutions
- Sharing data across threads
- Synchronization and Deadlock
- Immutable objects

Concurrency

- Creating Atomic variables
- Using Read-Write Locks
- Thread-safe collections
- Concurrenct synchronizers (Semaphore, Phaser, and others)
- Executors and ThreadPools to concurrently schedule tasks
- Parallelism and the Fork-Join framework

Database Application with JDBC

- Layout of the JDBC API
- JDBC divers
- Queries and results
- PreparedStatement and CallableStatement
- Transactions

- RowSet 1.1 RowSetProvider and RowSetFactory
- The DAO Pattern and JDBC Localization
- Advantages of localization
- Defining locale
- Read and set locale using the Locale object
- Resource bundles
- Format messages, dates and numbers

Java EE 6: Develop Web Components with Servlets JSPs

Course Objectives

- Write servlets using the Java programming language (Java servlets)
- Understand and manage HTTP sessions in a web application
- Create servlet filters and listeners
- Write pages created with JavaServer Pages technology (JSP pages)
- Create easy-to-maintain JSP pages using the Expression Language and the JSP Standard Tag Library (JSTL)
- Use integrated development environments (IDEs) and application servers for Java EE development and deployment **Course Outline**

Introducing the Course

- Reviewing the Java SE and Java EE Curriculum
- Getting Acquainted with Other Students
- Reviewing Course Objectives
- Discussing 5 Day Course Schedule
- Describing the Format that the Class will Use
- Introducing Web Application Technologies
- Describing the Java EE 6 Web Profile

Web Application Essentials

- Describing Java Servlet Technology
- Describing JavaServer Pages Technology

- Understanting the Model-View-Controller (MVC) Architecture
- Explaining Java EE Containers and Java Application Servers
- Describing the Web Application Development Process
- Identifying the Essential Structure of a WAR File

Developing a Servlet

- Describing the HTTP Headers and Their Function
- Explaining the Request and Response Processes
- Understanding the Life Cycle of a Servlet
- Listing Injection and Lifecycle Method Annotations
- Understanding the Threading Model of a Servlet
- Developing a Servlet to Respond to Requests from the Client Browser

Handling Form Requests in Servlets

- Using HTML Forms To Collect Data From Users and Send it To a Servlet
- Understanding How Form Data Is Sent in an HTTP Request
- Developing a Servlet that Retrieves Form Parameters
- Understanding and Using HttpSession Objects
- Using Cookies for Session Management
- Using URL Rewriting for Session Management

Configuring Your Web Application

- Describing the Purpose of Deployment Descriptors
- Creating Servlet Mappings to Allow Invocation of a Servlet
- Creating and Access Context and Init Parameters
- Using the @WebServlet and @WebInitParam Annotations
- Using the ServletContextListener Interface
- Describing the Different Scopes in a Web Application
- Handling Errors Using a Deployment Descriptor

Implementing an MVC Design

- Implementing the Controller Design Element Using a Servlet
- Implementing the Model Design Element Using a POJO
- Implementing the View Design Element Using a JSP and Expression Language (EL)

- Connecting the model, View, and Controller Elements to Implement a Working MVC Solution
- Injecting a Service in a Controller

Developing Components with JavaServer Pages Technology

- Describing JSP Page Technology
- Writing JSP Code Using Scripting Elements
- Writing JSP Code Using the Page Directive
- Writing JSP Code Using Standard Tags
- Writing JSP code using Expression Language
- Configuring the JSP Page Environment in the web.xml File
- Writing an Error Page by Using JSP

Developing JSP Pages by Using Custom Tags

- Designing JSP Pages with Custom Tag Libraries
- Using a Custom Tag Library in JSP Pages
- Describing JSTL Tags

Using Filters in Web Applications

- Describing the Web Container Request Cycle
- Describing the Filter API
- Developing a Filter Class
- Configuring a Filter in the web.xml File

More Servlet Features

- Using the Asynchronous Servlet Mechanism
- Using JavaScript to Send an HTTP Request from a Client
- Processing an HTTP Response Entirely in JavaScript
- Combining These Techniques to Create the Effect of Server-push
- Handling Multipart Form Data

