

# SpringCloud 完整案例第二季

## 一、开发环境准备

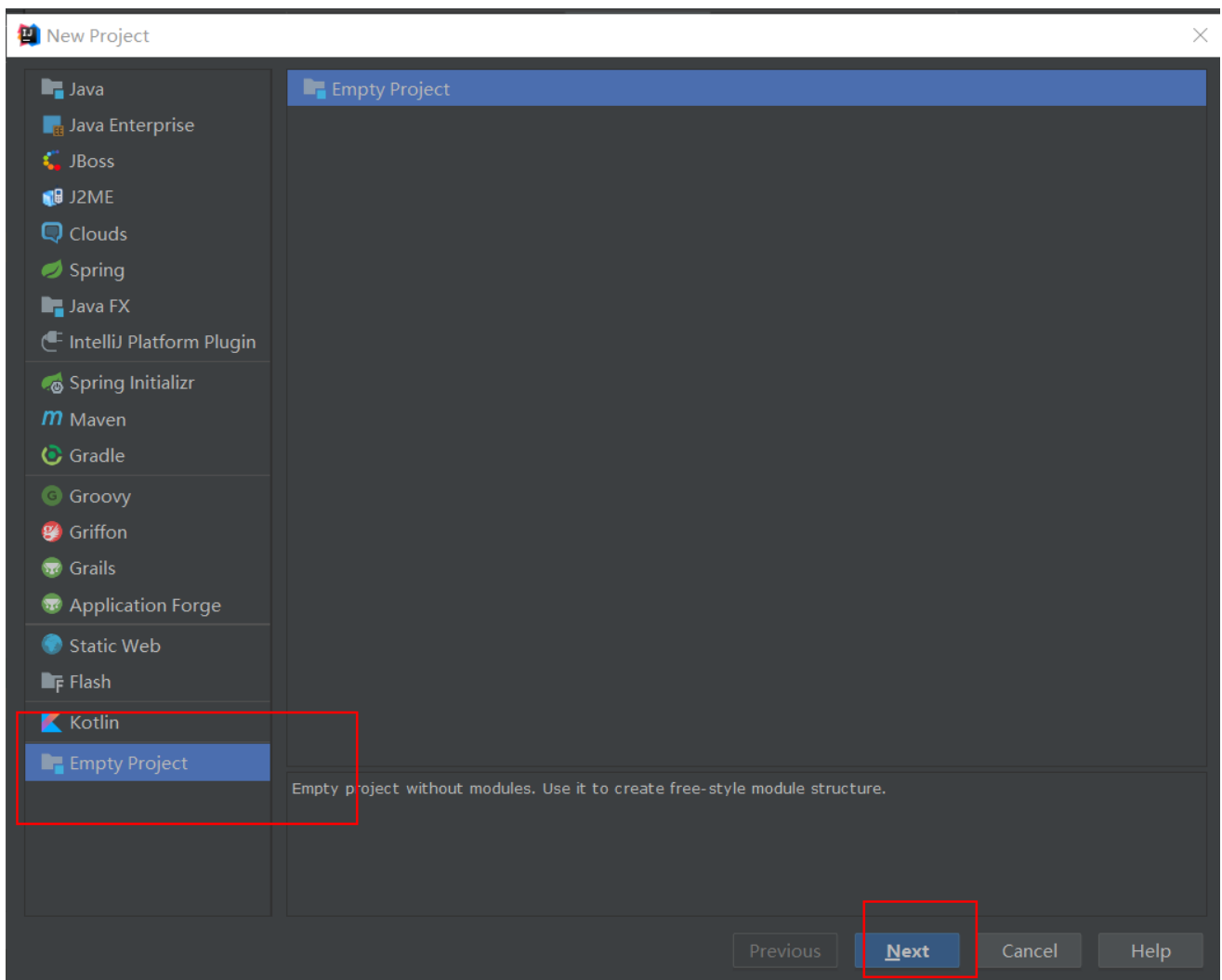
idea: IDEA ULTIMATE 2018.1

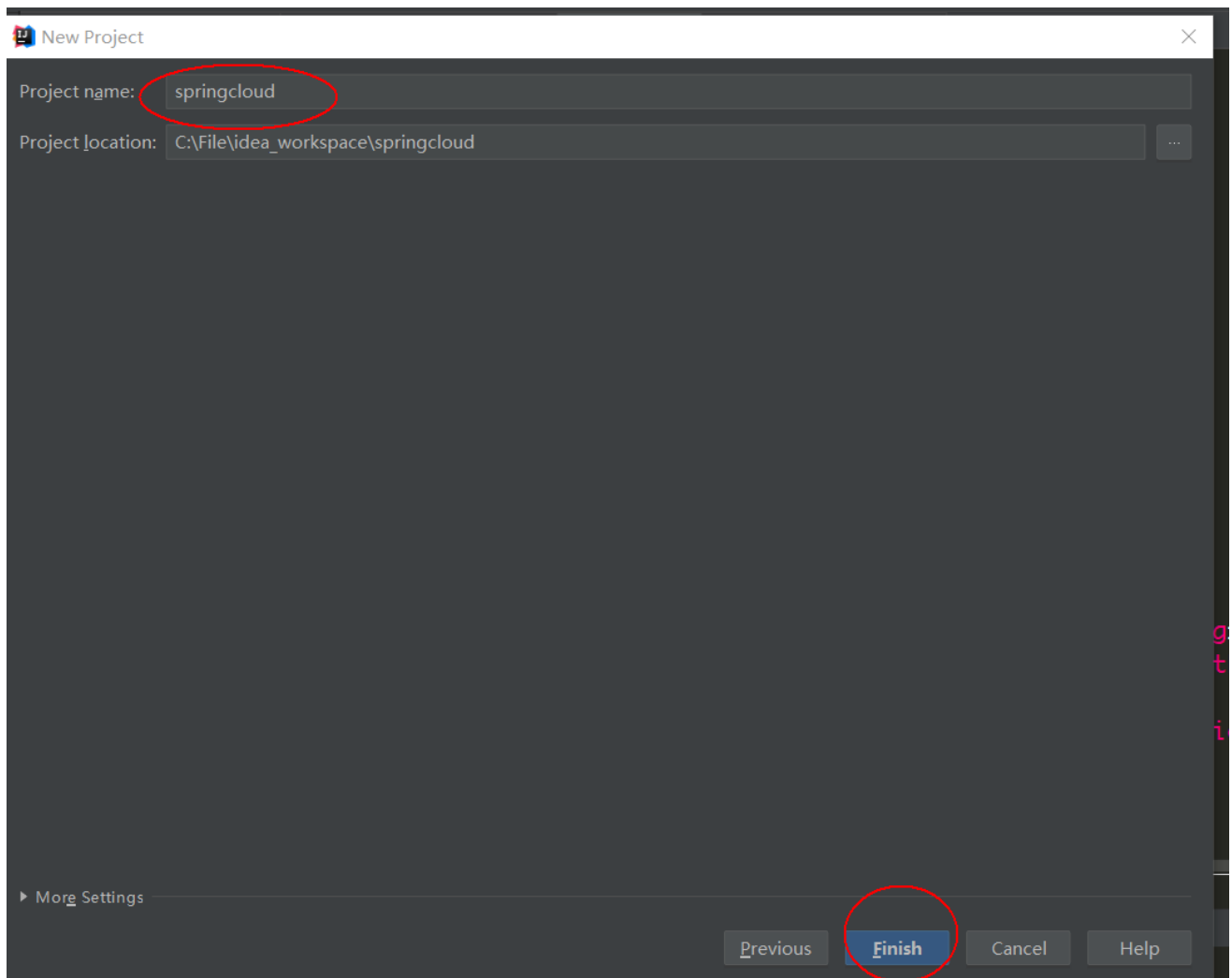
java: 1.8

maven: 3.5

## 二、创建工程（注意有坑）

### 1. 首先创建一个空工程





## 2. 创建并测试Eureka Server 的module

### 2.1 创建module

Structure

common

eureka

provider

Name: common

Sources Paths Dependencies

Language level: 8 - Lambdas, type annotations etc.

Mark as: Sources Tests Resources Test Resources Excluded

C:\File\idea\_workspace\springcloud\common

src

target

+ Add Content

C:\File\idea\_won

Source Folders

src\main\java...

Test Source Folders

src\test\java...

