Enterprise Integration Using Spring 2.5

Duration:	4 days
Type:	intermediate

Description

This course provides a comprehensive introduction to the Spring framework. Delegates start with Springs role as a Lightweight Container, progress to its support for Aspect Oriented Programming and finish with its benefits as an Integration Platform. A wide range integration modules can be covered as required, including database access, remoting and web services.

Prerequisites

Delegates should have a minimum of two years Java programming experience. Additionally for each type of integration to be covered delegates must have experience in the relevant underlying frameworks. For example in the area of integrating ORM frameworks delegates must have used Hibernate and/or the JPA.

List of Modules

Introduction to Spring

- Problems with normal JEE development
- JEE patterns and anti-patterns
- Spring as a lightweight container
- Spring as an integration platform
- Spring as a Web Framework

Introduction to Dependency Injection

- Why conventional classes cant be unit tested
- Injecting dependencies via constructor arguments
- Injecting dependencies via JavaBean properties
- Externalizing dependencies in a configuration file

Spring as a Lightweight (IoC) Container

- Reading bean definitions from an XML file
- Configuring beans via constructor arguments
- Configuring beans via JavaBean properties
- Support in Spring for injecting collections
- Creating beans via factory methods and objects
- Nesting bean definitions and using auto-wiring
- Using inheritance to simplify bean definitions
- Choosing between singleton and prototype scope

Advanced Dependency Injection in Spring

- Overriding inherited settings in bean definitions
- Replacing method definitions in beans
- Loading entries from properties files
- Listening for Spring specific events
- Creating property editors to support user defined types
- Writing post-processors for beans and bean factories

Annotation Based Dependency Injection

- Configuring a Spring project to use annotation based injection
- Using *@Autowired* for autowiring by type
- Using *@Autowired* and *@Qualifier* for autowiring by name
- Using *@Resource* for autowiring by name
- Declaring beans using *@Component*, *@Service* and *@Controller*

Extending the Spring Configuration Format (Optional)

- Understanding the core concepts behind schema extensions
- Using the built in extensions for utility beans
- Writing your own schema extensions

Aspect Oriented Development in Spring

- The notion of cross-cutting concerns
- Understanding Aspects, Advice and Pointcuts
- Different possible approaches to weaving
- How Spring implements AOP using proxies
- Integration of AspectJ in Spring V2
- The AspectJ pointcut expression syntax

Springs Role as an Integration Platform

- How Spring implements services via AOP
- Benefits of allowing Spring to control transactions and security
- An overview of frameworks that can be integrated via Spring

Database Integration in Spring

- Using *SimpleJdbcTemplate* to simplify running SQL queries
- Support for creating and managing distributed transactions
- Configuring and using Hibernate via bean definitions and templates
- Configuring and using the Java Persistence API (JPA)

Web Application Development with Spring MVC

- Overview of the Spring MVC architecture
- Marking classes as Controllers
- Mapping URL's to methods
- Validating input and error handling
- Redirecting the request to a view
- Using the JSP tag libraries
- Overview of Spring Web Flow