Publishing MicroService to Eureka server (R & D Server)

=> Every Ms must be published/registered with Eureka Server (R&D Server) by becoming Eureka client => we need to develop MicroService as Spring RestController adding the the support

of @EnableEurekaClient (2.x)

(or)

@EnableDiscoveryClient (3.x)

PUBLISH

Spring Rest Project (#MS1)

(@EnableEurekaClient)

(@EnableDiscoveryClient)

RestController

Spring Boot Project (starter Project)

(Eureka server) [@Enable EurekaServer]

MS#1 - MS#2 - MS#3

(instance of the published

microServices)

Spring Rest Project (#MS3) (@EnableEurekaClient) (@EnableDiscoveryClient)

RestController

Every MS that will be published

in Eureka R&D Server is technically called Eureka Client Pureka DisoveryClient

(2.x)

(3.x)

=> Netflix Eureka R&D server is called as No Document server becoz

at max it will maintain the service instances of the registered Micro Services and does not maintain any docs related to that

=>The regular RestController (Restful service) that is taken as @EnableEureka Client and registered with Eureka server is called Restful MicroService

(SearchService)

PUBLISH

PUBLISH

...

Spring Rest Project (#MS2)

(@EnableEurekaClient)

(@EnableDiscovery Client)

RestController

it to

(CartService)

Procedure for MS Development and Publishing Eureka R&D Server

step1) Make sure One Spring boot Project alreay developed and running as Eureka Server (Previous class) (make sure that it is running on the Discovery

(Order Service)

The RestController App/Project

=> Spring cloud /Spring boot cloud is a module in spring/spring boot framework that gives set of tools and libraries to develop the microservice architecture based Apps/Projects

with @EnableEurekaClient (or) @EnableDiscoveryClient is called

MS

in

that is ready to publish Eureka R & D Server

port: 8761)

step2) Create Spring Boot Starter Project adding spring web, EurekaClient Dependencies (MicroService Development)

(@EnableDiscoveryClient)

step3) Place @Enable EurekaClient Annotation on top of main class.

@SpringBootApplication

@EnableEurekaClient (or) @EnableDiscoveryClient

public class Spring BootMsProj01SearchServiceApplication {

}

public static void main(String[] args) {

}

MicroService = @Restcontroller class enabled with @EnableEurekaClient or @EnableDiscoveryClient

if ur having plan of creating multiple instances for

MS running on different port numbers.. it is recommanded not to add spring boot dev tools starter to the Project

To make @RestController as the MS

in spring boot 2.x use @EnableEurekaClient (removed from spring boot 3.x)

In spring boot 3.x use @EnableDiscoveryClient

SpringApplication.run(SpringBootMsProj01SearchServiceApplication.class, args);

step4) add the following entries in application.properties file

step5)

in applicaiton.properties

#MS service port number

server.port=7171

# Service Id

spring.application.name=Search-Service

# Specify the Eureka server URL to publish the MS
eureka.client.service-url.default-zone=http://localhost:8761/eureka

Develop RestController representing the MicroService package com.nt.controller;

fixed path for publishing

import org.springframework.web.bind.annotation.GetMapping; import org.springframework.web.bind.annotation.RequestMapping; import org.springframework.web.bind.annotation. RestController;

@RestController

@RequestMapping("/search")

public class SearchServiceController {

@GetMapping("/display")

public String displayMessage() {

return "Welcome to Flipkart Search Service";

}

}

note: we can register same multiple instances of same or different microservices with Eureka server.. For single instance registration the Service id becomes instanced id and while working multiple instances we generate instanced ids dynamically

step6) Run the MicroService Project as spring boot App

(This process automatically publishes MS to Eureka server)

step7) Refresh the home page of eureka server (http://localhost:8761) and observe the instance section

Instances currently registered with Eureka

Application

SEARCH-SERVICE

becoz of @EnableDiscoveryClient the MS becomes ready to register with EurekaServer

AMIS

Availability Zones

Status

n/a (1)

(1)

UP (1) -DESKTOP-IUDAAVL:Search-Service:7171

if we register only one instance of MS Project with Eureka server then the service id/name of the project becomes Instance id

note::

becoz of the http://localhost: 8761/eureka url given in application.properties the step8) Collect url from status section and modify it to generate the request to MS

MS will be registered to Eureka server automatically

http://desktop-iudaavl:7171/search/display

(or)

http://localhost:7171/search/display

Inter communication between MicroServices

====

=> To see communication b/w two micro services .. both micro services must be published in the Eureka Server

=> The MS that provides services is called Provider/Server/Producer/Parent Service

=> The MS that cosumes services is called Cosumer/Client/Child Service

CartService <---------->

PaymentService

(consumer/Client)

Employee <-------

(consumer/Client)

(Procudure/Server)

Department

(Procudure/Server)

CartService

(consumer/Client)

Billing Service (Procudure/Server)

Generally the the Cosumer and Producer Apps will be represented as show below

(parent and child)

Payment service

(parent)

Doctor

(parent)

cart service

(child) Patient (child)

(b) Publish

note:: Here Provider

to Conusmer interaction

and consumer to provider interaction is possible

becoz both Microservices are published in EurekaService

MicroService style intra communication

Consumers (c)

EurekaServer

find(-)

