EUREKA – Service Discovery with Spring Boot

# **Introduction**

Service Discovery is one of the key tenets of a microservice-based architecture. Trying to hand-configure each client or some form of convention can be difficult to do and can be brittle. Eureka is the Netflix Service Discovery Server and Client. The server can be configured and deployed to be highly available, with each server replicating state about the registered services to the others.

Let us consider a situation, we have producer and consumer micro service. There is one end point in consumer service which pulls information from producer service. It means one microservice calls another microservice. In this case, we can not hard code or mention the REST end point details because deployment of service may change the end point details. In this situation, service discovery comes into picture.

Let us consider an example, we have **Producer** and **Consumer** microservice. To achieve Service Discovery, we have used here Netflix Eureka.

# **Eureka Server**

## **Maven Configuration (pom.xml)**

<project xmlns=*"http://maven.apache.org/POM/4.0.0"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>eureka-server</groupId>

<artifactId>eureka-server</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>eureka-server</name>

<url>http://maven.apache.org</url>

<parent>

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

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

<version>2.0.0.RELEASE</version>

</parent>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<dependencies>

**<dependency>**

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

**<artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>**

**</dependency>**

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<scope>test</scope>

</dependency>

</dependencies>

**<dependencyManagement>**

**<dependencies>**

**<dependency>**

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

**<artifactId>spring-cloud-dependencies</artifactId>**

**<version>Finchley.RELEASE</version>**

**<type>pom</type>**

**<scope>import</scope>**

**</dependency>**

**</dependencies>**

**</dependencyManagement>**

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

</plugin>

<plugin>

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

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

## **Spring Boot Application Layer**

**EurekaServerApplication.java**

**package** com.ddlab.rnd.eureka;

**import** org.springframework.boot.SpringApplication;

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

**import** org.springframework.cloud.netflix.eureka.server.EnableEurekaServer;

@SpringBootApplication

**@EnableEurekaServer**

**public** **class** EurekaServerApplication {

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

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

}

}

## **Spring Boot Eureka Configuration**

**application.properties**

#spring.application.name=eureka

#server.servlet.context-path=/eureka

#server.port=8090

server.error.whitelabel.enabled=false

server.port=8761

#You comment below two lines, you will see Eureka server as a client and

#It will be displayed as UNKNOWN

#The below the correct configuration

eureka.client.register-with-eureka=false

eureka.client.fetch-registry=false

#

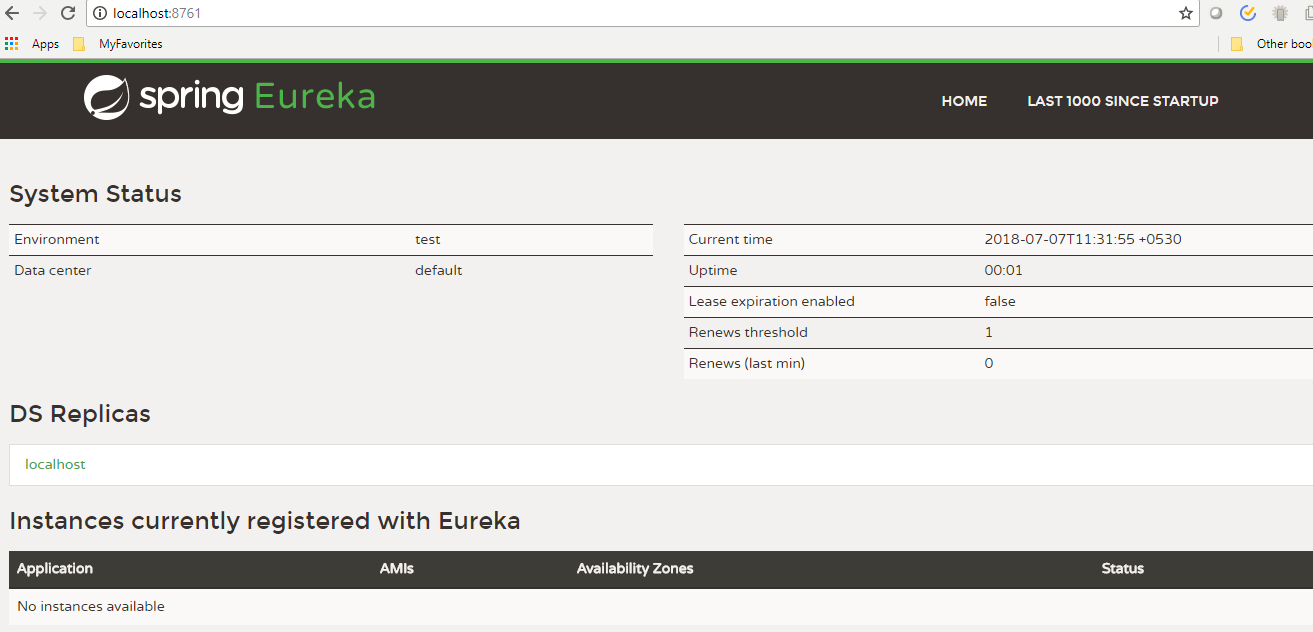
#logging.level.com.netflix.eureka=OFF

#logging.level.com.netflix.discovery=OFF

## **How to Use**

If you start the Spring boot application with Eureka in port 8761, you will see the following screen shot.

