Java EE Programming: Spring 3.0

Meet the expert: Greg Matus combines his practical programming skills and mentoring excellence to develop the highest quality educational programs available. His technical focus includes developing programs and projects focused around advanced application development, performance, and tuning skills and technologies. His specialties include Java, advanced J2EE / JEE, AJAX, XML, Web Services, JSP, SQL, JDBC and Database topics, Oracle, Eclipse, WebLogic, and more.

Prerequisites: This an intermediate- level Spring 3.0 training course, designed for developers who need to understand how and when to use Spring in Java and JEE applications. You should have practical basic Java development experience.

Runtime: 17:37:01

Course description: The Spring framework is an application framework that provides a lightweight container that supports the creation of simple to complex components in a non-invasive fashion. The Spring flexibility and transparency is congruent and supportive of incremental development and testing. The framework structure supports the layering of functionality such as persistence, transactions, view-oriented frameworks, and enterprise systems and capabilities. Spring makes JavaEE development easier. Spring simplifies common tasks and encourages good design based on programming to interfaces. Spring makes your application easier to configure and reduces the need for many JEE design patterns. Spring puts the OO design back into your JEE application. This Spring course will take an in-depth tour of the basic Spring framework, initially examining concepts such as Inversion of Control and Dependency Injection, and then working with the container and basic components. The improved Spring 3.0 configuration management options centered on the Spring Expression Language (SPEL) is covered. The course then moves into the areas of persistence and transactions, looking at various options including both JDBC and Hibernate. You will then look at options for handling the view portion of MVC web architecture.

Course outline:

Introduction
- Introduction
- Introduction to Spring
- Benefits of Spring
- Drawbacks: Traditional Java EE
- Spring Framework: Goals
- Nature of POJOs
- Spring Architecture
- Key Features of Spring
- Core Spring Principles
- Configuring Spring
- POJOs and Interfaces: Problems
- POJOs and Interfaces
- Spring is an Object Factory
- Dependency Injection
- Dependency Injection: Example
- Complete Example
- Demo: Intro to Spring
- Summary

Spring Configuration
- Introduction
- Dependency Injection & Testing
- Testing with Mocks
- Testing with Mocks: UML Diagram
- Spring Architecture
- Spring Jars
- Demo: Spring Jars
- Spring DI Container
- Initializing the Container
- Accessing Beans: Container
- Demo: Accessing Beans
- Configuring Objects: Spring
- XML Schema and DTD
- Core Structure of the XML File
- Defining and Naming Beans
- Typical Bean Creation
- Constructor Argument Matching
- Distinguishing ref and value
- Allowed conversions for value
- Examples of ref and value
- Summary

Property Injection
- Introduction
- DI: Bean Properties
- DI: Using Nested Syntax
- Special Handling for Collections
- Collection Based Properties
- Initializing to a Null Value
- Factories: Adv. Bean Creation
- Static Bean Factory
- Method Factories
- Bean Creation - Singletons
- Managed Bean Lifecycle
- Validation of Bean Dependencies
- Autowiring
- Demo: Property Injection
- Summary

Config: Advanced Features
- Introduction
- init and destroy Methods
- Other Advanced Features
- Spring 3 Annotations
- @Resource
- @Autowired
- @Component
- @Component Specializations
- @Required
- Demo: Config Advanced Features
- Summary

Config: Additional Features
- Introduction
- Overview of Factory Beans
- XSL Transform: Factory Beans
- Configuring the XSLFactory Bean
- Spring: Pre-built Factory Beans
- Bean Definition Re-use
- Using Property Files
- PropertyPlaceholderConfigurer
- Custom Property Editors
- Bean Post-Processors
- Demo: Additional Features
- Summary

AOP
- Introduction
- Spring Architecture
- Aspect Oriented Programming
- What is AOP
- Cross Cutting Concerns
- What is an Aspect
- Aspects and Decoupling
- Structure of Proxy - version 1
- Structure of Proxy - version 2
- Tradeoffs Between Styles
- Some AOP Vocabulary
- Cross Cutting Concerns
- Why is AOP so Important
- Summary Crosscutting Concerns
- Summary

AOP in Spring
- Introduction
- Spring: AOP in a Nutshell
- Complete AOP HelloWorld Example

(Continued on page 2)

Java EE Programming: Spring 3.0

- The Simplest Advice
- Demo: Ice Cream
- Complete Advice & Config File
- Demo: Adding Welcome Advice
- Summary

