**1.** **What is @SpringBootApplication annotation in spring boot project?** @SpringBootApplication annotation was introduced in Spring Boot 1.2.0 version. This annotation is equivalent to declaring these 3 annotations.

**1) @Configuration**

**2) @EnableAutoConfiguration**

**3) @ComponentScan**

The following are the parameters accepted in the @SpringBootApplication annotation:

* **exclude**: Exclude the list of classes from the auto configuration.
* **excludeNames**: Exclude the list of fully qualified class names from the auto configuration.
* **scanBasePackageClasses**: Provide the list of classes that has to be applied for the @ComponentScan.
* **scanBasePackages** Provide the list of packages that has to be applied for the @ComponentScan.

**2.** **What Embedded Containers Does Spring Boot Support?** Spring boot is a Java based framework that supports application services. It runs as a standalone jar with an embedded servlet container or as a WAR file inside a container. The spring boot framework supports three different types of embedded servlet containers: Tomcat (default), Jetty and Undertow.

**3.** **What is Spring Boot Actuator?** Spring Boot Actuator is a sub-project of Spring Boot. It adds several production grade services to your application with little effort on your part. Actuators enable production-ready features to a Spring Boot application, without having to actually implement these things yourself. The Spring Boot Actuator is mainly used to get the internals of running application like health, metrics, info, dump, environment, etc

* **/health**: It shows application health information.
* **/info**: It displays arbitrary application info.
* **/metrics**: It gives all metrics related information for the current application.
* **/trace**: It displays trace information for last few HTTP requests.

**4.** **What is Spring Boot Initializer?** The Spring Initializer is ultimately a web application that can generate a Spring Boot project structure for you. It doesn’t generate any application code, but it will give you a basic project structure and either a Maven or a Gradle build specification to build your code with. All you need to do is write the application code. Spring Initializer can be used several ways, including:

* A web-based interface.
* Via Spring Tool Suite.
* Using the Spring Boot CLI.

**5.** **How to reload Spring Boot Application without restarting server?** Include below dependency in your pom.xml file. By default, any entry on the classpath that points to a folder will be monitored for changes. With this dependency any changes you save, the embedded tomcat will restart. This helps developers to improve the productivity.

|  |
| --- |
| <dependency>      <groupId>org.springframework.boot</groupId>      <artifactId>spring-boot-devtools</artifactId>      <optional>true</optional>  </dependency> |

**6.** **What are the key components of Spring Boot framework?**

* **Spring Boot Starters:** The main responsibility of Spring Boot Starter is to combine a group of common or related dependencies into single dependencies. Spring Boot starters can help to reduce the number of manually added dependencies just by adding one dependency. Examples are spring-boot-starter-web, spring-boot-starter-test, spring-boot-starter-data-jpa, etc.
* **Spring Boot AutoConfigurator:**  Spring Boot AutoConfigurator will simplify all these XML based configurations. It also reduces the number of annotations.
* **Spring Boot CLI:** Spring Boot CLI (Command Line Interface) is a Spring Boot software to run and test Spring Boot applications from command prompt. When we run Spring Boot applications using CLI, then it internally uses Spring Boot Starter and Spring Boot AutoConfigurate components to resolve all dependencies and execute the application.
* **Spring Boot Actuator:** Spring Boot Actuator is a sub-project of Spring Boot. It adds several production grade services to your application with little effort on your part. Actuators enable production-ready features to a Spring Boot application, without having to actually implement these things yourself. The Spring Boot Actuator is mainly used to get the internals of running application like health, metrics, info, dump, environment, etc.

**7. What is Spring Boot?** Spring Boot is a Spring module which provides RAD (Rapid Application Development) feature to Spring framework. It is used to create standalone spring-based application that you can just run because it needs very little spring configuration. Spring Boot does not generate code and there is absolutely no requirement for XML configuration. It uses convention over configuration software design paradigm that means it decrease the effort of developer.

**8. Advantages of Spring Boot**

* Create stand-alone Spring applications that can be started using java -jar.
* Embed Tomcat, Jetty or Undertow directly. You don't need to deploy WAR files.
* It provides opinionated 'starter' POMs to simplify your Maven configuration.
* It automatically configures Spring whenever possible.
* It provides production-ready features such as metrics, health checks and externalized configuration.
* Absolutely no code generation and no requirement for XML configuration.

**9. Spring Boot Features**

* **Web Development: -** It is well suited Spring module for web application development. We can easily create a self-contained HTTP server using embedded Tomcat, Jetty or Undertow. We can use the spring-boot- starter-web module to start and running application quickly.
* **Spring Application :-** It is a class which provides the convenient way to bootstrap a spring application which can be started from main method. You can call start your application just by calling a static run() method.

