springcloud

随笔

```
spring Boot版本为2.0.3.RELEASE，Spring Cloud版本为Finchley.RELEASE。这两个一定要对应起来
pom文件中
   <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.0.3.RELEASE</version>
        <relativePath/>
    </parent>

<properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <project.reporting.outputEncoding>UTF-
8</project.reporting.outputEncoding>
        <java.version>1.8</java.version>
        <spring-cloud.version>Finchley.RELEASE</spring-cloud.version>
    </properties>
```

1. 服务注册与发现中心 Eureka server

```
1.引入spring-cloud-starter-netflix-eureka-server的依赖，
2.启动一个服务注册中心，只需要一个注解@EnableEurekaServer，这个注解需要在springboot工程的启动application类上
3.eureka是一个高可用的组件，它没有后端缓存，每一个实例注册之后需要向注册中心发送心跳（因此可以在内存中完成），在默认情况下erureka server也是一个eureka client ,必须要指定一个 server。
在appication.yml通过eureka.client.registerWithEureka：false和fetchRegistry：false来表明自己是一个eureka server.
```

```
server:
  port: 8761

eureka:
  instance:
    hostname: localhost
  client:
    registerWithEureka: false
    fetchRegistry: false
    serviceUrl:
      defaultZone: http://${eureka.instance.hostname}:${server.port}/eureka/

spring:
  application:
```

```
    name: eurka-server
```

2. 服务提供者 (eureka client)

当client向server注册时，它会提供一些元数据，例如主机和端口，URL，主页等。Eureka server 从每个client实例接收心跳消息。 如果心跳超时，则通常将该实例从注册server中删除。

需要指明spring.application.name,这个很重要，这在以后的服务与服务之间相互调用一般都是根据这个name 。

```
1. 引入依赖 <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
2. 通过注解@EnableEurekaClient 表明自己是一个eurekaclient.
3. 还需要在配置文件中注明自己的服务注册中心的地址，application.yml配置文件如下：
```

```
server:
  port: 8762

spring:
  application:
    name: service-hi

eureka:
  client:
    serviceUrl:
      defaultZone: http://localhost:8761/eureka/
```

3.Feign

Feign 采用的是基于接口的注解

Feign 整合了ribbon，具有负载均衡的能力

整合了Hystrix，具有熔断的能力

```
1. pom文件引入Feign的起步依赖spring-cloud-starter-openfeign、Eureka的起步依赖spring-cloud-starter-netflix-eureka-client、Web的起步依赖spring-boot-starter-web
2. 配置文件application.yml文件，指定程序名为service-feign，端口号为8765，服务注册地址为http://localhost:8761/eureka/
3. 在程序的启动类ServiceFeignApplication ，加上@EnableFeignClients注解开启Feign的功能
4. 定义一个feign接口，通过@ FeignClient（“服务名”），来指定调用哪个服务。比如在代码中调用了service-hi服务的“/hi”接口
5. 在Web层的controller层，对外暴露一个"/hi"的API接口，通过上面定义的Feign客户端 SchedualServiceHi 来消费服务。
```

```yaml
eureka:
  client:
    serviceUrl:
      defaultZone: http://localhost:8761/eureka/
server:
  port: 8765
spring:
  application:
    name: service-feign
```

```java
@FeignClient(value = "service-hi")
public interface SchedualServiceHi {
    @RequestMapping(value = "/hi",method = RequestMethod.GET)
    String sayHiFromClientOne(@RequestParam(value = "name") String name);
}
```

```java
@RestController
public class HiController {


    //编译器报错，无视。 因为这个Bean是在程序启动的时候注入的，编译器感知不到，所以报错。
    @Autowired
    SchedualServiceHi schedualServiceHi;

    @GetMapping(value = "/hi")
    public String sayHi(@RequestParam String name) {
        return schedualServiceHi.sayHiFromClientOne( name );
    }
}
```

4.断路器（Hystrix）

如果单个服务出现问题，调用这个服务就会出现线程阻塞，此时若有大量的请求涌入，Servlet容器的线程资源会被消耗完毕，导致服务瘫痪。服务与服务之间的依赖性，故障会传播，会对整个微服务系统造成灾难性的严重后果，这就是服务故障的“雪崩”效应。

当对特定的服务的调用的不可用达到一个阀值（Hystric 是5秒20次）断路器将会被打开。

断路打开后，可用避免连锁故障，fallback方法可以直接返回一个固定值。

1. 首先在pox.xml文件中加入spring-cloud-starter-netflix-hystrix的起步依赖：
2. 需要在配置文件中配置打开它，在配置文件加以下代码： feign.hystrix.enabled=true
3. 只需要在FeignClient的SchedualServiceHi接口的注解中加上fallback的指定类就行了：
4. SchedualServiceHiHystric需要实现SchedualServiceHi 接口，并注入到Ioc容器中，代码如下：

```
@Component
public class SchedualServiceHiHystric implements SchedualServiceHi {
    @Override
    public String sayHiFromClientOne(String name) {
        return "sorry "+name;
    }
}
```

5.路由网关(zuul)

Zuul的主要功能是路由转发和过滤器。路由功能是微服务的一部分，比如 / api/user转发到到user服务，/api/shop转发到到shop服务。zuul默认和Ribbon结合实现了负载均衡的功能。

1. pom 中引入依赖spring-cloud-starter-netflix-zuul，spring-cloud-starter-netflix-eureka-client，spring-boot-starter-web
2. 在其入口applicaton类加上注解@EnableZuulProxy，开启zuul的功能：，@EnableEurekaClient， @EnableDiscoveryClient
3. 加上配置文件application.yml加上以下的配置代码：
首先指定服务注册中心的地址为http://localhost:8761/eureka/，服务的端口为8769，服务名为service-zuul；以/api-a/ 开头的请求都转发给service-ribbon服务；以/api-b/开头的请求都转发给service-feign服务；
4. 访问 http://localhost:8769/api-a/hi?name=forezp，http://localhost:8769/api-b/hi?name=forezp
5. zuul不仅只是路由，并且还能过滤，做一些安全验证。

```
eureka:
  client:
    serviceUrl:
      defaultZone: http://localhost:8761/eureka/
server:
  port: 8769
spring:
  application:
    name: service-zuul
zuul:
  routes:
    api-a:
      path: /api-a/**
      serviceId: service-ribbon
    api-b:
      path: /api-b/**
      serviceId: service-feign
```