Mastering JEE with JSF2, EJB3, JPA, and Web Services

Course Summary

Description
In this course, we make participants productive JEE developers using the proven technologies in the Java enterprise stack. The course addresses all areas of modern JEE development: we start by diving into Java ServerFaces 2 (JSF2), a modern user interface framework. We next go to the business tier, working with Enterprise Java Beans, exploring the creation of Session Beans, transaction management, and Message Driven Beans. Next, we approach Java persistence with JPA. At the end we cover the application integration and interoperability with JAX-WS web services. For this lab exercises in this course we use IBM Rational Application Developer V8 and WebSphere Application Server. If desired, the course can be customized to use other middleware.

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

- Develop JEE applications.
- Develop JSF2 Web User Interfaces.
- Implement Business Logic using EJB3.
- Implement persistence using JPA.
- Achieve interoperability through JAX-WS web services.

Topics
- Overview of JSF 2
- The JSF 2 Lifecycle
- JSF 2 User Interface Component Model
- JSF 2 Event Handling
- JSF 2 Validation Framework
- Building JSF 2 Custom Components
- Building JSF 2 Renderers
- Internationalization (I18N) and Localization (L10N)
- JSF 2 and Ajax
- Java EE 5 Architecture
- Enterprise JavaBeans (EJB) Architecture
- Developing Session Beans
- Java Message Service (JMS)
- Message-Driven Beans and JMS destinations
- Aspect Oriented Programming (AOP) in JEE
- EJB Transactions
- Java Persistence API (JPA)
- Java Persistence Query Language (JPQL)
- Web Services with JAX-WS

Audience
This course is designed for anyone who develops enterprise Java applications.

Prerequisites
Students should have basic knowledge of Java and relational databases.

Duration
Five days
Mastering JEE with JSF2, EJB3, JPA, and Web Services

Course Outline

I. Overview of JSF 2
   A. Overview of JSF 2 Framework
   B. First look at a JSF 2 sample application
   C. Requirements to run a JSF 2-based Web application

II. The JSF 2 Lifecycle
    A. The JSF 2 controller servlet
    B. Request tree construction
    C. Applying request values
    D. Handling request events
    E. Processing validations
    F. Invoking application and updating models
    G. Rendering responses

III. JSF 2 User Interface Component Model
     A. Overview of JSF 2 component model classes
     B. Rendering model
     C. Conversion model
     D. Event model
     E. Validation model
     F. Standard user interface components
     G. JSF 2 per-request context info

IV. JSF 2 Event Handling
    A. JSF 2 event model
    B. Event objects and event listeners in JSF 2
    C. Writing listeners for UI components

V. JSF 2 Validation Framework
   A. Standard validation classes
   B. Validation class registration
   C. Custom validator implementation
   D. Data conversion

VI. Building JSF 2 Custom Components
    A. Custom components overview
    B. Java classes for building custom components
    C. Custom renderer
    D. Implementing custom components
    E. Using custom components in your page

VII. Building JSF 2 Renderers
     A. Overview of rendering process

VIII. Internationalization (I18N) and Localization (L10N)
     A. Overview of I18N and L10N in Java
     B. Supporting I18N and L10N in JSF 2
     C. Localizing error messages in validation

IX. JSF 2 and Ajax
    A. Overview of Ajax
    B. Ajax in JSF 2
    C. The <f:ajax> tag and its attributes

X. Java EE 5 Architecture
   A. Overview of the Java EE 5 Architecture
   B. Design issues in implementing e-business and enterprise applications
   C. Design patterns for implementing e-business and enterprise applications
   D. Java EE 5 Components, Containers and Connectors
   E. Overview of Java EE 5 APIs
   F. Major roles in designing, developing, and deploying Java EE 5 applications
   G. Application packaging and deployment using WAR, JAR and EAR files

XI. Enterprise JavaBeans (EJB) Architecture
    A. EJB Architecture
    B. Designing and implementing Enterprise JavaBeans
    C. Local vs. remote EJB
    D. Entities
    E. Session beans
    F. Message-driven beans
    G. Interceptors
    H. Annotations and code injection

XII. Developing Session Beans
     A. Session context
     B. Designing and implementing session beans
     C. The life cycle of a stateless session bean
     D. The life cycle of a stateful session bean
Mastering JEE with JSF2, EJB3, JPA, and Web Services

Course Outline

XIII. Java Message Service (JMS)
A. Introduction to messaging systems
B. Point-to-point model
C. Publish-and-subscribe model
D. Building custom finders
E. Polymorphism
F. Automatic storage of query results in plain old Java objects (POJO)
G. Lazy loading for performance

XIV. Message-Driven Beans and JMS destinations
A. Client view of MDB
B. MDB lifecycle
C. MDB with JCA resource adapter
D. Developing Message-Driven Beans (MDB)

XV. Aspect Oriented Programming (AOP) in JEE
A. Interception methods
B. Building Interceptor Classes

XVI. EJB Transactions
A. Container vs. Bean managed transactions
B. Container-managed transaction attributes
C. Transaction propagation
D. Accessing the UserTransaction service using injectors
E. Invoking the UserTransaction service using JTA
F. Transaction Isolation attributes
G. SessionSynchronized Stateful Session Beans
H. Transactions and exceptions
I. Using annotation to specify transaction attributes

XVII. Java Persistence API (JPA)
A. Java Persistence API (JPA)
B. Entity manager
C. Entity context
D. Modeling entity relationships
E. Modeling entity inheritance
F. Mapping persistent objects
G. Annotations
H. Transaction propagation

XVIII. Java Persistence Query Language (JPQL)
A. Overview of JPQL
B. Writing portable queries based on Entities
C. Building native queries for performance

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.