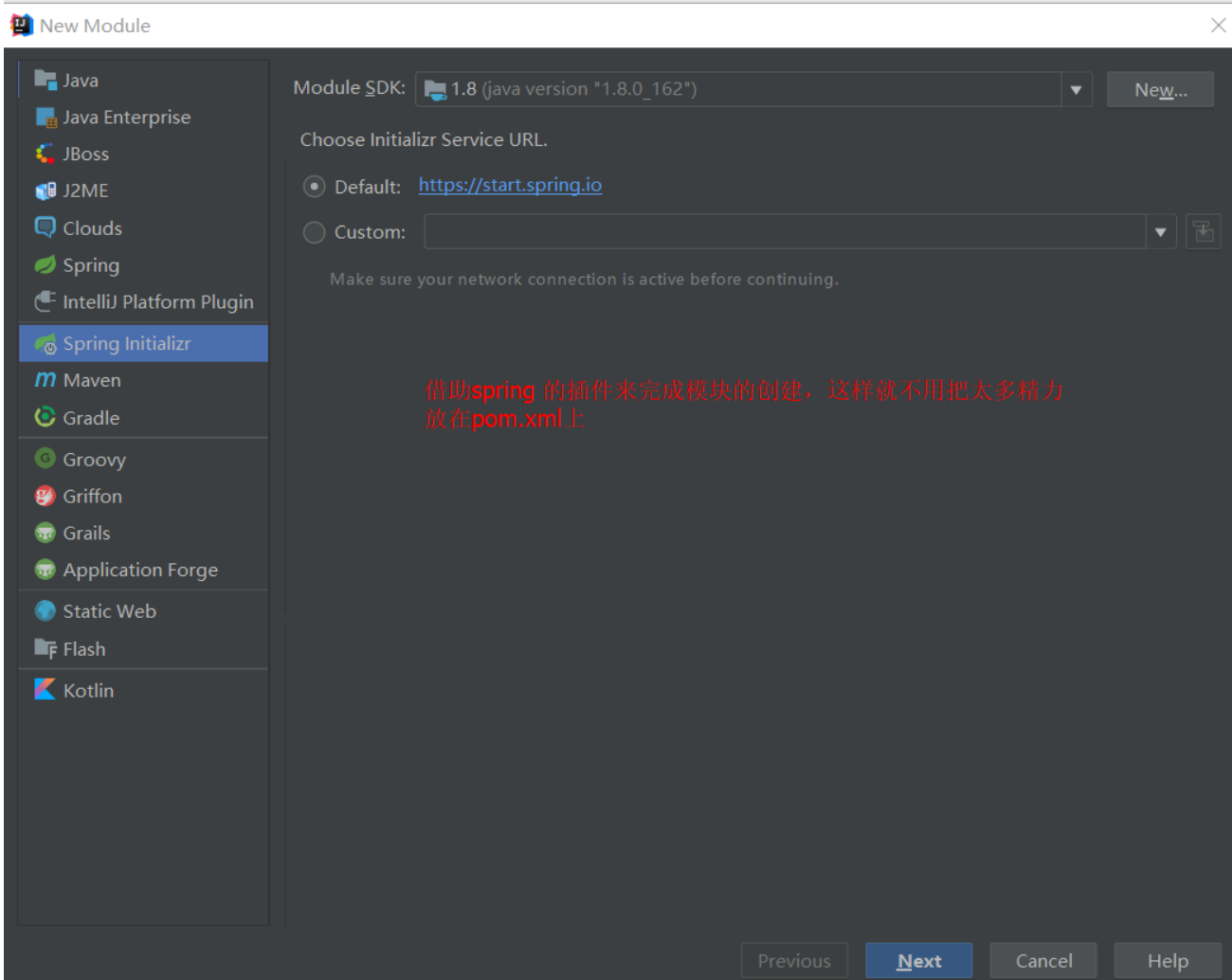
Resource Folders

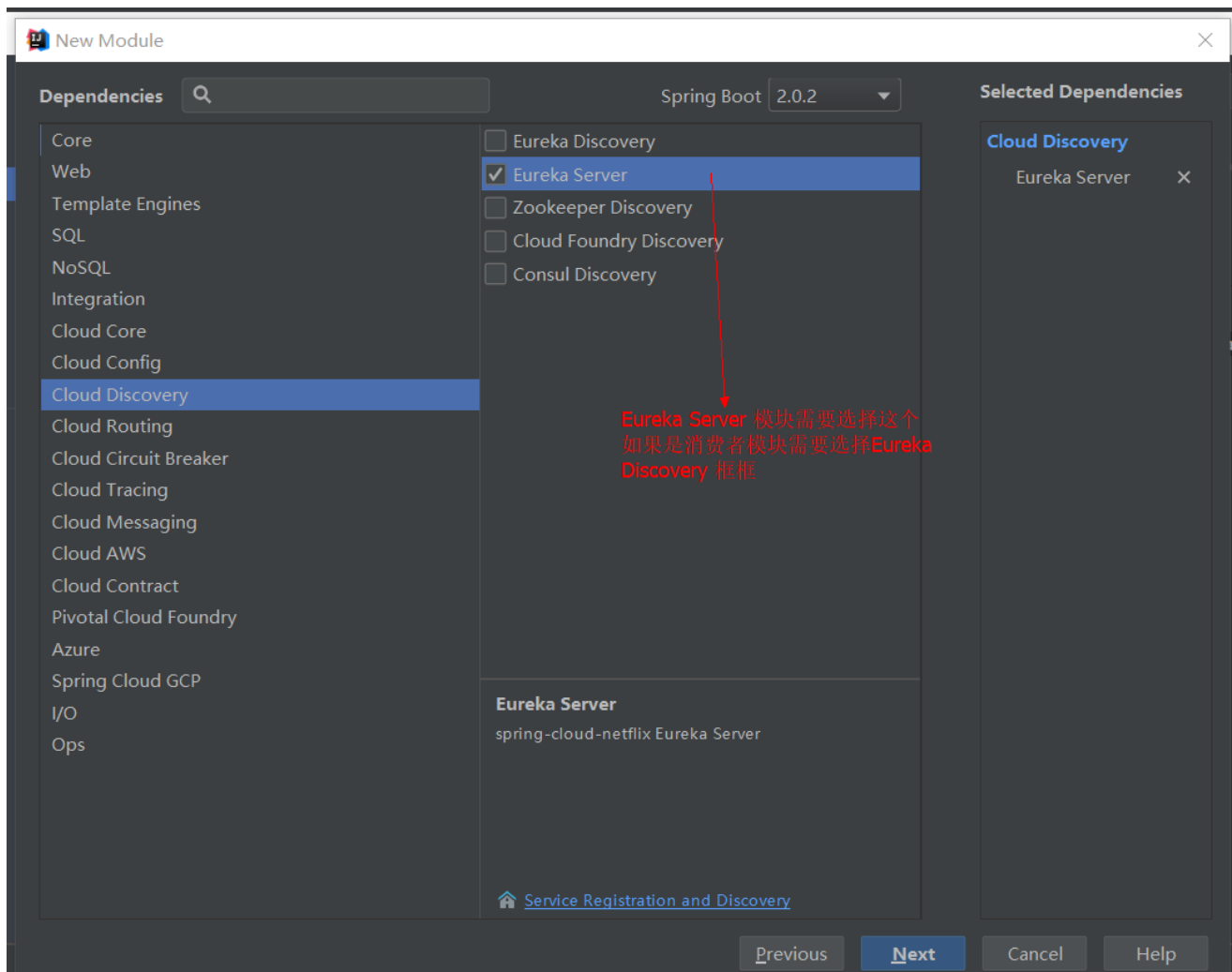
src\main\resou...

Excluded Folders

target

注意：最新版本的idea 为工程创建module的时候，最好使用这种方式，或者第一次进入空工程的时候创建模块，如果通过点击左上角的File按钮来添加module，会把新创建的module添加至当前选中的module。。。无语





## 2.2 yml

```
1 server:
2   port: 8761
3
4 eureka:
5   instance:
6     hostname: localhost
7   client:
8     registerWithEureka: false
9     fetchRegistry: false
10    serviceUrl:
11      defaultZone: http://${eureka.instance.hostname}:${server.port}/eureka/
```

## 2.3 pom.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
```

```
http://maven.apache.org/xsd/maven-4.0.0.xsd">
4   <modelVersion>4.0.0</modelVersion>
5
6   <groupId>com.cris</groupId>
7   <artifactId>eureka</artifactId>
8   <version>0.0.1-SNAPSHOT</version>
9   <packaging>jar</packaging>
10
11  <name>eureka</name>
12  <description>Demo project for Spring Boot</description>
13
14  <parent>
15    <groupId>org.springframework.boot</groupId>
16    <artifactId>spring-boot-starter-parent</artifactId>
17    <version>2.0.2.RELEASE</version>
18    <relativePath/> <!-- lookup parent from repository -->
19  </parent>
20
21  <properties>
22    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
23    <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
24    <java.version>1.8</java.version>
25    <spring-cloud.version>Finchley.BUILD-SNAPSHOT</spring-cloud.version>
26  </properties>
27
28  <dependencies>
29    <dependency>
30      <groupId>org.springframework.cloud</groupId>
31      <artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>
32    </dependency>
33
34    <dependency>
35      <groupId>org.springframework.boot</groupId>
36      <artifactId>spring-boot-starter-test</artifactId>
37      <scope>test</scope>
38    </dependency>
39  </dependencies>
40
41  <dependencyManagement>
42    <dependencies>
43      <dependency>
44        <groupId>org.springframework.cloud</groupId>
45        <artifactId>spring-cloud-dependencies</artifactId>
46        <version>${spring-cloud.version}</version>
47        <type>pom</type>
48        <scope>import</scope>
49      </dependency>
50    </dependencies>
51  </dependencyManagement>
52
53  <build>
54    <plugins>
55      <plugin>
```

```

56         <groupId>org.springframework.boot</groupId>
57         <artifactId>spring-boot-maven-plugin</artifactId>
58     </plugin>
59 </plugins>
60 </build>
61
62 <repositories>
63     <repository>
64         <id>spring-snapshots</id>
65         <name>Spring Snapshots</name>
66         <url>https://repo.spring.io/snapshot</url>
67         <snapshots>
68             <enabled>true</enabled>
69         </snapshots>
70     </repository>
71     <repository>
72         <id>spring-milestones</id>
73         <name>Spring Milestones</name>
74         <url>https://repo.spring.io/milestone</url>
75         <snapshots>
76             <enabled>false</enabled>
77         </snapshots>
78     </repository>
79 </repositories>
80
81
82 </project>

```

## 2.4 主启动类

