

---

# Master Spring and Spring Boot

**Spring and Spring Boot** Frameworks are the **No 1 frameworks** for building enterprise apps in the Java world.

In this course, you will **learn Spring and Spring Boot from ZERO**.

I'm a great believer that the best way to learn is by doing and we designed this course to be **hands-on**.

You will build a **web application, a REST API and full stack application** using Spring, Spring Boot, JPA, Hibernate, React, Spring Security, Maven and Gradle.

You will learn to containerise applications using Docker. You will learn to deploy these applications to AWS.

By the end of the course, you will know everything you would need to become a great Spring and Spring Boot Developer.

## Installing Tools

### Our Recommendations

- Use **latest version** of Java
- Use **latest version** of "Eclipse IDE for Enterprise Java Developers"
- Remember: Spring Boot 3+ works only with Java 17+

### Installing Java

- Windows - <https://www.youtube.com/watch?v=I0SBRWVS0ok>
- Linux - <https://www.youtube.com/watch?v=mHvFpyHK97A>
- Mac - <https://www.youtube.com/watch?v=U3kTdMPlgsY>

### Troubleshooting

- Troubleshooting Java Installation - [https://www.youtube.com/watch?v=UI\\_PabQ1YB0](https://www.youtube.com/watch?v=UI_PabQ1YB0)

### Installing Eclipse

- Windows - <https://www.youtube.com/watch?v=toY06tsME-M>
- Others - <https://www.youtube.com/watch?v=XveQ9Gq41UM>

---

## Troubleshooting

- Configuring Java in Eclipse - [https://www.youtube.com/watch?v=8iOr\\_fcE3L0](https://www.youtube.com/watch?v=8iOr_fcE3L0)

## Lectures

### Getting Started - Master Spring Framework and Spring Boot

- Getting Started - Master Spring Framework and Spring Boot

### Getting Started with Java Spring Framework

- Step 01 - Understanding the Need for Java Spring Framework
- Step 02 - Getting Started with Java Spring Framework
- Step 03 - Creating a New Spring Framework Project with Maven and Java
- Step 04 - Getting Started with Java Gaming Application
- Step 05 - Understanding Loose Coupling and Tight Coupling
- Step 06 - Introducing Java Interface to Make App Loosely Coupled
- Step 07 - Bringing in Spring Framework to Make Java App Loosely Coupled
- Step 08 - Your First Java Spring Bean and Launching Java Spring Configuration
- Step 09 - Creating More Java Spring Beans in Spring Java Configuration File
- Step 10 - Implementing Auto Wiring in Spring Framework Java Configuration File
- Step 11 - Questions about Spring Framework - What will we learn?
- Step 12 - Understanding Spring IOC Container - Application Context and Bean Factory
- Step 13 - Exploring Java Bean vs POJO vs Spring Bean
- Step 14 - Exploring Spring Framework Bean Auto Wiring - Primary and Qualifier Annotations
- Step 15 - Using Spring Framework to Manage Beans for Java Gaming App
- Step 16 - More Questions about Java Spring Framework - What will we learn?
- Step 17 - Exploring Spring Framework With Java - Section 1 - Review

### Using Spring Framework to Create and Manage Your Java Objects

- Step 01 - Getting Spring Framework to Create and Manage Your Java Objects
- Step 02 - Exploring Primary and Qualifier Annotations for Spring Components
- Step 03 - Primary and Qualifier - Which Spring Annotation Should You Use?
- Step 04 - Exploring Spring Framework - Different Types of Dependency Injection
- Step 05 - Java Spring Framework - Understanding Important Terminology
- Step 06 - Java Spring Framework - Comparing @Component vs @Bean

- 
- Step 07 - Why do we have dependencies in Java Spring Applications?
  - Step 08 - Exercise/ Solution for Real World Java Spring Framework Example
  - Step 09 - Exploring Spring Framework With Java - Section 2 - Review

