Introduction to Spring 5, Spring Boot and Spring REST

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

Price: $2495 *(Face-to-Face & Remote-Live)*, or $1495 *(On-Demand)*

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

Delivery Options: Attend face-to-face in the classroom, remote-live or on-demand training.

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
- Using Spring Boot to simplify dependency management and configuration
- Understanding and using Boot's auto-configuration
- Customizing Boot's behavior with properties and in other ways
- Working with the ORM (Object-Relational Mapping) module to integrate Spring with technologies such as JPA
- Using Spring Data to automatically generate JPA-based repository classes
- Understanding and using Spring's transaction support, including the easy-to-use Java annotation support
- Understanding REST, and using Spring REST to build RESTful services
- Using Ajax-based front ends with Spring REST
- Using RestTemplate to create Java REST clients

Course Description

Spring 5 provides an evolutionary advance of Spring's powerful capabilities. This course introduces the many Spring Core capabilities, as well as providing guidelines on when and how to use them. It also goes into considerable depth on Spring Boot for dependency management and auto-configuration, as well as Spring REST for creating RESTful resources.

This course has been completely revised to utilize Spring Boot's easy configuration and auto-configuration wherever possible. "Classic" Spring configuration (usually more verbose and complicated) is optionally covered in abbreviated form.

The course starts with in-depth coverage of Spring's Core module to reduce coupling and increase the flexibility, ease of maintenance, and testing of your applications. It goes on to cover many of the most important capabilities of Spring, including easing configuration with Spring Boot, integrating JPA persistence layers with Spring and Spring Data, and using Spring's declarative transaction capabilities.

The course includes a solid introduction to Spring REST, and coverage of building RESTful resources. It also covers many of the details of Spring Boot, including how to create Boot-based POMs (maven) for simplified dependency management, customizing Boot behavior, and understanding/managing Boot's auto-configuration.

Comprehensive hands on exercises are integrated throughout the course to reinforce learning and develop real competency. This course will enable students to build working Spring applications with Java and will give students an understanding of the important concepts and technology in a very short time.

Students desiring coverage of Spring MVC rather than Spring Boot may instead be interested in attending the Introduction to Spring 5, Spring MVC and Spring REST course.

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
  - Motivation for Spring, Spring Architecture
  - The Spring Framework
  - maven and Spring
- Spring Introduction
  - Declaring and Managing Beans
  - ApplicationContexts - The Spring Container
  - XML and @Component/@Named Config
- Dependencies and Dependency Injection (DI)
  - Examining Dependencies
  - Dependency Inversion / Dependency Injection (DI)
  - DI in Spring - XML and @Autowired
- Spring Boot Quickstart

Configuration in Depth

- Java Based Configuration (@Configuration)
  - Overview, @Configuration, @Bean
  - Dependency Injection
  - Resolving Dependencies
- Integrating Configuration Types
  - XML and @Component Pros/Cons
  - @Configuration Pros/Cons
  - Choosing a Configuration Style
  - Integrating with @Import and <import>
- Bean Scope and Lifecycle
  - Singleton, Prototype, and Other Scopes
  - Configuring Scope
  - Bean Lifecycle / Callbacks

Spring Boot Overview

Spring Testing

- Spring Boot Overview
- Spring POMs with Boot Parents
- Spring Boot Starters
- SpringApplication – Apps With main()
- CommandLineRunner and ApplicationRunner
- Working with Properties
  - Boot Property Files
  - Using Application Properties
  - Customizing Behavior with Boot Properties
- Testing and JUnit 5 Overview
  - Writing Tests - Test Classes, asserts, Naming Conventions
  - Running Tests - IDE, maven, ...
  - Test Fixtures - setup and teardown
- Spring TestContext Framework
  - Overview
  - Configuration
  - Running Tests

Database Access with Spring/Boot

- Overview of Spring/Boot database support
  - DataSources, Boot Auto-Configuration, and Custom Configuration
  - Boot - Embedded Database
- Using Spring/Boot with JPA
  - Spring Boot Auto-Configuration and Scanning
  - Customizing the Configuration
  - Creating a JPA Repository/DAO Bean - @PersistenceUnit, @PersistenceContext
- Spring Data Overview
  - Overview and Architecture
  - Configuring Spring Data
  - Repositories and JPA Repositories
  - Using CrudRepository
- Using Spring Data
  - Naming Conventions for Querying
  - Creating more Complex Queries
  - Query Configuration
- [Optional] Configuration Without Boot
  - Managing the EntityManager (EM)
  - LocalContainer EntityManagerFactoryBean and Container-managed EMs
  - JEE and JNDI Lookup of the EM
  - Configuration and Vendor Adaptors

Spring Transaction (TX) Management

- Overview
- Declarative TX Management (REQUIRED, etc.)
- TX Scope and Propagation
- Configuration and Boot Auto-Configuration
- Pointcut-Based Configuration of Transactions

RESTful Services with Spring

- REST Overview and Principles
- DispatcherServlet - Boot Auto-Config and Customization
- Requests and Responses - GET, POST, PUT, DELETE
- Spring's REST API

Working with JSON and XML

- Generating JSON
  - JSON Overview
  - JSON Representations for Resources
  - Message Converters
- [Optional] Generating XML

    Spring support for REST
    - @RequestMapping/@PathVariable, @RequestBody, @ResponseBody
    - URI Templates and @PathVariable
    - Controllers with @RestController
- Ajax Overview

    - JAXB and Jackson Message Converters for XML
    - JAXB / @XmlRootElement
- Content Negotiation

Java Clients for RESTful Services

- Client Requirements and Spring's RestTemplate
- getForObject() / getForEntity()
- Other RestTemplate Methods
- Accessing Headers / exchange()

Common REST Patterns

- GET: Read
- POST: Create
- PUT: Update
- DELETE: Delete
- Programming on server side and client side (with RestTemplate)

Boot Configuration and Customization

- Logging and its Configuration
- Profiles
- Other Configuration/li>

Spring Boot Web/Security

- Spring Boot Web
- Boot's Embedded Servers
- Classic Spring MVC Configuration
- Spring Boot Security
- Spring Boot Data REST

Additional Spring/Boot Features

- Updates to Spring Core
- WebFlux / Reactive Web Framework
- Boot Actuator