**Eureka Server Application**

You can think of it as a structure that keeps a list of applications running on the platform. Each microservice application registers itself to Eureka Server. Eureka Server has a mechanism that periodically sends requests to applications and checks whether they are alive or not. If the previously registered application is not standing, it is automatically deleted from the list.

The contents of the **pom.xml**file are as follows.

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?> |
|  | <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 https://maven.apache.org/xsd/maven-4.0.0.xsd"> |
|  | <modelVersion>4.0.0</modelVersion> |
|  | <parent> |
|  | <groupId>org.springframework.boot</groupId> |
|  | <artifactId>spring-boot-starter-parent</artifactId> |
|  | <version>3.1.1</version> |
|  | <relativePath /> |
|  | </parent> |
|  | <groupId>com.eurekaserver.application</groupId> |
|  | <artifactId>eureka-server</artifactId> |
|  | <version>1.0.0</version> |
|  | <name>eureka-server</name> |
|  | <description>Eureka Server Application</description> |
|  |  |
|  | <properties> |
|  | <java.version>17</java.version> |
|  | <spring-cloud.version>2022.0.3</spring-cloud.version> |
|  | </properties> |
|  |  |
|  | <dependencies> |
|  | <dependency> |
|  | <groupId>org.springframework.cloud</groupId> |
|  | <artifactId>spring-cloud-starter-netflix-eureka-server</artifactId> |
|  | </dependency> |
|  | </dependencies> |
|  |  |
|  | <dependencyManagement> |
|  | <dependencies> |
|  | <dependency> |
|  | <groupId>org.springframework.cloud</groupId> |
|  | <artifactId>spring-cloud-dependencies</artifactId> |
|  | <version>${spring-cloud.version}</version> |
|  | <type>pom</type> |
|  | <scope>import</scope> |
|  | </dependency> |
|  | </dependencies> |
|  | </dependencyManagement> |
|  |  |
|  | <build> |
|  | <plugins> |
|  | <plugin> |
|  | <groupId>org.springframework.boot</groupId> |
|  | <artifactId>spring-boot-maven-plugin</artifactId> |
|  | </plugin> |
|  | </plugins> |
|  | </build> |
|  | </project> |

The content of the **application.yml** file is as follows.

|  |
| --- |
| eureka: |
|  | client: |
|  | registerWithEureka: false |
|  | fetchRegistry: false |
|  | server: |
|  | port: 8761 |

***server.port:****It is specified on which port the application will serve. The default port number for Eureka Server is 8761.*

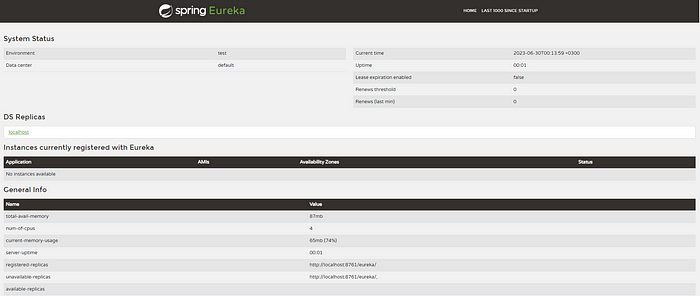
***eureka.client.registerWithEureka:****We set this property to false so that it does not register itself in the list. Because; Here it acts as a server, not a client.*

***eureka.client.fetchRegistry:****This value is set to false because it will not retrieve the registered microservices list from anywhere. It will create and maintain this list itself.*

|  |
| --- |
| @SpringBootApplication |
|  | @EnableEurekaServer |
|  | public class EurekaServerApplication { |
|  | public static void main(String[] args) { |
|  | SpringApplication.run(EurekaServerApplication.class, args); |
|  | } |
|  | } |

Here, we use the **EnableEurekaServer**annotation to define the application that will run as an Eureka Server application.

Now, when we wake up the application and call the address <http://localhost:8761/>, a page like the one below should open.



**Eureka Discovery Client Application**

It is a sample microservice application that will register to Eureka Server.

The contents of the **pom.xml**file are as follows.

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?> |
|  | <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 https://maven.apache.org/xsd/maven-4.0.0.xsd"> |
|  | <modelVersion>4.0.0</modelVersion> |
|  | <parent> |
|  | <groupId>org.springframework.boot</groupId> |
|  | <artifactId>spring-boot-starter-parent</artifactId> |
|  | <version>3.1.1</version> |
|  | <relativePath /> |
|  | </parent> |
|  | <groupId>com.eurekaclient.application</groupId> |
|  | <artifactId>eureka-client</artifactId> |
|  | <version>0.0.1-SNAPSHOT</version> |
|  | <name>eureka-client</name> |
|  | <description>Eureka Client Application</description> |
|  |  |
|  | <properties> |
|  | <java.version>17</java.version> |
|  | <spring-cloud.version>2022.0.3</spring-cloud.version> |
|  | </properties> |
|  |  |
|  | <dependencies> |
|  | <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> |
|  |  |
|  | </dependencies> |
|  |  |
|  | <dependencyManagement> |
|  | <dependencies> |
|  | <dependency> |
|  | <groupId>org.springframework.cloud</groupId> |
|  | <artifactId>spring-cloud-dependencies</artifactId> |
|  | <version>${spring-cloud.version}</version> |
|  | <type>pom</type> |
|  | <scope>import</scope> |
|  | </dependency> |
|  | </dependencies> |
|  | </dependencyManagement> |
|  |  |
|  | <build> |
|  | <plugins> |
|  | <plugin> |
|  | <groupId>org.springframework.boot</groupId> |
|  | <artifactId>spring-boot-maven-plugin</artifactId> |
|  | </plugin> |
|  | </plugins> |
|  | </build> |
|  | </project> |

The content of the **application.yml** file is as follows.

|  |
| --- |
| eureka: |
|  | client: |
|  | serviceUrl: |
|  | defaultZone: http://localhost:8761/eureka/ |
|  | server: |
|  | port: 8080 |
|  | spring: |
|  | application: |
|  | name: eureka-client |

***eureka.client.serviceUrl.defaultZone:****This information must be given to which Eureka Server address the application will connect to.*

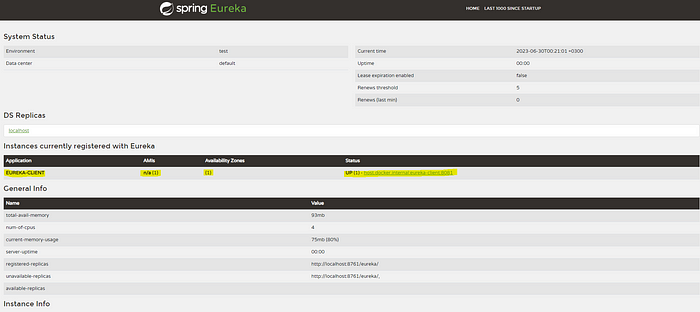
***server.port:****It is the port number information on which the application will run. Spring boot applications use 8080 by default. You can change it if you want.*

***spring.application.name:****The name of the application. It is saved in the Eureka Server with the name you give.*

|  |
| --- |
| @SpringBootApplication |
|  | @EnableEurekaClient |
|  | public class EurekaClientApplication { |
|  |  |
|  | public static void main(String[] args) { |
|  | SpringApplication.run(EurekaClientApplication.class, args); |
|  | } |
|  | } |

Here, we use the EnableEurekaClient annotation to define the application that will run as a client application.

Now, when we wake up the application and call the address <http://localhost:8761/>, we can see that the application named eureka-client that we created in the list is registered to the Eureka Server.



Eureka Discovery Client, Eureka Server registered.