Core JavaServer Faces (with Spring and Hibernate Overview) Developer's Workshop

Course Summary

Description

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 new official standard for simplifying Java web development, explaining what JSF is, how it works, and how it relates to other frameworks and technologies like Struts, Servlets, JSP, and JSTL.

Also provided is coverage of all the standard components, renderers, converters, and validators, along with advice on how to use them to create solid applications. Working in a hands-on, lab-intensive environment, students will explore and use complex user interface layouts, prototyping, and integrating templates with back-end model objects. Also covered are advanced techniques like internationalization, integration with Struts, and extending JSF with custom components, renderers, converters, and validators.

Topics

- Introduction to JavaServer Faces
- JSF Architecture Overview
- Request Processing
- Simple JSF User Interface Components
- JSP 2.0 EL Expression Language
- Event Handling
- Data Validation
- Advanced Data Validation
- Data Conversion and Rendering
- Custom Graphic controls
- Spring Overview

Audience

This course is designed for experienced Java developers who want to use and leverage JSF, Spring and Hibernate to build robust web applications.

Prerequisites

Students should have at least six months prior Java development experience, including some experience using Server Side technologies (Servlets/JSPs.) Students should be comfortable creating a servlet and JSP.

Duration

Five days

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically

ProTech
protechtraining.com

"Charting the Course ...

... to Your Success!"

Core JavaServer Faces (with Spring and Hibernate Overview) Developer's Workshop

Course Outline

I. Introduction to JavaServer Faces
- A. Introduction to JavaServer Faces
- B. What is JavaServer Faces?
- C. Benefits of JSF
- D. JSF Development Roles
- E. Where Does JSF Fit?
- F. JSF vs. Struts
- G. System Requirements
- H. Concerns With JSF
- I. JSF Framework Structure

II. JSF Architecture Overview
- A. JSF Architecture Overview
- B. Physical Components
- C. How Does JSF Work?
- D. The FacesServlet
- E. The Lifecycle Object
- F. Lifecycle Phases
- G. Writing a JSF Application
- H. A Sample JSF Application
- I. Structure of a Web Application

III. Request Processing
- A. Request Processing
- B. Page Navigation
- C. JSF Objects
- D. The FacesContext Object
- E. The UIViewRoot Object
- F. The UIComponent Interface
- G. The ExternalContext Object
- H. The Application Object
- I. Accessing Backing Beans

IV. Simple JSF User Interface Components
- A. Simple JSF User Interface Components
- B. JSF Custom Tag Libraries
- C. The HTML Tag Library
- D. The <h:commandButton> Tag
- E. The <h:commandLink> Tag
- F. The <h:inputText> Tag
- G. The <h:outputText> Tag
- H. The <h:outputLabel> Tag
- I. The <h:messages> Tag
- J. The <h:selectBooleanCheckbox> Tag
- K. Panels

V. JSP 2.0 EL Expression Language JSP 2.0 EL Expression Language
- A. What is EL?
- B. EL Basics
- C. EL Identifiers
- D. EL Implicit Objects
- E. EL Operators
- F. EL Notes
- G. JSTL
- H. The <c:out> tag
- I. The <c:set> Tag
- J. The <c:remove> Tag
- K. The <c:catch> Tag
- L. JSTL Logic Tags
- M. The <c:if> Tag
- N. The <c:choose> Tag
- O. The <c:forEach> Tag

VI. Event Handling
- A. Event Handling
- B. The Java 2 Event Model
- C. The JSF Event Model
- D. The ActionEvent Class
- E. The ValueChangeEvent Class
- F. Event Listeners in JSF
- G. The ActionListener
- H. The ValueChangeListener

VII. Data Validation
- A. Data Validation
- B. Validation
- C. Using Standard Validators
- D. Handling Error Output
- E. <f:validateLength>
- F. <f:validateLongRange> & <f:validateDoubleRange>
- G. Custom & Simple Validators
- H. Creating the Validator Class
- I. Creating the Tag Handler
- J. Registering the Validator Class
- K. Using the Validator Class

VIII. Advanced Data Validation
- A. Advanced Data Validation
- B. Typical Validator Problems
- C. The StateHolder Interface
- D. The saveState() Method

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically

Core JavaServer Faces (with Spring and Hibernate Overview) Developer's Workshop

Course Outline (cont'd)

E. The restoreState() Method
F. The transient Attribute
G. Validating Dependent Fields
H. The MultiFieldValidator
I. The DependentFieldValidator
J. The DependentFieldValidatorTag
K. Updating the faces-config.xml File
L. Using the New Validator
M. Implement MultiFieldValidator
N. Using the Custom JSP Tag

