Spring vs Springboot

By Siwareddy

Spring	SpringBoot
• Lot of boiler plate code need to implement(configure) to start the Spring Project using JDBC, JPA, Hibernate, MVC etc. • XML Configurations Required • Additional configuration is required to integrate AOP, Batch, Security	• It minimizes writing multiple boilerplate codes, XML configuration and annotation, ultimately enhancing productivity while reducing lots of development time. • Spring Boot makes it easy to create stand-alone, production-grade Spring-based Applications that you can run. • It is opinionated and follows the convention over configuration software design paradigm • It makes it easier to integrate the Spring Boot Application with the Spring Ecosystem that majorly includes Spring ORM, Spring JDBC, Spring Security, Spring Data and many other things.
Developer manually define dependencies for Spring project in pom.xml which is tedious job when Spring project upgrade to new version of the Spring Framework	• Spring Boot enables you to rapidly build and create Spring applications utilizing the Spring framework. • SpringBoot comes with concept of starter in pom.xml file which internally take care of downloading the dependencies jars based on your SpringBoot requirement.
To test the Spring project, Server environment setup needs to be done explicitly	• Spring Boot offers embedded http servers such as Jetty, Tomcat, etc. where developer deploy the code and test the functionality which improves the productivity of the developer • It offers Command Line Interface (CLI) tool for develop coding in Groovy to eliminate the boiler platform code.
Spring does not provide support for in memory databases	• It offers several plugins for working with embedded and in-memory databases easily (E.g. H2) • SpringBoot provides Actuator to monitor and manage our application • Springboot framework used to develop microservices.
