**Spring Cloud**

**22 Jul 2018**

* **Config Server**

1. Config server is used to store configuration files required for all the micro services
2. To check successful configuration - http://localhost:8888/config/default
3. Annotate Spring boot class with **@EnableConfigServer**
4. Add below configuration in **application.properties** file

spring.profiles.active=native

spring.cloud.config.server.native.search-locations=classpath:/common-config

management.security.enabled=false

server.port = 8888

1. Now in **common-config** folder create **application.properties** file and add below line for eureka server so that any micro service with config server dependency automatically access eureka server URL.

eureka.client.serviceUrl.defaultZone=http://localhost:8090/eureka

1. Below dependency is required

<dependencies>

<dependency>

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

<artifactId>spring-cloud-config-server</artifactId>

</dependency>

<dependency>

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

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

</dependency>

</dependencies>

<dependencyManagement>

<dependencies>

<dependency>

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

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

<version>Camden.SR6</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>

<resources>

<resource>

<filtering>true</filtering>

<directory>src/main/resources</directory>

<includes>

<include>\*.properties</include>

<include>common-config/</include>

</includes>

</resource>

</resources>

</build>

* **Eureka Server – Netflix**

1. Eureka Server is used to register all the services
2. Just annotate spring boot class with @EnableEurekaServer and Include below dependency

<dependency>

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

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

</dependency>

* **Service Registration in Eureka**

1. To include service in Eureka server just annotate spring boot class with @EnableDiscoveryClient and then Just include below two dependencies.

Config is dependency is used only if spring config server is used otherwise only eureka dependency is enough to register service in eureka server.

<dependency>

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

<artifactId>spring-cloud-starter-config</artifactId>

</dependency>

<dependency>

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

<artifactId>spring-cloud-starter-eureka</artifactId>

</dependency>

* **Accessing Service from Eureka**

1. Below code is used to access the service from eureka, Eureka Server and Config server dependency is required same as registration above.

@Autowired

private DiscoveryClient discoveryClient;

ServiceInstance serviceInstance=loadBalancer.choose("employee-producer"); System.out.println(serviceInstance.getUri());

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

baseUrl=baseUrl+"/employee";

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

headers.set("Accept", MediaType.***APPLICATION\_JSON\_VALUE***);

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

RestTemplate restTemplate = new RestTemplate();

ResponseEntity<String> response=null;

try{

response=restTemplate.exchange(baseUrl,

HttpMethod.GET, headers,String.class);

}catch (Exception ex)

{

System.out.println(ex);

}

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

* **Ribbon - Client Side Load balancing for accessing service from Eureka**

1. Just need to below dependency instead of DiscoveryClient and rest code to access service is same and also dependency required are eureka and ribbon and config server as per configuration whether config server is used.

@Autowired

private LoadBalancerClient loadBalancer;

<dependency>

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

<artifactId>spring-cloud-starter-ribbon</artifactId>

</dependency>

* **Feign Client for accessing service from Eureka**

1. Feign client access micro service directly and include load balancer client as per dependency in the pom.

@FeignClient(name="employee-producer") – name is the name of service

**public** **interface** RemoteCallService {

@RequestMapping(method=RequestMethod.GET, value="/employee")

public Employee getData();

}

Just include @EnableFeignClients in spring boot application and use

@Autowired

private RemoteCallService remoteCallService;

<dependency>

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

<artifactId>spring-cloud-starter-feign</artifactId>

</dependency>

* **Zuul service to as Gateway for accessing service from Eureka**

1. **Zuul service will act as gateway for all the services**
2. Include **@EnableZuulProxy** annotation in spring boot application. Create pre,post,error and route ZuulFilter in main application.

Include in application.properties file

zuul.routes.producer.url=http://localhost:8080

above line indicate that any url which include producer will redirect to localhost:8080

eureka.client.serviceUrl.defaultZone=http://localhost:8090/eureka

**<dependency>**

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

**<artifactId>spring-cloud-starter-zuul</artifactId>**

**</dependency>**

* **Spring Bus**

**Yogesh Dhawale**