**What is SpringBoot**

* SpringBoot is a java framework which is built on top of Spring framework.
* Using SpringBoot we can develop application fastly because developer do not have to do intial set-up and configuration, SpringBoot handles all essential set-ups and configuration implicitly so that a developer can focus on building application rather than focusing on setting up the project and configuring it.

**SpringBoot core principles**

1. Opinionated defaults – means initial setups of project is done by SpringBoot implicitly, programmer does not have to do it manually.

We select all the things in spring intializr like maven, jdk-version, dependencies to be added and tell project structure(like group,artifact,packageName) in springInitializr and our SpringBoot project will be ready, we do not have to manually set-up or configure the project.

It implicitly provide us the application.properties file for providing any other configuration in SpringBoot application.

1. Convention over configuration- In SpringBoot if we follow certain convention(rules) then we can avoid writing boiler plate code.

For ex- Using annotation like @RestController for creating RESTful APIs, @Service for service layers, or creating sub-package inside the main package so they also get scanned by IOC.

1. Auto-configuration- SpringBoot will configure lot of things implicitly by reading project dependencies so that developer does not have provide configuration explicitly

For ex- no need of making dispatcher servlet, handler mapper, view resolver ..SpringBoot will configure all things implicitly by reading dependencies, so programmer can focus on building application.

1. Embedded server – SpringBoot has embedded servers, so need of any external server while making SpringBoot project.

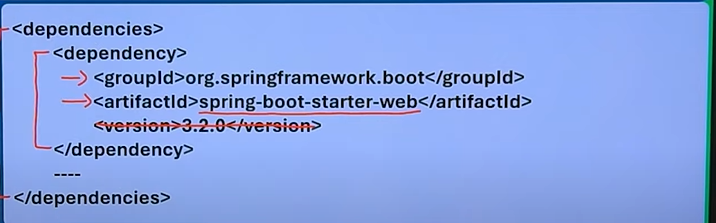
Use of SpringBoot

* Develop web application
* RESTful web services
* Microservices
* Batch processing
* Integrating with other technologies
* Testing
* Monitoring & Management

**SpringBoot Starter**

SpringBoot Starter is a single dependency which will have the collection of other dependencies related to that particular feature or technology.

For ex- if you add spring-boot-starter-web (this dependency will have all the necessary dependencies for developing web-application), spring-boot-starter-data-jpa(gives spring-data-jpa, hibernate, database connection pool)



**Advantage of SpringBoot starter**

* Convenience just add one dependency in pom.xml rest all dependencies SpringBoot Starter will add for that particular feature.
* Minimizing the version mismatch between dependencies.
* Helps developer to focus more on building application rather than on adding different dependencies.

@SpringBootApplication

* This annotation will tell this is our main class and this class will be entry point for our spring application.
* It is a combination of three annotations @Configuration + @ComponentScan + @EnableAutoConfiguration.

@Configuration

* This annotations marks the class as the configuration class where we define multiple bean definition using @bean annotation for the application context.

@ComponentScan

* This annotation will tell spring to scan the default package and its sub-packages for components like @Component, @Service, @Controller, @Repository (for object creation or dependency injection)

Note- All the package created in SpringBoot application should be the sub-package of the default package because @ComponentScan annotation will tell IOC to scan only the default package.

@EnableAutoConfiguration

* This annotation will automatically configure the spring application(project) based on its dependencies.

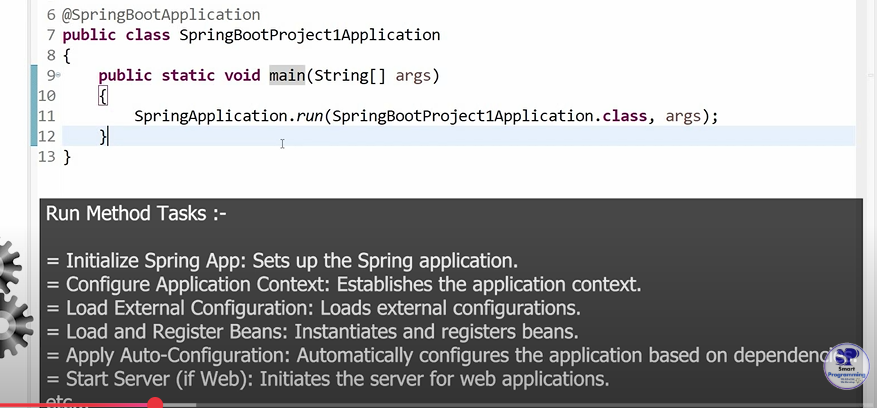
For example, if you include the spring-boot-starter-web dependency, Spring Boot will know programmer is developing a web-application so it will automatically configure(provide) all the things required for developing web-server. For ex - embedded web server (like Tomcat), dispatchers, view resolver etc which are essential for developing a web application.

