

The Spring Framework



Abstract

Spring is one of the most popular Java frameworks today. Unlike Java EE, Spring works well with any Java application from a simple applet to a sophisticated enterprise application, and increases development productivity, code maintainability and extensibility while improving code testability and application quality.

Spring is a layered framework that contains a complete lightweight container as its core and many additional packages that can flexibly integrate into Spring and extend its functionality.

This course teaches Spring's philosophy, focuses on the internals of the Spring engine and covers some of the most popular additional packages that extend the framework. The course includes comprehensive exercises that ensure participants gain all the information and hands-on needed to get familiar with the framework and apply it successfully in their projects.

Target Audience

JavaSE/EE developers, team leaders and Architects

Prerequisites

Familiarity with the Java language.

Content:

Introduction to Spring (1.5 hours):

- History.
- IoC and Dependency Injection.
- Spring philosophy.
- AOP
- Testability.

Inversion of Control (4 hours):

- Introduction.
- The importance of decoupling.
- Spring IoC container and the BeanFactory.
- Dependancies.



Injection types.
Bean Scopes.
Bean Lifecycle and Environment.
Bean definition inheritance.

Extending the Container (2 hours):

Controlling the Bean creation using FactoryBeans.
Bean post processors.
Bean factory post processors.

The Application Context(3 hours):

Introduction to ApplicationContext.
BeanFactory vs. ApplicationContext.
Annotation based configuration.
ApplicationContext as I18N facility.
ApplicationContext as Event facility.
ApplicationContext as Resource loader.
Integrating the ApplicationContext into complex JavaEE applications.
Stereotypes and Auto Detected Components.

Sophisticated Bean Features (2 hours):

Validation.
Working with BeanWrappers.
Property Editors.

Aspect Oriented Programming - AOP (1.5 hours):

Introduction.
Motivation.
Concepts and Terminology.

The Spring AOP Model (4 hours):

Introduction.
Limitations.
Spring's Dynamic Proxy model.
AspectJ annotation support.
Aspect/Pointcut/Advise declarations.
The AspectJ pointcut definition language.



Introductions.
Advisors.
Aspect Lifecycle.

Advanced Spring AOP (2 hours):

Understanding Spring Proxies.
Working with full AspectJ support.
Choosing the correct AOP model.
Load-Time Weaving.

Testing with Spring (2 hours):

Introduction to Unit and Integration Testing.
Unit Testing and Dependency Injection.
Spring Mock Objects.
Unit Testing utilities.
The Spring TestContext Framework

Spring Remoting - Optional (3 hours):

Introduction.
Remoting by RMI.
Remoting by Hessian or Burlap.
Remoting by HTTP Invokers.
WebServices.
Remoting by JMS.

Data Access Using JDBC - Optional (2 hours):

Introduction.
Using the JdbcTemplate.
JDBC Batch Operations.
SimpleJDBC Classes.
Operation Objects.

Data Access Using ORM(JPA, Hibernate) - Optional (2 hours):

Introduction to ORM
Introduction to JPA and Hibernate
Spring integration with JPA.
Configuring Declarative Transactions.



Transaction Management - Optional (3 hours):

- Introduction to Transactions.
- Declarative Transactions.
- Programmatic Transactions.
- Declarative vs. Programmatic.
- Resource Synchronization.

Spring MVC - Optional (4 hours):

- Introduction.
- The MVC pattern.
- Spring MVC Implementation.
- Controllers.
- Mapping Request Handlers.
- Views.
- I18N and Locale Resolvers.
- Customizing Look & Feel using Themes.
- File Uploading.
- Error Handling.
- Using Annotations.

Spring JMS integration - Optional (2 hours):

- Introduction to JMS
- JMS Configuration Using Spring.
- Sending/Receiving Messages.
- Message Driven POJOs.
- Messaging and Transactions.

Scheduling and Executors - Optional (2 hours):

- Java Scheduling.
- Using Java Timers.
- Using Quartz Timers.
- Using Task Executors.

Duration: 3 days + (2 optional days)