Spring Boot

02 August 2025 10:37

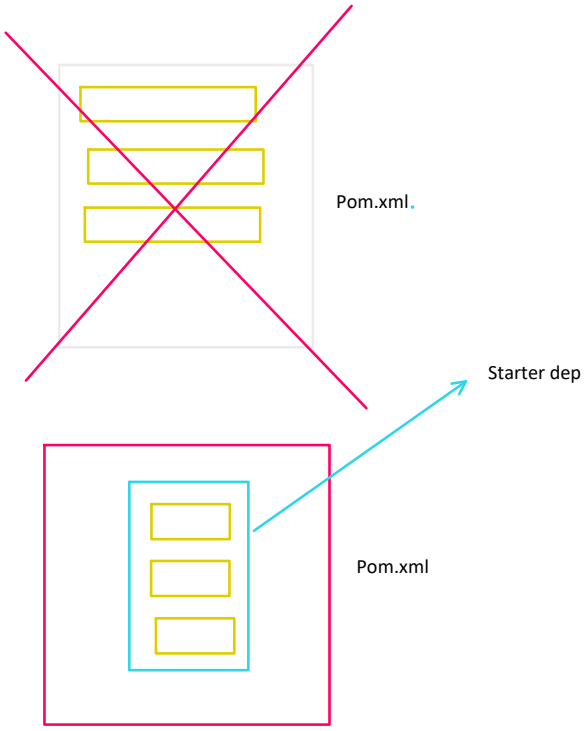
=> Spring boot is one approach to develop spring based applications with less configurations.
Spring boot = Spring framework - xml configuration

Spring boot is not replacement for spring core, Spring boot developed on top of spring core.

Note: All Spring Framework concepts can be used in spring boot.

Spring Boot - Advantages
---

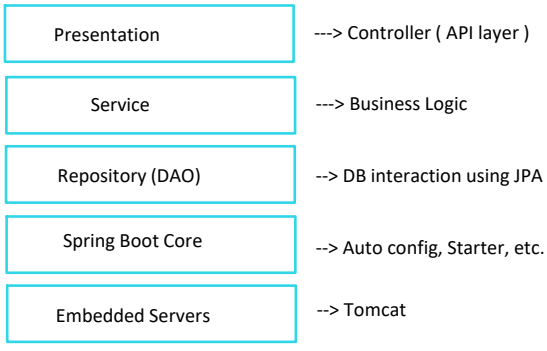
1. Less configuration & No xml configurations
2. Pom starters to simplify dependencies configuration
   a. Spring-boot-starter-web
   b. Spring-boot-starter-data-jpa
   c. Spring-boot-starter-security
   d. Spring-boot-starter-mail
3. Auto configuration
4. Embedded servers (ex: tomcat, jetty, netty)
5. Actuators (production ready feature)



Note: Spring boot makes it easy to create stand-alone, production-grade spring based application that you can "just run."