The URL for the browser : **http://localhost:8761/**



You can see that No instances are available.

# **Producer Microservice**

## **Maven Configuration (pom.xml)**

<project xmlns=*"http://maven.apache.org/POM/4.0.0"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>producer</groupId>

<artifactId>producer</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>producer</name>

<url>http://maven.apache.org</url>

<parent>

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

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

<version>2.0.0.RELEASE</version>

</parent>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<dependencies>

**<dependency>**

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

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

**</dependency>**

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<scope>test</scope>

</dependency>

**<dependency> <!-- For Eureka -->**

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

**<artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>**

**</dependency>**

</dependencies>

**<dependencyManagement> <!-- For Eureka -->**

**<dependencies>**

**<dependency>**

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

**<artifactId>spring-cloud-dependencies</artifactId>**

**<version>Finchley.RELEASE</version>**

**<type>pom</type>**

**<scope>import</scope>**

**</dependency>**

**</dependencies>**

**</dependencyManagement>**

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

</plugin>

<plugin>

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

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

## **Spring Boot Application Layer**

**ProducerBootApplication.java**

**package** com.ddlab.rnd.boot;

**import** org.springframework.boot.SpringApplication;

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

**import** org.springframework.cloud.client.discovery.EnableDiscoveryClient;

**@SpringBootApplication**

**@EnableDiscoveryClient**

**public** **class** ProducerBootApplication {

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

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

}

}

**BootConfig.java**

**package** com.ddlab.rnd.boot;

**import** org.springframework.context.annotation.ComponentScan;

**import** org.springframework.context.annotation.Configuration;

@ComponentScan(basePackages= {"com.ddlab.rnd.\*","com.ddlab.boot.\*"})

@Configuration

**public** **class** BootConfig {

}

## **Controller Layer**

**ProducerController.java**

**package** com.ddlab.boot.controller;

**import** java.util.Arrays;

**import** java.util.List;

**import** org.springframework.http.HttpStatus;

**import** org.springframework.http.MediaType;

**import** org.springframework.http.ResponseEntity;

**import** org.springframework.web.bind.annotation.CrossOrigin;

**import** org.springframework.web.bind.annotation.GetMapping;

**import** org.springframework.web.bind.annotation.PathVariable;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** org.springframework.web.bind.annotation.RestController;

@RestController

@RequestMapping("/home")

@CrossOrigin

**public** **class** ProducerController {

@GetMapping(path = "/item/{id}", produces = MediaType.***TEXT\_PLAIN\_VALUE***)

**public** ResponseEntity<String> getItemsById(@PathVariable("id") String id) {

List<String> soapList = Arrays.*asList*("Lux soap", "Rexona Soap", "Patanjali Soap");

**return** **new** ResponseEntity<String>(soapList.toString(), HttpStatus.***OK***);

}

}

## **Spring Boot and Eureka Configuration**

**application.properties**

spring.application.name=producer

server.servlet.context-path=/producer

server.port=8081

server.error.whitelabel.enabled=false

#Eureka Configuration

**eureka.client.serviceUrl.defaultZone=http://localhost:8761/eureka**

**bootstrap.properties**

spring.application.name=producer

# **Consumer Microservice**

## **Maven Configuration (pom.xml)**

<project xmlns=*"http://maven.apache.org/POM/4.0.0"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>consumer</groupId>

<artifactId>consumer</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>consumer</name>

<url>http://maven.apache.org</url>

<parent>

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

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

<version>2.0.0.RELEASE</version>

</parent>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<dependencies>

<dependency>

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

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

</dependency>

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<scope>test</scope>

</dependency>

**<dependency>**

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

**<artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>**

**</dependency>**

</dependencies>

**<dependencyManagement>**

**<dependencies>**

**<dependency>**

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

**<artifactId>spring-cloud-dependencies</artifactId>**

**<version>Finchley.RELEASE</version>**

**<type>pom</type>**

**<scope>import</scope>**

**</dependency>**

**</dependencies>**

**</dependencyManagement>**

<build>

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<configuration>

<source>1.8</source>

<target>1.8</target>

</configuration>

</plugin>

<plugin>

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

<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

</project>

## **Spring Boot Application Layer**

**ConsumerBootApplication.java**

**package** com.ddlab.rnd.boot;

**import** org.springframework.boot.SpringApplication;

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

@SpringBootApplication

**public** **class** ConsumerBootApplication {

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

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

}

}

**BootConfig.java**

**package** com.ddlab.rnd.boot;

**import** org.springframework.context.annotation.ComponentScan;

**import** org.springframework.context.annotation.Configuration;

@ComponentScan(basePackages = {"com.ddlab.rnd.\*", "com.ddlab.boot.\*"})