## **Exploring Spring Framework Advanced Features**

- Step 01 - Exploring Lazy and Eager Initialization of Spring Framework Beans
- Step 02 - Comparing Lazy Initialization vs Eager Initialization
- Step 03 - Exploring Java Spring Framework Bean Scopes - Prototype and Singleton
- Step 04 - Comparing Prototype vs Singleton - Spring Framework Bean Scopes
- Step 05 - Exploring Spring Beans - PostConstruct and PreDestroy
- Step 06 - Evolution of Jakarta EE - Comparing with J2EE and Java EE
- Step 07 - Exploring Jakarta CDI with Spring Framework and Java
- Step 08 - Exploring Java Spring XML Configuration
- Step 09 - Exploring Java Annotations vs XML Configuration for Java Spring Framework
- Step 10 - Exploring Spring Framework Stereotype Annotations - Component and more
- Step 11 - Quick Review - Important Spring Framework Annotations
- Step 12 - Quick Review - Important Spring Framework Concepts
- Step 13 - Exploring Spring Big Picture - Framework, Modules and Projects

## **Getting Started with Spring Boot**

- Step 01 - Getting Started with Spring Boot - Goals
- Step 02 - Understanding the World Before Spring Boot - 10000 Feet Overview
- Step 03 - Setting up New Spring Boot Project with Spring Initializr
- Step 04 - Build a Hello World API with Spring Boot
- Step 05 - Understanding the Goal of Spring Boot
- Step 06 - Understanding Spring Boot Magic - Spring Boot Starter Projects
- Step 07 - Understanding Spring Boot Magic - Auto Configuration
- Step 08 - Build Faster with Spring Boot DevTools
- Step 09 - Get Production Ready with Spring Boot - 1 - Profiles
- Step 10 - Get Production Ready with Spring Boot - 2 - ConfigurationProperties
- Step 11 - Get Production Ready with Spring Boot - 3 - Embedded Servers
- Step 12 - Get Production Ready with Spring Boot - 4 - Actuator
- Step 13 - Understanding Spring Boot vs Spring vs Spring MVC
- Step 14 - Getting Started with Spring Boot - Review

---

## **Getting Started with JPA and Hibernate with Spring and Spring Boot**

- Step 01 - Getting Started with JPA and Hibernate - Goals
- Step 02 - Setting up New Spring Boot Project for JPA and Hibernate
- Step 03 - Launching up H2 Console and Creating Course Table in H2
- Step 04 - Getting Started with Spring JDBC
- Step 05 - Inserting Hardcoded Data using Spring JDBC
- Step 06 - Inserting and Deleting Data using Spring JDBC
- Step 07 - Querying Data using Spring JDBC
- Step 08 - Getting Started with JPA and EntityManager
- Step 09 - Exploring the Magic of JPA
- Step 10 - Getting Started with Spring Data JPA
- Step 11 - Exploring features of Spring Data JPA
- Step 12 - Understanding difference between Hibernate and JPA

## **Build Java Web Application with Spring Framework, Spring Boot and Hibernate**

- Step 00 - Introduction to Building Web App with Spring Boot
- Step 01 - Creating Spring Boot Web Application with Spring Initializr
- Step 02 - Quick overview of Spring Boot Project
- Step 03 - First Spring MVC Controller, @ResponseBody, @Controller
- Step 04 - 01 - Enhancing Spring MVC Controller to provide HTML response
- Step 04 - 02 - Exploring Step By Step Coding and Debugging Guide
- Step 05 - Redirect to a JSP using Spring Boot - Controller, @ResponseBody and View Resolver
- Step 06 - Exercise - Creating LoginController and login view
- Step 07 - Quick Overview - How does web work - Request and Response
- Step 08 - Capturing QueryParams using RequestParam and First Look at Model
- Step 09 - Quick Overview - Importance of Logging with Spring Boot
- Step 10 - Understanding DispatcherServlet, Model 1, Model 2 and Front Controller
- Step 11 - Creating a Login Form
- Step 12 - Displaying Login Credentials in a JSP using Model
- Step 13 - Add hard coded validation of userid and password
- Step 14 - Getting started with Todo Features - Creating Todo and TodoService
- Step 15 - Creating first version of List Todos Page
- Step 16 - Understanding Session vs Model vs Request - @SessionAttributes
- Step 17 - Adding JSTL to Spring Boot Project and Showing Todos in a Table
- Step 18 - Adding Bootstrap CSS framework to Spring Boot Project using webjars
- Step 19 - Formatting JSP pages with Bootstrap CSS framework