```


1  @SpringBootApplication
2  @EnableEurekaServer
3  public class EurekaApplication {
4
5      public static void main(String[] args) {
6          SpringApplication.run(EurekaApplication.class, args);
7      }
8  }

```

## 2.5 启动测试

← → ↻ 🏠 localhost:8761

Oracle JAVA5E Hibernate Spring-SpringMVC windows环境配置 eclipse操作指南 navicat操作mysql 工具使用指南 总结了一部分口 Ma Android学习之路 我的CSDN SpringCloud

 HOME LAST 1000 SINCE STARTUP

### System Status

Environment	test	Current time	2018-05-23T16:16:22 +0800
Data center	default	Uptime	00:09
		Lease expiration enabled	false
		Renews threshold	1
		Renews (last min)	0

EMERGENCY! EUREKA MAY BE INCORRECTLY CLAIMING INSTANCES ARE UP WHEN THEY'RE NOT. RENEWALS ARE LESSER THAN THRESHOLD AND HENCE THE INSTANCES ARE NOT BEING EXPIRED JUST TO BE SAFE.

### DS Replicas

### Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
No instances available			

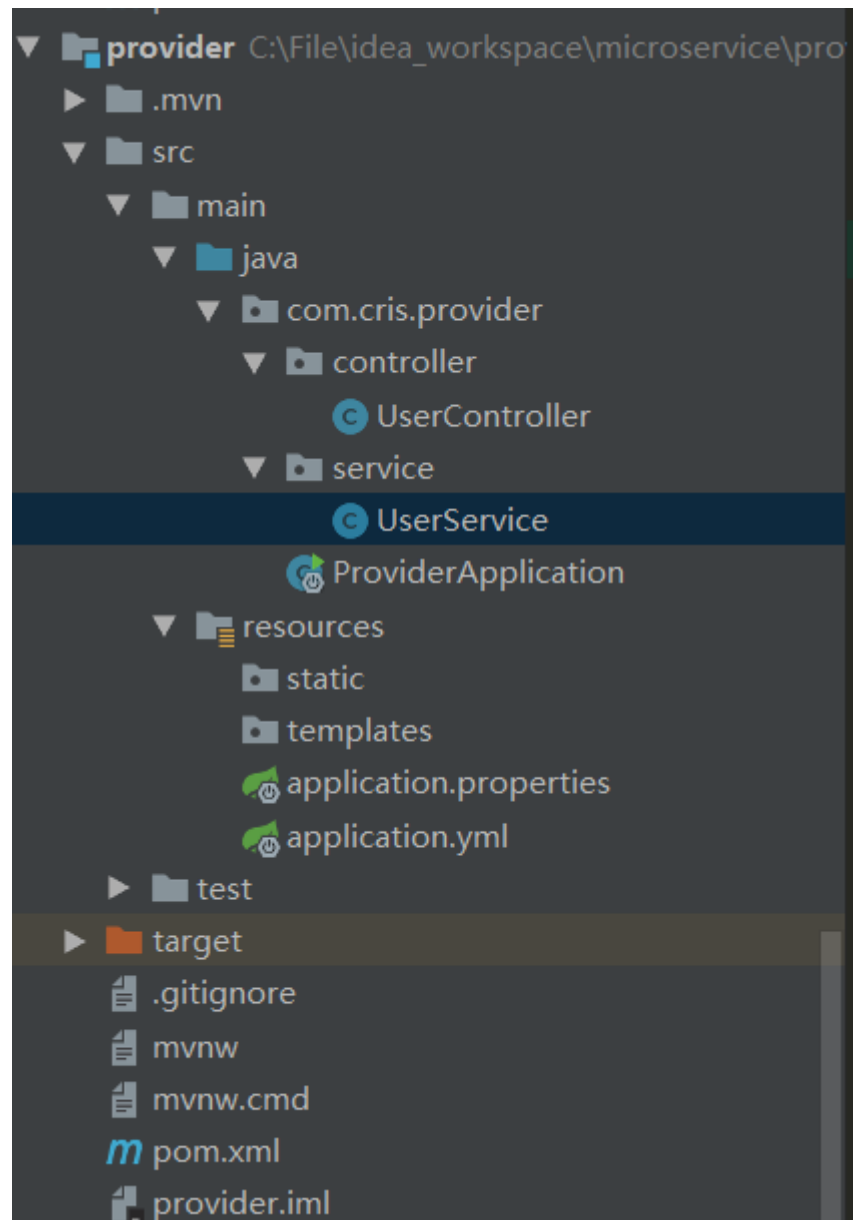
### General Info

Name	Value
total-avail-memory	401mb
environment	test

- [IDEA 安装lombok插件](#)

## 4. 创建provider 模块 (添加web和eureka discovery 依赖)





#### 4.1 application.yml

```
1 eureka:
2   client:
3     serviceUrl:
4       defaultZone: http://localhost:8761/eureka/    # 注册到指定的eureka server 服务器上
5   server:
6     port: 8762
7   spring:
8     application:
9       name: service-provider    # 服务名 (很重要)
```

#### 4.2 pom.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
```

```

2 <project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
  http://maven.apache.org/xsd/maven-4.0.0.xsd">
4     <modelVersion>4.0.0</modelVersion>
5
6     <groupId>com.cris</groupId>
7     <artifactId>provider</artifactId>
8     <version>0.0.1-SNAPSHOT</version>
9     <packaging>jar</packaging>
10
11     <name>provider</name>
12     <description>Demo project for Spring Boot</description>
13
14     <parent>
15         <groupId>org.springframework.boot</groupId>
16         <artifactId>spring-boot-starter-parent</artifactId>
17         <version>2.0.2.RELEASE</version>
18         <relativePath/> <!-- lookup parent from repository -->
19     </parent>
20
21     <properties>
22         <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
23         <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
24         <java.version>1.8</java.version>
25         <spring-cloud.version>Finchley.BUILD-SNAPSHOT</spring-cloud.version>
26     </properties>
27
28     <dependencies>
29         <dependency>
30             <groupId>org.springframework.boot</groupId>
31             <artifactId>spring-boot-starter-web</artifactId>
32         </dependency>
33         <dependency>
34             <groupId>org.springframework.cloud</groupId>
35             <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
36         </dependency>
37
38         <dependency>
39             <groupId>org.springframework.boot</groupId>
40             <artifactId>spring-boot-starter-test</artifactId>
41             <scope>test</scope>
42         </dependency>
43     </dependencies>
44
45     <dependencyManagement>
46         <dependencies>
47             <dependency>
48                 <groupId>org.springframework.cloud</groupId>
49                 <artifactId>spring-cloud-dependencies</artifactId>
50                 <version>${spring-cloud.version}</version>
51                 <type>pom</type>
52
53                 <scope>import</scope>

```

```

53         </dependency>
54     </dependencies>
55 </dependencyManagement>
56
57 <build>
58     <plugins>
59         <plugin>
60             <groupId>org.springframework.boot</groupId>
61             <artifactId>spring-boot-maven-plugin</artifactId>
62         </plugin>
63     </plugins>
64 </build>
65
66 <repositories>
67     <repository>
68         <id>spring-snapshots</id>
69         <name>Spring Snapshots</name>
70         <url>https://repo.spring.io/snapshot</url>
71         <snapshots>
72             <enabled>true</enabled>
73         </snapshots>
74     </repository>
75     <repository>
76         <id>spring-milestones</id>
77         <name>Spring Milestones</name>
78         <url>https://repo.spring.io/milestone</url>
79         <snapshots>
80             <enabled>false</enabled>
81         </snapshots>
82     </repository>
83 </repositories>
84
85
86 </project>

```

### 4.3 service

```

1  /**
2   * @ClassName UserService
3   * @Description TODO
4   * @Author zc-cris
5   * @Version 1.0
6   */
7  @Service
8  public class UserService {
9      public String get(){
10         return "cris";
11     }
12 }

```

### 4.4 controller

```

1  /**
2   * @ClassName UserController
3   * @Description TODO
4   * @Author zc-cris
5   * @Version 1.0
6   **/
7  @RestController
8  public class UserController {
9
10     @Autowired
11     private UserService service;
12
13     @Value("${server.port}")
14     private String port;
15
16     @GetMapping("/provider/get")
17     public String get(){
18         return service.get() + port;
19     }
20 }

```

## 4.5 主启动类

```

1  @SpringBootApplication
2  @EnableEurekaClient
3  public class ProviderApplication {
4
5      public static void main(String[] args) {
6          SpringApplication.run(ProviderApplication.class, args);
7      }
8  }

```

## 4.5 启动并测试

The screenshot shows the Spring Eureka web interface at localhost:8761. The top navigation bar includes links for HOME and LAST 1000 SINCE STARTUP. The main content area is titled "System Status" and contains two tables. The left table shows Environment: test and Data center: default. The right table shows Current time: 2018-05-23T16:49:26 +0800, Uptime: 00:42, Lease expiration enabled: false, Renewals threshold: 3, and Renewals (last min): 2. Below these tables, a red warning message states: "EMERGENCY! EUREKA MAY BE INCORRECTLY CLAIMING INSTANCES ARE UP WHEN THEY'RE NOT. RENEWALS ARE LESSER THAN THRESHOLD. INSTANCES ARE NOT BEING EXPIRED JUST TO BE SAFE." Under the "DS Replicas" section, the "Instances currently registered with Eureka" table is displayed. The table has columns for Application, AMIs, Availability Zones, and Status. The first row shows "SERVICE-PROVIDER" (circled in red) with AMIs: n/a (1), Availability Zones: (1), and Status: UP (1) - localhost:service-provider:8762. A red arrow points from the text "服务模块名称" to the "SERVICE-PROVIDER" entry.

