# [Spring Boot Micro service Interview Questions & Answers](https://www.topjavatutorial.com/microservice/spring-boot-microservice-interview-questions-answers/)

https://www.topjavatutorial.com/microservice/spring-boot-microservice-interview-questions-answers/

## **Spring Boot Micro service Interview Questions**

### **Q 1: What is Spring Boot? Why is Spring Boot popular for Java microservice development?**

Spring Boot provides a lot of boiler plate code and configurations that we were manually doing for application development.  
It abstracts the maven dependencies, configurations, application server and reduces development time.  
Spring Boot makes it easier to create production ready applications in no time.

Since it follows convention over configuration, it comes with default configurations for most of the spring projects including microservice development.  
So, we don’t need to do much to bootstrap the spring boot microservice starter project.

### **Q 2: What are the advantages of Spring Boot?**

Spring Boot is popular for microservice development because of following advantages:

– Reduces boiler plate code  
– Embed Tomcat, Jetty or Undertow directly (no need to deploy WAR files)  
– Provide production-ready features such as metrics, health checks and externalized configuration  
– Provide opinionated ‘starter’ dependencies to simplify your build configuration  
– Automatically configure Spring and 3rd party libraries whenever possible  
– Provide production-ready features such as metrics, health checks and externalized configuration  
– Absolutely no code generation and no requirement for XML configuration

### **Q 3: What is YAML?**

YAML is a convenient syntax for storing external properties in a hierarchical format.

We can also use application.yaml or application.yml file as an alternative to application.properties.

For example, let’s consider the following property in application.properties:

spring.application.name=topjavatutorial

The same can be written in application.yaml as:

spring:

  application:

    name: topjavatutorial

### **Q 4: How to configure the port for spring boot application?**

In order to run a spring boot application on a custom port, you can do the following:

– Set the port as a command line parameter

-Dserver.port=8090 or --server.port=8090

– Set the sever.port in application.properties.

server.port=8090

– Set the port in application.yml

server:

    port: 9999

**Note**: If we set the port to 0, random ports will be assigned.

### **Q 5: What is Swagger? How to integrate it with Spring Boot?**

Swagger is set of open source tools that helps with creating documentation for your REST services.

Refer this article for integrating Swagger with Spring Boot:

[How to integrate Swagger with Spring Boot](http://www.topjavatutorial.com/frameworks/spring/spring-core/integrate-swagger-2-spring-boot/)

### **Q 6: How to add a context path to Spring Boot application?**

By default, the context path is “/”.

We can update the **contextPath**in application.properties file.

For example, if we are trying to add contextPath as “/mycontext”, we can add it as:

server.contextPath=/mycontext

With Spring Boot 2.0, the same can be added as:

server.servlet.contextPath=/mycontext

### **Q 7: How to set the logging level with application.properties?**

We can add the log levels TRACE, DEBUG, INFO, WARN, ERROR, FATAL, OFF in application.properties.

The syntax is:

logging.level.<logger-name>=<level>

The root logger can be configured by using **logging.level.root**.

For example,

logging.level.root=WARN

logging.level.org.springframework.web: DEBUG

logging.level.org.hibernate: ERROR

### **Q 8: What are Spring boot Starter POMs?**

Starter POMs are convenient descriptors that you can add in your pom.xml. They contain lot of dependencies that help you setup and run your project quickly.

They follow naming convention **spring-boot-starter-\***, where \* is a particular type of application.

Here are some examples:

spring-boot-starter is the core Spring Boot starter, including auto-configuration support, logging and YAML.

Similarly, spring-boot-starter-web has Tomcat and spring-webmvc and you can add this to create Spring MVC application.

### **Q 9: What is actuator in spring boot?**

Actuators allow you to monitor and interact with your application.

Spring Boot includes a number of built-in endpoints and you can also add your own.

For example, the “health” endpoint provides basic application health information. Similarly, we can use “metrics” too get metrics information for the current application.

To enable actuators, add a dependency to the spring-boot-starter-actuator Starter POM.

<dependency>

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

        <artifactId>spring-boot-starter-actuator</artifactId>

</dependency>

### **Q 10: How to access a value defined in the application.properties file in Spring Boot?**

We can access properties defined in application.properties file inside the our bean using the **@Value** tag.

For example, the following will read a property “name” from the application.properties:

import org.springframework.stereotype.\*;

import org.springframework.beans.factory.annotation.\*;

@Component

public class MyBean {

    @Value("${name}")

    private String name;

    // ...

### **Q 11: How to log Sql queries and values in Spring Boot Hibernate application?**

To log the queries, add the following properties in application.properties:

To log the values, add the following as well:

spring.jpa.properties.hibernate.show\_sql=true

spring.jpa.properties.hibernate.use\_sql\_comments=true

spring.jpa.properties.hibernate.format\_sql=true

spring.jpa.properties.hibernate.type=trace

# MICROSERVICES INTERVIEW QUESTIONS & ANSWERS

1. **Question 1. What Are Microservices?**

**Answer :**

Microservices is a variant of the service-oriented architecture (SOA) architectural style that structures an application as a collection of loosely coupled services. In a Microservices architecture, services should be fine-grained and the protocols should be lightweight. The benefit of decomposing an application into different smaller services is that it improves modularity and makes the application easier to understand, develop and test. It also parallelism development by enabling small autonomous teams to develop, deploy and scale their respective services independently. It also allows the architecture of an individual service to emerge through continuous refactoring. Microservices-based architectures enable continuous delivery and deployment.

