V1.0

Training Program on

Java Spring Boot

Java Spring Boot Training Program

What is Java?

Java Spring Boot is an open-source framework for building web applications and microservices using the Java programming language. It is designed to simplify the process of creating standalone, production-grade Spring-based applications that can be deployed with minimal configuration.

Spring Boot provides a set of pre-configured components, such as the web server, database connectivity, security, and data access, which makes it easy for developers to focus on writing business logic and application functionality rather than worrying about configuring the infrastructure.

Spring Boot also comes with a command-line tool, the Spring Boot CLI, that enables rapid application prototyping and development. It supports various web frameworks, such as Spring MVC, and it integrates seamlessly with other popular Java frameworks, such as Hibernate and Apache Kafka.

Overall, Java Spring Boot is a powerful tool for Java developers who want to build scalable, efficient, and production-ready web applications and microservices

Advantages of Java Spring Boot in Application Development:

- Rapid development: Spring Boot allows developers to quickly and easily create Spring-based applications with minimal setup and configuration. This speeds up the development process and enables developers to focus on writing business logic and application functionality rather than infrastructure code.
- Simplified configuration: Spring Boot provides pre-configured components that can be easily customized using simple configuration files or annotations. This makes it easy to configure and deploy applications in different environments without the need for complex XML configuration files.
- Microservices architecture: Spring Boot is well-suited for building microservices-based architectures, where each service can be developed and deployed independently. It supports various protocols for communication between microservices, such as REST, SOAP, and Apache Kafka.
- Security: Spring Boot provides built-in security features such as authentication and authorization, which can be easily customized to meet specific application requirements. It also integrates with popular security frameworks such as OAuth and Spring Security.
- Performance: Spring Boot applications are designed to be highly performant and scalable. It includes features such as caching, connection pooling, and reactive programming that help improve the performance of applications.
- Large ecosystem: Spring Boot has a large and active community of developers, which provides access to a wide range of plugins, tools, and libraries that can be used to extend its functionality.

Java Spring Boot offers many advantages for application development, including rapid development, simplified configuration, support for microservices architectures, security, performance, and a large ecosystem of developers and tools.

.Audience

The Java Spring Boot training is a 4-days course designed for:

- Software Professionals
- Developers working in an IT environment
- Java developers, software engineers, and web application developers who are familiar with Java programming language and want to learn how to use Spring Boot to build scalable, efficient, and production-ready web applications and microservices.
- The training program may also be suitable for professionals who are responsible for deploying and managing Spring Boot applications in a production environment, such as DevOps engineers, system administrators, and technical leads.

Overall, anyone who is interested in learning about Java Spring Boot and how it can be used to build modern web applications and microservices can benefit from a training program on this topic.

Learning Objectives:

- Understand the fundamentals of Spring Boot, including its architecture, key features, and benefits.
- Set up a development environment and create a new Spring Boot project.
- Build RESTful web services using Spring Boot's built-in web server.
- Work with databases in Spring Boot, including configuring data sources, creating JPA entities, and using Spring
- Use Spring Boot Actuator to monitor and manage Spring Boot applications.
- Use Spring Boot's caching and transaction management features to improve application performance.
- Deploy Spring Boot applications to different environments, including on-premise and cloud-based platforms.
- Configure Spring Boot for production environments, including logging, error handling, and monitoring.

Overall, an understanding of Spring Boot and its various features and benefits, and equips them with the skills and knowledge required to develop, test, deploy, and manage Spring Boot applications in a production environment.

Candidate Prerequisites

- Basic computer skills: Participants should be comfortable using a computer and have a basic understanding of computer programming concepts.
- Proficiency in Java programming language, including object-oriented programming concepts such as classes, objects, inheritance, and polymorphism.
- Knowledge of web development concepts, such as HTML, CSS, and JavaScript.
- Familiarity with the basics of database design and SQL.
- Basic understanding of software development principles and practices, such as version control, testing, and debugging.

Additionally, participants should have a laptop or desktop computer with the necessary software and tools installed, such as a Java Development Kit (JDK), an integrated development environment (IDE) such as Eclipse or IntelliJ IDEA, and a web browser. The specific software and tools required may vary depending on the course and the instructor's recommendations.

Infrastructure Prerequisites

Software:

Java Development Kit (JDK) version 11 or later.
An integrated development environment (IDE) such as Eclipse or IntelliJ IDEA.
Spring Tool Suite (STS), a popular IDE for Spring Boot development.
A database management system such as MySQL or PostgreSQL.
A web server such as Apache Tomcat or Jetty.
Git, a version control system for managing source code.
Postman or a similar API testing tool.

Hardware:

A laptop or desktop computer with at least 8GB of RAM and a multi-core processor.
A reliable internet connection.

Cloud Infrastructure:

Optional, but beneficial for practising deployment and scaling of applications.
Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform (GCP) provide free trials and services for learning and practice purposes.
Note that the specific software and cloud infrastructure requirements may vary depending on the course and the instructor's recommendations.

Overall, participants should have access to a computer with the necessary software and hardware requirements to complete the exercises and activities within the training program. Cloud infrastructure is optional but recommended for learning cloud-based Java spring boot concepts.