Environment	test	Current time	2018-05-23T16:49:26 +0800
Data center	default	Uptime	00:42
		Lease expiration enabled	false
		Renews threshold	3
		Renews (last min)	2

EMERGENCY! EUREKA MAY BE INCORRECTLY CLAIMING INSTANCES ARE UP WHEN THEY'RE NOT. RENEWALS ARE LESSER THAN THRESHOLD. INSTANCES ARE NOT BEING EXPIRED JUST TO BE SAFE.

DS Replicas

Instances currently registered with Eureka

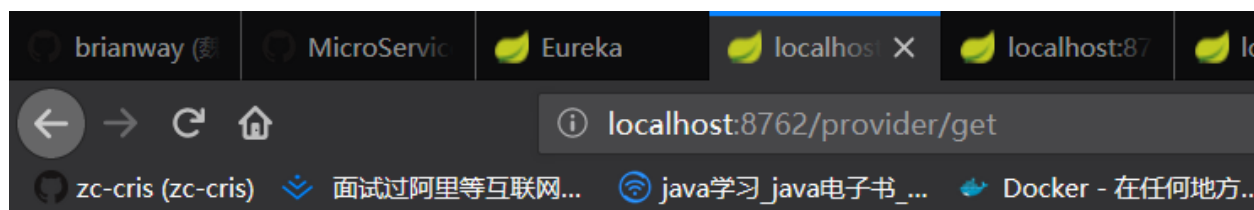
Application	AMIs	Availability Zones	Status
SERVICE-PROVIDER	n/a (1)	(1)	UP (1) - localhost:service-provider:8762

## 5. 快速启动多个不同端口的Provider 模块

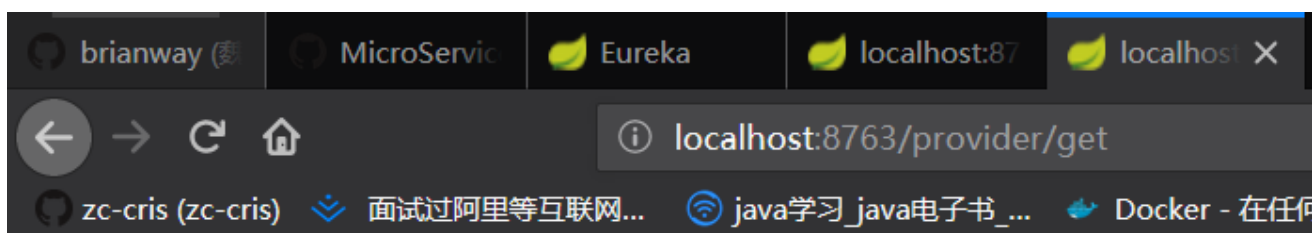
The screenshot shows the IntelliJ IDEA Run/Debug Configurations dialog for the "ProviderApplication" configuration. The "Configuration" tab is selected. The "Main class" is set to "com.cris.provider.ProviderApplication". The "VM options" field contains "-Dserver.port=8763". The "Program arguments" field is empty. The "Working directory" is set to "C:\File\idea\_workspace\springcloud". The "Environment variables" field is empty. The "Use classpath of module" is set to "provider". The "JRE" is set to "Default (1.8 - SDK of 'provider' module)". The "Shorten command line" is set to "user-local default: none - java [options] classname [args]". The "Before launch" section shows "Build" as the first step. A red arrow points from the text "点击Edit Configuration" to the "Edit Configuration" button in the top right corner of the dialog. Below the arrow, red text says: "然后输入这行参数，端口号不能重复，即可开启多个不同的服务模块".