**public static void main(String[] args){**

**SpringApplication.run(className.class, args);**

**}**

* **Application Events and Listeners :-** Spring Boot uses events to handle variety of tasks. It allows us to create factories file that are used to add listeners. we can refer it by using ApplicationListener key. Always create factories file in META-INF folder like: **META-INF/spring.factories**
* **Admin Support :-** Spring Boot provides the facility to enable admin related features for the application. It is used to access and manage application remotely. We can enable it by simply using spring.application.admin.enabled property.
* **Externalized Configuration :-** Spring Boot allows us to externalize our configuration so that we can work with the same application in different environments. Application use YAML files to externalize configuration.
* **Properties Files :-** Spring Boot provides rich set of Application Properties. So, we can use that in properties file of our project. Properties file is used to set properties like: **server-port = 8082** and many others. It helps to organize application properties.
* **YAML Support :-** It provides convenient way for specifying hierarchical configuration. It is a superset of JSON. The SpringApplication class automatically support YAML. It is successful alternative of properties.
* **Type-safe Configuration :-** Strong type-safe configuration is provided to govern and validate the configuration of application. Application configuration is always a crucial task which should be type-safe. We can also use annotation provided by this library.
* **Logging :-** Spring Boot uses Common logging for all internal logging. Logging dependencies are managed by default. We should not change logging dependencies, if there is no required customization is needed.
* **Security :-** Spring Boot applications are spring bases web applications. So, it is secure by default with basic authentication on all HTTP endpoints. A rich set of Endpoints are available for develop a secure Spring Boot application.

**10. Spring Boot Project :-** There are multiple approaches to create Spring Boot project. We can use any of the following approach to create application.

* Spring Maven Project
* Spring Starter Project Wizard
* Spring Initializr
* Spring Boot CLI

**11. Spring Boot Maven Project :-** Creating Spring Boot project by creating maven project. It includes the following steps.

* Select project type.
* Configure project by providing project name.
* After clicking finish, Spring boot project has been created. Our new project looks like the following screen shot.
* we are adding parent to our Maven project. It is used to declare that our project is a child to this parent project.

|  |
| --- |
| **<parent>**  **<groupId>**org.springframework.boot**</groupId>**  **<artifactId>**spring-boot-starter-parent**</artifactId>**  **<version>**1.4.2.RELEASE**</version>**  **</parent>** |

* we are adding web dependency by adding spring-boot-starter-web.

|  |
| --- |
| **<dependencies>**  **<dependency>**  **<groupId>**org.springframework.boot**</groupId>**  **<artifactId>**spring-boot-starter-web**</artifactId>**  **</dependency>**  **</dependencies>** |

* we are calling run method and passing class name as argument.

|  |
| --- |
| **import** org.springframework.boot.SpringApplication;  **import** org.springframework.boot.autoconfigure.SpringBootApplication;  @SpringBootApplication  **public** **class** SpringBootExample {  **public** **static** **void** main(String[] args) {  SpringApplication.run(SpringBootExample.**class**, args);  }  } |

**12. What are the Spring Boot Annotations?** The @RestController is a stereotype annotation. It adds @Controller and @ResponseBody annotations to the class. We need to import org.springframework.web.bind.annotation package in our file, in order to implement it.

The **@RestController** annotation informs to the Spring to render the result back to the caller.

The **@RequestMapping** annotation is used to provide routing information. It tells the Spring that any HTTP request should map to the corresponding method. We need to import org.springframework.web.annotation package in our file.

|  |
| --- |
| **import** org.springframework.web.bind.annotation.RequestMapping;  **import** org.springframework.web.bind.annotation.RestController;  @RestController  **public** **class** HomeController {      @RequestMapping("/hello")  **public** String hello(){  **return**"Hello!";      }  } |

**13. What is Spring Boot dependency management?** Spring Boot manages dependencies and configuration automatically. You don't need to specify version for any of that dependencies. Spring Boot upgrades all dependencies automatically when you upgrade Spring Boot.

**14. What are the Spring Boot properties?** Spring Boot provides various properties which can be specified inside our project's **application.properties** file. These properties have default values and you can set that inside the properties file. Properties are used to set values like: server-port number, database connection configuration etc.

**15. What is thymeleaf?** It is a server side Java template engine for web application. It's main goal is to bring elegant natural templates to your web application. It can be integrate with Spring Framework and ideal for HTML5 Java web applications.

**16. How to use thymeleaf?** In order to use Thymeleaf we must add it into our pom.xml file like:

**<dependency>**

**<groupId>**org.springframework.boot**</groupId>**

**<artifactId>**spring-boot-starter-thymeleaf**</artifactId>**

**</dependency>**

**17. How to connect Spring Boot to the database using JPA?** Spring Boot provides **spring-boot-starter-data-jpa** starter to connect Spring application with relational database efficiently. You can use it into project POM file.

**18. How to connect Spring Boot application to database using JDBC?** Spring Boot provides starter and libraries for connecting to our application with JDBC. Here, we are creating an application which connects with Mysql database.

**19. What is @RestController annotation in Spring Boot?** The **@RestController** is a stereotype annotation. It adds @Controller and @ResponseBody annotations to the class. We need to import org.springframework.web.bind.annotation package in our file, in order to implement it.

**20 . What is @RequestMapping annotation in Spring Boot?** The **@RequestMapping** annotation is used to provide routing information. It tells the Spring that any HTTP request should map to the corresponding method. We need to import org.springframework.web.annotation package in our file.

**21. How to create Spring Boot application using Spring Starter Project Wizard?** There is one more way to create Spring Boot project in STS (Spring Tool Suite). Creating project by using IDE is always a convenient way.

**22. Spring Vs Spring Boot?** Spring is a web application framework based on Java. It provides tools and libraries to create a complete customized web application. Whereas Spring Boot is a spring module which is used to create spring application project that can just run.