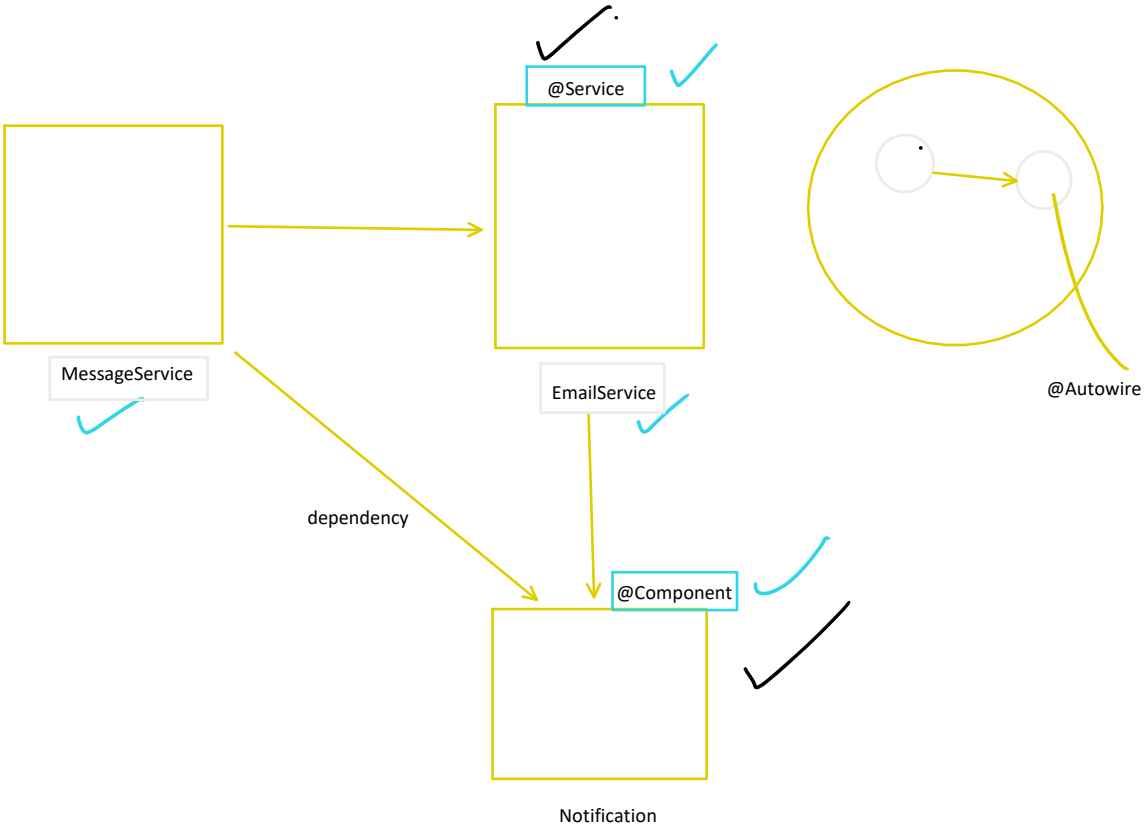
Spring boot 1.0 released in 2014
Current version of spring boot is 4.x 2025 may
Note: java 17 is mandatory to work with spring boot 3.x version

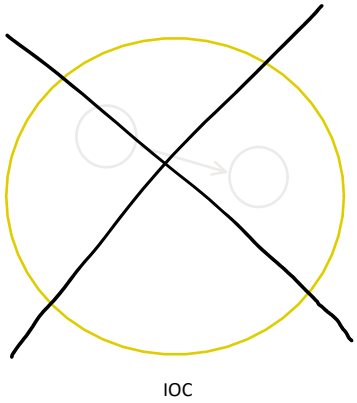
Spring boot Architecture:



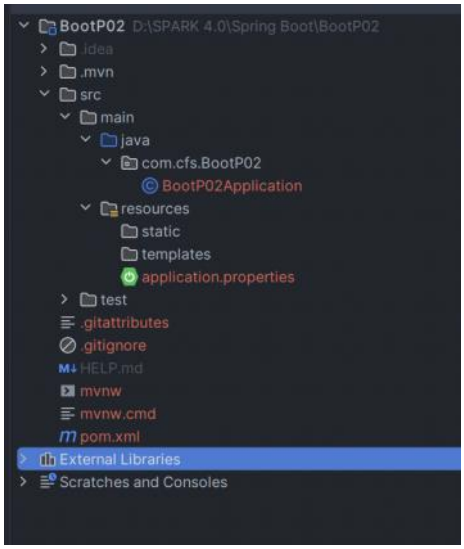
Spring boot Application creation

-> We can create boot application in 2 ways
1. Initializer website (start.spring.io)
2. IDE





200 %



Src/main/java -> to keep our project source code
-Application.java

Src/main/resource -> to keep project configuration files
-application.properties /yml

Src/test/java -> to keep junit code (unit testing)
-ApplicationTest.java

Src/test/resources -> unit testing releated config

External libraries -> dependencies required for project

Pom.xml -> maven configuration file.

In-depth understanding of @SpringBootApplication

```java
@SpringBootApplication
public class BootP02Application {

    public static void main(String[] args) {
        SpringApplication.run(BootP02Application.class, args);
    }

}
```

Internal flow of spring boot application

---

1. It is entry point for spring boot application
2. SpringApplication.run(….) -> Triggers
3. Creates IOC container
4. Scan for @component / @RestController
5. Auto config beans (like tomcat)
6. Register dispatcher Servlet
7. Start embedded tomcat server

2. @SpringBootApplication annotation is equal to below 3 annotations
   - @SpringBootConfiguration
   - @EnableAutoConfiguration
   - @ComponentScan

Note:
1. all spring annotations tell IOC that please handle me
2. Each annotation has there it own meaning.

