

Spring® is a lightweight Java™ framework for building enterprise applications. Its Core module allows you to manage the lifecycle of your objects, and the dependencies between them, via configuration metadata (either XML or annotations) and Dependency Injection / Inversion of Control. Its advanced capabilities provide support for JDBC and persistence frameworks like Hibernate® (DAO and ORM modules), Aspect-Oriented Programming (AOP module), integration with Java EE Web technologies (MVC module), security, transactions, and more.

This course is a new course based on the Spring 3 release. It includes complete coverage of the annotation based approach to configuration and the use of Java-5 capabilities that was first introduced in Spring 2.x, and which has been greatly enhanced in Spring 3. It also provides coverage of the traditional XML-based configuration that can still play an important role in existing and new projects. The course starts with the basics of Spring and in-depth coverage on using the powerful capabilities of the Core module to reduce coupling, and increase the flexibility, ease of maintenance, and testing of your applications. It goes on to cover all the important capabilities of Spring 3, including using Spring to simplify the creation of a persistence layer with JDBC and/or persistence frameworks like Hibernate and JPA. It includes coverage of advanced capabilities such as using Spring's Aspect Oriented Programming (AOP) to program cross-cutting concerns such as transactions and security. The course includes integration of Spring with Java EE Web applications, and an introduction to Spring Security v3, its architecture, and how to use it to secure both Web application request and bean invocations.

#### Course Objectives:

- Understand the core principles of Spring, and of Dependency Injection (DI)/Inversion of Control
- Use the Spring Core module and DI to configure and wire application objects (beans) together
- Understand and use the complete capabilities of the Core module, such as lifecycle events, bean scopes, and the Spring API
- Work with the DAO and/or ORM modules to create a well structured persistence layer with JDBC
- Use Springs Data Integration with JDBC and technologies such as Hibernate or JPA.
- Understand and use Spring's powerful new AOP capabilities for programming cross-cutting concerns across multiple points in an application
- Understand and use Spring's transaction support, including its easy to use tx/aop XML configuration elements and Java 5 annotations
- Integrate Spring with Java EE Web applications
- Understand the basics of Spring Security, and how to secure Web apps and Spring managed beans with it
- Understand and use Spring Web Flow 2 to define complex user interface flow in Web applications.

**Audience:** Java developers who need to work with Spring based applications.

**Prerequisites:** A good working knowledge of basic Java, JDBC, and Servlets/JSP.

**Number of Days:** 3 days

## 1. Introduction to Spring

The Challenge of Enterprise Applications  
 Shortcomings of Java/Java EE  
 What is Spring?  
 The Spring Modules  
 The Spring Distribution  
 Spring Introduction  
 Managing Beans  
 A Basic Spring Application  
 Some Bean Classes  
 Configuration Metadata  
 Declaring Beans  
 The Spring Container  
 Working with Spring  
 Why Bother?  
 Some BeanFactory Methods  
 Dependencies and Dependency Injection  
 Dependencies Between Objects  
 Dependency Inversion Principal  
 Dependency Injection (DI) in Spring  
 Dependency Injection Configuration  
 Advantages of Dependency Injection  
 Dependency Injection Reduces Coupling

## 2. More about Bean Properties

Working with Properties  
 Configuring Value Based Properties  
 Using Value Based Properties  
 Property Conversions  
 Constructor Injections  
 Constructor Argument Resolution  
 Setter Injection vs. Constructor Injection  
 Collection Valued Properties  
 Working with Collections  
 Configuring <list> and <set> Properties  
 Configuring Collections of Bean  
 References  
 Map Valued Properties  
 java.util.Properties Valued Properties  
 Additional Capabilities  
 Factory Methods  
 Instance Factory Methods  
 Bean Aliases  
 Bean Definition Inheritance  
 Autowiring  
 Autowiring byType

Pros and Cons of Autowiring  
 To Autowire or Not to Autowire

## 3. The Spring Container and API

ApplicationContext  
 ApplicationContext Interface  
 ApplicationContext Implementations  
 Constructors  
 Using an ApplicationContext  
 Spring Resource Access  
 Built-in Resource Implementations  
 Bean Scope and Lifecycle  
 Bean Scope  
 Specifying Bean Scope  
 Inner Beans  
 Compound Names  
 Depends On  
 Bean Creation Lifecycle  
 Using the Lifecycle Interfaces for Beans  
 Bean Creation Lifecycle Details  
 Bean Destruction Lifecycle  
 BeanPostProcessor  
 @PostConstruct and @PreDestroy  
 Event Handling  
 MessageSources  
 Issues with Messages  
 Resource Bundles  
 Defining Resource Bundles  
 Using Resource Bundles and  
 MessageSource  
 Localization/Internationalization  
 Paramaterizing Messages  
 Annotation Driven Configuration  
 Annotations in Spring  
 Enabling Spring Annotations  
 @Component and Auto-Detecting Beans  
 DI Using @Resource  
 Complete Declarations Using  
 Annotations  
 @Resource – Additional Uses  
 @AutoWired  
 Qualifiers  
 Lifecycle Annotations  
 XML Config – Annotations and  
 Scanning  
 Annotation Configuration – Pro/Con  
 A Note on the XML Configuration

