About Spring 5.x Training

The Spring Framework is an open source application framework for Java. This framework has taken the Java software community by storm. Spring provided the technology to develop everything from small, stand-alone applications to large complex, enterprise systems out of simple POJOs (plain old Java objects).

In this class, students are exposed to the light-weight Spring container, configuration, foundational API, and general Spring architecture. Not just a class that focuses on theory, this course is loaded with practical labs and deals with configuration, maintenance and architectural issues. After taking the class, developers will immediately be able to utilize the Spring Framework in their new or existing applications.

Spring 5.x Course Prerequisite

- Students should have a good understanding of the Java programming language. A basic understanding of relational databases and SQL is very helpful. A basic understanding of XML is also useful. Students that have attended Intertech’s Complete Java have the necessary background for this course.

Spring 5.x Training Course Objective

- Learn how to download, setup and configure the Spring Framework
- Explore the Spring Container and Modules
- Discover the Spring philosophies and principles and how they impact application development
- Understand dependency injection
- Learn aspect oriented programming and how it is used to provide cross cutting concerns
- See how to accomplish data access with Spring’s DAO Module
- Understand how Spring deals with transaction management
- Examine Spring’s unit testing framework

Spring 5.x Training Course Duration

- 60 Working days, daily one and half hours

Spring 5.x Training Overview

Spring Introduction

- Differences between programming language, software technology and framework
- Introduction to Spring Framework
- Evolution of spring Framework
- Modules of Spring in Spring 1.x,2.x,3.x,4.x and 5.x
- MVC Architecture
- Role of spring framework in MVC Architecture application development
- Definition of spring framework
- POJO Class, POJI, JavaBean, Component Class ,spring bean classes

Spring Core Module

- Introduction to IOC
- Introduction to Spring Container/IOC Container
- Types of Dependency Injections

- Setter injection
- Constructor injection
- Aware injection
- Method injection
- Lookup method injection
- Introduction to Design patterns
- Factory DesignPattern
- Strategy Design pattern
- Layered Application demonstrating real time dependency injection
- Resolving/identifying params in constructor injection
- Bean Inheritance
- Collection Merging
- Null injections
- Bean alias
- Default bean ids
- Performing dependency lookup by using IOC container
- <Idref>tag
- Understanding Factory Methods
- Factory Method Bean Instantiation
- Static factory method bean Instantiation
- instance factory method bean Instantiation
- Singleton java class and its usecases
- **Bean Scopes**
- Singleton
- Prototype
- Request
- Session
- Application
- Websocket
- **Bean Wiring**
- Explicit wiring
- Implicit wiring or auto wiring
- ByType
- ByName
- Constructor
- Autodetect
- **p-namespace,C-namespace**
- **ApplicationContext Container**
- Preinstantiation of singleton scope beans
- Working with properties file
- I18n (Internationalization)
- Event Handling
- BeanFactory Vs ApplicationContext
- Automatic registration of bean post processor and bean factory post processor
- **Bean Life Cycle**
- Declarative approach
- Programmatic approach
- Annotation driven approach
- **Nested IOC Containers**
- Presentation tier
- Business tier
- Various attributes of <ref> tag
- FactoryBean
- ServiceLocator as factory bean

- FactoryMethod bean Instantation based service locator
- Method Replacement/Method Injection
- Aware Injection
- Lookup method injection
- BeanPostProcessor
- BeanFactoryPostProcessors
- PropertyEditors
- Custom property editors
- Spring Expression Language(SpEL)

Spring Core Module with Annotations

- Spring stereo type annotations
- @Component,@Service,@controller,@Repositry and etc…
- @Autowired,@Qualifier,@Lazy and etc…
- Working with Java config annotations
- @Named,@Inject,@Resource and etc…
- Working with properties file in annotations environment
- Developing Layered applications in annotations environment

Spring Core Module with 100% Code/Java Config Approach.