- 
- Step 20 - Lets Add a New Todo - Create a new View
  - Step 21 - Enhancing TodoService to add the todo
  - Step 22 - Adding Validations using Spring Boot Starter Validation
  - Step 23 - Using Command Beans to implement New Todo Page Validations
  - Step 24 - Implementing Delete Todo Feature - New View
  - Step 25 - Implementing Update Todo - 1 - Show Update Todo Page
  - Step 26 - Implementing Update Todo - 1 - Save changes to Todo
  - Step 27 - Adding Target Date Field to Todo Page
  - Step 28 - Adding a Navigation Bar and Implementing JSP Fragments
  - Step 29 - Preparing for Spring Security
  - Step 30 - Setting up Spring Security with Spring Boot Starter Security
  - Step 31 - Configuring Spring Security with Custom User and Password Encoder
  - Step 32 - Refactoring and Removing Hardcoding of User Id
  - Step 33 - Setting up a New User for Todo Application
  - Step 34 - Adding Spring Boot Starter Data JPA and Getting H2 database ready
  - Step 35 - 01 - Configuring Spring Security to Get H2 console Working
  - Step 36 - Making Todo an Entity and Population Todo Data into H2
  - Step 37 - Creating TodoRepository and Connecting List Todos page from H2 database
  - Step 38 - 01 - Connecting All Todo App Features to H2 Database
  - Step 38 - 02 - Exploring Magic of Spring Boot Starter JPA and JpaRepository
  - Step 39 - OPTIONAL - Overview of Connecting Todo App to MySQL database
  - Step 40 - OPTIONAL - Installing Docker
  - Step 41 - OPTIONAL - Connecting Todo App to MySQL database

## **Creating a Java REST API with Spring Boot, Spring Framework and Hibernate**

- Step 00 - Creating a REST API with Spring Boot - An Overview
- Step 01 - Initializing a REST API Project with Spring Boot
- Step 02 - Creating a Hello World REST API with Spring Boot
- Step 03 - Enhancing the Hello World REST API to return a Bean
- Step 04 - What's happening in the background? Spring Boot Starters and Autoconfiguration
- Step 05 - Enhancing the Hello World REST API with a Path Variable
- Step 06 - Designing the REST API for Social Media Application
- Step 07 - Creating User Bean and UserDaoService
- Step 08 - Implementing GET Methods for User Resource
- Step 09 - Implementing POST Method to create User Resource
- Step 10 - Enhancing POST Method to return correct HTTP Status Code and Location URI
- Step 11 - Implementing Exception Handling - 404 Resource Not Found

- 
- Step 12 - Implementing Generic Exception Handling for all Resources
  - Step 13 - Implementing DELETE Method to delete a User Resource
  - Step 14 - Implementing Validations for REST API
  - Step 15 - Overview of Advanced REST API Features
  - Step 16 - Understanding Open API Specification and Swagger
  - Step 17 - Configuring Auto Generation of Swagger Documentation
  - Step 18 - Exploring Content Negotiation - Implementing Support for XML
  - Step 19 - Exploring Internationalization for REST API
  - Step 20 - Versioning REST API - URI Versioning
  - Step 21 - Versioning REST API - Request Param, Header and Content Negotiation
  - Step 22 - Implementing HATEOAS for REST API
  - Step 23 - Implementing Static Filtering for REST API
  - Step 24 - Implementing Dynamic Filtering for REST API
  - Step 25 - Monitoring APIs with Spring Boot Actuator
  - Step 26 - Exploring APIs with Spring Boot HAL Explorer
  - Step 27 - Connecting REST API to H2 using JPA and Hibernate - An Overview
  - Step 28 - Creating User Entity and some test data
  - Step 29 - Enhancing REST API to connect to H2 using JPA and Hibernate
  - Step 30 - Creating Post Entity with Many to One Relationship with User Entity
  - Step 31 - Implementing a GET API to retrieve all Posts of a User
  - Step 32 - Implementing a POST API to create a Post for a User
  - Step 33 - Exploring JPA and Hibernate Queries for REST API
  - Step 34 - Connecting REST API to MySQL Database - An Overview
  - Step 34z - OPTIONAL - Installing Docker
  - Step 35 - OPTIONAL - Connecting REST API to MySQL Database - Implementation
  - Step 36 - Implementing Basic Authentication with Spring Security
  - Step 37 - Enhancing Spring Security Configuration for Basic Authentication

### **Building Java Full Stack Application with Spring Boot and React**

- Step 01 - Getting Started - Full Stack Spring Boot and React Application
- Step 02 - Exploring What and Why of Full Stack Architectures
- Step 03 - Understanding JavaScript and EcmaScript History
- Step 04 - Installing Visual Studio Code
- Step 05 - Installing nodejs and npm
- Step 06 - Creating React App with Create React App
- Step 07 - Exploring Important nodejs Commands - Create React App
- Step 08 - Exploring Visual Studio Code and Create React App

