# [incomplete] Spring Boot 3: The Big Picture

History of Java –

A diagram of a computer

Description automatically generated

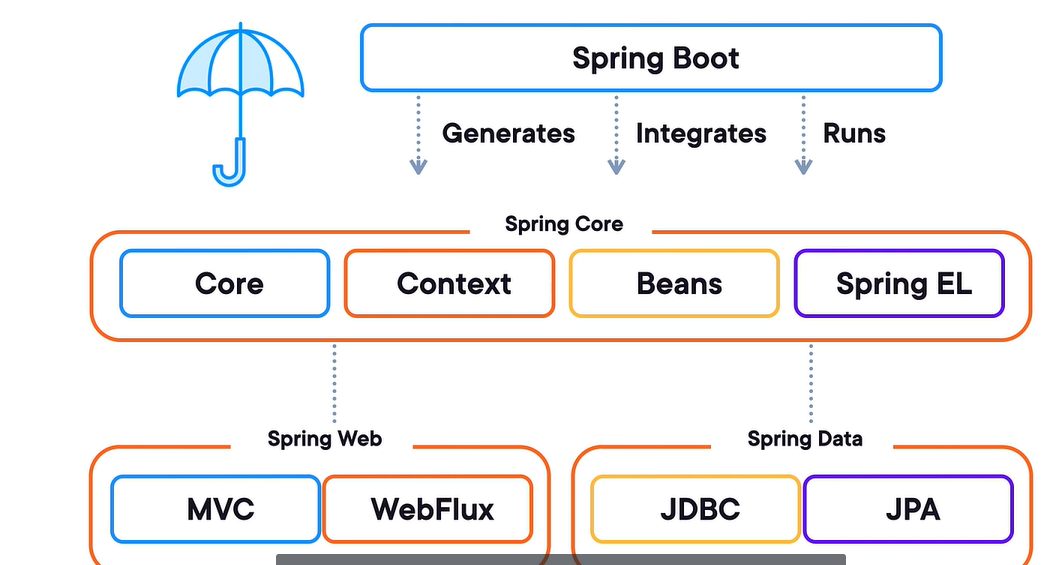
Java was created in the early nineteenth by James Gosling, Mike Sheriden and Patrick Naughton while working at Sun Microsystems.

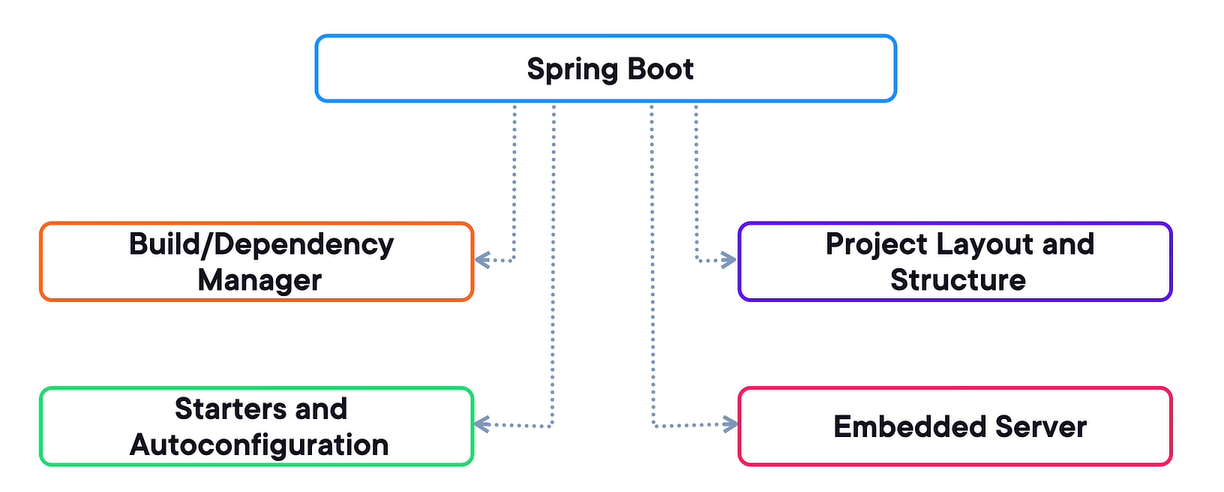
JEE provided a specification on how to build applications and deploy them on prebuilt applications servers. Now, instead of building servers, you could build Enterprise Java beans, servlets and other application-related pieces and deploy them on existing application servers. WebLogic and WebSphere servers were costly due to which open-source servers like JBoss and Tomcat popped up. Spring wasn’t an application server but an alternative way to build java applications. No more EJB’s or persistent Beans. Spring’s goal was to reduce cost, integrate more frameworks, and run entire enterprise systems in the open-source world.