- Working with @Bean, @Configuration,@Lazy,@PropertySource and etc…
- Developing Layered application
- Working with AnnotationConfigApplicationContext

Spring Boot Core

- Introduction to Spring Boot
- Spring Boot primary goals
- Spring boot features
- Spring Boot Starters
- Understanding @SpringBootApplication
- Auto configuration
- Example Applications
- Spring Profiles
- properties Vs application.yml
- Spring Boot Standalone flow
- Working with sts plugins in eclipse to develop spring boot application

Spring JDBC/DAO

- Introduction
- Plain JDBC limitations
- Spring JDBC/DAO Advantages
- Working with different Data Sources
- JdbcTemplate
- JNDI Registry and ServerManaged Jdbc connection pool
- Callback Interfaces
- Batch processing/Updating
- NamedParameterJdbcTemplate

- Working with SimpleJdbcInsert, SimpleJdbcCall
- SimpleJdbcCall to call PL/SQL procedures
- Mapping SQL operations as Sub Classes
- Spring JDBC/DAO with Annotations
- Spring JDBC/DAO with 100% Code Approach
- Spring Boot JDBC/DAO
- Spring Boot DAO-JdbcTemplate
- Spring Boot DAO-Simple Jdbc Insert
- Spring Boot Application Flow
- Working with DataSources through AutoConfiguration in Spring Boot 1.x and 2.x

Spring AOP Module

- Introduction
- Need of AOP
- Proxy design patterns
- AOP Terminologies/Principles
- Aspect, Advice, Joinpoint, Pointcut
- Target Class, Proxy Class, Weaing
- Types of Advices
- Before Advice, After Advice, Around Advice
- Throws Advice
- Types of Pointcuts
- Static pointcuts, Dynamic pointcuts
- Programmatic Spring AOP
- Declarative Spring AOP
- @AspectJ Style AOP support
- Spring AOP/AspectJ AOP with Annotations
- Spring AOP/AspectJ AOP with 100% Code Approach
- Spring AOP/Aspectj AOP with Spring Boot AOP

Spring Transaction Management

- Introduction to Transaction Management
- Local Transaction Vs Distributed Trasanction
- 2pc Principle
- Transaction models
- Flat Transaction Model
- Nested Transaction Model
- Need of Spring Transaction Management
- Choosing Spring Transaction Manager
- DataSourceTransactionManager
- Hibernate TransactionManager
- JTATransactionManager and etc…
- Different ways of implementing of Spring Transaction management
- Programatic approach
- Declarative approach using Spring AOP/AspectJ AOP
- Annotation Driven Approach using AspectJ AOP
- 100% code Driven approach using AspectJ AOP
- Spring Boot Driven approach using AspectJ AOP
- Transaction Attributes
- Transactions and integration testing

- Distributed TransactionManagement implementation using webLogic server,Atomikos API
- Configuring Transaction isolation Levels
- Read uncommitted, Read Committed
- Repeatable Read, Serializable
- Working with RollBackfor,noRollBackfor,Timeout and etc…in Transaction Management

Spring MVC

- Introduction To MVC
- Understanding MVC1,MVC2 Architectures
- Front Controller Design Pattern
- Intercepting Filter Vs Front Controller
- Different types of Servlet URL patterns
- Spring MVC Resources
- Spring MVC flow
- Structural Flow
- Strategy Flow(Code based Flow)
- DispatcherServlet
- Different Controller Classes
- ParamaterizableViewController
- UrlFileNameViewController
- AbstractController
- AbstractComandController
- SimpleFormController
- MultiActionController
- AbstractWizardFormController and etc…
- Developing Mini Project with CURD operation
- ContextLoaderListener
- Working with Two Containers
- HadlerMappings
- BeanNameUrlHandlerMapping
- SimpleUrlHandlerMapping
- ControllerClassNameHandlerMapping
- DefaultAnnotationHandlerMapping
- RequestMappingHandlerMapping and etc…
- HandlerMappingChaining
- Form Validations
- Enabling Server side Validations only when client side validation are not enabled
- <form:errors>
- ViewResolvers
- InternalResourceViewResolver
- UrlBasedViewResolver
- ResourceBundleViewResolver
- XmlViewResolver
- TilesViewResolver
- BeanNameViewResolver and etc…
- ViewResolverChaining
- Views
- InternalResourceView
- JstlView, TilesView, AbstractPdfView
- AbstractXlsView and etc….
- Exception Handling in Spring MVC
- Tiles integration with Spring MVC