- 
- Step 09 - Exploring Create React App Folder Structure
  - Step 10 - Getting started with React Components
  - Step 11 - Creating Your First React Component and more
  - Step 12 - Getting Started with State in React - useState hook
  - Step 13 - Exploring JSX - React Views
  - Step 14 - Following JavaScript Best Practices - Refactoring to Modules
  - Step 15 - Exploring JavaScript further

### **Exploring React Components with Counter Example**

- Step 01 - Exploring React Components with Counter Example
- Step 02 - Getting Started with React Application - Counter
- Step 03 - Getting Started with React Application - Counter - 2
- Step 04 - Exploring React State with useState hook - Adding state to Counter
- Step 05 - Exploring React State - What is happening in Background?
- Step 06 - Exploring React Props - Setting Counter increment value
- Step 07 - Creating Multiple Counter Buttons
- Step 08 - Moving React State Up - Setting up Counter and Counter Button
- Step 09 - Moving React State Up - Calling Parent Component Methods
- Step 10 - Exploring React Developer Tools
- Step 11 - Adding Reset Button to Counter
- Step 12 - Refactoring React Counter Component

### **Building Java Todo Full Stack Application with Spring Boot and React**

- Step 01 - Getting Started with React Todo Management App
- Step 02 - Getting Started with Login Component - Todo React App
- Step 03 - Improving Login Component Further - Todo React App
- Step 04 - Adding Hardcoded Authentication - Todo React App
- Step 05 - Conditionally Displaying Messages in Login Component - Todo React App
- Step 06 - Adding React Router Dom and Routing from Login to Welcome Component
- Step 07 - Adding Error Component to our React App
- Step 08 - Removing Hard Coding from Welcome Component
- Step 09 - Getting Started with React List Todo Component
- Step 10 - Displaying More Todo Details in React List Todo Component
- Step 11 - Creating React Header, Footer and Logout Components
- Step 12 - Adding Bootstrap to React Front End Application

- 
- Step 13 - Using Bootstrap to Style Todo React Front End Application
  - Step 14 - Refactoring React Components to Individual JavaScript Modules
  - Step 15 - Sharing React State with Multiple Components with Auth Context
  - Step 16 - Updating React State and Verifying Updates through Auth Context
  - Step 17 - Setting isAuthenticated into React State - Auth Context
  - Step 18 - Protecting Secure React Routes using Authenticated Route - 1
  - Step 19 - Protecting Secure React Routes using Authenticated Route - 2

## **Connecting Spring Boot REST API with React Frontend - Java Full Stack Application**

- Step 01 - Setting Todo REST API Project for React Full Stack Application
- Step 02 - Calling Spring Boot Hello World REST API from React Hello World Component
- Step 03 - Enabling CORS Requests for Spring Boot REST API
- Step 04 - Invoking Spring Boot Hello World Bean and Path Param REST API from React
- Step 05 - Refactoring Spring Boot REST API Invocation Code to New Module
- Step 06 - Following Axios Best Practices in Spring Boot REST API
- Step 07 - Creating Retrieve Todos Spring Boot REST API Get Method
- Step 08 - Displaying Todos from Spring Boot REST API in React App
- Step 09 - Creating Retrieve Todo and Delete Todo Spring Boot REST API Methods
- Step 10 - Adding Delete Feature to React Frontend
- Step 11 - Setting Username into React Auth Context
- Step 12 - Creating Todo React Component to display Todo Page
- Step 13 - Adding Formik and Moment Libraries to Display Todo React Component
- Step 14 - Adding Validation to Todo React Component using Formik
- Step 15 - Adding Update Todo and Create Todo REST API to Spring Boot Backend API
- Step 16 - Adding Update Feature to React Frontend
- Step 17 - Adding Create New Todo Feature to React Frontend
- Step 18 - Securing Spring Boot REST API with Spring Security
- Step 19 - Adding Authorization Header in React to Spring Boot REST API calls
- Step 20 - Configuring Spring Security to allow all Options Requests
- Step 21 - Calling Basic Authentication Service when Logging into React App
- Step 22 - Using async and await to invoke Basic Auth API
- Step 23 - Setting Basic Auth Token into Auth Context
- Step 24 - Setting up Axios Interceptor to add Authorization Header
- Step 24A - Debugging Problems with Basic Auth and Spring Boot
- Step 25 - Getting Started with JWT and Spring Security
- Step 26 - Integrating Spring Security JWT REST API with React Frontend
- Step 27 - Debugging Problems with JWT Auth and Spring Boot



