

Search Tutorials

# Spring Boot Microservices + ELK(Elasticsearch, Logstash, and Kibana) Stack Hello World Example

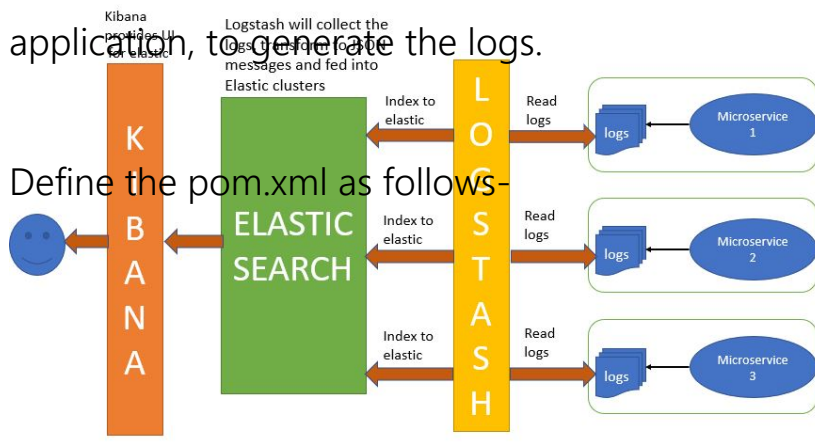
(<https://srv.carbonads.net/ads/click/x/GTND42JIC6YD45QICESLYKQNCKYDsegment=placement:www.javainuse.com/>)

In this tutorial we will be using ELK stack along with Spring Boot Microservice for analyzing the generated logs. In next tutorial we will see how use FileBeat along with the ELK stack. (<https://www.javainuse.com/elasticsearch/filebeat-elk>)

You can make use of the Online Grok Pattern Generator Tool (/grok) for creating, testing and debugging grok patterns required for logstash.

This site uses cookies to deliver our services and to show you relevant ads. By continuing to visit this site you agree to our use of cookies. Learn more (<http://www.javainuse.com/privacy>)

This implementation is simple to create. We will be creating a simple spring boot application, to generate the logs.



## What is ELK? Need for it?

The ELK Stack consists of three open-source products - Elasticsearch, Logstash, and Kibana from Elastic.

- Elasticsearch is a NoSQL database that is based on the Lucene search engine.
- Logstash is a log pipeline tool that accepts inputs from various sources, executes different transformations, and exports the data to various targets. It is a dynamic data collection pipeline with an extensible plugin ecosystem and strong Elasticsearch synergy
- Kibana is a visualization UI layer that works on top of Elasticsearch.

These three projects are used together for log analysis in various environments. So Logstash collects and parses logs, Elastic search indexes and store this information while Kibana provides a UI layer that provide actionable insights.

### Use Cases-

- Consider you have a single application running and it produces logs. Now Create the Spring Boot Bootstrap class as follows- suppose you want analyze the logs generated. One option is to manually analyze them. But suppose these logs are large, then manually analyzing them is not feasible.
- Suppose we have multiple Application running and all these applications produce logs. If we have to analyze the logs manually we will need to go through all the log files. These may run into hundreds.

This site uses cookies to deliver our services and to show you relevant ads. By continuing to visit this site you agree to our use of cookies. Learn more (<http://www.javainuse.com/privacy>)

```
package com.javainuse;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class HelloWorldSpringBootApplication {

    public static void main(String[] args) {
        SpringApplication.run(HelloWorldSpringBootApplication.class, args);
    }
}
```

Next define the controller to expose REST API. We will be making use of these calls to write content to the log file.

```
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-parent</artifactId>
<version>1.5.6.RELEASE</version>
<relativePath /> <!-- lookup parent from repository -->
</parent>

<properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
    <java.version>1.8</java.version>
</properties>

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

<build>
    <plugins>
        <plugin>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-maven-plugin</artifactId>
        </plugin>
    </plugins>
</build>
```

