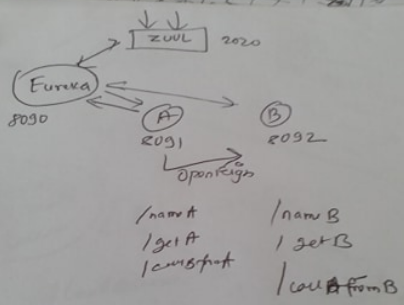
**ZUUL\_EUREKA\_OPENFEIGN\_CIRCUITBREAKER**



**FromZuul Api gateway**

<http://192.168.0.19:2020/B-api/callAFromB>

<http://192.168.0.19:2020/A-api/callBFromA>

**MainClass**

@SpringBootApplication  
@EnableDiscoveryClient  
@EnableZuulProxy  
**public class** ZuulApiGatewayApplication {  
  
 **public static void** main(String[] args) {  
 SpringApplication.*run*(ZuulApiGatewayApplication.**class**, args);  
 }  
  
}

**Application.properties**

**spring.application.name**= **api-gateway  
server.port** = **2020  
  
  
zuul.routes.A-service.path**=**/A-api/\*\*  
zuul.routes.A-service.serviceId**=**A-SERVICE  
  
  
zuul.routes.B-service.path**=**/B-api/\*\*  
zuul.routes.B-service.serviceId**=**B-SERVICE**

**serviceA**

<http://192.168.0.19:8091/nameA>

<http://192.168.0.19:8091/callBFromA>

**MainClass**

@EnableDiscoveryClient  
@SpringBootApplication  
@EnableFeignClients  
**public class** ServiceAApplication {  
 **public static void** main(String[] args) {  
 SpringApplication.*run*(ServiceAApplication.**class**, args);  
 }  
}

**MyFeignClientA.java**@FeignClient(value=**"B-SERVICE"**)  
**public interface** MyFeignClientA {  
  
 @GetMapping(value=**"getB"**)  
 String getB();  
  
}

**ControllerA.java**@RestController  
**public class** ControllerA {  
  
 @Autowired  
 MyFeignClientA **myFeignClientA**;  
  
 @GetMapping(**"/nameA"**)  
 **public** String getControllerName() {  
 **return "Name A From A Service IP Address: "**;  
 }  
  
 @GetMapping(**"/getA"**)  
 **public** String getAaa() {  
 **return "THIS IS SENT FROM A SERVICE"**;  
 }  
  
 @GetMapping(**"/callBFromA"**)  
*// @HystrixCommand(fallbackMethod = "serviceMayBeDown") // This is usually used in service class OR used where  
 // restTemplate is used to communicate with another  
 // micro-service* @HystrixCommand(fallbackMethod = **"serviceMayBeDown"**, commandProperties = {  
 @HystrixProperty(name = **"execution.isolation.thread.timeoutInMilliseconds"**, value = **"1500"**)  
 })  
 **public** String communicate() {  
  
 **return myFeignClientA**.getB();  
  
 }  
  
 **public** String serviceMayBeDown() {  
 **return "Requested B Service May be down or too busy"**;  
 }  
  
 }

**Application.properties**

**server.port**=**8091  
  
spring.application.name**= **A-SERVICE  
eureka.instance.leaseRenewalIntervalInSeconds** = **28  
eureka.instance.leaseExpirationDurationInSeconds** = **1  
  
eureka.client.serviceUrl.defaultZone** = **http://localhost:8761/eureka/  
  
eureka.client.healthcheck.enabled** = **true**

**serviceB**

<http://192.168.0.19:8092/nameB>

<http://192.168.0.19:8092/callAFromB>

**MainClass**

@EnableDiscoveryClient  
@SpringBootApplication  
@EnableFeignClients  
**public class** ServiceBApplication {  
 **public static void** main(String[] args) {  
 SpringApplication.*run*(ServiceBApplication.**class**, args);  
 }  
  
 }

**MyFeignClientB.java**

@FeignClient(value=**"A-SERVICE"**)  
**public interface** MyFeignClientB {  
  
 @GetMapping(value=**"getA"**)  
 String getA();  
  
}

**ControllerB.java**

@RestController  
**public class** ControllerB {  
  
 @Autowired  
 MyFeignClientB **myFeignClientB**;  
  
 @GetMapping(**"/nameB"**)  
 **public** String getControllerName() {  
 **return "Name B From B Service IP Address: "**;  
 }  
  
 @GetMapping(**"/getB"**)  
 **public** String getBbb() {  
  
*// try {  
// Thread.sleep(200);  
// } catch (InterruptedException e) {  
// //* ***TODO Auto-generated catch block****// e.printStackTrace();  
// }* **return "THIS IS SENT FROM B SERVICE"**;  
 }  
  
  
 @GetMapping(**"/callAFromB"**)  
 **public** String communicate() {  
 **return myFeignClientB**.getA();  
  
 }  
  
}

**Application.propertiesFile  
server.port**=**8092  
  
spring.application.name**= **B-SERVICE  
eureka.instance.leaseRenewalIntervalInSeconds** = **28  
eureka.instance.leaseExpirationDurationInSeconds** = **1  
  
eureka.client.serviceUrl.defaultZone** = **http://localhost:8761/eureka/  
  
eureka.client.healthcheck.enabled** = **true**

**EurekaServer**

**Mainclass**

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

**Application.propertiesFile**

**server.port**=**8761  
spring.application.name**= **eureka-server  
eureka.client.register-with-eureka**=**false  
eureka.client.fetch-registry**=**false  
  
logging.level.com.netflix.eureka**=**OFF  
logging.level.com.netflix.discovery**=**OFF**