点击Edit Configuration  
然后输入这行参数，端口号不能重复，即可开启多个不同的服务模块

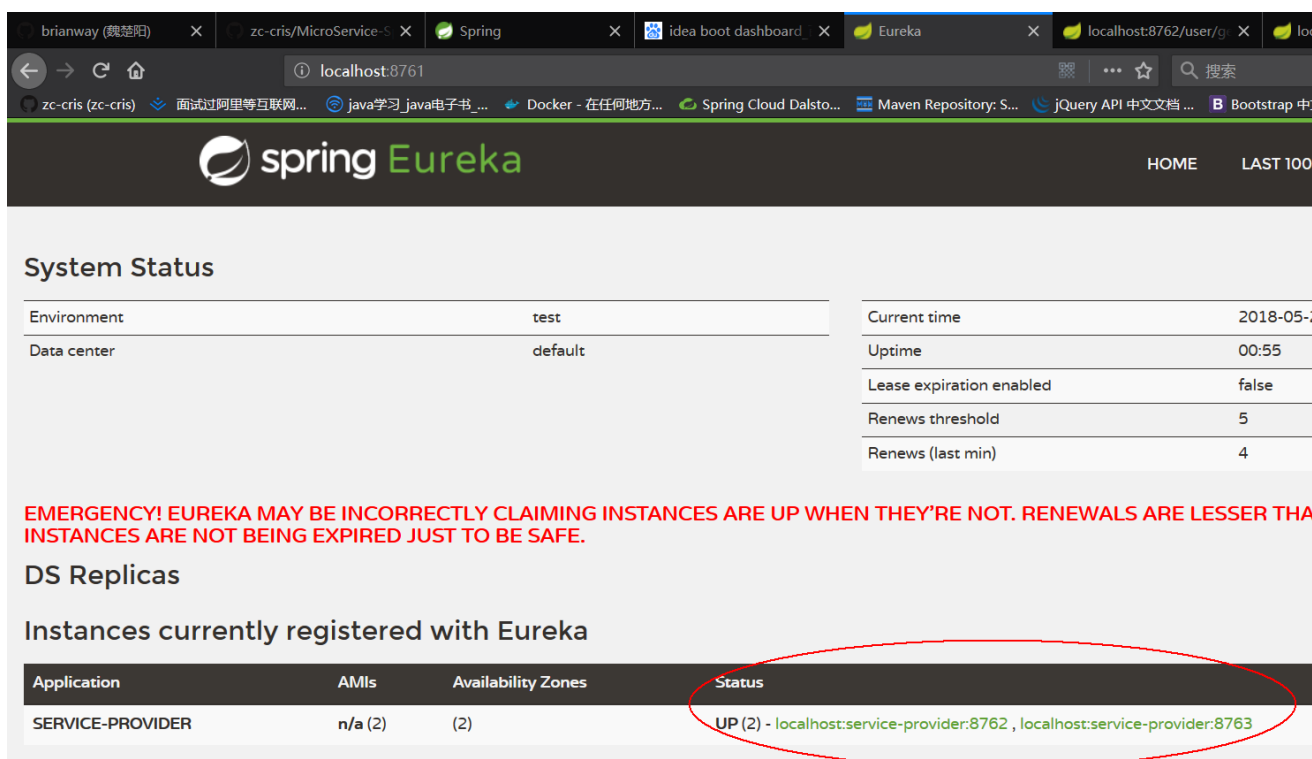
- 启动并测试



cris8762



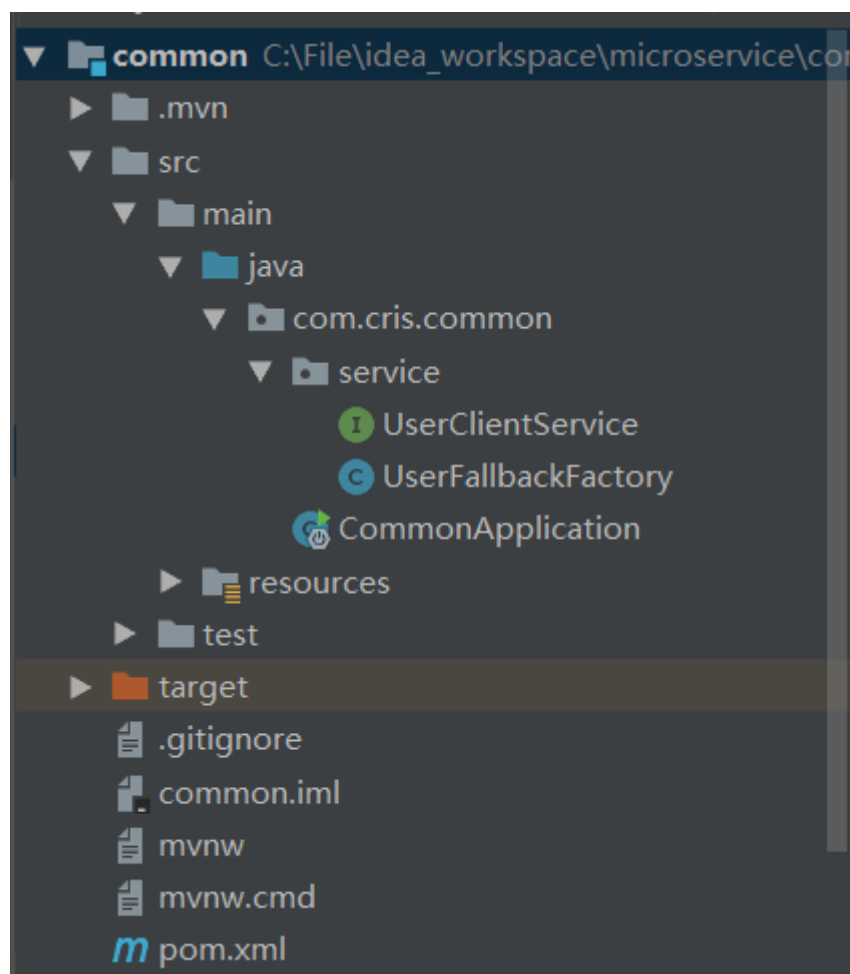
cris8763



## 6. 引入Feign + hystrix组件的公共模块

注意：这里不建议使用Ribbon 组件+RestTemplate 的方式来构建消费模块，原因在第一季已经阐述，这里直接上手Feign 组件以及hystrix

## 6.1 创建公共模块（添加Feign 依赖）



## 6.2 pom.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
5     http://maven.apache.org/xsd/maven-4.0.0.xsd">
6     <modelVersion>4.0.0</modelVersion>
7
8     <groupId>com.cris</groupId>
9     <artifactId>common</artifactId>
10    <version>0.0.1-SNAPSHOT</version>
11    <packaging>jar</packaging>
12
13    <name>common</name>
14    <description>Demo project for Spring Boot</description>
15
16    <parent>
17        <groupId>org.springframework.boot</groupId>
18        <artifactId>spring-boot-starter-parent</artifactId>
```

```
17     <version>2.0.2.RELEASE</version>
18     <relativePath/> <!-- lookup parent from repository -->
19 </parent>
20
21 <properties>
22     <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
23     <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
24     <java.version>1.8</java.version>
25     <spring-cloud.version>Finchley.BUILD-SNAPSHOT</spring-cloud.version>
26 </properties>
27
28 <dependencies>
29     <dependency>
30         <groupId>org.springframework.cloud</groupId>
31         <artifactId>spring-cloud-starter-openfeign</artifactId>
32     </dependency>
33
34     <dependency>
35         <groupId>org.springframework.boot</groupId>
36         <artifactId>spring-boot-starter-test</artifactId>
37         <scope>test</scope>
38     </dependency>
39 </dependencies>
40
41 <dependencyManagement>
42     <dependencies>
43         <dependency>
44             <groupId>org.springframework.cloud</groupId>
45             <artifactId>spring-cloud-dependencies</artifactId>
46             <version>${spring-cloud.version}</version>
47             <type>pom</type>
48             <scope>import</scope>
49         </dependency>
50     </dependencies>
51 </dependencyManagement>
52
53 <build>
54     <plugins>
55         <plugin>
56             <groupId>org.springframework.boot</groupId>
57             <artifactId>spring-boot-maven-plugin</artifactId>
58         </plugin>
59     </plugins>
60 </build>
61
62 <repositories>
63     <repository>
64         <id>spring-snapshots</id>
65         <name>Spring Snapshots</name>
66         <url>https://repo.spring.io/snapshot</url>
67         <snapshots>
68             <enabled>true</enabled>
69
70         </snapshots>
```



```

70     </repository>
71     <repository>
72         <id>spring-milestones</id>
73         <name>Spring Milestones</name>
74         <url>https://repo.spring.io/milestone</url>
75         <snapshots>
76             <enabled>false</enabled>
77         </snapshots>
78     </repository>
79 </repositories>
80
81
82 </project>
83

```

- 注意，这里创建公共模块的时候直接使用spring initial工具创建，其他的模块（消费模块，生产模块）需要引用公共模块直接导入GAV 即可，而不是传统意义上的maven模块

## 6.2 新建映射到服务模块controller 的client并且进行hystrix 处理

```

1  /**
2   * @ClassName UserClientService
3   * @Description TODO
4   * @Author zc-cris
5   * @Version 1.0
6   */
7  @FeignClient(value = "SERVICE-PROVIDER", fallbackFactory = UserFallbackFactory.class)
8  public interface UserClientService {
9
10     @GetMapping("/provider/get")
11     public String get();
12
13 }

```

## 6.3 hystrix 处理逻辑(AOP)

```

1  /**
2   * @ClassName UserFallbackFactory
3   * @Description TODO
4   * @Author zc-cris
5   * @Version 1.0
6   */
7  @Component
8  public class UserFallbackFactory implements FallbackFactory<UserClientService> {
9
10     @Override
11     public UserClientService create(Throwable throwable) {
12         return new UserClientService() {
13             @Override
14             public String get() {
15
16                 return "当前服务模块不可用";
17             }
18         };
19     }
20 }

```

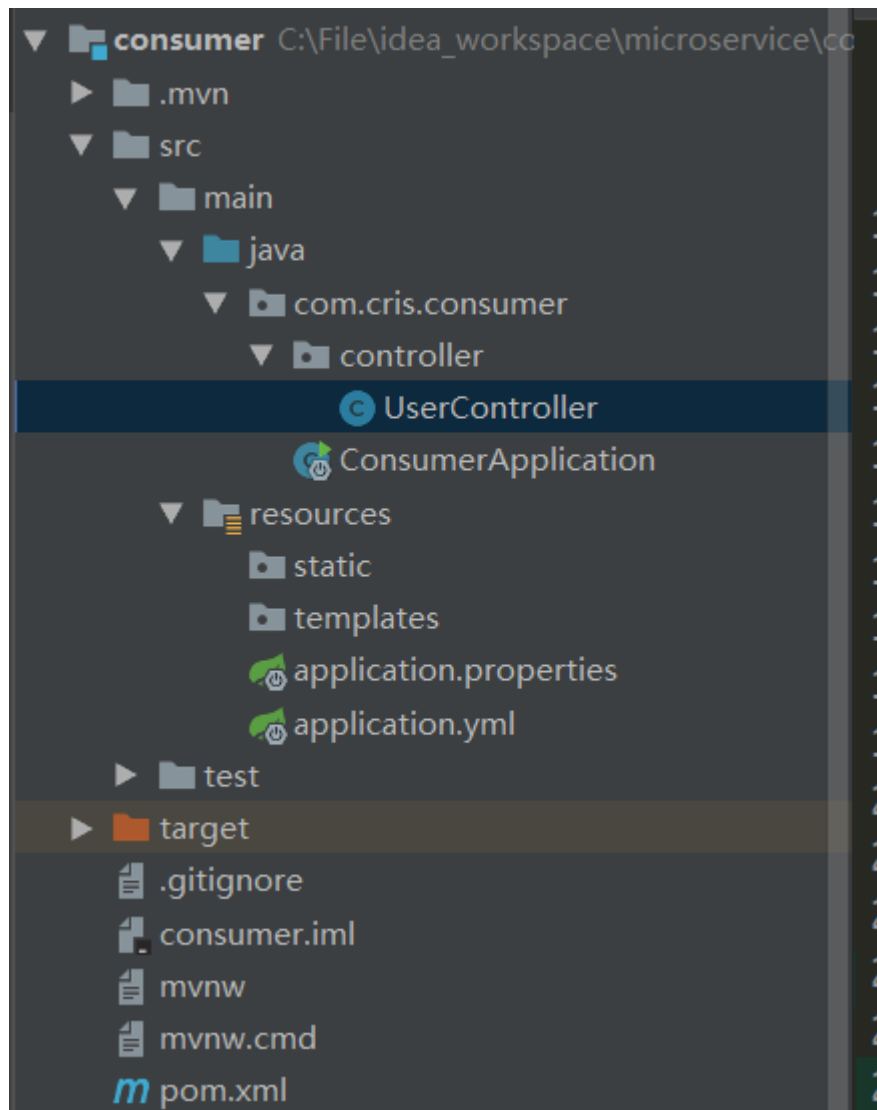
```
16     }  
17     };  
18 }  
19 }
```

## 6.4 主启动类

```
1 @SpringBootApplication  
2 public class CommonApplication {  
3  
4     public static void main(String[] args) {  
5         SpringApplication.run(CommonApplication.class, args);  
6     }  
7 }
```

## 6.5 mvn install

## 7. 新建消费模块 (添加Feign, web, eureka discovery 依赖)



## 7.1 pom.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
5   http://maven.apache.org/xsd/maven-4.0.0.xsd">
6   <modelVersion>4.0.0</modelVersion>
7
8   <groupId>com.cris</groupId>
9   <artifactId>consumer</artifactId>
10  <version>0.0.1-SNAPSHOT</version>
11  <packaging>jar</packaging>
12
13  <name>consumer</name>
14  <description>Demo project for Spring Boot</description>
15
16  <parent>
17    <groupId>org.springframework.boot</groupId>
18    <artifactId>spring-boot-starter-parent</artifactId>
```

```

17     <version>2.0.2.RELEASE</version>
18     <relativePath/> <!-- lookup parent from repository -->
19 </parent>
20
21 <properties>
22     <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
23     <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
24     <java.version>1.8</java.version>
25     <spring-cloud.version>Finchley.BUILD-SNAPSHOT</spring-cloud.version>
26 </properties>
27
28 <dependencies>
29     <dependency>
30         <groupId>com.cris</groupId>
31         <artifactId>common</artifactId>
32         <version>0.0.1-SNAPSHOT</version>
33     </dependency>
34     <dependency>
35         <groupId>org.springframework.boot</groupId>
36         <artifactId>spring-boot-starter-web</artifactId>
37     </dependency>
38     <dependency>
39         <groupId>org.springframework.cloud</groupId>
40         <artifactId>spring-cloud-starter-openfeign</artifactId>
41     </dependency>
42     <dependency>
43         <groupId>org.springframework.cloud</groupId>
44         <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
45     </dependency>
46
47     <dependency>
48         <groupId>org.springframework.boot</groupId>
49         <artifactId>spring-boot-starter-test</artifactId>
50         <scope>test</scope>
51     </dependency>
52 </dependencies>
53
54 <dependencyManagement>
55     <dependencies>
56         <dependency>
57             <groupId>org.springframework.cloud</groupId>
58             <artifactId>spring-cloud-dependencies</artifactId>
59             <version>${spring-cloud.version}</version>
60             <type>pom</type>
61             <scope>import</scope>
62         </dependency>
63     </dependencies>
64 </dependencyManagement>
65
66 <build>
67     <plugins>
68         <plugin>
69
            <groupId>org.springframework.boot</groupId>

