Table of Contents

UNIT 1: Spring Boot - Introduction

Objectives

In this Unit you will learn to –

- Define Spring Boot
- Identify the need for Spring Boot and its advantages over other frameworks
- Describe the features of Spring Boot and its components
- Compare and contrast Spring Boot with Spring and Spring MVC
- Create a new Spring Boot project using Spring Initializr and import it into an IDE
- Develop and test a "Hello World" application using Spring Boot

Learning Outcome:

Gain an introduction to Spring Boot.

1. Spring Boot - Introduction

1. Introduction to Spring Boot

Spring Boot is an open-source Java-based framework that is used to create stand-alone, production-grade Spring-based applications with minimal configuration. Spring Boot is built on top of the Spring framework and provides additional features and functionality that simplifies the development of Spring-based applications. The primary goal of Spring Boot is to make it easier to create Spring applications by reducing the amount of boilerplate code required and providing defaults for many configuration options.

2. Need for Spring Boot

Challenges with traditional Spring framework

- The traditional Spring framework requires a lot of configuration and setup, which can be time-consuming and error-prone.
- Developers need to configure various XML files, annotations, and properties to get a Spring application up and running.
- This complexity can be overwhelming, especially for beginners, and can lead to errors and bugs in the application.

How Spring Boot addresses these challenges

- Spring Boot provides a simpler and faster way to create Spring-based applications by reducing the amount of configuration required.
- It offers a convention-over-configuration approach, which means that developers only need to define what is different from the defaults.
- Spring Boot also provides a wide range of pre-built and pre-configured components, such as embedded servers, security, and data access, that can be easily integrated into the application.

Use cases for Spring Boot:

- Spring Boot is widely used for developing microservices and RESTful APIs.
- It is also used for developing web applications, data processing applications, and batch applications.
- Spring Boot can be used in both small and large-scale projects, and its flexibility makes it an ideal choice for modern software development.

3. Spring Boot Features

Spring Boot provides a wide range of features that make it easier to develop Spring-based applications. Some of the key features of Spring Boot include:

Auto-configuration

- Spring Boot provides auto-configuration, which automatically configures the application based on the dependencies added to the project.
- It eliminates the need for manual configuration and saves a lot of time.

Embedded servers

- Spring Boot comes with embedded servers, such as Tomcat, Jetty, and Undertow, which allows developers to package the server with the application and run it as a standalone executable JAR file.
- This simplifies the deployment process and makes it easy to deploy the application to various environments.

Production-ready features

- Spring Boot provides several production-ready features, such as health checks, metrics, and monitoring, which make it easy to monitor and manage the application in production.

- It also supports various logging frameworks and provides error handling and exception reporting.

Simplified dependency management

- Spring Boot provides simplified dependency management, which makes it easy to manage dependencies and resolve conflicts.
- It uses a curated set of dependencies, which are compatible with each other and reduces the risk of version conflicts.

Spring Boot CLI

- Spring Boot provides a CLI (Command Line Interface) tool, which allows developers to quickly create and test Spring Boot applications from the command line.
- It provides a convenient way to develop and test applications without the need for an IDE.

Actuator

- Spring Boot Actuator provides several endpoints that expose information about the application, such as health, metrics, and environment variables.
- It allows developers to monitor and manage the application in production.

Developer tools

- Spring Boot provides several developer tools, such as live reload, which automatically reloads the application when changes are made to the code.
- It also provides a development-time database, which makes it easy to test the application without the need for an external database.

4. Spring vs Spring Boot vs Spring MVC

Spring

- Spring is a framework for building enterprise applications.
- It provides several modules, such as Spring Core, Spring JDBC, and Spring MVC, which can be used individually or together to build applications.
- Spring provides inversion of control (IoC) and dependency injection (DI) features, which make it easy to manage dependencies and promote loose coupling.
- Spring is highly customizable and allows developers to configure the application according to their requirements.

Spring Boot

- Spring Boot is a framework for building standalone Spring applications.
- It provides several features, such as auto-configuration, embedded servers, and simplified dependency management, which makes it easy to develop and deploy Spring applications.
- Spring Boot reduces the time and effort required to configure the application and promotes convention over configuration.
- Spring Boot is opinionated and provides a curated set of dependencies, which are compatible with each other and reduces the risk of version conflicts.

Spring MVC

- Spring MVC is a module of the Spring framework for building web applications.
- It provides several features, such as model-view-controller (MVC) architecture, request mapping, and view resolution, which make it easy to build web applications.
- Spring MVC promotes separation of concerns and allows developers to write clean and modular code.