Requirement : if we need to represent java class as configuration class then we will use @Configuration

@SpringBootConfiguration

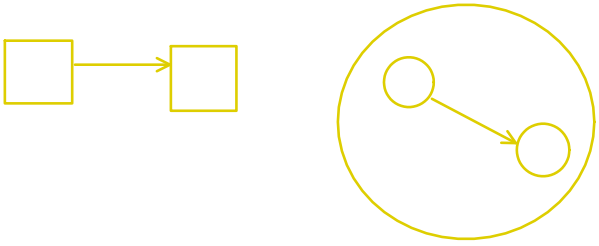
By looking this prove we can say that class where @SpringBootApplication act as configuration.

@Configuration

=> Spring boot start class will act as Configuration class because of @SpringBootApplication annotation

=> In Spring boot application auto configuration feature will be available because of @EnableAutoConfiguration

What is Autowiring ?



Annnotation

Note: for Autowiring beans are required

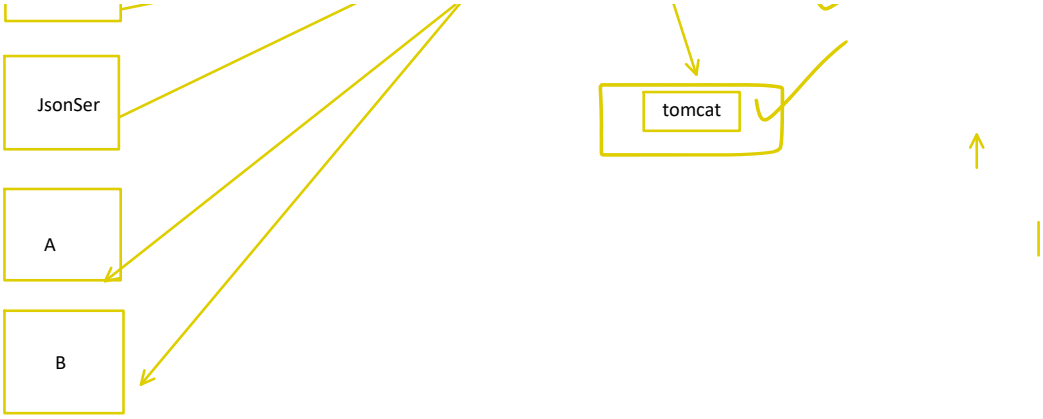
What is auto config ?


```xml
<parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>3.5.4</version>
    <relativePath/> <!-- lookup parent from repository -->
</parent>
```

```xml
<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>

</dependencies>
```

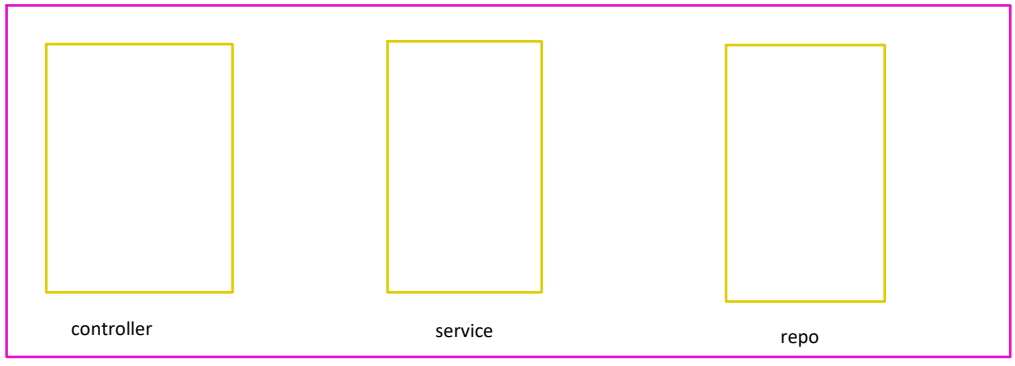
Tomcat
Spring mvc
Json
dispatcherServlet



Component scan will be performed in spring boot because of @ComponentScan annotation

Note: package naming convention will important role in component scanning.

Eg: eg.cfs
Eg.cfs.controller
Eg.cfs.service



StereoType Annotation

This is a spring-manged component - please detect it and register it as bean in the container.

Annotation	Purpose	Layer
@Component	Generic spring managed bean	Any

@Service	Make a service (business logic)	Service layer
@Repository	Make a DAO (data access) class	Repo
@Controller	Marks a web MVC controller (Spring MVC)	Web Layer
@RestController	Combines @Controller + @ResponseBody	Rest APIs

Note: All of them are meta-annotated with @Component, which is why spring picks them up during @ComponentScan

Spring boot + REST API