Training Outline:

Note: The duration and the contents are approximate. The trainer may change the approach and the duration, based on the response of the participants.

Day 1

Session 1: Getting Started with Spring Boot

- Overview of Spring Boot
- Setting up a Spring Boot development environment
- Creating a new Spring Boot project
- Understanding the project structure

Session 2: Building RESTful APIs with Spring Boot

- Understanding RESTful web services
- Creating a RESTful API using Spring Boot
- Implementing GET, POST, PUT, and DELETE requests
- Testing the API using Postman

Session 3: Working with Data in Spring Boot

- Understanding Spring Data
- Configuring a data source
- Creating and querying a database
- Mapping database tables to Java objects

Session 4: Hands-On Activity

- Build a simple RESTful API using Spring Boot
- Implement GET, POST, PUT, and DELETE requests
- Store and retrieve data from a database

Day 2: Advanced Spring Boot Topics

Session 5: Security and Authentication with Spring Boot

- Understanding Spring Security
- Configuring authentication and authorization
- Using JSON Web Tokens (JWT) for authentication
- Securing RESTful APIs\

Session 6: Spring Boot Testing

- Understanding different types of testing
- Writing unit tests for Spring Boot applications
- Integration testing with Spring Boot
- Mocking dependencies for testing

Session 7: Spring Boot DevTools

- Introduction to Spring Boot DevTools
- Understanding hot reloading
- Using DevTools for faster development cycles
- Debugging Spring Boot applications

Session 8: Hands-On Activity

- Implement authentication and authorization in the RESTful API
- Write unit and integration tests for the API
- Use DevTools to improve development workflow

Day 3: Microservices with Spring Boot

Session 9: Introduction to Microservices

- Understanding microservices architecture
- Advantages and disadvantages of microservices
- Comparison with monolithic architecture

Session 10: Building Microservices with Spring Boot

- Designing microservices
- Creating multiple Spring Boot applications
- Communicating between microservices using RESTful APIs
- Implementing service discovery and registration with Eureka

Session 11: Load Balancing and Resilience

- Implementing load balancing with Ribbon
- Handling failures and errors with Hystrix
- Using circuit breakers for resilience

Session 12: Hands-On Activity

- Design and build microservices using Spring Boot
- Implement communication between microservices
- Use Ribbon and Hystrix for load balancing and resilience

Day 4: Deploying Spring Boot Applications

Session 13: Packaging Spring Boot Applications

- Understanding different packaging options
- Creating an executable JAR file
- Building and deploying a Docker container

Session 14: Continuous Integration and Deployment

- Setting up a continuous integration (CI) pipeline
- Automating deployment with CI tools
- Best practices for CI/CD

Session 15: Performance Tuning

- Identifying performance bottlenecks
- Tuning Spring Boot applications for performance
- Caching data to improve performance

Session 16: Hands-On Activity

- Package the microservices into an executable JAR file
- Deploy the microservices to a Docker container
- Set up a CI/CD pipeline for automated deployment

OPTIONAL PROJECT

Note: Additional Days Will Be Needed
Depending on the organization, the project may change

Project: Simple e-commerce web application project

Description:

A simple e-commerce web application that allows users to browse products, add them to a cart, and place orders. The application should consist of multiple microservices that communicate with each other using RESTful APIs. The microservices should be deployed in Docker containers, and the application should be deployed to a cloud platform such as AWS or Azure. This project will allow candidates to apply their knowledge of Spring Boot, microservices, and deploy into a real / production environment.

Requirements:

- The application should allow users to browse products, add them to a cart, and place orders.
- The application should consist of multiple microservices that communicate with each other using RESTful APIs.
- The application should be scalable and fault-tolerant.
- The application should be deployed to a cloud platform such as AWS or Azure.

Tools/Technologies:

- Java Spring Boot for developing the microservices and RESTful APIs.
- React.js for developing the frontend of the web application.
- Docker for containerization and deployment of the microservices.
- Kubernetes or Docker Swarm for orchestration of the microservices.
- AWS or Azure for cloud deployment and hosting of the application.

Required Coding:

- Develop a product microservice that allows users to browse products.
- Develop a shopping cart microservice that allows users to add products to a cart and checkout.
- Develop an order microservice that allows users to place orders.
- Develop a user authentication microservice that handles user login and registration.
- Develop RESTful APIs for communication between the microservices.
- Develop a React.js frontend for the web application that consumes the RESTful APIs.

Databases:

- Each microservice can have its own database or they can share a common database.
- A NoSQL database such as MongoDB or Cassandra can be used to store product information.
- A relational database such as MySQL or PostgreSQL can be used to store user information and orders.
- Overall, this project will require candidates to apply their knowledge of Java Spring Boot, microservices, RESTful APIs, Docker, and cloud deployment to develop a scalable and fault-tolerant e-commerce web application. It will also require knowledge of frontend development using React.js and database design and implementation.

This project can be completed individually or in a group and the trainer can use the performance, to assess the knowledge and skills of the participants, which they acquired from the training program