Capstone Courseware, LLC
33 Boylston Street
Jamaica Plain, MA 02130
877-227-2477
capstonecourseware.com

117A. Spring-MVC Web Applications

Version 2.5

This course enables the experienced Java developer to use the Spring application framework to manage objects in a lightweight "IoC" (inversion-of-control) container; to create simple and complex web applications; and to manage persistent objects using Spring's support for DAOs and transaction control. Spring is a far-reaching framework that aims to facilitate all sorts of Java development, including every level of multi-tier distributed systems. Here we focus on the Core and Web/MVC modules, with a lighter (but by no means dismissive) touch on persistence through DAO and ORM modules.

The Core module gives the developer declarative control over object creation and assembly; this is useful for any tier of any Java application. So is Spring's validation framework, and so we study these things in a mix of standalone (Java SE) applications and web applications. Then students build web applications that use the Spring MVC framework to rationalize their designs into coherent request/response cycles. They use Spring command objects to manage HTML forms and their data, and connect these to the validation framework. We connect our applications to persistent stores and study the DAO and ORM modules, to better understand JDBC and JPA persistence models and declarative transaction control.

Prerequisites

- Java programming -- Course 103 is excellent preparation.
- Basic knowledge of XML -- Course 501.
- Servlets programming -- Course 110.
- JSP -- Course 112.

Learning Objectives

- Understand the scope, purpose, and architecture of Spring
- Use Spring's bean factories and application contexts to declare application components, rather than hard-coding their states and lifecycles
- Use dependency injection to further control object relationships from outside the Java code base
- Use annotations to take advantage of Spring post-processors for automated bean instantiation and wiring
- Create validators for business objects, and associate them for application-level and unit-testing uses
- Build a web application as a Spring DispatcherServlet and associated application context, with declared beans acting as controllers, command objects, and view resolvers
- Build and manage HTML forms with Spring command objects and custom tags
- Use Spring interceptors to implement horizontal features in the web application
- Connect business objects to persistent stores using Spring's DAO and ORM modules

Timeline: 5 days.

IDE Support: SpringIDE 2.2

In addition to the primary lab files, an optional overlay is available that adds support for SpringIDE 2.2. Students can code, build, deploy, and test all exercises from within Eclipse, and take advantage of Eclipse WTP's built-in editors and wizards for web applications, XML files, JSPs, and more. An appendix in the coursebook introduces SpringIDE features. See also our orientation to Using Capstone's Eclipse Overlays, and please be advised that this is an optional feature; it is not a separate version of the course, and the course itself does not contain explicit Eclipse-specific lab instructions.

Module 1. Introduction to Spring

Chapter 1. Overview of Spring

Java EE: The Good, The Bad, and the Ugly
Enter the Framework
Spring Value Proposition
The Spring Container
Web Applications
Persistence Support
Aspect-Oriented Programming
The Java EE Module(s)
Integrating Other Frameworks

Chapter 2. The Container

JavaBeans, Reconsidered
The Factory Pattern
Inversion of Control
XML View: Declaring Beans
Java View: Using Beans
Singletons and Prototypes

Chapter 3. Instantiation and Configuration

Configuring Through Properties
Configuration Namespaces
The p: Notation
Bean (Configuration) Inheritance
Configuring Through Constructors
Bean Post-Processors
Lifecycle Hooks
Integrating Existing Factory Code

Chapter 4. Dependency Injection

Complex Systems
Assembling Object Graphs
Dependency Injection
Single and Multiple Relationships
The Utility Schema
Bean Aliases

Inner Beans
Autowiring
Auto-Detecting Beans
@Autowired Properties
Best Practices with Spring 2.5 Annotations

Chapter 5. Assembling Object Models

Collections and Maps
Support for Generics
The Spring Utility Schema (util:)
Autowiring to Multiple Beans
Order of Instantiation
Bean Factory vs. Application Context

Chapter 6. Validation

Validators
The Errors Object
ValidationUtils
Error Messages and Localization
Nested Property Paths

Module 2. Web Applications

Chapter 1. The Web Module

Servlets and JSPs: What's Missing
The MVC Pattern
The Front Controller Pattern
DispatcherServlet
A Request/Response Cycle
The Strategy Pattern
JavaBeans as Web Components
Web Application Contexts
Handler Mappings
"Creating" a Model
View Resolvers

Chapter 2. Customizing Control Flow

HandlerMapping Options
ViewResolver Options
Chaining View Resolvers
Triggering Redirects

Chapter 3. Controllers and Commands

Working with Forms
Command Objects
The Template Method Pattern
Command Controllers
Data Binding
MultiActionController
Scope and Granularity of Command Objects
Auto-Detecting @Controllers
The @RequestMapping Annotation

Chapter 4. Binding and Validation

Property Editors
Custom Property Editors
Registrars
Validating Form Input

Chapter 5. Form Controllers

Form Controllers
AbstractFormController
SimpleFormController
Spring Custom Tags
<form:form> and Friends
<form:errors>
Reporting Errors
@RequestMapping for Form Controllers
@RequestMapping for Multi-Action Controllers
Other Handler Annotations
Controller vs. @Controller

Chapter 6. Refining the Handling Cycle

The Intercepting Filter Pattern
Exception Handling
Interceptors
The Decorator Pattern
Context and Lifecycle
Awareness Interfaces
Support and Utility Classes
"Death By XML"

Module 3. Persistence

Chapter 1. Templates and DAOs

The DAO Pattern
The DaoSupport Hierarchy
The DataAccessException Hierarchy
JDBC DAOs
JdbcTemplate and RowMapper

Chapter 2. Working with JPA

Object/Relational Mapping
The Java Persistence API
Blending Spring and JPA
Entity Manager Factories
Configuration Issues

Chapter 3. Transactions

Transaction Managers
Transaction Advice
AOP vs. Annotations
JDBC Transaction Manager
JPA Transaction Manager

Appendix A. Spring IDE

- Installing Spring IDE
- Naviation Features
- Auto-Completes
- Validation
- Support for Spring WebFlow

Appendix B. Learning Resources

System Requirements

Hardware Requirements (Minimum)	1.0 GHz, 256 meg RAM, 500 meg disk space.
Hardware Requirements (Recommended)	1.5 GHz, 512 meg RAM, 1 gig disk space.
Operating System	Tested on Windows XP Professional. Course software should be viable on all systems which

	support a J2SE 5.0 JDK.
Network and Security	Limited privileges required -- please see our standard security requirements at **http://capcourse.com/Guides/Security.html**.
Software Requirements	All free downloadable tools.