

Course 437: Integrating JavaServer Faces, Hibernate, and Spring (5 days)

Course Description...

Created in collaboration by several leading J2EE and JSF authors and industry experts, this intensive course will give you the skills to design and build scalable, secure, maintainable web applications.

Helping front-end developers, back-end developers, and architects understand how they can get the most out of JavaServer Faces (JSF), this course explores the official standard for simplifying Java web development. Your ROI instructor will explain what JSF is, how it works, and how it relates to other frameworks and technologies like Struts, Servlets, JSP, and JSTL.

JavaServer Faces (JSF) provides event driven, component –based technology for developing J2EE web applications. This technology greatly simplifies developing web applications.

Hibernate is a powerful enabler that addresses object/relational persistence in the Java world. Hibernate offers all the advantages of developing in Java plus a comprehensive suite of capabilities for mapping object-oriented features to the relational model. This course tells you what you need to know to design and build your own Hibernate-enabled applications. You'll learn the details of the key Hibernate capabilities and how to leverage their strengths.

Spring makes J2EE development easier by simplifying commons tasks and encouraging strong design based on programming to interfaces. Spring makes your application easier to configure and reduces the need for many J2EE design patterns. Spring puts the OO design back into your J2EE application, and it integrates nicely with JSF.

The combination of JSF, Spring and Hibernate is a powerful web development stack.

You will learn to...

- Write web applications that take advantage of the FacesServlet, FacesContext and Action Java classes to control the user experience of the web application.
- Write JSF applications that gather and update information from external application servers such as EJBs, CORBA servers, and database servers.
- Create and use custom Tag Libraries in JavaServer Pages.



- Understand the use of the standard JSF Validators.
- Understand the use of the standard JSF Data Conversion classes.
- Take advantage of the JSF architecture that supports rendering output in several formats from the same application. Such as: HTML, WML, XML, etc.
- Explain how the issues associated with object persistence in a relational model are addressed by Hibernate
- Understand the relationships between Java, JSF, Spring, and Hibernate
- Map Java classes to relational tables.
- Capture both relational and inheritance associations in metadata using XML.
- Create and use mappings between Java classes and relational databases.
- Understand how identity and keys are handled in Hibernate.
- Understand the persistent object lifecycle and how that relates to transactions and concurrency.
- Explain the issues associated with complex frameworks such as J2EE and how Spring addresses those issues
- Write applications that take advantage of the Spring container and the declarative nature of assembling simple components into applications.
- Understand how to use Hibernate and JSF within the Spring framework

Who should attend...

This an advanced level training course, designed for J2EE developers that need to further extend their skills in web development.

Prerequisites...

Attendees should have extensive experience with developing J2EE applications. The following courses are recommended prerequisites for this course:

- Essential Java (Crs. # 430)
- Building J2EE Web Applications using Servlets, JSPs, JDBC, and Security (Crs. #432)

See next page for a detailed course outline...



Course Outline...

Introduction to JavaServer Faces7

- Benefits & Concerns of JSF
- JSF Development Roles
- JSF vs. Struts
- JSF Framework Structure

JSF Architecture

- JSF Architecture Overview & Physical Components
- Physical Components
- How Does JSF Work?
- Lifecycle Phases
- Writing a JSF Application

Request Processing

- Request Processing
- Page Navigation
- JSF Objects
- The FacesContext Object; The UIViewRoot Object
- The UIComponent Interface
- The ExternalContext Object; The Application Object
- Accessing Backing Beans

Simple JSF User Interface Components

- Simple JSF User Interface Components
- JSF Custom Tag Libraries
- The HTML Tag Library
- The <h:commandButton> Tag; The <h:commandLink> Tag; The <h:inputText> Tag; The <h:outputText> Tag; The ; <h:outputLabel> Tag; The <h:messages> Tag; The <h:selectBooleanCheckbox> Tag
- Panels

JSP 2.0 EL Expression Language

- JSP 2.0 EL Expression Language
- What is EL?
- EL Basics & EL Identifiers
- EL Implicit Objects
- EL Operators & EL Notes
- JSTL
- The <c:out> tag; The <c:set> Tag; The <c:remove> Tag; The <c:catch> Tag; JSTL Logic Tags; The <c:if> Tag; The <c:choose> Tag; The <c:forEach> Tag

Event Handling

- The Java 2 Event Model
- The JSF Event Model
- The ActionEvent Class
- The ValueChangeEvent Class
- Event Listeners in JSF
- The ActionListener
- The ValueChangeListener



Data Validation

- Data Validation & Validation
- Using Standard Validators
- Handling Error Output
- `<f:validateLength>`; `<f:validateLongRange>` & `<f:validateDoubleRange>`
- Custom Validators; Simple Validators
- Creating the Validator Class
- Creating the Tag Handler
- Registering the Validator Class
- Using the Validator Class

Advanced Data Validation

- Advanced Data Validation
- Typical Validator Problems
- The StateHolder Interface
- The `saveState()` Method
- The `restoreState()` Method
- The transient Attribute
- Validating Dependent Fields
- Class Diagram
- The MultiFieldValidator
- The DependentFieldValidator & DependentFieldValidatorTag
- Updating the `faces-config.xml` File
- Using the New Validator
- Implement MultiFieldValidator
- Using the Custom JSP Tag

Data Conversion and Rendering

- Data Conversion and Rendering
- Converters & Custom Converters
- Writing the Converter Class
- Renderers
- Renderers in the “Real World”
- Creating the Renderer Class
- Creating the Tag Handler
- The TLD for the Tag Handler
- Registering the Renderer
- Using the Component

Hibernate Overview

- ORM Mapping Issues
- Hibernate Architecture
- Persistence, Identity, and Equality in Hibernate
- Domain Models and Metadata Options

Hibernate QuickStart

- Basic Mapping
 - Class/properties to Table/Columns
- Basic Configuration
- Mapping a POJO to a Database



- CRUD operations
- Basics of Hibernate Session
- Working with Persisted Objects

Types in Hibernate

- Hibernate Type System
- Entity and Value Types
- Components
- Collections
- Implementing Complex Types

Associations in Hibernate

- Relational Associations
 - Composition
 - Bidirectional
 - Unidirectional
- Inheritance Associations
 - Mapping Strategies
 - Single, Class, and Concrete Tables

Working with Persisted Objects

- Object states and lifecycles
- Hibernate Persistence Manager API

Introduction to the Spring Framework

- Inversion of Control
- Dependency Injection
- Spring Overview
- Spring Application Architectures
- Spring Container
 - Managing the Container
 - Access to Services and Resources
 - Application Contexts
- Beans as Components
 - Beans and Factories
 - XML Bean Configuration
 - Bean Definition and Dependencies
 - Bean Lifecycle
- Customization Options
 - Post-Processors
 - Property Editors

Spring and Persistence

- Data Access Pattern
- Overview of Persistence Layer and Transactions
- Hibernate
 - Spring - Hibernate Architecture
 - ORM Mapping Overview
 - DAO Implementation
 - Working with Hibernate DAOs in Spring
 - Hibernate Template



Spring and the Web

- Spring/Web Framework Architecture
- Spring and JSF
 - Spring/JSF Architecture
 - Integrating JSF into Spring

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