**MicroServices with Spring Boot**

[**Eureka Server Spring Boot MicroService**](https://www.javadream.in/eureka-server-spring-boot-microservice/)

Eureka Server is used to monitor all the application. With the help of Eureka Server we don’t need to worry about information like which application running on which port and how many instances of application are running, all this are take cares by Eureka Server.So to Create a Eureka Server we have to follow below steps:

1. Add required dependency in you pom.xml file
2. Enable Your main class as a Eureka Server using **@EnableEurekaServer**annotation.
3. Define applicationName and portNo in application.properties file.
4. By Default Eureka act as a client but now we want this application to work as Eureka Server so we make Eureka registry as false in **application.properties** file.

1- Add the Eureka Server dependency in your pom.xml file.

|  |  |
| --- | --- |
| 1  2  3  4 | <dependency>   <groupId>org.springframework.cloud</groupId>   <artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>  </dependency> |

2- Enable Eureka Server by using **@EnableEurekaServer** annotation in your main class.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | @EnableEurekaServer  @SpringBootApplication  public class EurekaServerApplication {     public static void main(String[] args) {      SpringApplication.run(EurekaServerApplication.class, args);      }  } |

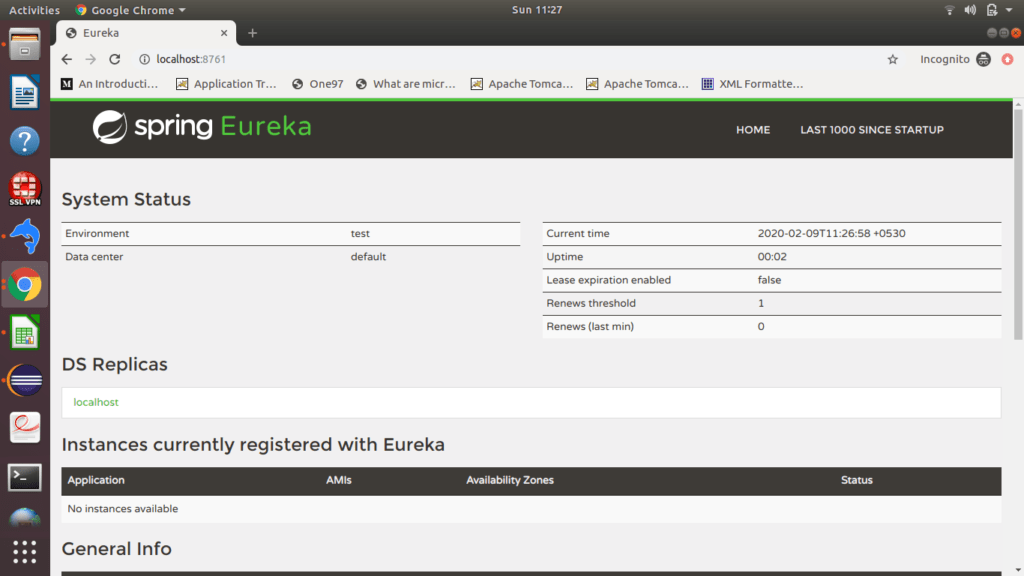
3- By Default Eureka Server application running as a client so if we have want to act this application as a server then we have to add below properties in **application.properties** file

|  |  |
| --- | --- |
| 1  2  3  4 | spring.application.name=netflix-eureka-server  server.port=8761  eureka.client.register-with-eureka=false  eureka.client.fetch-registry=false |

In above application.properties file

server.port define the port on which Eureka Server running in our case it is running on 8761 port so to access your eureka server just hit below URL in your browser.

<http://localhost:8761/>



[**Spring MicroService-1 registration with Eureka Server**](https://www.javadream.in/spring-microservice-registration-with-eureka-server/)

We know that in Monolithic application complete Code is at one place, in MicroService we divide this big monolithic application in small small microservices.

Suppose we create a separate application named **EurekaClient-1**with some rest end points and host this application on Eureka Server so we don’t need to worry about details of this application.