Spring also contains a well thought out approach to handling presentation data and persistent information in the spring MVC and data projects.

Spring contains the Spring Boot project, which is the foundational project that boots all other Spring projects and libraries, including the core Spring framework.

Spring boot generates, integrates and runs your Spring app so that you can start working on your app business logic rather than nonfunctional software effort for your project.





Spring Boot was all about making server-side development in the enterprise world easier and faster. It is like Ruby on Rails, Python/Django, Javascript/Node.

Java EE moved to Jakarta EE.

Alternative Frameworks –

A screen shot of a computer screen

Description automatically generated

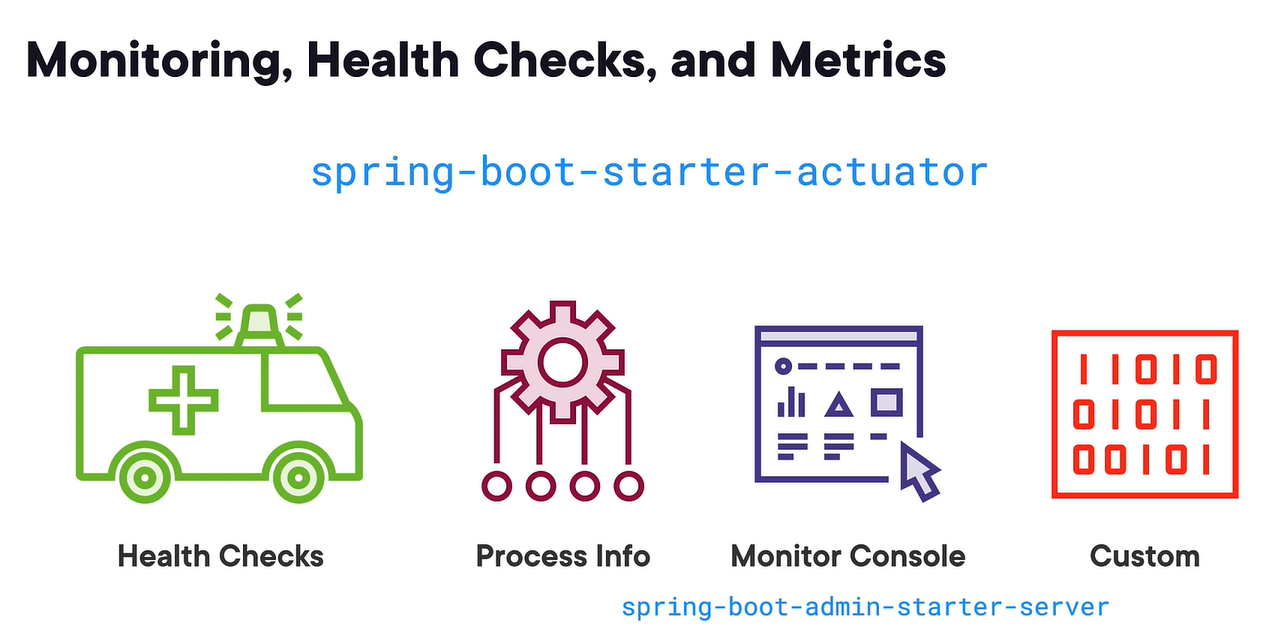
With Spring boot, we don’t need to deploy the WAR file on Tomcat server, instead Spring Boot embedded the application container inside of itself and deployed the Java app to the container under the convers. By this we can simply create a basic Java application using the public static void main method and Spring Boot will then bootstrap our Spring application, prepare it, and get it deployed. It will take our code and deploy it to the embedded Tomcat servlet container, and then fire up the server and make it available for use.