Implementing Security

- Describing a Common Failure Mode in Security
- Requiring that a User Log in Before Accessing Specific Pages in Your Web Application

- Describing the Java EE Security Model
- Requiring SSL Encrypted Communication for Certain URLs or Servlets

Integrating Web Applications with Databases

- Understanding the Nature of the Model as a Macro-pattern
- Implementing Persistent Storage for Your Web Applications Using JDBC or Java Persistence API

Spring MVC Content

Getting started with Spring

What is Spring?

- Overview of the Spring Framework
- Spring Modules and architecture
- A Simple Example
- Wiring Beans
- Configuring a Properties File
- Beans and Containers
- Spring Containers
- Spring Configuration File
- Spring Beans
- Using the Container
- The BeanFactory Interface
- The ApplicationContext Interface
- Singleton vs. Prototype
- Bean Naming
- Dependency Injection
- Setter Injection
- Constructor Injection
- Autowiring
- Autowiring through configuration

- Autowiring by type and by name
- Aspect-Oriented Programming

Building Spring web applications

Following the life of a request

- Setting up Spring MVC
- Writing a simple controller
- Testing the controller
- Defining class-level request handling
- Passing model data to the view
- Accepting request input
- Taking query parameters
- Taking input via path parameters
- Processing forms
- Writing a form-handling controller
- Validating forms

Rendering web views

Understanding view resolution

- Creating JSP views
- Defining a layout with ApacheTiles views

Advanced Spring MVC

- Alternate Spring MVC configuration
- Customizing DispatcherServlet configuration
- Adding additional servlets and filters
- Declaring DispatcherServlet in web.xml

Processing multipart form data

- Configuring a multipart resolver

- Handling multipart requests

Handling exceptions

- Mapping exceptions to HTTP status codes
- Writing exception-handling methods

Securing web applications

Getting started with Spring Security

- Intercepting requests
- Authenticating users

Spring in the back end

- Hitting the database with Spring and JDBC

Configuring a data source

- Using JNDI data sources
- Using a pooled data source

Using JDBC with Spring

- Working with JDBC templates

Java Persistence with Hibernate

Overview

Hibernate is an object-relational mapping (ORM) library for the Java language, providing a framework for mapping an object-oriented domain model to a traditional relational database. Hibernate solves Object-Relational impedance mismatch problems by replacing direct persistence-related database accesses with high-level object handling functions. **Benefits**

- Understand ORM and basics of Hibernate.
- Understand and implement life cycle of Hibernate Persistence and Session Factory.

- Implement Hibernate Mappings, Inheritance and Types.
- Understand Hibernate Criteria and Query Language.
- Exploring Hibernate Transactions,
- Filter and Performance. Implement Hibernate Search and Validations.
- Hibernate with NoSQL and Spring.

React JS

1. Welcome to React

• Obstacles and Roadblocks
• React's Future
• Working with Files
i. React Developer Tools
ii. Installation Node JS

2. Emerging JavaScript

• Declaring Variable in ES6
• Arrow function
• Transpiling ES6
• ES6 Objects and Arrays
• Promises
• Classes

3. Pure React

• Page Setup
• The Virtual DOM
• React Elements
• React DOM
• Children
• Constructing Elements with Data
• React Components
• DOM rendering

4. React with JSX

• React Elements as JSX
• Babel
• Intro to Webpack

5. Props, State and the Component Tree

• Property Validation
• Refs
• React State Management
• State within component Tree

6. Enhancing Components

• Component Lifecycle
• JavaScript Library Integration
• Higher-Order Components
• Flux

7. Redux

• State
• Actions
• Reducers
• The Store
• Action creators
• Middleware

8. React Redux

• Explicitly Passing the Store
• Passing Store via Context
• Presentation Versus Container Components
• The React Redux Provider
• React Redux Connect
9. React Router

• Incorporating the Router

• Nesting Routes
• Route Parameters

10. React on the Server