IX. Data Conversion and Rendering
A. Data Conversion and Rendering
B. Data Conversion vs. Rendering
C. Converters & Custom Converters
D. Writing the Converter Class
E. Renderers
F. Renderers in the "Real World"
G. Creating the Renderer Class
H. Creating the Tag Handler
I. The TLD for the Tag Handler
J. Registering the Renderer
K. Using the Component

X. Custom Graphic Controls
A. Custom Graphic Controls
B. Dynamic Graphics and the Web
C. The <img> Tag
D. Example – Creating a Bar Chart
E. Creating the ChartServlet
F. Update web.xml
G. Create the <chartItem> Component
H. Creating the ChartItemTag Class
I. The UIChartItem Class
J. Create the <barChart> Component
K. The UIBarChart Class
L. The BarChartRenderer Class
M. Final Clean Up

XI. Spring Overview
A. The Spring Framework
1. What is Spring?
2. Benefits of Spring
3. Spring Architecture
4. Spring Basics
5. Application Context
6. Example using Classpath Application Context
7. Configuring a Bean
8. Defining simple beans
9. Accessing beans from application context
10. Configuring Collaborators
11. Configuration Properties (non collaborators)
12. More complex standard properties
13. Spring Property Editors
14. Properties, Lists, Maps, and Sets are Supported
15. Create and Destroy methods
16. Three Ways to Wire up Collaborators
17. Configuring Collaborators via constructor
18. Overview of Aspect-oriented Programming247
19. Join Points
20. Pointcuts
21. Inter-type declarations
22. Creating around advice with an interceptor
23. Binding the around interceptor
24. Using the AOP class
25. Much more to AOP
26. Proxy Factories
27. JDBC support & template
28. JDBC Helper Objects
29. Defining base SQL Query Object
30. Create concrete Query classes
31. Using SQL Update for Deleting
32. Using SQL Update for Inserting
33. JdbcDAO
34. Defining a JdbcDAO class
35. JdbcDAO initializing collaborators
36. JdbcDAO using collaborator
37. Configuring JdbcDAO object
38. Configuring JdbcDAO object: passing datasource
39. Using AOP to apply Spring transaction support
40. Types of Transaction Management
41. Configuring a JTA transaction manager

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically

Core JavaServer Faces (with Spring and Hibernate Overview) Developer's Workshop

Course Outline (cont'd)

B. Spring Basics
   1. Spring Basics
   2. Inversion of Control (IoC)
   3. Dependency Pull IoC
   4. Contextualized Dependency Lookup
   5. Our First Spring Application
   6. Creating the Business Objects
      Creating the Dependent Objects
      Registering Objects
   7. Developing Workflow Objects Using Contextualized Dependency
   8. A Better Use of IoC
   9. The BeanFactory
   10. The ApplicationContext Interface
   11. The Spring XML Configuration File
   12. The <bean> Element
   13. Injecting Dependencies
   14. Collections Properties

C. Advanced Spring Beans
   1. Advanced Spring Beans
   2. Constructor Dependency Injection
   3. The Autowire Attribute
   4. Lifecycle Interfaces
   5. The InitializingBean Interface
   6. The DisposableBean Interface
   7. Knowing Who You Are
   8. Lifecycle Interfaces of the BeanFactory
      Property Editors
   9. What is Method Injection?
   10. Method Injection

XII. Hibernate

A. Overview of Hibernate Hibernate Overview & Features
   1. Down to Business
   2. OR mapping with Hibernate
   3. Plain old java object no Hibernate
   4. Mapping file
   5. Setting up Relationships
   6. Group contains Users
   7. User has Contact Info
   8. User associated with roles
   9. Employee is a User
   10. Types of queries in Hibernate

XIII. DBUnit

A. DBUnit Overview
   1. Testing DAO object that access this
   2. Using DBUnit (JUnit) Step By Step
   3. Step 1, Subclass org.dbunit.DatabaseTestCase
   4. Step 2, Override the setUp method
   5. Step 3, Create Dataset XML file
      Step 4, getConnection() to provide db connection
   6. Step 5, getSetUpOperation() and getTearDownOperation()
   7. Step 6, Define one or more testXXX methods
   8. Step 7, Release any resources by overriding tearDown()
   9. Common Pitfalls and Strategy for Integration Testing with Cactus
      DBUnit Ant Support

B. Spring and Hibernate
   1. Using Queries with Spring/Hibernate
   2. Spring IOC and Hibernate
   3. Using named queries
   4. Problem
   5. Managing Transactions

XIV. Appendix A

A. (Short) Introduction to ANT
B. What is ANT?
C. What are the benefits of ANT? Installing ANT
D. Using ANT
E. Writing a build.xml File
F. Running an ANT Script

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically