**Spring Boot Interview Questions**

**Q1. How to disable a specific auto-configuration?**

If we want to disable a specific auto-configuration, we can indicate it using the *exclude* attribute of the *@EnableAutoConfiguration* annotation. For instance, this code snippet neutralizes *DataSourceAutoConfiguration*:

|  |  |
| --- | --- |
| 1  2  3 | .// other annotations  . @EnableAutoConfiguration(exclude = DataSourceAutoConfiguration.class)  . public class MyConfiguration { } |

**Q2. How to register a custom auto-configuration?**

To register an auto-configuration class, we must have its fully-qualified name listed under the *EnableAutoConfiguration* key in the *META-INF/spring.factories* file:

|  |  |
| --- | --- |
| 1 | org.springframework.boot.autoconfigure.EnableAutoConfiguration=com.baeldung.autoconfigure.CustomAutoConfiguration |

### ****Q7. How to deploy Spring Boot web applications as JAR and WAR files?****

Notice that the *packaging* element in the *pom.xml* file must be set to *jar* to build a JAR file:

|  |  |
| --- | --- |
| 1 | **<packaging>jar</packaging>** |

If we don’t include this element, it also defaults to *jar*.

In case we want to build a WAR file, change the *packaging* element to *war*:

|  |  |
| --- | --- |
| 1 | **<packaging>war</packaging>** |

And leave the container dependency off the packaged file:

|  |  |
| --- | --- |
| 1  2  3  4  5 | **<dependency>**  **<groupId>org.springframework.boot</groupId>**  **<artifactId>spring-boot-starter-tomcat</artifactId>**  **<scope>provided</scope>**  **</dependency>** |

After executing the Maven *package* phase, we’ll have a deployable WAR file.

## 4. What are disadvantages of Spring boot?

If you want to convert your old spring application to Spring boot application, it may not be straight forward and can be time consuming.

## 5. How can you override default properties in Spring boot Project?

Spring boot provides a lot of properties which can be overridden by specifying them in application.properties.

**For example:** You want to specify prefix and suffix in Spring MVC applications. You can simply do it by putting below properties in application.properties.

spring.mvc.view.prefix: /WEB-INF/  
spring.mvc.view.suffix: .jsp

## 7. Can we use Spring boot with applications which are not using Spring?

No, it is not possible as of now. Spring boot is limited to Spring applications only.

**What is auto-configuration in Spring boot?**

it automatically configures a lot of things based upon what is present in the classpath.

the point is auto-configuration does a lot of work for you with respect to configuring beans, controllers, view resolvers etc, hence it helps a lot in creating a Java application.

**4. What is starter dependency in Spring Boot? how does it help?**

After examining several Spring-based projects Spring guys notice that there is always some set of libraries which are used together e.g. [Spring MVC](http://javarevisited.blogspot.sg/2018/01/7-reasons-for-using-spring-to-develop-RESTful-web-service.html#axzz55a8rTeu7) with [Jackson](http://javarevisited.blogspot.sg/2018/01/how-to-ignore-unknown-properties-parsing-json-java-jackson.html) for creating RESTful web services. Since declaring a dependency in Maven's pom.xml is the pain, they combined many libraries into one based upon functionality and created this starter package.  
  
This not only frees you from declaring many dependencies but also fees you from compatibility and version mismatch issue. Spring Boot starter automatically pulls compatible version of other libraries so that you can use them without worrying about any compatibility issue.

=> @SpringBootTest is used to run unit test on Spring Boot environment.

**15. Can you control logging with Spring Boot? How?**  
Yes, we can control logging with Spring Boot by specifying log levels on application.properties file. Spring Boot loads this file when it exists in the [classpath](http://www.java67.com/2012/08/what-is-path-and-classpath-in-java-difference.html) and it can be used to configure both Spring Boot and application code.  
  
Spring Boot uses Commons Logging for all internal logging and you can change log levels by adding following lines in the application.properties file:  
  
logging.level.org.springframework=DEBUG  
logging.level.com.demo=INFO

### Spring Boot Disadvantages

### It is very tough and time consuming process to convert existing or legacy Spring Framework projects into Spring Boot Applications. It is applicable only for brand new/Greenfield Spring Projects.

1. Spring boot may unnecessarily increase the deployment binary size with unused dependencies.
2. If you are a control freak, I doubt Spring Boot would fit your needs.
3. Spring Boot sticks good with micro services. The Spring Boot artifacts can be deployed directly into Docker containers. In a large and monolithic based applications, I would not encourage you to use Spring Boot.