Hands On Technology Transfer
The Best Way to Transfer Technology Skills

Contact Us
1-978-250-4299

Introduction to Spring 4, Spring MVC and Spring REST

Duration: 5 Days *(Face-to-Face & Remote-Live)*, or 35 Hours *(On-Demand)*

Price: CDN$3,275 *(Face-to-Face & Remote-Live)*, or CDN$1,995 *(On-Demand)*

Discounts: We offer multiple discount options. Click here for more info.

Delivery Options: Attend face-to-face in the classroom or remote-live attendance.

Students Will Learn

- Understanding the core principles of Spring and of Dependency Injection (DI) / Inversion of Control
- Using the Spring Core module and DI to configure and wire application objects (beans) together
- Knowing the different types of metadata (XML, annotations/@Component, and Java Configuration/@Configuration) and how and when to use them
- Understanding and using the complete capabilities of the Core module, such as lifecycle events, bean scopes and the Spring API
- Working with the ORM (Object-Relational Mapping) module to integrate Spring with technologies such as Hibernate or JPA
- Understanding and using Spring's powerful AOP capabilities for programming cross-cutting concerns across multiple points in an application
- Learning safe and maintainable techniques for programming with AOP
- Understanding and using Spring's transaction support, including the easy-to-use Java annotation support, as well as the tx/aop XML configuration elements
- Integrating Spring with Java EE Web applications
- Building Web applications with Spring MVC, including configuration using Java config and Servlet 3 capabilities
- Understanding REST, and use Spring MVC to build RESTful services
- Using Ajax-based front ends with Spring MVC / Spring REST

Course Description

This course introduces the techniques for using the powerful capabilities of Spring 4 including the three main configuration styles: Java-based (`@Configuration`), annotation-based (`@Component`), and the traditional XML-based configuration that may still play an important role in existing and new projects. It also provides guidelines for when and how to use each one.

The course starts with in-depth coverage on using the powerful capabilities of Spring's Core module to reduce coupling and increase the flexibility, ease of maintenance, and testing of your applications. Coverage also includes integrating persistence layers (e.g. Hibernate/JPA) with Spring, using Spring's powerful Aspect Oriented Programming (AOP) to program cross-cutting concerns in a safe and maintainable way and using Spring's declarative transaction capabilities.

The course includes integrating Spring with Java EE Web applications and an introduction to Spring MVC. Spring MVC utilizes a Model-View-Controller pattern for building Web applications and the introduction covers the basics of Spring MVC and how it supports organizing your Web applications in a highly structured, loosely coupled manner. This includes an introduction to REST (Representational state transfer) and how to use Spring MVC to build RESTful resources and invoke them from Ajax-based front ends.

This course will enable you to build working Spring applications and give you an understanding of the important concepts and technology. Comprehensive hands-on labs provide reinforcement of the topics covered in the course and practical experience deploying solutions.

Students who do not require coverage of Spring MVC and RESTful Web Services may want to take the 3-day Introduction to the Spring 4 Framework class instead.

Students requiring an introduction to JEE Web Development, JDBC, JNDI, and JSP as well as Spring and Hibernate, may want to take the Web Application Development Using JEE, Frameworks, Web Services and AJAX class instead.

Course Prerequisites

Java SE programming experience and an understanding of object-oriented design principles. Fundamental knowledge of XML is helpful but not required. HOTT's course Java Programming or equivalent knowledge provides a solid foundation.

Course Overview

Introduction to Spring

- Overview of Spring Technology
    - Challenges for Modern Applications
    - Motivation for Spring, Spring Architecture
    - The Spring Framework
- Spring Introduction
    - Managing Beans
    - Inversion of Control / IoC, Dependency Injection / DI
    - Configuration Metadata Overview, Configuring Beans (XML)
- The Spring Container
    - Overview of the Spring Container
    - `ApplicationContext` Overview
    - `ClassPathXmlApplicationContext`, `FileSystemXmlApplicationContext`, `AnnotationConfigApplicationContext`
    - API and Usage
- Dependencies and Dependency Injection (DI)
    - Examining Dependencies
    - Dependency Inversion
    - Configuration and Usage of Dependency Injection (DI) in Spring

Configuration in Depth

- Annotation Driven Configuration
    - JSR 330 (`@Named`) and Spring (`@Component`) Annotation Styles
    - `@Named/@Component`, `@Inject/@Autowired`, `@Repository`, `@Service`
    - Configuring Beans and Autowiring with Annotations
    - Enabling Annotations - `context:component-scan`
- Java Based Configuration (`@Configuration`)
    - Overview - code-centric Configuration
    - `@Configuration` and `@Bean`
    - Dependency Injection
    - Resolving Dependencies on Other Beans
    - Injecting Configuration Classes
- Integrating Configuration Types
    - Choosing a Configuration Style
    - Integrating Configuration Styles
    - Importing: `@Import` and
    - Scanning with `@Configuration` style
- Bean Scope and Lifecycle
    - Bean Scope Defined - Singleton, Prototype, and Other Scopes
    - Configuring Scope
    - Bean Creation Lifecycle
    - Lifecycle Callbacks
    - `BeanPostProcessor`