@Configuration

**public** **class** BootConfig {}

## **Controller Layer**

**ConsumerController.java**

**package** com.ddlab.boot.controller;

**import** java.io.IOException;

**import** java.util.List;

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

**import** org.springframework.cloud.client.ServiceInstance;

**import** org.springframework.cloud.client.discovery.DiscoveryClient;

**import** org.springframework.http.HttpEntity;

**import** org.springframework.http.HttpHeaders;

**import** org.springframework.http.HttpMethod;

**import** org.springframework.http.MediaType;

**import** org.springframework.http.ResponseEntity;

**import** org.springframework.web.bind.annotation.CrossOrigin;

**import** org.springframework.web.bind.annotation.GetMapping;

**import** org.springframework.web.bind.annotation.RequestMapping;

**import** org.springframework.web.bind.annotation.RestController;

**import** org.springframework.web.client.RestClientException;

**import** org.springframework.web.client.RestTemplate;

@RestController

@RequestMapping("/home")

@CrossOrigin

**public** **class** ConsumerController {

@GetMapping(path = "/item/{id}", produces = MediaType.***TEXT\_PLAIN\_VALUE***)

**public** ResponseEntity<String> getItemsById() **throws** RestClientException, IOException {

String baseUrl = "http://localhost:8081/producer/home/item/11";

RestTemplate restTemplate = **new** RestTemplate();

ResponseEntity<String> response = **null**;

**try** {

response = restTemplate.exchange(baseUrl, HttpMethod.***GET***, *getHeaders*(), String.**class**);

} **catch** (Exception ex) {

System.***out***.println(ex);

}

System.***out***.println(response.getBody());

**return** response;

}

**private** **static** HttpEntity<?> getHeaders() **throws** IOException {

HttpHeaders headers = **new** HttpHeaders();

headers.set("Accept", MediaType.***TEXT\_PLAIN\_VALUE***);

**return** **new** HttpEntity<>(headers);

}

**@Autowired private DiscoveryClient discoveryClient;**

**@GetMapping(path = "/items/{id}", produces = MediaType.*TEXT\_PLAIN\_VALUE*)**

**public** ResponseEntity<String> getNewItemsById() **throws** RestClientException, IOException {

System.***out***.println("discoveryClient----------->" + discoveryClient);

**discoveryClient.getServices().forEach(System.*out*::println);**

**List<ServiceInstance> instances = discoveryClient.getInstances("producer");**

System.***out***.println("instances----------->" + instances);

ServiceInstance serviceInstance = instances.get(0);

System.***out***.println("serviceInstance----------->" + serviceInstance);

String baseUrl = serviceInstance.getUri().toString();

System.***out***.println("baseUrl----------->" + baseUrl);

**baseUrl = baseUrl + "/producer/home/item/11";**

System.***out***.println("baseUrl----------->" + baseUrl);

**RestTemplate restTemplate = new RestTemplate();**

ResponseEntity<String> response = **null**;

**try** {

**response = restTemplate.exchange(baseUrl, HttpMethod.*GET*, *getHeaders*(), String.class);**

} **catch** (Exception ex) {

System.***out***.println(ex);

}

System.***out***.println(response.getBody());

**return** response;

}

}

## **Spring Eureka Configuration**

**application.properties**

spring.application.name=consumer

server.servlet.context-path=/consumer

server.port=8082

server.error.whitelabel.enabled=false

#Eureka Configuration

**eureka.client.serviceUrl.defaultZone=http://localhost:8761/eureka**

**bootstrap.properties**

**spring.application.name=consumer**

# **How to use**

Start the Eureka server, start Producer and Consumer Service.

Use the following URLs in browser, you will get responses.

**http://localhost:8761/ (For Eureka Server)**

<http://localhost:8081/producer/home/item/11> (To access producer micro service)

<http://localhost:8082/consumer/home/item/11> (To access consumer micro service)

[http://localhost:8082/consumer/home/item**s**/11](http://localhost:8082/consumer/home/items/11) (Consumer service call Producer through Eureka)

After accessing Eureka Server, you will see the following screenshot.

