# Spring Boot for Microservices: Operation

In this lesson, we'll be looking at how the Spring framework handles the operation of microservices. Let's begin!

#### WE'LL COVER THE FOLLOWING ^

- Operation
  - Deployment in Spring
  - Configuration in Spring
  - Logs in Spring
  - Metrics in Spring

# Operation #

Spring Boot also has some interesting approaches for operation.

### Deployment in Spring #

 To deploy a Spring Boot application, it is enough to just copy the JAR file to the server and start it. Deploying a Java application can't be further simplified.

#### Configuration in Spring #

• Spring Boot offers numerous options for the configuration. For example, a **Spring Boot** application can read the configuration from a configuration file or from an environment variable. **Spring Cloud** offers support for **Consul** as a server for configurations. The examples in this course use application.properties files for configuration because they are relatively easy to handle.

#### Logs in Spring #

• Spring Boot applications can generate logs in many different ways.

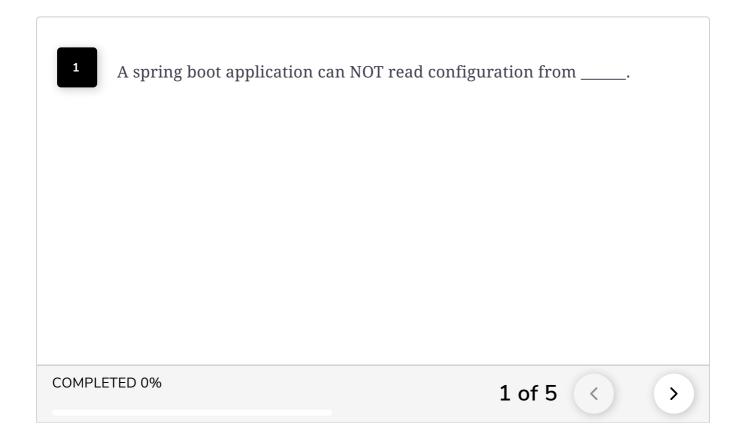
Usually, a Spring Boot application displays the logs in the console. Output

to a file is also possible. A Spring Boot application can also send the logs as **JSON** data to a central server instead of using a simple human-readable text format. JSON facilitates the processing of log data on this server.

### Metrics in Spring #

• For metrics, Spring Boot offers a **special starter**, namely the Actuator. After adding a dependency to spring-boot-starter-actuator, the application collects metrics, for example about the HTTP requests. In addition, **Spring Boot Actuator** provides **REST endpoints** under which the metrics are available as JSON documents.

# QUIZ



In the *next lesson*, we'll discuss Spring Boot with regards to resilience and the creation of new microservices.

Stay tuned!