**Spring boot**

1. Spring Boot is a java based framework.
2. Spring boot is use for a faster development and also it reduce the development efforts.
3. Using a Spring Boot you can create a production ready application.
4. Spring Boot provide embedded servers like Tomcat server, Data Base server such as H2 Database.
5. Spring boot is a flexible framework where you can customize as per your requirement.
6. Spring Boot is based on Spring Framework.
7. Spring Boot is majorly used for Full stack application and microservices.
8. Spring framework is divided into multiple module. You can use a module as per you project requirement.
   1. Spring IOC/Core
   2. Spring JDBC
   3. Spring ORM
   4. Spring REST
   5. Spring batch
   6. Spring MVC
   7. Spring Cloud
   8. Spring JMS
   9. Spring Security
9. How to implement Module in spring boot project
   1. Add the dependencies of Spring Module into project.
   2. Configure the model into project using predefine configuration.
   3. Use a Spring APIs to implement the functionality.
10. Spring Documentation

<https://docs.spring.io/spring-framework/docs/current/reference/html/>

<https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/>

1. Spring boot is a wrapper of spring framework. Spring framework issue such as setup, configuration and dependency management etc. has been resolve in this spring boot.
   1. Spring Boot provides a starter project which contains the required dependency for the modules.
   2. Spring Boot provides the auto configuration feature using which the module configuration will be happened automatically. You can customize the configuration as per your requirement.
   3. Spring provides the embedded tomcat server and H2 server by default.

**Spring Boot Project Creation**

1. Spring CLI (Command Line Interface)
   1. In this approach you have to install the Spring CLI tool to create spring boot application.

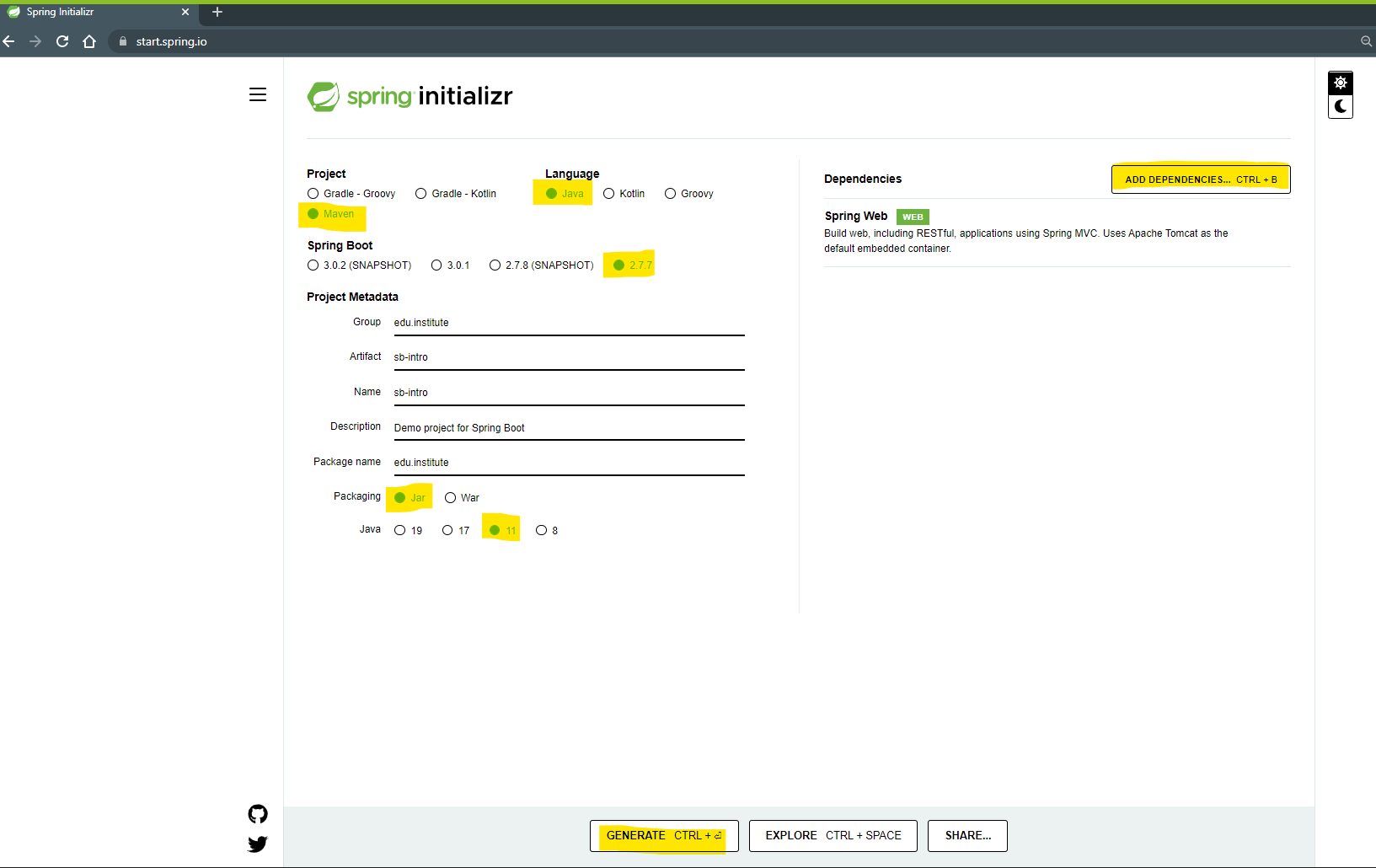
<https://docs.spring.io/spring-boot/docs/current/reference/html/cli.html>

1. Spring STS (Spring Tool Suit)
   1. This is the IDE provided by Spring community. This is an extension for eclipse, IntelliJ etc. IDE.