Comparison Table

Features	Spring	Spring Boot	Spring MVC
Purpose	Framework for building enterprise applications	Framework for building standalone Spring applications	Module of Spring for building web applications
Configuration	Requires manual configuration	Provides auto-configuration and convention over configuration	Requires manual configuration
Dependency Management	Requires manual dependency management	Provides simplified dependency management	Requires manual dependency management
Embedded Server	Not included	Provides embedded servers, such as Tomcat and Jetty	Not included
Production-Ready Features	Not included	Provides several production-ready features, such as health checks and metrics	Not included
Opinionated	No	Yes	No
IoC and DI	Yes	Yes	Yes
MVC architecture	Yes	No	Yes
Request Mapping	Yes	No	Yes
View Resolution	Yes	No	Yes

As seen from the table, Spring is a framework for building enterprise applications, while Spring Boot is a framework for building standalone Spring applications.

Spring MVC is a module of the Spring framework for building web applications. Spring requires manual configuration and dependency management, while Spring Boot provides auto-configuration and simplified dependency management. Spring MVC provides several features, such as MVC architecture, request mapping, and view resolution, which make it easy to build web applications.

5. Creating Project - Spring Initializr

Spring Initializr is a web-based tool that allows developers to quickly create and configure new Spring Boot projects. It provides a simple interface where developers can choose the project type, language, build tool, and dependencies for their project.

The different components of Spring Initializr include:

1. **Project Metadata:** This includes information such as project name, group ID, artifact ID, package name, and description.
2. **Project Type:** Spring Initializr supports different project types such as Maven and Gradle.
3. **Language:** Spring Initializr supports different programming languages, including Java and Kotlin.
4. **Build Tool:** Spring Initializr supports different build tools, including Maven and Gradle.
5. **Dependencies:** Spring Initializr provides a list of dependencies that developers can add to their projects. This includes Spring Boot starter dependencies and other third-party dependencies.

Creating a new Spring Boot project using Spring Initializr is a straightforward process. Here's a step-by-step guide on how to create a new Spring Boot project using Spring Initializr:

1. Open your web browser and navigate to https://start.spring.io/.
2. Choose the necessary options for your project in the "Project Metadata" section. This includes the Group, Artifact, and Version of the project.

3. Select the dependencies that you want to include in your project by clicking the "Add Dependencies" button. Spring Initializr provides a wide range of dependencies for your Spring Boot project.
4. Choose the build tool that you want to use for your project. You can choose between Maven and Gradle.
5. Select the programming language that you want to use for your project. Spring Initializr supports Java, Kotlin, and Groovy.
6. Choose the Spring Boot version that you want to use for your project.
7. Click the "Generate" button to create your Spring Boot project.
8. Once the project has been generated, you can download the zip file containing the project or directly copy the generated code and paste it into your local IDE.

The different configurations and options available during project creation include:

1. Project Metadata: This includes the Group, Artifact, and Version of the project.
2. Dependencies: Spring Initializr provides a list of dependencies that developers can add to their projects. This includes Spring Boot starter dependencies and other third-party dependencies.
3. Build Tool: Spring Initializr supports different build tools, including Maven and Gradle.
4. Programming Language: Spring Initializr supports Java, Kotlin, and Groovy.
5. Spring Boot Version: Spring Initializr provides the option to select the Spring Boot version to use.

Once the project has been generated, you can import it into an IDE such as Eclipse or IntelliJ. To import the project, follow these steps:

1. Open your IDE.

2. Click on "File" > "Import".
3. Select "Existing Maven Projects" or "Existing Gradle Projects" and click "Next".
4. Browse to the directory where you have saved the downloaded Spring Boot project and click "Next".
5. Follow the prompts to import the project into your IDE.

Once the project is imported, you can run it using the main method or create a Spring Boot run configuration in your IDE.

6. Spring Boot Hello World Application

1. Setting up the development environment:
    - Install Java and any preferred IDE (e.g. Eclipse, IntelliJ)
    - Install Spring Boot
2. Creating a new Spring Boot project:
    - Use Spring Initializr to create a new project
    - Select the required dependencies, such as Spring Web
3. Creating the controller:
    - Create a new Java class for the controller
    - Use the @RestController annotation to mark the class as a controller
    - Create a method that returns the string "Hello World" using the @GetMapping annotation
4. Running the application:
    - Run the application using the IDE or the command line
    - Open a web browser and go to the URL http://localhost:8080/
5. Testing the application:

- Verify that the browser displays "Hello World"
- Test the application using a REST client (e.g. Postman)

Here's the code for the "Hello World" application:

```
import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.RestController;


@RestController

public class HelloWorldController {


    @GetMapping("/")

    public String hello() {

        return "Hello World";

    }

}
```

This code creates a new controller that returns the string "Hello World" when the application is accessed at the root URL (http://localhost:8080/). When you run the application and access the URL in a web browser, you should see "Hello World" displayed on the page.

Conclusion

In this unit, we have learned about Spring Boot, its features, and the benefits it provides over traditional Spring and Spring MVC. We have also learned how to create a Spring Boot project using Spring Initializr and how to create a "Hello World" application.