- MessageSources and I18N
- Formatting Labels, Formatting Numbers
- Formatting Dates,
- Formatting Currency Symbols
- Locale
- MVC namespace
- Handler interceptors/Adapters
- Checking Browser Type
- Checking Timeout period
- Preventing double posting problem
- PDF Views and Excel Views
- File Uploading and Downloading
- Spring MVC with Annotations
- Annotation driven Controllers
- @RequestMapping, @Controller
- @ModelAttribute,@SessionAttribute,@RequestParam
- RequestToViewNameTranslator
- MVC NameSpace
- Annotation driven Form validation using Hibernate Validator API,JEE validator API
- Spring MVC with 100% Code Approach
- Dynamic Registration of Servlet
- WebApplicationInitializer
- SpringServletContainerInitializer
- @EnableWebMVC,@Import
- AbstractAnnotationConfigDispatcherServletInitializer
- Spring Boot MVC
- SpringServletInitializer
- Working with embedded TomcatServer
- Spring Boot MVC flow
- Spring Boot dev tools
- Properties,application.yml in Spring Boot
- Developing Mini Project with CURD operation
- Solving double posting problems in Spring Boot MVC using PostRedirectGetPattern
- Profiles in spring ,Spring boot

Spring Security

- Introduction
- Authentication authorization
- Authentication Manager and authentication info provider
- Need of Spring Security
- DeliagatingFilterProxy
- SecurityNameSpace
- Form Login
- Remember Me
- Session Concurrency
- Logout and etc…
- Working with Different Authentication Providers
- Xml File,Properties File,DataBase,LDAP Server
- Security Examples
- Using Xml Configuratuions
- Using Annotation Configurations
- 100% Code Driven Configurations

- Spring Boot Configuration
- Using LDAP Server as Authentication Provider

Spring ORM

- Introduction to ORM
- Spring ORM Advantage
- Integrating with Hibernate
- Spring with hibernate using HibernateTemplate
- HiberanteTemplate and its methods
- HibernateDAOSupport
- HibernateCallback interfaces

Spring Data and Spring Data JPA

- Need of Spring Data
- Spring Data JPA
- Finder Methods of Finder API
- Repositories
- JpaRepository
- CURDRepository
- Paging and Sorting Repository and etc…
- Spring Data Custom Query
- Automatic custom Query
- Manual custom Query(@Query)
- Spring Data Jpa Exception Translator
- Difference b/w Hibernate and Spring Data JPA
- Interacting with Mongo DB
- JUnit Test Cases

Spring Batch

- Need of Batch Processing
- Need of Spring Batch
- Understanding Spring Batch Architecture
- Working with Batch NameSpace
- Working with different ItemReaders, ItemWriters and ItemProcessors
- Converting Database Data to CSV File
- Converting CSV File to XML File
- Converting XML file to CSV File
- Spring Batch Application using Spring Boot
- JobBuilderFactory
- StepBuilderFactory
- Step point TaskLet and etc….
- Working with scheduler

Spring Mail

- Understanding Java mail API
- Understanding SMTP,POP3,IMAP Protocols
- Understanding Mail server and Mail Clients

- Understanding Email Message Structure
- Spring Mail abstraction over Java Mail
- Spring Mail using Spring Boot

Introduction to Spring MicroServices

How to Explain Project Architectures

- Servlet, Jsp Project Architecture
- Spring MVC Project Architecture
- Spring Boot with Micro service Project Architecture
- Adapting Agile Methodology