```

```

70         <artifactId>spring-boot-maven-plugin</artifactId>
71     </plugin>
72 </plugins>
73 </build>
74
75 <repositories>
76     <repository>
77         <id>spring-snapshots</id>
78         <name>Spring Snapshots</name>
79         <url>https://repo.spring.io/snapshot</url>
80         <snapshots>
81             <enabled>true</enabled>
82         </snapshots>
83     </repository>
84     <repository>
85         <id>spring-milestones</id>
86         <name>Spring Milestones</name>
87         <url>https://repo.spring.io/milestone</url>
88         <snapshots>
89             <enabled>false</enabled>
90         </snapshots>
91     </repository>
92 </repositories>
93
94
95 </project>

```

## 7.2 application.yml

```

1  eureka:
2    client:
3      register-with-eureka: false  # 试试不把自己注册到eureka server 中，是否可以拿到eureka
server上的服务信息列表（答案是可以的）
4      serviceUrl:
5        defaultZone: http://localhost:8761/eureka/
6  server:
7    port: 8765
8  spring:
9    application:
10     name: service-feign
11  feign:
12    hystrix:
13     enabled: true  // 支持熔断处理

```

## 7.3 controller

```

1  /**
2   * @ClassName UserController
3   * @Description TODO
4   * @Author zc-cris
5   * @Version 1.0

```

```

6  /**
7  @RestController
8  public class UserController {
9
10     @Autowired
11     private UserClientService service;
12
13     @GetMapping("/consumer/user/get")
14     public String get(){
15         return service.get();
16     }
17
18     @GetMapping("/consumer/get")
19     public String getUser(){
20         return "cris";
21     }
22
23 }

```

## 7.4 主启动类

```

1  @SpringBootApplication
2  @EnableEurekaClient
3  @EnableFeignClients(basePackages = {"com.cris"})
4  @ComponentScan("com.cris")
5  public class ConsumerApplication {
6
7      public static void main(String[] args) {
8          SpringApplication.run(ConsumerApplication.class, args);
9      }
10 }

```

## 7.5 启动并测试

- 先启动eureka server，然后启动我们集成了feign 组件的消费模块

brianway (魏楚阳) x MicroService-Spr... Eureka x localhost:8762/us... localhost:8763/us... localhost:8764/co... 找不到服务器 x Spring

localhost:8761

spring Eureka

HOME LAST 1000 SING

### System Status

Environment	test	Current time	2018-05-23T20:
Data center	default	Uptime	00:05
		Lease expiration enabled	false
		Renews threshold	5
		Renews (last min)	4

### DS Replicas

#### Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
SERVICE-PROVIDER	n/a (2)	(2)	UP (2) - localhost:service-provider:8762 , localhost:service-provider:8763

General Info

注意，这里只有我们的两个服务模块，没有集成feign组件的消费模块

brianway (魏) MicroService Eureka localhost:87 localhost:87 localhost:87

localhost:8765/consumer/user/get

zc-cris (zc-cris) 面试过阿里等互联网... java学习 java电子书... Docker - 在任何地方... Spring C

当前服务模块不可用

可以看到我们的hystrix 处理单元起作用了，因为服务模块并没有启动，模拟降级或者熔断处理

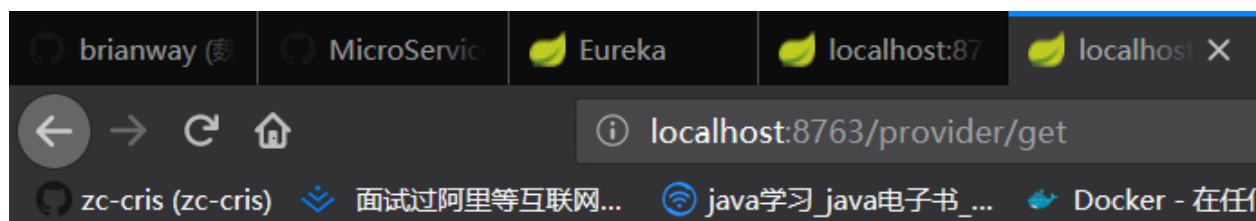
- 启动我们的服务模块（多端口）再测试

brianway (魏) MicroService Eureka localhost X localhost

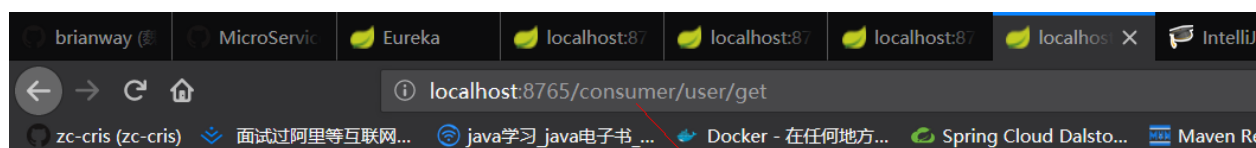
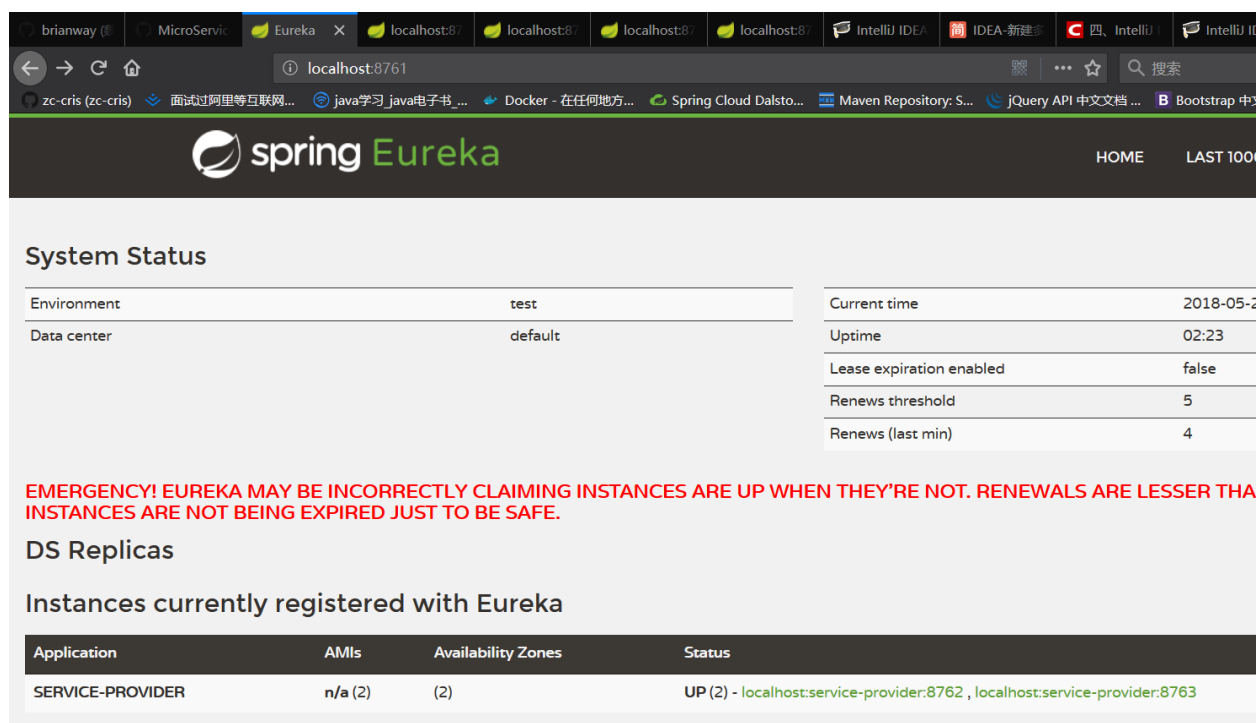
localhost:8762/provider/get

