JavaServer Faces v2.0

Course Summary

Description

This comprehensive course shows Java programmers how to build web applications with JavaServer Faces 2.0. We develop the best-practice concepts that are formalized by the JSF architecture, from model/view/controller to the UI component framework and request-handling lifecycle. Students start to discover that there is a "JSF way" of doing things, and we learn not just APIs and tag libraries but the habit of slicing application logic into its most reusable forms: managed beans, event listeners, converters, validators, and more.

Students acquire a firm command of JSF development, learning to work with JSF's list and table components, building reusable composite components, and building Ajax applications. Simple, high-level Ajax functionality is covered, and students also work more directly with JSF's JavaScript API and resource-management framework.

Objectives
At the end of this course, students will be able to:

- Understand the purpose and scope of the JSF architecture
- Build web applications using JSF's FacesServlet, faces-config.xml, and the JSF request/response lifecycle
- Use Facelets tag libraries to build JSF views
- Use managed beans to encapsulate form handling and server-side presentation logic
- Implement control logic as JSF event listeners or action methods
- Use validators and converters to implement a validation phase for a JSF application
- Build composite UI fragments or custom components using Facelets
- Build Ajax applications with JSF: client-side behaviors and partial requests and responses followed by DOM updates

Topics

- Overview
- Lifecycle
- UI Components
- Page Navigation
- Managed Beans
- Scopes
- Dependency Injection
- Facelets
- Events and Listeners
- Lists and Tables
- Converters
- Validators
- Resources
- Composites
- Ajax
- The JSF JavaScript API

Audience

This course is intended primarily for experienced Java application developers. Page authors, component developers, and others who may have little or no Java experience (but perhaps are stronger on HTML, JavaScript, and JSP) may well find this to be a valuable training experience, though without solid Java skills many of the coding exercises will be difficult to follow.

Prerequisites

- Java programming experience is essential to understanding the JSF API as presented
- General understanding of servlets and JSP is recommended, but not required
- Basic knowledge of XML will be helpful, as will any previous experience with HTML.

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!"

JavaServer Faces v2.0

Course Outline

I. Overview
- A. Java EE and Web Applications
- B. Perspectives: Servlets and JSP
- C. Perspectives: MVC Frameworks
- D. Perspectives: AWT and JFC
- E. JSF Value Proposition
- F. JSF Configuration
- G. Issues with JSP and JSF
- H. Facelets

II. Lifecycle
- A. The JSF Request/Response Cycle
- B. Lifecycle Phases
- C. Phase Listeners
- D. The FacesContext Class
- E. Who Does What
- F. Partial Request Cycles

III. UI Components
- A. The UIComponent Class
- B. Behavioral Interfaces
- C. The Core and HTML Tag Libraries
- D. Relationship to CSS
- E. ID, Client ID, and Label
- F. UISelectItem(s)
- G. Navigating the UI Tree
- H. The binding Attribute

IV. Page Navigation
- A. View Selection
- B. Navigation Rules
- C. Implicit Navigation
- D. Problems with POSTback
- E. Post/Redirect/Get
- F. Support for HTTP GET
- G. Conditional Navigation

V. Managed Beans
- A. JavaBeans and JSF
- B. Backing Beans
- C. Configuring Managed Beans
- D. @ManagedBean and Related Annotations
- E. The Unified Expression Language
- F. Value and Method Expressions
- G. Implicit Objects

VI. Scopes
- A. Managed-Bean Scopes
- B. Lifecycle Annotations
- C. View Parameters
- D. The Flash

VII. Dependency Injection
- A. Managed Properties
- B. Values, Lists, and Maps
- C. Using Dynamic Expressions
- D. Dependencies and Bean Scopes
- E. The @ManagedProperty Annotation

VIII. Facelets
- A. Migrating from JSP
- B. View Definition Languages
- C. Facelets
- D. Tag Libraries
- E. Writing and Using Custom Tags

IX. Events and Listeners
- A. JSF Event Model
- B. Event Types and Timing
- C. Event Queueing
- D. ActionEvent and ActionListener
- E. Action Methods
- F. Connecting Controllers to Beans
- G. ValueChangeEvent and ValueChangeListener
- H. Deferring Event Processing
- I. Limitations of FacesListeners

X. Lists and Tables
- A. Working with Collections
- B. Why We Don't Use <c:anything> <c:anyMore>
- C. <ui:repeat> vs. <c:forEach>
- D. <h:dataTable>
- E. Defining Columns and Facets
- F. One Command Per Row
- G. Reading the Row Number
- H. Pseudo-Maps
- I. Working with Persistent Data
- J. Concurrency and Caching
- K. Limiting the Scope of Queries
- L. Paging

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

JavaServer Faces v2.0

Course Outline (cont'd)

XI. Converters
- A. The Converter Interface
- B. Life of a Datum
- C. Standard Converters
- D. Custom Converters
- E. The @FacesConverter Annotation
- F. Timing of Conversion
- G. Representing Persistent Objects by ID

XII. Validators
- A. The Validator Interface
- B. Standard Validators
- C. Using Regular Expressions
- D. Producing Error Messages
- E. Message Keys
- F. Presenting Error Messages
- G. Posting Error Messages from Anywhere
- H. Custom Validators
- I. The @FacesValidator Annotation
- J. Validating Multiple Inputs
- K. JSR-303 Support: "Bean Validation"

XIII. Resources
- A. Resource Libraries
- B. Deploying Images, Scripts, and Stylesheets
- C. Addressing Resources

XIV. Composites
- A. Limitations of Custom Tags
- B. Composite Components
- C. Encapsulation
- D. Deploying and Using Composites
- E. Interface and Implementation
- F. Impact on the UI Tree
- G. Attributes
- H. Retargeting

XV. Ajax
- A. What is Ajax?
- B. The XMLHttpRequest Object
- C. Ajax and the JSF Lifecycle
- D. Using <f:ajax>
- E. execute and render Attributes
- F. Ajax Listeners

XVI. The JSF JavaScript API
- A. The JSF JavaScript API
- B. Trigering Ajax Requests
- C. Refining <f:ajax> with Callbacks
- D. onevent and onerror Attributes
- E. The Ajax Request/Response Process
- F. Using Hidden Inputs
- G. Other JavaScript Functions

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