SPPRIN BOOT (Introduction)

Definition: *spring is open source java based spring module or framework that simplifies the development of an application by it's the some pre-defined configuration (auto-configuration) that reduces the need of boiler plate codes.*

Why Spring Boot? :-

- *Very simple to create spring based application*
- *It cuts down the development time.*
- *Eliminates to write repetitive codes*
- *Its very simple to combine the spring boot application with its eco system like spring JDBC, Spring ORM, Spring Data, Spring Security etc..*
- *Spring boot follows opiniated default configuration approach.*
- *Provides embedded servers like tomcat.*

Difference between dependency and plugins:

Dependency: *The third party module or library that our application need to provide additional functionality.*

- *Spring boot provides sets of starter dependencies known as "Spring-boot-starter"(pre-configured).*

Plugins: *Application oriented add-on features. <artifactId>spring-boot-maven-plugin</artifactId> :- it is responsible for build packaging and running spring boot framework.*

NOTE: (V.V.I)

When my requirement is to run the spring boot application as normal java main class then in this condition we need @SpringBootApplication , in this case there is no need of spring-boot-maven-plugin because we know that both the things are responsible to run the spring boot application. When my reqirement is to run the spring boot application on command-prompt in this we need spring-boot-maven-plugin as mvn spring-boot:run

During development or testing time, there is no need of spring-boot-maven-plugin but when I deployed the application on production server I need this because to deploy on server I need war or jar file, and this plugin help us to convert our application as jar or war file.

IMPORTANT ANNOTATIONS

1. ***@SpringBootApplication:*** *Entry point of Spring boot application.it is a combination of three annotation.*
   - *@Configuartion : used for declaring the class as configuration class and also used to configure spring application context. Generally used for custom configuration.*
   - *@EnableAutoConfiguration: It triggers the auto configuration feature of spring boot.*
   - *@ComponentScan: specified the base package to scan the components whatever present in base pakage to for spring Application Context configuration.*
2. ***@controller****: used to indicate that this class is capable of handling protocol request and provide response in the form of view or template. Used in Spring MVC*
3. ***@RestController****:used to indicate that this class is capable of handling protocol request and provide response in the form JSON or XML data directly. Used to build RestFul web services.*
4. ***@RequestMapping****; specify the URL pattern or path for the controller or RestController methods. This a generalised annotation for various specified annoattions:*
   - *@PostMapping: specify the path for POST http method for creation of data in db.*
   - *@PutMapping: specify the path for PUT http method for updation of data in db*
   - *@GetMapping: specify the path for GET http method for retrieval of data from db*
   - *@DeleteMapping: specify the path for DELETE http method for deletion of data from db*
5. ***@PathVariable****: Bind value from the path to the parameter of the method.*
6. ***@RequestParam****: Bind query String as query parameter from the path to parameter of the method.*
7. ***@RequestBody****: Bind Body of the request to the parameter of the method.*
8. ***@ResponseBody****: Bind the return value of a controller method directly to the Http response body.*
9. ***@Autowired****: used for automatic dependency injection*
10. ***@Component****: generalized annotation used for indicating that this is my Spring bean*
    - *@service: specialised annotation for indicating a class as a service class which consist business logic*
    - *@Repository: specialized annotation for indiacating a class as a repository class which is responsible for RDBMS connection and processing.*
    - *@controller: already discussed.*
11. ***@Bean:****when any method returns a object and I want to resgister it in to application context then we should use @Bean.*
12. ***@Value****: inject values from properties file.*
13. ***@Profile****: indicating that which component is belongs to which profile. Profile is nothing but the different-different environment for development and testing*
14. ***@conditional****: based on our requirement we can enable or disable the components*

Basic required dependencies:

1. *Spring-Boot-Starter :* foundation of spring boot application, for every application it works internally whether we are use it in pom.xml file manually or not.
2. *Spring-Boot-Starter-web :* for building web application, it include spring MVC and embedded server for handling HTTP request and response.
3. *Spring-Boot-Starter-data-jpa* :for working with RDBMS using JPA specification. It include Hibernate, Spring data Jpa and database drivers.
4. *Spring-Boot-devtools* : for automatic application restart and live relod when changes detected.
5. *Spring-Boot-Starter-test* :for support for testing framework like JUNIT, spring test.
6. *Spring-Boot-Starter-Security*: for security purpose of application.
7. *Spring-Boot-Starter-logging:* for common logging library like log4j.

Extra dependencies:

- *Lombok*: for reducing boiler plate code in pojo like @data @builder.
- *springdoc-openapi-ui :* for supporting for generation of documentation of application like swagger api
- commons-text: apache library use because it consist various methods and class for text manipulation like random password generator.
- *Spring-Boot-Starter-mail:* support for sending mail

Spring Actuator:

A production ready feature provided by Spring boot which helps to monitor and manage the application using its various pre-defined endpoints like: "/actuator/health"," "/actuator/info". We have to add a dependency for it "Spring-Boot-Starter-Acuator".