Otherwise in spring application programmer has to manually configure the tomcat server, or dispatcher servlet class or view resolver class.

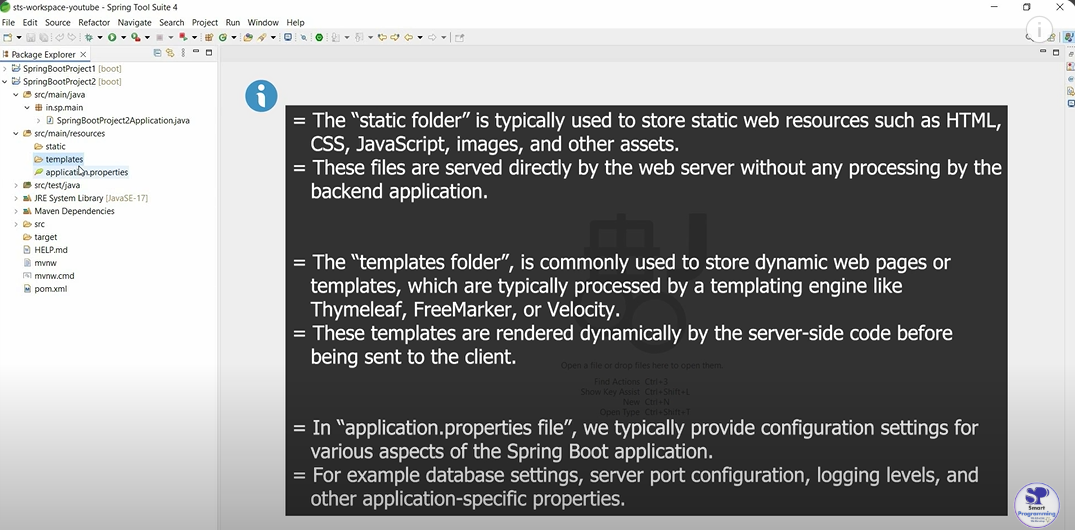
(just for knowledge)

When you add dependencies, jar files are downloaded you can go and check in maven dependencies there will be one jar file spring-boot-auto-configuration which will provide different autoconfiguration to our Spring Application at different stages of program exexution.For ex -there will be one package with name DispatcherServlet auto-configuration which will have DispatcherServlet class will be annoted with certain conditional annotation, if those conditions will be satisfied then this dispatcherServlet class will get excuted. So at different stages of our program execution whenever our SpringBoot application needs some auto-configuration, the respective auto-configuration class’ s, conditional annotation will be checked, if it is satisfying that class will get executed.

**run()**



**Folder structure**



**CommandLineRunner Interface**

* It is a functional interface because it has only one abstract method called run().
* run() method will be executed after main method.
* CommandLine Interface should be used when you need to perform some task at start-up after the SpringBoot application has fully initialized.



But this makes our main class heavy as other methods are also present with the main method.

So to avoid this problem we can make another class which will be marked with @Configuration, so we will have two configuration class and make all other methods in another configuration class.

**Connecting SpringBoot Application with database using JDBC**

By using SpringBoot JDBC, we can connect our SpringBoot Application with database very easily.

How?

Because Springboot automatically provides many configuration task implicitly, helps programmer to focus on building application logic.

Steps for connecting SpringBoot Application with database

1. Add jdbc and mySql starter dependencies
2. Provide database properties in application.properties file

spring.datasource.url = jdbc:mysql://localhost:3306/mydb

spring.datasource.driver-class-name = com.mysql.jdbc.Driver

spring.datasource.username=root

spring.datasource.password=admin

Datasource is a interface in Spring application we manually configure this but in SpringBoot application, SpringBoot will automatically configure datasource, based on the properties provided in application.properties files.

1. Use “JdbcTemplate” class for CRUD operation on D.B.

Note- This JdbcTemplate class is given by Spring.

JpaRepository

* JpaRepository is a interface provided by Spring Data JPA, it has various methods for performing CRUD operation, and also helps us in creating custom queries and various other functionalities.
* The implementation to JpaRepository interface is provided by Spring-Data-JPA at run-time.