This site uses cookies to deliver our services and to show you relevant ads. By continuing to visit this site you agree to our use of cookies. Learn more (<http://www.javainuse.com/privacy>)

```
package com.javainuse;

import java.io.PrintWriter;
import java.io.StringWriter;
import java.util.Date;

import org.apache.log4j.Level;
import org.apache.log4j.Logger;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Bean;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.client.RestTemplate;

@RestController
class ELKController {
    private static final Logger LOG = Logger.getLogger(ELKCon

    @Autowired
    RestTemplate restTemplate;

    @Bean
    RestTemplate restTemplate() {
        return new RestTemplate();
    }

    @RequestMapping(value = "/elk")
    public String helloWorld() {
        String response = "Welcome to JavaInUse" + new Da
        LOG.log(Level.INFO, response);

        return response;
    }

    @RequestMapping(value = "/exception")
    public String exception() {
        String response = "";
        try {
            throw new Exception("Exception has occurred");
        } catch (Exception e) {
            LOG.error(e.getMessage());
        }
    }
}
```

This site uses cookies to deliver our services and to show you relevant ads. By continuing to visit this site you agree to our use of cookies. Learn more (<http://www.javainuse.com/privacy>)

```
        } catch (Exception e) {
            e.printStackTrace();
            LOG.error(e);

            StringWriter sw = new StringWriter();
            PrintWriter pw = new PrintWriter(sw);
            e.printStackTrace(pw);
            String stackTrace = sw.toString();
            LOG.error("Exception - " + stackTrace);
            response = stackTrace;
        }

        return response;
    }
}
```

Finally specify the name and location of the log file to be created in the application.properties file.

```
logging.file=C:/elk/spring-boot-elk.log
```

Next we will configure the logstash pipeline. When using the ELK stack we are ingesting the data to elasticsearch, the data is initially unstructured. We first need to break the data into structured format and then ingest it to elasticsearch. Such data can then be later used for analysis. This data manipulation of unstructured data to structured is done by Logstash. Logstash itself makes use of grok filter to achieve this.

You can make use of the Online Grok Pattern Generator Tool (/grok) for creating, testing and debugging grok patterns required for logstash.

This is done using the logstash.conf-

```
input {
  file {
    type => "java"
    path => "C:/elk/spring-boot-elk.log"
    codec => multiline {
      pattern => "^%{YEAR}-%{MONTHNUM}-%{MONTHDAY} %{TIME}.*"
      negate => "true"
      what => "previous"
    }
  }
}

filter {
  #If log line contains tab character followed by 'at' then we will
  if [message] =~ "\tat" {
    grok {
      match => ["message", "^(\\tat)"]
      add_tag => ["stacktrace"]
    }
  }
}

output {

  stdout {
    codec => rubydebug
  }

  # Sending properly parsed log events to elasticsearch
  elasticsearch {
    hosts => ["localhost:9200"]
  }
}
```

Start logstash using the command prompt as follows-

```
logstash -f logstash.conf
```

This site uses cookies to deliver our services and to show you relevant ads. By continuing to visit this site you agree to our use of cookies. Learn more (<http://www.javainuse.com/privacy>)

Start the spring boot application by running the HelloWorldSpringBootApplication as a java application.

Logs will be generated in C:/elk folder.

- goto localhost:8080/elk
- goto localhost:8080/exception
- go to kibana UI console- localhost and create an index pattern logstash-\* to see the indexed data-


## Download Source Code

Download it -

Spring Boot Microservice+ ELK stack (/zip/spring/boot/spring-boot-elk.zip)


### See Also

Spring Boot Hello World Application- Create simple controller and jsp view using Spring Boot Tutorial-Spring Data JPA (/spring/SpringBoot\_DataJPA)  
Spring Boot + Simple Security Configuration (/spring/sprboot\_sec)  
Pagination using Spring Boot Simple Example (/spring/SpringBootUsingPagination)  
Spring Boot + ActiveMQ Hello world Example (/spring/sprboot\_activemq)  
Spring Boot + Swagger Example Hello World Example (/spring/boot\_swagger)  
Spring Boot + Swagger- Understanding the various Swagger Annotations (/spring/boo  
Spring Boot Main Menu (/spring/sprboot)  
Spring Boot Interview Questions (/spring/SpringBootInterviewQuest

This site uses cookies to deliver our services and to show you relevant ads. By continuing to visit this site you  agree to our use of cookies. Learn more (<http://www.javainuse.com/privacy>)