A screenshot of a computer program

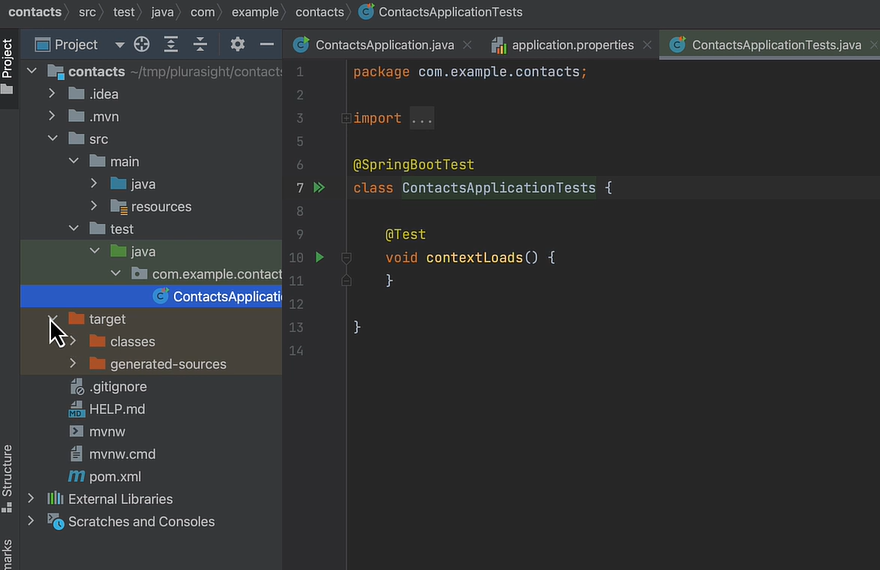
Description automatically generated

Spring boot manages all our application dependencies using our choice of Maven or Gradle as the build tool.

We can use spring-boot-starter-actuator for health checks, process info, monitor console and custom.



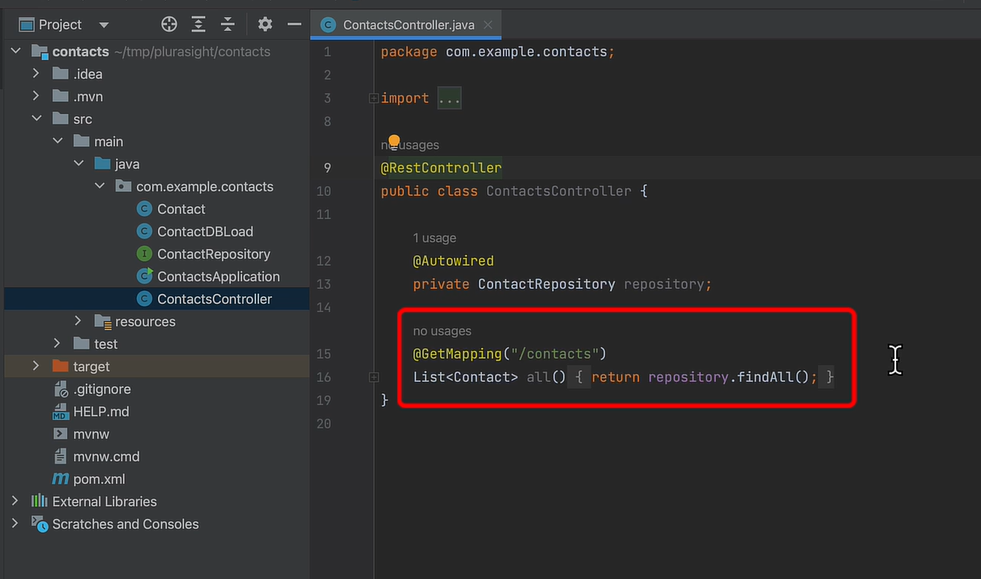
In the target folder Maven build artifacts are placed and everything compiles down inside of there.