MS#1 MS#2 (consumerMS) (provider Ms)

publish (a)

Stent Comp

RestController

request (d)

(g)

RestTemplate

MS#1

response (f)

Provider MS

RestController (e)

MS#2

if ur planning to register multiple instances of the MS with Eureka server.. then we need to place special entries in the application.properties file MS Project to generate instanced ids randomly

EurekaServer

(b)

Publish

find(-)

MS#1 (consumer),

MS#2

publish (a)

(provider)

Consumer Ms (c)

RestController

with RestTemplate

request (d)

MS#1

response (f)

Provider MS

RestController (e) MS#2

SOA style Intra communication (SOAP based

=======

==================

Registry (UDDI)

wsdl

WebServices)

Here we need to have

1 Publish

fixed Provider and

fixed Cosumer Serivces/Apps

Here both Consumer and Provider services will be published

the

to Eureka server before starting interaction

=> Here any MS can becom provider MS or Consumer MS

=> In the given situation, the MS can become both provider and consumer

2

find

Service Provider

Service Consumer

req

(stub)

Bind (3)

(Skelton)

res

Here only provider Service will be published.

=>In the intra communication of MicroService through Eureka Server the Consumer/Client/Child MS

should find and get the details of Provider/Producer/Server MS by submitting its service Id .. For this the Cosumer helper MS must use one special comp "Client Comp/Client Type comp". (provider MS)

=>The work of "Client comp" in Consumer Ms is

(It is helper or supporting comp in Consumer App)

a) getting Producer MS service Instance from Eureka SErver by submitting its Service Id /instance ID

b) Gathers Producer MS details like URL/URI,method type, PATH and etc.. from Service Instance (endpoint details) Passing the above details to RestTemplate of Consumer MS to make RestTemplate to send

http request to Producer Ms and to recieve http response from provider MS

As of now In Eureka server env.. 3 types of "Client Comps/Client type comps" are possible

a) Discovery Client Comp (Very Basic Client -- Legacy)

b) LoadBalance Client comp (good)

c) Feign Client Comp (Abstract Client -- best) (best)

(Does not support Load Balancing)

supports Load Balancing

service id

Eureka Server

Ms#1 ms#2

(provider MS)

MS#1--->Consumer MS

(b)

MS#2 ----> Producer MS/Provider MS

service find("ms#2")

(a)

(c)

Ms2 service

Consumer Ms Instance obj (RestController)

details from Service instance (URL, PATH,.....]

Client Comp (d) RestTemplate

(e)

http request to Provider App

(f)

MS#2

MS#1

http response from provider App (g)

provider Ms logics

Here only one way communication is possible that is Consumer to provider

In Consumer MS Project

and we link with Consumer MS

as separate class

=> For modularity purpose, we try to develop Client comp out side of Consumer MS as seperate class

=> Application name given in MS project becomes Service ID

=> if Win the MS for multiple times then we need to give/get multiple instance ids

=The Application/Service is having only one instance then the service Id itself becomes instance id

How do u create multiple instances of one Ms (MicroService)?

Ans) Add service id + random value as the instance id in application.properties and run the same MS Project App for multiple times with different port numbers

step1) Develop the MS having instance id in application.properties file

application.properties (In MicroService Project)

# service id

note:: before running the app /project for multiple times change port numbers like 9090, 9091,9092,9093 and etc..

spring.application.name=FirstMs

# Instance id

spring.application.instance-id=${spring.application.name}; ${random.value}

#Embbed Server port number

server.port=4041

#registration url of Eureka server

eureka.client.service-url.default-zone=http://localhost:8761/eureka

step2) Devleop the @RestController class as the MicroService

//SeasonFinder.java

package com.nt.ms;

```
import java.time.LocalDateTime;

import org.springframework.beans.factory.annotation.Value; import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity; import
org.springframework.web.bind.annotation.GetMapping;

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

@RestController

@RequestMapping("/season-api")

public class Season FinderOperationsMs {

@Value("${spring.application.instance-id}")

private String instance_id;

}

//Rest operation/ endpoint

@GetMapping("/show-season")

public ResponseEntity<String> findSeason(){

//get Local date and time

LocalDateTime ldt=LocalDateTime.now();

//get current month

int month=ldt.getMonthValue();

String seasonName=null;

if(month>=7&& month<=10)

seasonName="Rainy Season";

else if(month>=3 && month<=6)

seasonName="Summer Season";

else

seasonName="Winter Season";

// improve the output

seasonName=season Name+"---"+instance_id;

//return the ResponseEntity object

return new ResponseEntity<String>(season Name, HttpStatus.OK);

}
```

step3) make sure that Eureka server project is already available

step4) run the applications/project is the following order

a) run Eureka server

b) run MS Project multiple times,but the change the MS port number

every time 4041,4042,4043 and etc..

c) launch and observe the Eureka server console

http://localhost:8761

Instances currently registered with Eureka

Availability Zones

Status

Application AMIs FIRSTMS n/a (3) (3)

4be4c75d2bd720373c236c5c, Search-

UP (3) - DESKTOP-JEUKT9C:FirstMs:4043, DESKTOP-JEUKT9C:FirstMs:4042, DESKTOP-JEUKT9C:FirstMs:4041
instance2

instance1

instance3