## Popular Posts

- Spring Boot Interview Questions  
(/spring/SpringBootInterviewQuestions)
- Implement Spring Boot Security and understand Spring Security Architecture (/webseries/spring-security-jwt/chap3)
- E-commerce Website - Online Book Store using Angular 8 + Spring Boot (/fullstack/ecommerce)
- Spring Boot +JSON Web Token(JWT) Hello World Example (/spring/boot-jwt)
- Angular 7 + Spring Boot Application Hello World Example (/spring/ang7-hello)
- Build a Real Time Chat Application using Spring Boot + WebSocket + RabbitMQ (/spring/boot-websocket-chat)
- Pivotal Cloud Foundry Tutorial - Deploy Spring Boot Application Hello World Example (/pcf/pcf-hello)
- Deploying Spring Based WAR Application to Docker (/devOps/docker/docker-war)
- EIP patterns using Apache Camel (/camel/camel\_EIP)
- Spring Cloud- Netflix Eureka + Ribbon Simple Example (/spring/spring\_ribbon)
- Spring Cloud- Netflix Hystrix Circuit Breaker Simple Example (/spring/spring\_hystrix\_circuitbreaker)


This site uses cookies to deliver our services and to show you relevant ads. By continuing to visit this site you agree to our use of cookies. Learn more (<http://www.javainuse.com/privacy>) 



- Spring Boot + Swagger Example Hello World Example (/spring/boot\_swagger)
- Spring Boot Batch Simple example (/spring/bootbatch)
- Spring Boot + Apache Kafka Example (/spring/spring-boot-apache-kafka-hello-world)
- Spring Boot Admin Simple Example (/spring/boot-admin)
- Spring Boot Security - Introduction to OAuth (/spring/spring-boot-oauth-introduction)
- Spring Boot OAuth2 Part 1 - Getting The Authorization Code (/spring/spring-boot-oauth-authorization-code)
- Spring Boot OAuth2 Part 2 - Getting The Access Token And Using it to Fetch Data. (/spring/spring-boot-oauth-access-token)
- JBoss Drools Hello World-Stateful Knowledge Session using KieSession (/drools\_hello\_kie)
- Understand Drools Stateful vs Stateless Knowledge Session (/drools\_states)
- JBoss Drools- Understanding Drools Decision Table using Simple Example (/drools/drools\_decision)

## See Also

- [Spring Batch Interview Questions \(/spring/sprbatch\\_interview\)](/spring/sprbatch_interview)
- [Spring AOP Interview Questions \(/spring/spring-AOP-interview-questions\)](/spring/spring-AOP-interview-questions)
- [Angular 2 Interview Questions \(/angular/ang2\\_intvw\)](/angular/ang2_intvw)
- [Apache Camel Interview Questions \(/camel/Apache\\_Camel\\_Questions\)](/camel/Apache_Camel_Questions)
- [JBoss Fuse Interview Questions \(/camel/JBoss\\_Fuse\\_Questions\)](/camel/JBoss_Fuse_Questions)
- [Drools Interview Questions \(/drools/drools\\_intvw\)](/drools/drools_intvw)
- [Java 8 Interview Questions \(/java/java8\\_intvw\)](/java/java8_intvw)
- [Spring Cloud Interview Questions \(/spring/spring-cloud-interview-questions\)](/spring/spring-cloud-interview-questions)
- [Microservices Interview Questions \(/spring/microservices-interview-questions\)](/spring/microservices-interview-questions)
- [Java HashMap and ConcurrentHashMap Interview Questions \(/java/java\\_map\\_intvw\)](/java/java_map_intvw)
- [Snowflake frequently asked interview questions \(/prep/snowflake\)](/prep/snowflake)
- [SAP FI - Accounts Receivable frequently asked interview questions \(/prep/sap1\)](/prep/sap1)
- [Top SAP ALV Interview Questions \(/prep/sap3\)](/prep/sap3)
- [Top SAP Business Objects Administration Interview Questions \(/prep/sap2\)](/prep/sap2)
- [EC2 frequently asked interview questions \(/prep/ec2\)](/prep/ec2)

This site uses cookies to deliver our services and to show you relevant ads. By continuing to visit this site you agree to our use of cookies. Learn more (<http://www.javainuse.com/privacy>) 

- Mule ESB frequently asked interview questions (/misc/muleintvw)
- Apache Kafka Interview Questions (/misc/apache-kafka-interview-questions)
- Tosca Testing Tool Interview Questions (/misc/tosca-testing-tool-interview-questions)
- Top Maven Build Tool Interview Questions (/misc/maven-interview-questions)
- Top Gradle Build Tool Interview Questions (/misc/gradle-interview-questions)
- Miscellaneous Topics (/misc)