1. **Question 2. What Is Spring Cloud?**

**Answer:**

Spring Cloud Stream App Starters are Spring Boot based Spring Integration applications that provide integration with external systems. Spring Cloud Task. A short-lived Microservices framework to quickly build applications that perform finite amounts of data processing.

1. **Question 3. What Are The Advantages Of Using Spring Cloud?**

**Answer:**

When developing distributed Microservices with Spring Boot we face the following issues-

* + **Complexity associated with distributed systems-**  
    This overhead includes network issues, Latency overhead, Bandwidth issues, security issues.
  + **Service Discovery-**  
    Service discovery tools manage how processes and services in a cluster can find and talk to one another. It involves a directory of services, registering services in that directory, and then being able to lookup and connect to services in that directory.
  + **Redundancy-**  
    Redundancy issues in distributed systems.
  + **Loadbalancing-**  
    Load balancing improves the distribution of workloads across multiple computing resources, such as computers, a computer cluster, network links, central processing units, or disk drives.
  + **Performance issues-**  
    Performance issues due to various operational overheads.
  + **Deployment complexities-**  
    Requirement of Devops skills.

1. **Question 4. What Is A Microservices Architecture?**

**Answer:**

Microservices architecture allows to avoid monolith application for large system. It provide loose coupling between collaborating processes which running independently in different environments with tight cohesion.

1. **Question 5. What Are The Advantages And Disadvantages Of Microservices?**

**Answer:**

**Microservices Advantages**

* + Smaller code base is easy to maintain.
  + Easy to scale as individual component.
  + Technology diversity i.e. we can mix libraries, databases, frameworks etc.
  + Fault isolation i.e. a process failure should not bring whole system down.
  + Better support for smaller and parallel team.
  + Independent deployment
  + Deployment time reduce

**Microservices Disadvantages**

* + Difficult to achieve strong consistency across services
  + ACID transactions do not span multiple processes.
  + Distributed System so hard to debug and trace the issues
  + Greater need for end to end testing
  + Required cultural changes in across teams like Dev and Ops working together even in same team.

1. **Question 6. What Netflix Projects Did We Use?**

**Answer:**

Eureka created by Netflix, it is the Netflix Service Discovery Server and Client. Netflix Ribbon, it provide several algorithm for Client-Side Load Balancing. Spring provide smart RestTemplate for service discovery and load balancing by using @LoadBalanced annotation with RestTemplate instance.

1. **Question 7. How Will You Monitor Multiple Microservices For Various Indicators Like Health?**

**Answer:**

Spring Boot provides actuator endpoints to monitor metrics of individual Microservices. These endpoints are very helpful for getting information about applications like if they are up, if their components like database etc are working good. But a major drawback or difficulty about using actuator endpoints is that we have to individually hit the endpoints for applications to know their status or health. Imagine Microservices involving 50 applications, the admin will have to hit the actuator endpoints of all 50 applications. To help us deal with this situation, we will be using open source project located at Built on top of Spring Boot Actuator, it provides a web UI to enable us visualize the metrics of multiple applications.

1. **Question 8. What Does One Mean By Service Registration And Discovery? How Is It Implemented In Spring Cloud?**

**Answer:**

When we start a project, we usually have all the configurations in the properties file. As more and more services are developed and deployed, adding and modifying these properties become more complex. Some services might go down, while some the location might change. This manual changing of properties may create issues. Eureka Service Registration and Discovery helps in such scenarios. As all services are registered to the Eureka server and lookup done by calling the Eureka Server, any change in service locations need not be handled and is taken care of.

1. **Question 9. How Do You Setup Service Discovery?**

**Answer:**

Spring Cloud support several ways to implement service discovery but for this I am going to use Eureka created by Netflix. Spring Cloud provide several annotation to make it use easy and hiding lots of complexity.

1. **Question 10. How Do You Access A Restful Microservice?**

**Answer:**

* + Load Balanced RestTemplate.
  + If there are multiple RestTemplate you get the right one.
  + It can used to access multiple Microservices.

1. **Question 11. How to Achieve Server Side Load Balancing Using Spring Cloud?**

**Answer:**

Server side load balancing can be achieved using Netflix Zuul. Zuul is a JVM based router and server side load balancing by Netflix. It provides a single entry to our system, which allows a browser, mobile app, or other user interface to consume services from multiple hosts without managing cross-origin resource sharing (CORS) and authentication for each one. We can integrate Zuul with other Netflix projects like Hystrix for fault tolerance and Eureka for service discovery, or use it to manage routing rules, filters, and load balancing across your system.

1. **Question 12. What Is Eureka?**

**Answer:**

Eureka is the Netflix Service Discovery Server and Client. Eureka Server is using Spring Cloud.