zc-cris (zc-cris) 面试过阿里等互联网... java学习 java电子书... D

cris8762



cris8763



cris8763

轮流返回两个服务模块的处理内容，说明如下：

1. feign 组件集成了ribbon 组件进行LB（负载均衡）
2. Ribbon 组件采用的是轮询（默认）
3. 如果要修改LB算法可以参考我的第一季文档

3. 及其方便hystrix 熔断处理（AOP）

注意，我们这里通过公共模块的Feign 代理接口进行访问服务模块的业务

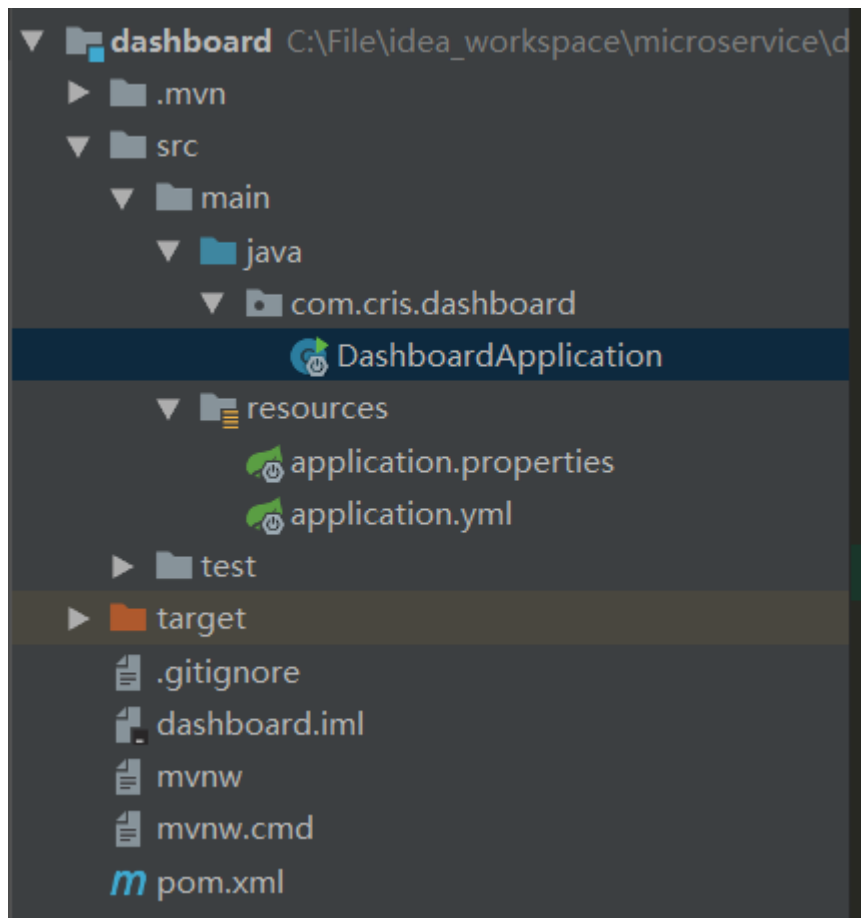
简单说：就是消费模块注入公共模块的Feign 代理接口，这个代理接口指向的是服务模块的controller 请求路径（然后调用服务模块的service服务）

好处：

1. 面向feign 接口编程；2. 又实现了服务模块的代理接口公共化，及其方便调用

## 8. 新建监控模块（引入hystrix Dashboard依赖）





## 8.1 pom.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
5     http://maven.apache.org/xsd/maven-4.0.0.xsd">
6     <modelVersion>4.0.0</modelVersion>
7
8     <groupId>com.cris</groupId>
9     <artifactId>dashboard</artifactId>
10    <version>0.0.1-SNAPSHOT</version>
11    <packaging>jar</packaging>
12
13    <name>dashboard</name>
14    <description>Demo project for Spring Boot</description>
15
16    <parent>
17        <groupId>org.springframework.boot</groupId>
18        <artifactId>spring-boot-starter-parent</artifactId>
19        <version>2.0.2.RELEASE</version>
20        <relativePath/> <!-- lookup parent from repository -->
21    </parent>
22
23    <properties>
```

```

22     <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
23     <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
24     <java.version>1.8</java.version>
25     <spring-cloud.version>Finchley.BUILD-SNAPSHOT</spring-cloud.version>
26 </properties>
27
28 <dependencies>
29
30     <dependency>
31         <groupId>org.springframework.cloud</groupId>
32         <artifactId>spring-cloud-starter-netflix-hystrix</artifactId>
33     </dependency>
34     <dependency>
35         <groupId>org.springframework.cloud</groupId>
36         <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
37     </dependency>
38     <dependency>
39         <groupId>org.springframework.boot</groupId>
40         <artifactId>spring-boot-starter-web</artifactId>
41     </dependency>
42     <dependency>
43         <groupId>org.springframework.cloud</groupId>
44         <artifactId>spring-cloud-starter-netflix-hystrix-dashboard</artifactId>
45     </dependency>
46
47     <dependency>
48         <groupId>org.springframework.boot</groupId>
49         <artifactId>spring-boot-starter-test</artifactId>
50         <scope>test</scope>
51     </dependency>
52 </dependencies>
53
54 <dependencyManagement>
55     <dependencies>
56         <dependency>
57             <groupId>org.springframework.cloud</groupId>
58             <artifactId>spring-cloud-dependencies</artifactId>
59             <version>${spring-cloud.version}</version>
60             <type>pom</type>
61             <scope>import</scope>
62         </dependency>
63     </dependencies>
64 </dependencyManagement>
65
66 <build>
67     <plugins>
68         <plugin>
69             <groupId>org.springframework.boot</groupId>
70             <artifactId>spring-boot-maven-plugin</artifactId>
71         </plugin>
72     </plugins>
73 </build>

```

```

75     <repositories>
76         <repository>
77             <id>spring-snapshots</id>
78             <name>Spring Snapshots</name>
79             <url>https://repo.spring.io/snapshot</url>
80             <snapshots>
81                 <enabled>true</enabled>
82             </snapshots>
83         </repository>
84         <repository>
85             <id>spring-milestones</id>
86             <name>Spring Milestones</name>
87             <url>https://repo.spring.io/milestone</url>
88             <snapshots>
89                 <enabled>false</enabled>
90             </snapshots>
91         </repository>
92     </repositories>
93
94
95 </project>

```

## 8.2 主启动类

```

1  @SpringBootApplication
2  @EnableHystrixDashboard
3  @EnableDiscoveryClient // 其实和@EnableEurekaClient 功能相似，但是多用于其他服务中心
4  public class DashboardApplication {
5
6      public static void main(String[] args) {
7          SpringApplication.run(DashboardApplication.class, args);
8      }
9  }

```

## 8.3 yml文件

```

1  server:
2      port: 8766
3  spring:
4      application:
5          name: service-dashboard

```

## 8.4 启动并自测



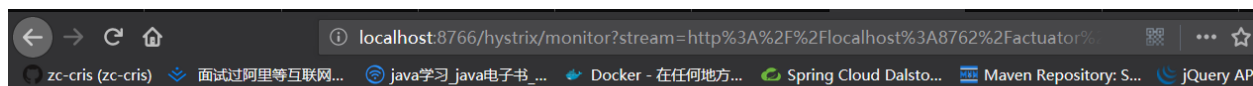
- 我的解决方案是：

服务模块的ym1文件

```
1 eureka:
2   client:
3     serviceUrl:
4       defaultZone: http://localhost:8761/eureka/
5 server:
6   port: 8762
7 spring:
8   application:
9     name: service-provider
10
11 management:
12   endpoints:
13     web:
14       exposure:
15         include: "*"

```

- 然后如果出现进入dashboard 管理界面一直loading 的情况, [请参考这篇文章](#)
- 但是问题又来了, 因为我是把熔断, 降级处理全部抽取出来放在公共的feign 代理接口中处理的, 所以我配置完以后还是一直loading... --!



## Hystrix Stream: test

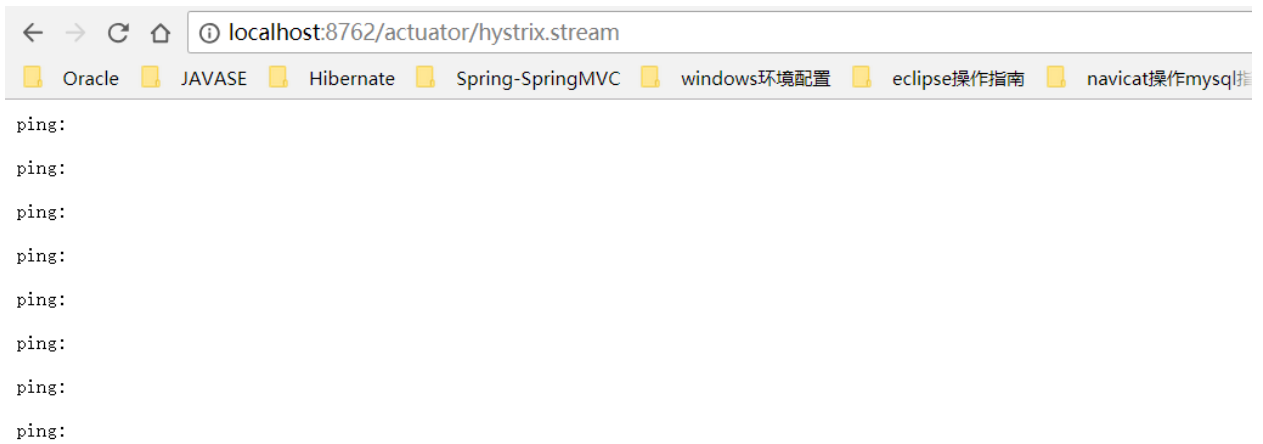
**Circuit** Sort: [Error then Volume](#) | [Alphabetical](#) | [Volume](#) | [Error](#) | [Mean](#) | [Median](#) | [90](#) | [99](#) | [99.5](#)

一直loading。。。。

Loading ...

**Thread Pools** Sort: [Alphabetical](#) | [Volume](#) |

Loading ...



- 解决方案：其实很简单，直接修改服务模块的服务方法，将其标记为@HystrixCommand(), 因为我们的 hystrix dashboard 不是所有业务方法都会监控的，只有标记了需要熔断或是降级处理的方法才会被监控起来