- Event Handling

Wiring in Depth

- Value Injection
  - Configuring Value Properties
  - Property Conversions
  - Externalizing Values in Properties Files
- Constructor Injection
  - Constructor Injection Overview
  - Configuration - `@Configuration` and XML
  - `p:` and `c:` namespaces for XML configuration
- Qualifiers / Domain Specific Language (DSL)
  - Limitations of Autowiring
  - Qualifiers and DSL
  - Creating and Using an Annotation-Based DSL for Bean Configuration
  - Benefits of Qualifiers for Bean Configuration
- Profiles
  - Profiles Overview
  - Configuring Profiles (XML and `@Configuration`)
  - Activating Profiles
- Overview of SpEL

Database Access with Spring

- Overview of Spring Database Support
- Configuring a DataSource
- Using Spring with Hibernate
  - High Level Hibernate Overview
  - `SessionFactory` configuration
  - `LocalSessionFactoryBean`
  - Contextual Sessions and Spring Integration
- Using Spring with JPA
  - Managing the EntityManager (EM)
  - `LocalContainerEntityManagerFactoryBean` and Container-managed EMs
  - JEE and JNDI Lookup of the EM
  - Configuration and Vendor Adaptors
  - Creating a JPA Repository/DAO Bean - `@PersistenceUnit`, `@PersistenceContext`

Aspect Oriented Programming (AOP)

- Overview of AOP
  - Crosscutting Concerns
  - AOP Basics, Aspect, Joinpoint, Advice, Pointcut
- Spring AOP Introduction
  - Configuration - XML and `@AspectJ`
  - Defining an Aspect, Pointcut, and Advice
  - How Advice is Triggered
- Pointcut Expressions and Advice
  - Pointcut Expression Overview
  - The `execution()` Designator
  - Other Designators (within, target, args, `@target`, ...)
  - Kinds of Advice - before, after, around, after-returning, after-throwing
- Marker Annotations (Rubber Stamp AOP)
  - Issue with AOP Configuration
  - Defining an AOP Marker / Rubber Stamp
  - Configuring AOP Using a Marker
  - Advantages of Marker Annotations
- `@AspectJ` Based AOP Support
  - `@AspectJ` Annotations Overview
  - Defining an Aspect, Pointcut, and Advice
- Other Considerations

Spring Transaction (TX) Management

- Introduction to Spring Transaction Management
  - Spring Transaction Managers
  - Spring Declarative TX Management
  - Spring TX Scope and Propagation
  - Spring TX Attributes (REQUIRED, SUPPORTS, etc)
- XML Configuration of Transactions
  - Specifying Advice, TX Attributes, and Methods
  - Linking Advice with Pointcuts
  - Benefits of XML Configuration of TX Behavior

Spring Web Integration and Introduction to Spring MVC

- Integrating Spring with Java EE Web Apps
  - `ContextLoaderListener`
  - `WebApplicationContext`
- Spring Web MVC Overview
  - Capabilities
  - Architecture (Front Controller, MVC Pattern)
- Spring MVC Basics
  - `DispatcherServlet`, Configuration (`@EnableWebMvc`, Servlet 3 initialization),

    - Spring AOP Proxies and Self-Invocation Issues
    - Load-Time Weaving
    - Caveats of AOP

Additional Spring MVC Capabilities

- Reference Data with `@ModelAttribute`
- Forms and Binding, Spring Form Tags
- Session Attributes, `@SessionAttributes`
- Validation / JSR-303

XML Specific Configuration

- Collection Valued Properties
    - Configuring and Using Lists and Sets
- Factory Classes and Factory Methods
- Definition Inheritance (Parent Beans)
- AutoWiring with XML
- Inner Beans
- Compound Names

    mvc Namespace
    - Controllers, `@Controller`, `@RequestMapping` (Handler Methods)
    - `@RequestParam` and Parameter Binding
    - View Resolvers
    - Controller Details, `@RequestMapping`, `@RequestParam`, `@PathVariable`
    - Model Data, `@ModelAttribute`, Model/ModelAndView Classes

RESTful Services with Spring

- REST Overview
    - Characteristics/Capabilities
    - URI Templates
    - REST vs SOAP
- REST and Spring MVC
    - Spring support for REST
    - `@RequestMapping/@PathVariable`, `@RequestBody`, `@ResponseBody`, HTTP Method Conversion
    - Writing RESTful Controllers
    - Returning XML and JSON Data
- Client-Side Access to RESTful Services
    - Ajax Access (Browser/JavaScript/jQuery)
    - Using Spring's `RestTemplate`
- Programming Common REST Patterns
    - GET: Read
    - POST: Create
    - PUT: Update
    - DELETE: Delete