More About Interceptors
- Introduction
- MethodBeforeAdvice
- The Method Class
- Three Technologies of Weaving
- The Generated Proxy Object
- The Cost of Using a Proxy
- Aspects/Proxies Costs/Benefits
- Four Kinds of Advice
- Spring Advice Types
- MethodBeforeAdvice
- AfterReturningAdvice
- ThrowsAdvice
- Demo: Interceptors
- Summary

Method Interceptor
- Introduction
- MethodInterceptor
- ExampleInterceptor
- Demo: Around Advice
- Demo: Initial Test
- Demo: Debugging
- Summary

Join Points and Pointcuts
- Introduction
- Joinpoints
- Pointcuts - Specify Joinpoints
- Where do Joinpoints fit
- AOP Definitions
- PointCuts
- Static PointCuts
- Dynamic PointCuts
- Pre-built Spring PointCuts
- Regular Expression PointCuts
- Custom PointCut Implementations
- Pointcuts util class
- The Advisor
- DefaultPointcutAdvisor
- RegexpMethodPointcutAdvisor
- Overview of the AOP Components
- Proxy Configuration
- ProxyConfig
- ProxyFactoryBean
- BeanNameAutoProxyCreator
- DefaultAdvisorAutoProxyCreator
- Overview of the AOP Components
- Summary

IOC and Annotation Based AOP
- Introduction
- Demo: Adding Statistics
- Demo: Search Statistics Advice
- Demo: BeanNameAutoProxyCreator
- Demo: Test Client
- Summary

Annotations
- Introduction
- Relationship to AspectJ
- Review of Java Annotations
- Example: Java Annotation
- Using Annotations
- Annotations: More Info
- Three Forms of Annotations
- Annotations to Create Aspects
- The Spring Config File
- Intro to Annotation Syntax
- Other Advice Types
- ProceedingJoinPoint
- Around advice
- More About Aspect Parameters
- Using args(argName)
- 3 Types of Pointcuts
- Details of: name-pattern
- Examples: execution Pointcut
- Examples: bean(beanName)
- Pointcut Pattern Expressions
- Demo: Annotations
- Summary

AOP Annotations
- Introduction
- Advice: Introductions
- What is an Introduction
- Introduction: Example
- Difference from Around Advice
- Complete Working Example
- Example: The Lockable Interface
- Example: Underlying POJO
- Example: Client Code
- Example: What are the Challenges
- Generated Proxy Object Structure
- Create a LockableImpl
- The Aspect Itself
- Overview of the Code
- Declaring the Aspect: mixin
- Declaring the Aspect: interaction
- Declaring the Interaction
- @Before Interaction Behavior
- @Before Aspect Declaration
- Pointcut Specification
- IsLockable Annotation
- Spring Config File
- Introduction Advice
- Summary

Persistence In Spring
- Introduction
- Spring Architecture
- DAO Implementations
- Transaction Support
- Continued: Transaction Support
- Isolation Level Concepts
- Propagation Behavior
- Transaction Config
- Programmatic Transaction
- Declarative Transactions
- Benefits of Declarative Trans
- Two Forms of Declarative Trans
- The Java Code is Simple
- Example: Declarative Annotations
- Summary

Spring JDBC
- Introduction
- Overview
- Switching Between DataSources
- JDBC DAO Implementation
- The jdbcTemplate
- RowMapper
- RowCallbackHandler
- Sending SQL
- Exception Handling
- Operation Classes
- Demo: Spring JDBC
- Demo: Extending jdbcDao
- Summary

Spring MVC
- Introduction
- Spring MVC
- Overview of Spring MVC
- The DispatcherServlet
- The WebApplicationContext
- Workflow of Request Handling
- Mapping URLs to Controllers
- Using Handler Mappings
- Handler Interceptors
- Review of Architecture
- ModelAndView & View
- The View in ModelAndView
- Demo: Spring MVC
- Summary

Spring MVC Views
- Introduction
- ModelAndView & View
- A Custom View Example
- ViewResolvers
- Controllers
- Using the MultiActionController
- Handling Form Requests
- Demo: MVC Views
- Demo: Annotations Approach
- Demo: Debugging
- Demo: Enabling View
- Summary

Spring MVC Form Handling
- Introduction
- Spring MVC Annotations
- ClassPath Scanning
- annotation-driven
- Annotation Controller Mapping
- How @RequestMapping Works
- MVC Handler Method Parameters
- The Form
- The Spring Form Tags
- Using a PropertyEditor
- Adding Validation
- Additional Functionality
- Demo: Spring MVC Form Handling
- Demo: Implement AddDVD
- Summary