To create this separate client application names **EurekaClient-1**we have to follow below steps:

1- Add below dependency in your pom.xml file

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | <dependency>   <groupId>org.springframework.boot</groupId>   <artifactId>spring-boot-starter-web</artifactId>  </dependency>    <dependency>   <groupId>org.springframework.cloud</groupId>   <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>  </dependency> |

2- Enable Eureka client using **@EnableEurekaClient**in your main class.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | @EnableEurekaClient  @SpringBootApplication  public class EurekaClient1Application {     public static void main(String[] args) {      SpringApplication.run(EurekaClient1Application.class, args);      }  } |

3- Give this application name, and a Port on which it will run, and path of the Eureka Server.

|  |  |
| --- | --- |
| 1  2  3 | spring.application.name=client1  server.port=8081  eureka.client.service-url.default-zone=<http://localhost:8761/eureka> |

4- Create Controller Class and make some end points.

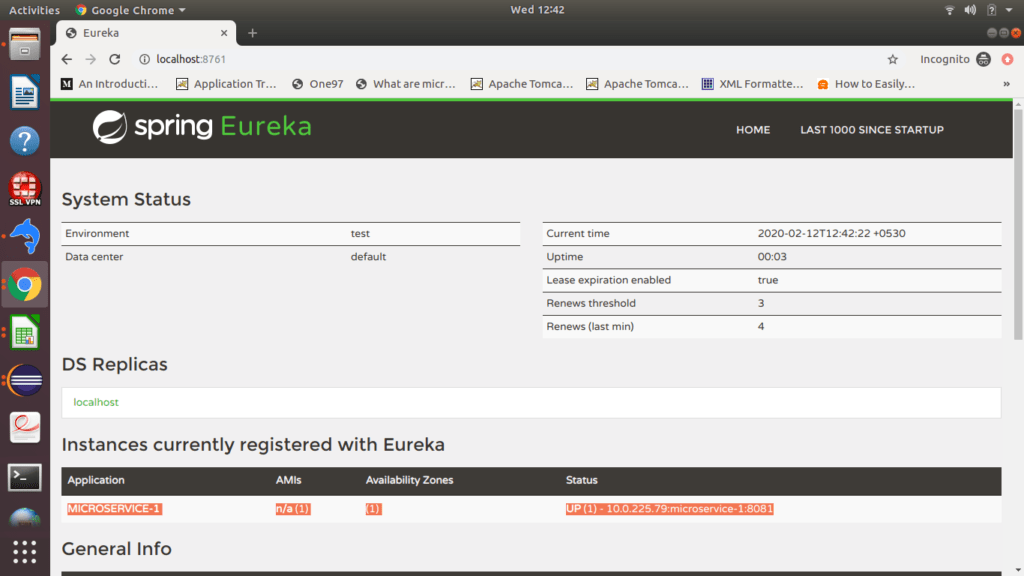
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | @RestController  public class ClientController1 {        @GetMapping("/client1")      public String client1() {          return "I am client 1";      }    } |

Run your application and see your eureka server you will see your application hosted there.

**Note**: Before running your application make sure your eureka server is running.

Now access below url for getting your end point response

<http://localhost:8081/client1>



[**Spring MicroService-2 registration with Eureka Server**](https://www.javadream.in/spring-microservice-registration-with-eureka-server/)

Suppose we create a separate application named **EurekaClient-1**with some rest end points and host this application on Eureka Server so we don’t need to worry about details of this application.

To create this separate client application names **EurekaClient-1**we have to follow below steps:

1- Add below dependency in your pom.xml file

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | <dependency>   <groupId>org.springframework.boot</groupId>   <artifactId>spring-boot-starter-web</artifactId>  </dependency>    <dependency>   <groupId>org.springframework.cloud</groupId>   <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>  </dependency> |

2- Enable Eureka client using **@EnableEurekaClient**in your main class.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | @EnableEurekaClient  @SpringBootApplication  public class EurekaClient1Application {     public static void main(String[] args) {      SpringApplication.run(EurekaClient1Application.class, args);      }  } |

3- Give this application name, and a Port on which it will run, and path of the Eureka Server.

|  |  |
| --- | --- |
| 1  2  3 | spring.application.name=client2  server.port=8082  eureka.client.service-url.default-zone=<http://localhost:8761/eureka> |

4- Create Controller Class and make some end points.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | @RestController  public class ClientController2 {        @GetMapping("/client2")      public String client2() {          return "I am client 2";      }    } |

Run your application and see your eureka server you will see your application hosted there.

**Note**: Before running your application make sure your eureka server is running.

Now access below url for getting your end point response

<http://localhost:8081/client2>

