

Microservicios – Arquitectura y Desarrollo

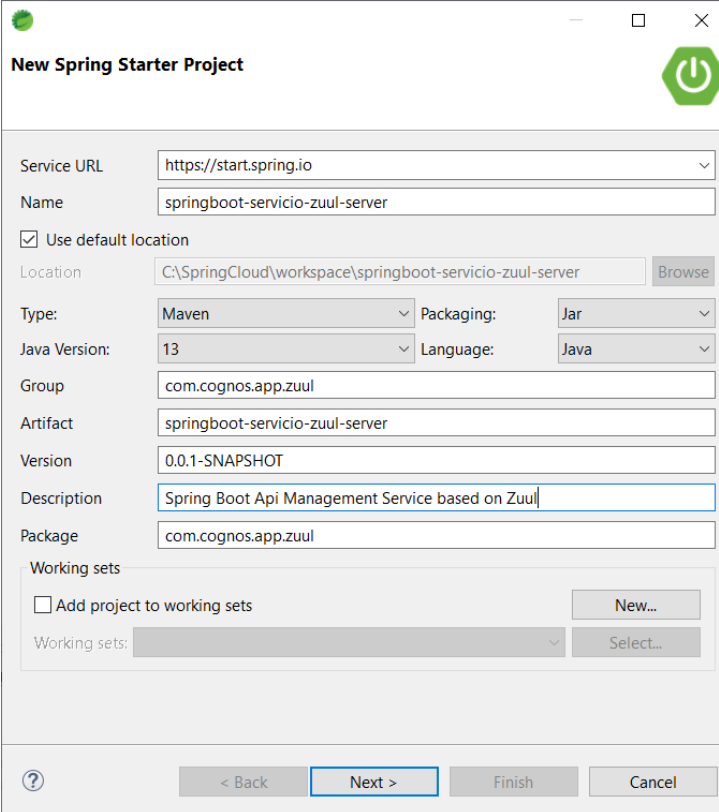
LAB Api Gateway y Microservicios con Netflix Zuul

Objetivos

- Mostrar al participante el procedimiento para el desarrollo de Microservicios usando el servicio de Api Management con **Netflix Zuul**

Procedimiento

1. Crea un nuevo proyecto llamado **springboot-servicio-zuul-server** y agrega las dependencias indicadas.

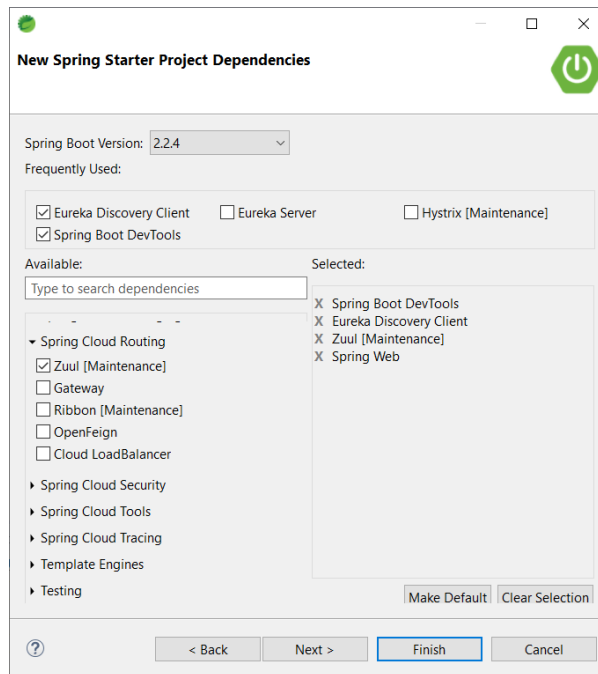


The screenshot shows the 'New Spring Starter Project' dialog box. The fields are filled as follows:

- Service URL: <https://start.spring.io>
- Name: `springboot-servicio-zuul-server`
- ☒ Use default location
- Location: `C:\SpringCloud\workspace\springboot-servicio-zuul-server`
- Type: `Maven`
- Packaging: `Jar`
- Java Version: `13`
- Language: `Java`
- Group: `com.cognos.app.zuul`
- Artifact: `springboot-servicio-zuul-server`
- Version: `0.0.1-SNAPSHOT`
- Description: `Spring Boot Api Management Service based on Zuul`
- Package: `com.cognos.app.zuul`

At the bottom, the 'Next >' button is highlighted with a blue border.

Clic en Next.



Agrega las dependencias **Spring Boot DevTools**, **Eureka Discovery Client**, **Zuul** y **Spring Web**. Clic en Finish.

2. Edita la clase **SpringbootServicioZuulApplication** y habilita Zuul en el Proyecto agrega la anotación **@EnableZuulProxy**

```

1 package com.cognos.app.zuul;
2
3 import org.springframework.boot.SpringApplication;
4
5
6
7 @EnableZuulProxy
8 @SpringBootApplication
9 public class SpringbootServicioZuulServerApplication {
10
11     public static void main(String[] args) {
12         SpringApplication.run(SpringbootServicioZuulServerApplication.class, args);
13     }
14 }
15
16

```

3. Edita el archivo **application.properties**, define el nombre y puerto del servicio, configura las rutas indicadas.

```

1
2 spring.application.name=servicio-zuul-server
3 server.port=8090
4
5 eureka.client.service-url.defaultZone=http://localhost:8761/eureka
6
7 zuul.routes.productos.service-id=servicio-productos
8 zuul.routes.productos.path=/api/productos/**
9
10 zuul.routes.items.service-id=servicio-items
11 zuul.routes.items.path=/api/items/**
12

```

4. Inicia todos los servicios y verifica con Eureka que los servicios están activos (up).

The screenshot shows the Spring Eureka web interface in a browser. The page has a dark header with the 'spring Eureka' logo and navigation links for 'HOME' and 'LAST 1000 SINCE STARTUP'. The main content area is divided into several sections:

- System Status:** A table showing environment details. The environment is 'test' and the data center is 'default'. Other metrics include current time (2020-02-05T13:06:06-0400), uptime (01:59), lease expiration enabled (true), renew threshold (6), and renew interval (8).
- DS Replicas:** A section showing the local data center 'localhost'.
- Instances currently registered with Eureka:** A table listing three services: 'SERVICIO-ITEMS', 'SERVICIO-PRODUCTOS', and 'SERVICIO-ZUUL-SERVER'. Each service is shown with its application name, AMIs, availability zones, and status (UP).
- General Info:** A table showing system metrics such as total available memory (94mb), environment (test), number of CPUs (8), and current memory usage (56mb (59%)).

5. Realiza una prueba usando Postman.

Probando el servicio **SERVICIO-ITEMS**: accede a <http://localhost:8090/api/items/listar>

The screenshot shows the Postman application interface. A GET request is configured to the URL 'http://localhost:8090/api/items/listar'. The request is sent, and the response is displayed in the 'Body' tab, which is formatted as JSON. The response shows a list of two items:

```
1 {
2   "producto": {
3     "id": 1,
4     "nombre": "Cup",
5     "precio": 800.0,
6     "creado": "2020-02-05T00:00:00-0000",
7     "port": 0
8   },
9   "cantidad": 1,
10  "total": 800.0
11 },
12 {
13   "producto": {
14     "id": 2,
15     "nombre": "T-Shirt",
16     "precio": 500.0,
17     "creado": "2020-02-05T00:00:00-0000",
18     "port": 0
19   },
20   "cantidad": 1,
21   "total": 500.0
22 }
23 }
```

Probando el servicio **SERVICIO-PRODUCTOS**, accede a <http://localhost:8090/api/productos/mostrar/3>