<https://spring.io/tools>

1. Spring Initializer
   1. Is a web application which is use to create a spring boot project. Using this you can create a spring project which will be downloaded as a zip file.

<https://start.spring.io/>



**Extract and Open project into Eclipse**

1. Copy and paste the downloaded project zip into workspace.
2. Extract a zip File.
3. Open Project into eclipse
   1. Open an Eclipse workspace.
   2. Go To “File” Menu - > Select “Import…” option
   3. You can search for Maven option into the wizard of the new window



* 1. Select “Existing Maven Project” option in the list and click on “Next”
  2. Select the extracted folder as a Root Directory (Make sure that select the folder which has pom.xml)
  3. Click on “Finish”

**Spring boot Application Execution**

1. All the Spring Boot application starts from the main method
2. There are two main statements in the main class
   1. @SpringBootApplication annotation
      1. This annotation is a combination of @Configuration, @EnableAutoConfiguration and @ComponentScan annotation.
      2. **@Configuration** annotation is use to load the as a configuration class, this classes loaded inside spring at the start of the application.
      3. **@EnableAutoConfiguration** annotation is use to enable the spring boot default configurations. Such as Tomcat configuration or module configuration which is added inside application.
      4. **@ComponentScan** annotation is use to load an object of spring bean classes which will be store inside a Spring Container.
   2. Calling a run method to start spring boot application 
      1. In this step the run method will be invoke and the spring boot application will be started.
      2. This method return the Object of Spring Container which can be further use to get the Object of spring bean.

**Spring Container**

1. Container are use to create and hold the objects.
2. Container will maintain the spring objects and you can get the objects whenever required.
3. In the Spring Boot ApplicationContext container is used.

**Spring IOC/Core**

1. Spring IOC is the inverse of control.
2. In this module you can learn to create a spring bean classes.
3. Spring container which is use to create and maintain the object of spring bean class.
4. In this module you perform the dependency Inject (DI) process which is equivalent to HAS-A relation in java.
5. There are multiple annotation you can use in this module

**Spring bean Classes**

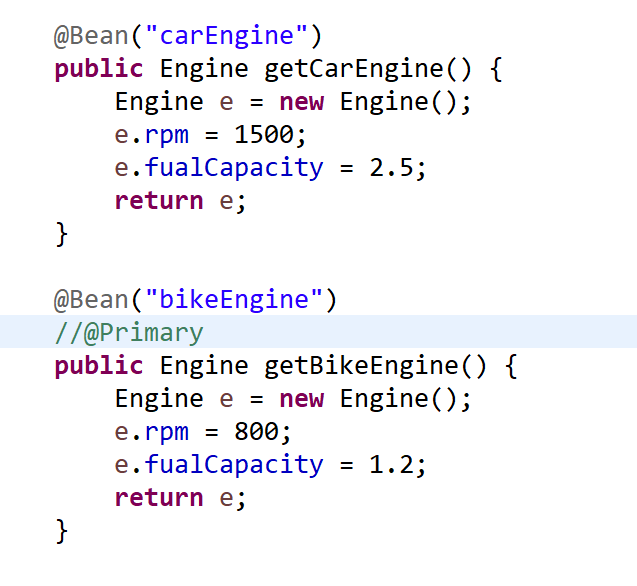
1. It is a java class (build-in or custom) whose object will be created and maintain by spring.
2. To mark any custom class as a spring bean class you have to user the following annotation on the class level
   1. @Component
   2. @Controller
   3. @RestController
   4. @Service
   5. @Repository
   6. @ControllerAdvice

**Spring Dependency Injection and Autowiring**

1. Dependency Injection is a process in which you can create an object of one class into another class In Java it is known as HAS-A relation.
2. To Automate the Dependency Injection process you can use an autowiring.
3. To do autowire process you have to use **@Autowire** annotation on the reference variable.
4. Also Make sure that the autuowire class is a spring bean class.

**Creating Object manually**

1. Using java approach you can create an object class manually but it can be handover to the spring for further managing.
2. Using @Bean you can achieve this.
3. While creating Object you can also specify the Object name so that while injecting it you can inject the appropriate object.



1. If you have multiple object created for same class then you can mark any one of them as primary using @Primary annotation.
2. By making any object of primary you can ask spring to by default give priority to inject the primary object.
3. To specify the object needs to be inject while DI process you can use @Qualifier(“beanname”) along with @Autowire

**REST APIs Based Application**