- A Brief Note on Annotations
- Java-Based Configuratio
- Java Configuration Overview
- Using Java-Based Configuration
- Dependency Injection
- More on How it Works
- Dependencies Between Configuration Classes
- Other Usage Scenarios
- Classpath Scanning
- Other @Bean Capabilities
- Java-Based Configuration – Pro / Con
- Other Capabilities
- SpEL – Spring Expression Language
- Other SpEL Capabilities
- Validation
- Using Validation
- Configuring Validation
- Validation Constraints
- Additional Capabilities

#### 4. Database Access with Spring

- Issues with JDBC
- Problems Using JDBC Directly
- Let's Review Some Simple JDBC Usage
- Simple Query on the Database
- Problems with the Previous Approach
- Spring Support for the DAO Pattern
- Spring DAO Support
- The Spring Database API
- The JdbcTemplate Class
- The JdbcDaoSupport Class
- DataSources
- Spring Jdbc Exception Hierarchy
- DAO Based on Spring Classes
- Configuring a DataSource
- Looking up a DataSource in JNDI
- Building a DAO Without the Support Class
- Queries and Updates
- Querying with JdbcTemplate
- Mapping Result Rows to Objects
- Defining a RowMapper Class
- Inserting/Updating
- SimpleJdbcTemplate
- The SimpleJdbcTemplate Class
- Using Spring with Hibernate

- Hibernate Overview
- Typical Hibernate Configuration File
- Using Hibernate Directly
- Spring Support for Hibernate
- Template Support for Hibernate
- LocalSessionFactoryBean
- Configuring a Hibernate Session Factory
- Contextual Sessions
- Spring-Free DAO
- What Approach to Use
- Using Spring with JPA
- Template Support for JPA
- Support for Managing EntityManager
- LocalEntityManagerFactoryBean
- Obtaining an EntityManager from JNDI
- LocalContainerEntityManagerFactoryBean
- Container-Managed EntityManager
- Additional Spring Configuration
- JPA Data Access Object
- Extended Persistence Context

#### 5. Aspect Oriented Programming

- AOP Overview
- The Issue with Crosscutting Concerns
- Crosscutting Illustrated
- Aspect Oriented Programming (AOP) Defined
- Spring AOP Introduction
- Spring AOP with AspectJ Annotations
- Defining an Aspect with @AspectJ
- Defining a Pointcut
- Defining Advice
- Configuring Spring
- A Program that Triggers Advice
- More on How Spring AOP Works
- Pointcut Expressions and Advice
- Pointcut Expressions
- Sample Execution Designator Patterns
- Other Designators Available in Spring AOP
- Combining Pointcut Expressions
- Kinds of Advice
- XML Based AOP Support
- Defining Aspects Using XML
- Specifying Advice with XML
- Other Considerations

- Spring Proxies and Direct Invocation
- More on Spring Proxies
- Issues with AOP
- Is AOP Worth It
- Other AOP Capabilities and Functionality

## 6. **Transactions**

- Transaction Lifecycle
- Transaction Managers
- Configuring Transaction Managers
- JTA Transaction Manager
- Spring Declarative Transaction Management
- Transactional Scope
- Transaction Attributes for Propagation
- MANDATORY
- NESTED
- NEVER
- NOT\_SUPPORTED
- REQUIRED
- REQUIRES\_NEW
- SUPPORTS
- Transaction Attributes – Some Choices
- Specifying Transaction Attributes
- Additional Transactional Attributes
- Rolling Back and Exceptions
- XML Configuration
- Specifying Transactions Using XML
- Linking Advice with Pointcuts
- <tx:method> Attributes

## 7. **Integration with Java Web Applications**

- Spring and Java EE
- Java EE Web Applications
- Web Application Structure
- Web Application Components
- ApplicationContext and Web Apps
- Configuring ContextLoaderListener
- Using the Application Context

## 8. **Spring Security Overview**

- Spring Security
- Spring Web Security – web.xml
- Spring Web Security – Spring Configuration
- More <http> Capabilities
- Other Authentication Providers

- Method Security
- Method Security – Annotations
- Method Security – Pointcut Expressions
- Method Security – XML Configuration