Sample REST API code –

A screenshot of a computer

Description automatically generated



A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

To run the spring boot application, we just need to execute the package Spring Boot JAR file by using the “java-jar <boot\_jarfile> command.

# Building an SPA Using Spring Framework 5 and Angular

High level architecture –

A diagram of a server

Description automatically generated

Spring boot is the starter kind of bootstrap for spring projects.

Java starts its life from “public static void main” function.

In application.properties we store configuration like database credentials or other environment variables.

We will create a JPA model which will be based on ORM. JPA will use Hibernate undercover which is an implementation of JPA by this it provides some handler methods and other methods on our entity so that the JPA will work properly.

To prevent the Hibernate methods from getting marshaled when we convert it to JSON we need to use the JsonIgnoreProperties annotation.

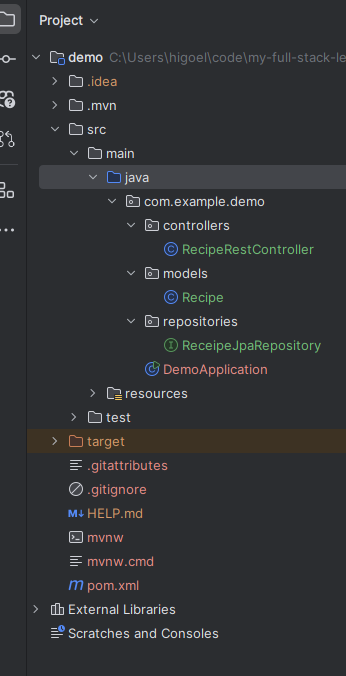
Spring Data JPA auto builds a lot of our query needs for us based off of our entity model. So, by this we will get all of our CRUD queries built for us automatically. These are simply a Java interface.

This is called dependency injection in Spring framework –

A screenshot of a computer program

Description automatically generated

A Spring Data JPA repository is an interface that connects to the recipe JPA entity. Once connected, Spring Data JPA will provide a good portion of our persistence needs out of the box i.e. basic CRUD operations. Spring has provided us with a decoupled three-tiered back-end architecture i.e. our REST or our service or access layer, then our model or database layer.



Using proxy config, we can even directly hit the server URL on 8080 through 4200 angular application directly on the browser as-well –

A screenshot of a computer

Description automatically generated

We can use Flyway DB while choosing a tool for database version migration.

# Building an SPA Using Spring Framework 6 and Angular

At its core, spring handles various backend concerns like dependency injection, data access, transaction management and security, allowing us to focus more on writing business logic rather than boilerplate code. With Spring Boot, an extension of Spring, developers can rapidly build REST APIs and microservices with minimal configuration.

Spring Boot and Angular compliments each other very well –

A close-up of a pink card

Description automatically generated

Convention over configuration approach simplifies the development by providing sensible defaults, reducing the need for excessive configuration. In Spring Boot, the framework automatically scans for components, controllers and services without requiring developers to configure these classes explicitly in XML or other configuration files. If a class flows a convention, like being annotated with a controller annotation, then Spring Boot automatically wires into the application context. Allowing for rapid REST API development, Spring also integrates easily with Spring Data for database management and Spring Security for handling user authentication and authorization.

Both frameworks (angular and spring boot) are built with modularity and scalability in mind. Spring handles backend concerns, such as REST APIs, data access, and security, while angular focuses on building dynamic, interactive UIs. Combination of both really shine in enterprise web applications where both the front end and the backend need to handle large-scale operations and data, real-time data apps and microservice architecture.

The Spring can take care of Data processing and business logic and angular can take care of responsive and interactive user experience.