Developing Business Components With Enterprise JavaBeans

Duration:	3 days
Type:	advanced

Description

This is an advanced JEE course which covers all aspects of Enterprise JavaBean development. Both traditional (pre version 3) and modern models for developing EJB's are covered in depth.

The course can be delivered using a wide range of containers, including WebLogic, GlassFish, JBoss and WebSphere. By default Eclipse and WebLogic are used.

Prerequisites

Delegates must have several years Java programming experience, preferably with prior experience of distributed application development.

List of Modules

JEE Architecture

- The evolution of JEE from CORBA
- JEE component types and services
- Support for Web Services in JEE5
- Responsibilities of an EJB Container
- Packaging and deploying JEE modules
- The JEE architecture for Enterprise Apps
- Alternative architectures for JEE
- Third party tools and frameworks

Core Concepts of Remote Invocation

- The need for distributed components in the JEE Architecture
- The evolution of Enterprise JavaBeans from CORBA
- Creating a Remote interface and implementation
- Rules for parameter passing in remote methods
- Registering a server-side object in the RMI Registry
- Writing a client which finds and uses remote objects
- Generating and handling remote exceptions
- Problems and anti-patterns when designing components

The Architecture of Enterprise JavaBeans

- A historical perspective on the evolution of the EJB specification
- Lifecycle and threading are managed by the container
- Enterprise services are configured declaratively
- Support for persistence and messaging is integrated

The Traditional Model for Developing EJB's

- Creating the Remote and Home interfaces
- Writing the bean implementation class
- Configuring the bean in the deployment descriptor
- Adding Local and LocalHome interfaces
- Passing setup info via the deployment descriptor
- Making the bean stateful and adding *create* methods
- Configuring and packaging cooperating beans

Enterprise JavaBeans in EJB3 and JEE5

- Problems with EJB development prior to JEE5
- The drive for simplification in EJB 3
- Replacing XML configuration with annotations
- Using sensible defaults to reduce code and XML
- Injecting dependencies via the *Resource* annotation
- Enabling Aspect Oriented Programming via Interceptors
- How the Java Persistence API replaces Entity Beans

Upcoming features in EJB 3.1:

- All interfaces become optional
- Beans can be declared as singletons
- The timer service is enhanced
- Packaging is simplified

Writing Session Beans in EJB3

- Annotations defined by the EJB3 specification
- Creating local and remote interfaces in EJB3
- Creating an EJB3 bean implementation class
- Writing and annotating lifecycle methods
- Automatically injecting resources into the bean
- Using the methods provided by the context object
- Using the Timer Service to schedule events
- Finding and using other session beans

Writing Message Driven Beans in EJB3

- Point to Point verses Publish/Subscribe messaging
- The structure of a JMS message and supported payloads
- Sending or publishing a message in a JEE5 component
- Creating a Message Driven Bean and processing messages
- Understanding the lifecycle and limitations of an MDB
- Using message linking to route messages to the same instance

Using the Java Persistence API

- Comparing the JPA to Hibernate
- Creating simple persistent classes
- Mapping classes to the database via annotations
- Mapping classes to the database via XML
- Packaging and deploying a Persistence Unit
- Acquiring an 'EntityManager' in JSE and JEE code
- Using an extended persistence context in Stateful Session Beans
- Creating, persisting, removing and detaching objects
- Writing simple and complex queries using JPA Query Language
- Mapping logical units of work to business transactions

Advanced Mapping Tasks in the JPA

- Choosing and specifying a key generation strategy
- Mapping a single class to multiple tables
- Mapping multiple classes to the same table
- Using collections to hold basic values
- Implementing one-to-many relationships
- Implementing many-to-many relationships
- Mapping inheritance between persistent classes
- Optimizing associations between objects

Interceptors in EJB3

- Introducing interceptors and Aspect Oriented Programming
- Using interceptors to implement cross cutting concerns across EJB's
- Attaching interceptors to EJB's via annotations and/or XML
- Intercepting business methods, life cycle events and exceptions
- Using dependency injection to pass resources into interceptors

Building Web Services using EJB3

- Review of Web Services core concepts
- Namespaces, XMLSchema, SOAP and WSDL
- Understanding how JEE5 supports Web Services via JAX-WS
- Configuring a POJO as a Web Service Endpoint
- Configuring a Session Bean as a Web Service Endpoint
- Using JSR 181 annotations to configure the WSDL
- Using JAXB to serialize parameters