---

## **Connecting Java Full Stack Application (Spring Boot and React) with JPA and Hibernate**

- Step 01 - Full Stack React and Spring Boot with JPA and Hibernate
- Step 02 - Full Stack React & Spring Boot with JPA & Hibernate - Getting Tables Ready
- Step 03 - Full Stack React & Spring Boot with JPA & Hibernate - Todo CRUD operations
- Step 04 - Full Stack React & Spring Boot with JPA & Hibernate - Add New Todo
- Step 05 - Full Stack React & Spring Boot with JPA & Hibernate - Connect with MySQL

## **Exploring Unit Testing with JUnit**

- Step 01 - What is JUnit and Unit Testing\_
- Step 02 - Your First JUnit Project and Green Bar
- Step 03 - Your First Code and First Unit Test
- Step 04 - Exploring other assert methods
- Step 05 - Exploring few important JUnit annotations

## **Exploring Mocking with Mockito for Spring Boot Projects**

- Step 00 - Introduction to Section - Mockito in 5 Steps
- Step 01 - Setting up a Spring Boot Project
- Step 02 - Understanding problems with Stubs
- Step 03 - Writing your first Mockito test with Mocks
- Step 04 - Simplifying Tests with Mockito Annotations - @Mock, @InjectMocks
- Step 05 - Exploring Mocks further by Mocking List interface

## **Securing Spring Boot Applications with Spring Security**

- Step 00 - Getting started with Spring Security
- Step 01 - Understanding Security Fundamentals
- Step 02 - Understanding Security Principles
- Step 03 - Getting Started with Spring Security
- Step 04 - Exploring Default Spring Security Configuration
- Step 05 - Creating Spring Boot Project for Spring Security
- Step 06 - Exploring Spring Security - Form Authentication
- Step 07 - Exploring Spring Security - Basic Authentication
- Step 08 - Exploring Spring Security - Cross Site Request Forgery - CSRF
- Step 09 - Exploring Spring Security - CSRF for REST API

- 
- Step 10 - Creating Spring Security Configuration to Disable CSRF
  - Step 11 - Exploring Spring Security - Getting Started with CORS
  - Step 12 - Exploring Spring Security - Storing User Credentials in memory
  - Step 13 - Exploring Spring Security - Storing User Credentials using JDBC
  - Step 14 - Understanding Encoding vs Hashing vs Encryption
  - Step 15 - Exploring Spring Security - Storing Bcrypt Encoded Passwords
  - Step 16 - Getting Started with JWT Authentication
  - Step 17 - Setting up JWT Auth with Spring Security and Spring Boot - 1
  - Step 18 - Setting up JWT Auth with Spring Security and Spring Boot - 2
  - Step 19 - Setting up JWT Resource with Spring Security and Spring Boot - 1
  - Step 20 - Setting up JWT Resource with Spring Security and Spring Boot - 2
  - Step 21 - Understanding Spring Security Authentication
  - Step 22 - Exploring Spring Security Authorization
  - Step 23 - Creating a Spring Boot Project for OAuth with Spring Security
  - Step 24 - Getting Started with Spring Boot and OAuth2 - Login with Google
  - Step 25 - Quick Review - Securing Spring Boot Apps with Spring Security

### **Learning Spring AOP with Spring Boot**

- Step 01 - Getting Started with Spring AOP - An overview
- Step 02 - What is Aspect Oriented Programming?
- Step 03 - Creating a Spring Boot Project for Spring AOP
- Step 04 - Setting up Spring Components for Spring AOP
- Step 05 - Creating AOP Logging Aspect and Pointcut
- Step 06 - Understanding AOP Terminology
- Step 07 - Exploring @After, @AfterReturning, @AfterThrowing AOP Annotations
- Step 08 - Exploring Around AOP annotations with a Timer class
- Step 09 - AOP Best Practice - Creating Common Pointcut Definitions
- Step 10 - Creating Track Time Annotation
- Step 11 - Getting Started with Spring AOP - Thank You

### **Learning Maven with Spring and Spring Boot**

- Step 01 - Introduction to Maven
- Step 02 - Creating a Spring Boot Project with Maven
- Step 03 - Exploring Maven pom.xml for Spring Boot Project
- Step 04 - Exploring Maven Parent Pom for Spring Boot Project

- 
- Step 05 - Exploring Maven Further
  - Step 06 - Exploring Maven Build Lifecycle with a Spring Boot Project
  - Step 07 - How does Maven Work?
  - Step 08 - Playing with Maven Commands
  - Step 09 - How are Spring Projects Versioned?

