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



**Create Spring Boot Project**

1. Go to Web Application to create spring boot project.

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



1. Get the Project Zip file and extract it into specific location.
2. Import the 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 Core/IOC**

1. Spring Bean classes.
2. SI, CI and DI
3. Spring Annotations
4. Autowire
5. Spring Container

@SpringBootApplication

1. It is a combination of 3 annotations internally
2. @Configuration
   1. To Declare class as a configuration class.
   2. These classes will be loaded inside spring container at the initial stages.
3. @EnableAutoConfiguration
   1. This use to enable the auto configuration of the spring boot application.
   2. This configuration will be perform by spring boot internally by looking into the dependencies added inside the project.
4. @ComponentScan
   1. To scan the spring bean classes from the project.
   2. This will scan the project (given package) and create and set object of spring bean classes inside container.

**Spring Bean Classes**

1. Spring bean classes are the java classes for which spring will create a object and also manages the java objects.
2. These classes can be a build-in class or custom class.
3. To create and maintain the java objects spring will make a use of Spring Container which is application context.
4. To declare any java class as a spring bean class you can use the following annotation
   1. @Component
   2. @RestController
   3. @Controller
   4. @Service
   5. @Repository
   6. @ControllerAdvice etc.

**Dependency Injection**

1. One class object will be created inside another class is the Dependency Injection.
2. This is also known as HAS-A relation in java.
3. This Dependency Injection can be automated by autowire process.
4. @Autowire: it is a process in which spring will identify the dependency and inject those object into the java class internally (To make dependency injection process internally/automatically).

**Spring Container**

1. Spring container will scan the project for the spring bean class.
2. It will create and hold the spring bean object.
3. It also manages the life cycle of the object.
4. It will also provides the object whenever required.

**Creating object manually for spring container**

1. The Objects of the java classes can be created manually and then those object can be assign to spring to for management.
2. To do this you have to create a method which will return the object of these type of classes and annotate that method with @Bean Annotation.
3. **Example:**

@Bean

**public** DbSetup getDbSetupObject() {

DbSetup setup = **new** DbSetup();

setup.setDBConnection();

**return** setup;

}

**JSON**

1. JSON stands for **J**ava**S**cript **O**bject **N**otation
2. JSON is use as a common language to communicate between the different applications based on different language or platform.
3. JSON used in a key and value pair format. One key and value pair also known as element.
4. In JSON key are always in String format and vales can be in String, Numeric, Boolean, JSON object or JSON array format.
5. JSON can be represented in 2 form
   1. JSON Object
   2. JSON Array

**JSON Object:**

1. JSON object can have a key and value pairs.
2. JSON Object will be represented by curly brackets.
3. Symbol:

{

“Key”: value

}

**JSON Array:**

1. JSON array can have a group of values or group of JSON Objects
2. JSON array will be represented by square bracket
3. Symbol:

[ value1, value2, … ]

[

{

Key:value

},

{

Key:value

}

]

Examples of JSON

**JSON Object**

{

“name”: “Abc”,

“nickName”:”Abc”,

“email”: “[abc@gmail.com](mailto:abc@gmail.com)”,

“contact”:998877667788,

“salary”: 45645.45,

“status”: “Active”,

“isCurrentEmp”: true

}

**JSON Array**

[76.67, 45.67, 76.5, 88,44]

[“Abc”, “Xyz”, “Pqr”]

{

“name”: “Xyz”,

“contact”: 8987799797,

“skills”: [“Java” ,”Html” ,”CSS” ,”JS”],

“address”: {

“city”:”pune”,

“pincode”:”998877”,

“state”: “MH”

},

“projects”: [

{

“name”:”Lib Management”,

“technologies”: [“Java”, “Html”, “CSS”],

“desc”: “This application is used to automate”

},

{

“name”:”Student Management”,

“technologies”: [“Aps.net”, “Html”, “CSS”],

“desc”: “This application is used to automate”

}

]

}