### **Learning Gradle with Spring and Spring Boot**

- Step 01 - Getting Started with Gradle
- Step 02 - Creating a Spring Boot Project with Gradle
- Step 03 - Exploring Gradle Build and Settings Files
- Step 04 - Exploring Gradle Plugins for Java and Spring Boot
- Step 05 - Maven or Gradle - Which one to use for Spring Boot Projects?

### **Learning Docker with Spring and Spring Boot**

- Step 01 - Getting Started with Docker
- Step 02 - Understanding Docker Fundamentals
- Step 03 - Understanding How Docker Works
- Step 04 - Understanding Docker Terminology
- Step 05 - Creating Docker Image for a Spring Boot Project - Dockerfile
- Step 06 - Building Spring Boot Docker Image using Multi Stage Dockerfile
- Step 07 - Building Spring Boot Docker Image - Optimizing Dockerfile
- Step 08 - Building Docker Image with Spring Boot Maven Plugin
- Step 09 - Quick Review of Docker with Spring Boot

### **Getting Started with Cloud and AWS**

- Step 02 - Introduction to Cloud and AWS - Advantages
- Step 03 - Creating Your AWS Account
- Step 04 - Creating Your First IAM User
- Step 05 - Understanding the Need for Regions and Zones
- Step 06 - Exploring Regions and Availability Zones in AWS

### **Getting Started with Compute Services in AWS**

- Step 01 - Getting Started with EC2 - Virtual Servers in AWS

- 
- Step 02 - Demo - Creating Virtual Machines with Amazon EC2
  - Step 02z - Demo - Setting up a Web Server in an Amazon EC2 Instance
  - Step 03 - Quick Review of Important EC2 Concepts
  - Step 04 - Exploring IaaS vs PaaS - Cloud Computing with AWS
  - Step 05 - Getting Started with AWS Elastic Beanstalk
  - Step 06 - Demo - Setting up Web Application with AWS Elastic Beanstalk
  - Step 07 - Demo - Playing with AWS Elastic Beanstalk
  - Step 08 - Understanding the Need for Docker and Containers
  - Step 09 - Exploring Container Orchestration in AWS
  - Step 10 - Demo - Setting up ECS Cluster with AWS Fargate
  - Step 11 - Demo - Playing with Amazon ECS
  - Step 12 - Getting Started with Serverless in AWS - AWS Lambda
  - Step 13 - Demo - Creating Your First Lambda Function
  - Step 14 - Demo - Playing with Lambda Functions
  - Step 15 - Cloud Computing in AWS - Quick Review of Compute Services

## **Deploying Spring Boot Applications to AWS**

- Step 01 - Deploying Hello World Spring Boot App to AWS
- Step 02 - Exploring AWS Elastic Beanstalk - Your First Spring Boot App in AWS
- Step 03 - Running Spring Boot REST API with MySQL Database as Docker Container
- Step 04 - Deploying Spring Boot REST API with MySQL to AWS Elastic Beanstalk and RDS
- Step 05 - Exploring AWS Elastic Beanstalk and Amazon RDS - Spring Boot REST API
- Step 06 - Exploring Spring Boot and React Full Stack App
- Step 07 - Deploying Full Stack Spring Boot REST API to AWS Elastic Beanstalk
- Step 08 - Deploying Full Stack React App to Amazon S3

## **Introduction to Functional Programming with Java**

- Step 00 - Introduction to Functional Programming - Overview
- Step 01 - Getting Started with Functional Programming with Java
- Step 02 - Writing Your First Java Functional Program
- Step 03 - Improving Java Functional Program with filter
- Step 04 - Using Lambda Expression to enhance your Functional Program
- Step 05 - Do Functional Programming Exercises with Streams, Filters and Lambdas
- Step 06 - Using map in Functional Programs - with Exercises
- Step 07 - Understanding Optional class in Java

- 
- Step 08 - Quick Review of Functional Programming Basics

### **Congratulations - Master Spring Framework and Spring Boot**

- Congratulations - Master Spring Framework and Spring Boot