```
1  /**
2   * @ClassName UserController
3   * @Description TODO
4   * @Author zc-cris
5   * @Version 1.0
6   */
7  @RestController
8  public class UserController {
9
10     @Autowired
11     private UserService service;
12
13     @Value("${server.port}")
14     private String port;
15
16     @GetMapping("/provider/get")
17     @HystrixCommand()
18     public String get(){
19         return service.get() + port;
20     }
21 }
```

- 然后修改provider 模块的主启动类

```

1  @EnableEurekaClient // 这个服务模块启动后自动注册进Eureka Server
2  @SpringBootApplication
3  @EnableCircuitBreaker // 对hystrix 熔断器的支持
4  @EnableHystrix
5  public class ProviderApplication {
6
7      public static void main(String[] args) {
8          SpringApplication.run(ProviderApplication.class, args);
9      }
10 }

```

- 再修改provider 模块的pom.xml

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <project xmlns="http://maven.apache.org/POM/4.0.0"
3      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4      xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
5      http://maven.apache.org/xsd/maven-4.0.0.xsd">
6      <modelVersion>4.0.0</modelVersion>
7
8      <groupId>com.cris</groupId>
9      <artifactId>provider</artifactId>
10     <version>0.0.1-SNAPSHOT</version>
11     <packaging>jar</packaging>
12
13     <name>provider</name>
14     <description>Demo project for Spring Boot</description>
15
16     <parent>
17         <groupId>org.springframework.boot</groupId>
18         <artifactId>spring-boot-starter-parent</artifactId>
19         <version>2.0.2.RELEASE</version>
20         <relativePath/> <!-- lookup parent from repository -->
21     </parent>
22
23     <properties>
24         <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
25         <project.reporting.outputEncoding>UTF-
26     8</project.reporting.outputEncoding>
27         <java.version>1.8</java.version>
28         <spring-cloud.version>Finchley.BUILD-SNAPSHOT</spring-cloud.version>
29     </properties>
30
31     <dependencies>
32         <dependency>
33             <groupId>org.springframework.boot</groupId>
34             <artifactId>spring-boot-starter-web</artifactId>
35         </dependency>
36         <dependency>
37             <groupId>org.springframework.cloud</groupId>
38             <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
39         </dependency>

```

```
1      <!-- 引入hystrix 熔断器组件 -->
2      <dependency>
3          <groupId>org.springframework.cloud</groupId>
4          <artifactId>spring-cloud-starter-netflix-hystrix</artifactId>
5      </dependency>
6
7      <dependency>
8          <groupId>org.springframework.boot</groupId>
9          <artifactId>spring-boot-starter-web</artifactId>
10     </dependency>
11
12     <dependency>
13         <groupId>org.springframework.boot</groupId>
14         <artifactId>spring-boot-starter-test</artifactId>
15     </dependency>
16     <!-- actuator 监控信息完善 -->
17     <dependency>
18         <groupId>org.springframework.boot</groupId>
19         <artifactId>spring-boot-starter-actuator</artifactId>
20     </dependency>
21
22 </dependencies>
23
24 <dependencyManagement>
25     <dependencies>
26         <dependency>
27             <groupId>org.springframework.cloud</groupId>
28             <artifactId>spring-cloud-dependencies</artifactId>
29             <version>${spring-cloud.version}</version>
30             <type>pom</type>
31             <scope>import</scope>
32         </dependency>
33     </dependencies>
34 </dependencyManagement>
35
36 <build>
37     <plugins>
38         <plugin>
39             <groupId>org.springframework.boot</groupId>
40             <artifactId>spring-boot-maven-plugin</artifactId>
41         </plugin>
42     </plugins>
43 </build>
44
45 <repositories>
46     <repository>
47         <id>spring-snapshots</id>
48         <name>Spring Snapshots</name>
49         <url>https://repo.spring.io/snapshot</url>
50         <snapshots>
51             <enabled>true</enabled>
```



```

52         </snapshots>
53     </repository>
54     <repository>
55         <id>spring-milestones</id>
56         <name>Spring Milestones</name>
57         <url>https://repo.spring.io/milestone</url>
58         <snapshots>
59             <enabled>false</enabled>
60         </snapshots>
61     </repository>
62 </repositories>
63 </project>
64 ...
65
66 .

```

- 最后启动eureka server 模块, customer 模块, provider 模块, dashboard 模块

**System Status**

Environment	test	Current time	2018-05-24T20:35
Data center	default	Uptime	00:35
		Lease expiration enabled	false
		Renews threshold	5
		Renews (last min)	4

**EMERGENCY! EUREKA MAY BE INCORRECTLY CLAIMING INSTANCES ARE UP WHEN THEY'RE NOT. RENEWALS ARE LESSER THAN T**  
**INSTANCES ARE NOT BEING EXPIRED JUST TO BE SAFE.**

**DS Replicas**

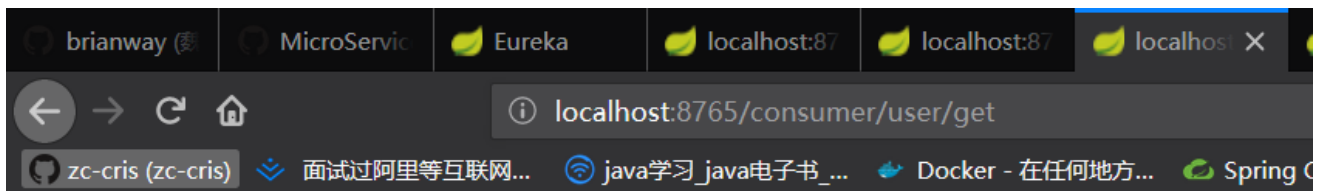
**Instances currently registered with Eureka**

Application	AMIs	Availability Zones	Status
SERVICE-DASHBOARD → 监控模块	n/a (1)	(1)	UP (1) - localhost:service-dashboard:8766
SERVICE-PROVIDER → 服务提供模块	n/a (1)	(1)	UP (1) - localhost:service-provider:8762

localhost:8762/provider/get

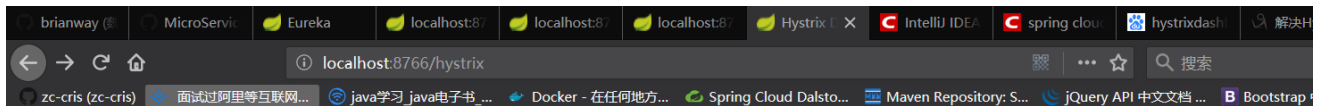
cris8762

服务模块自测没有问题



cris8762

消费模块调用没有问题



监控路径和之前版本不一样

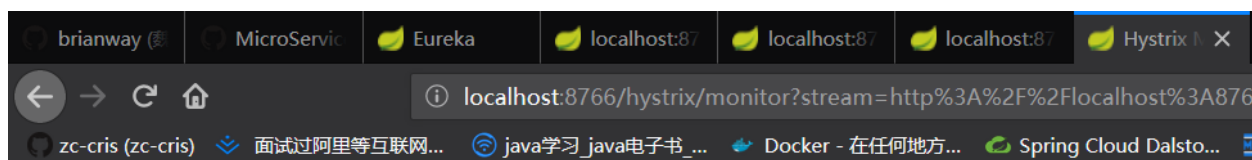
## Hystrix Dashboard

Cluster via Turbine (default cluster): <http://turbine-hostname:port/turbine.stream>  
Cluster via Turbine (custom cluster): [http://turbine-hostname:port/turbine.stream?cluster=\[clusterName\]](http://turbine-hostname:port/turbine.stream?cluster=[clusterName])  
Single Hystrix App: <http://hystrix-app:port/hystrix.stream>

Delay:  ms    Title:

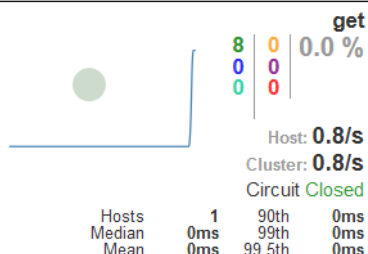
没有要求，尽量见名知意即可

- 进入dashboard 管理界面，如果出现loading...，刷新消费模块调用服务模块的url即可



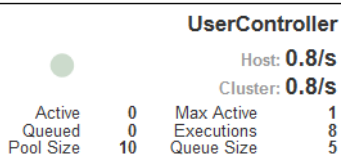
## Hystrix Stream: test

**Circuit** Sort: [Error then Volume](#) | [Alphabetical](#) | [Volume](#) | [Error](#) | [Mean](#) | [Median](#) | [90](#) | [99](#) | [99.5](#)



狂点服务模块调用消费模块服务的url，可以看见明显的峰值变化

**Thread Pools** Sort: [Alphabetical](#) | [Volume](#) |



## 9. 总结

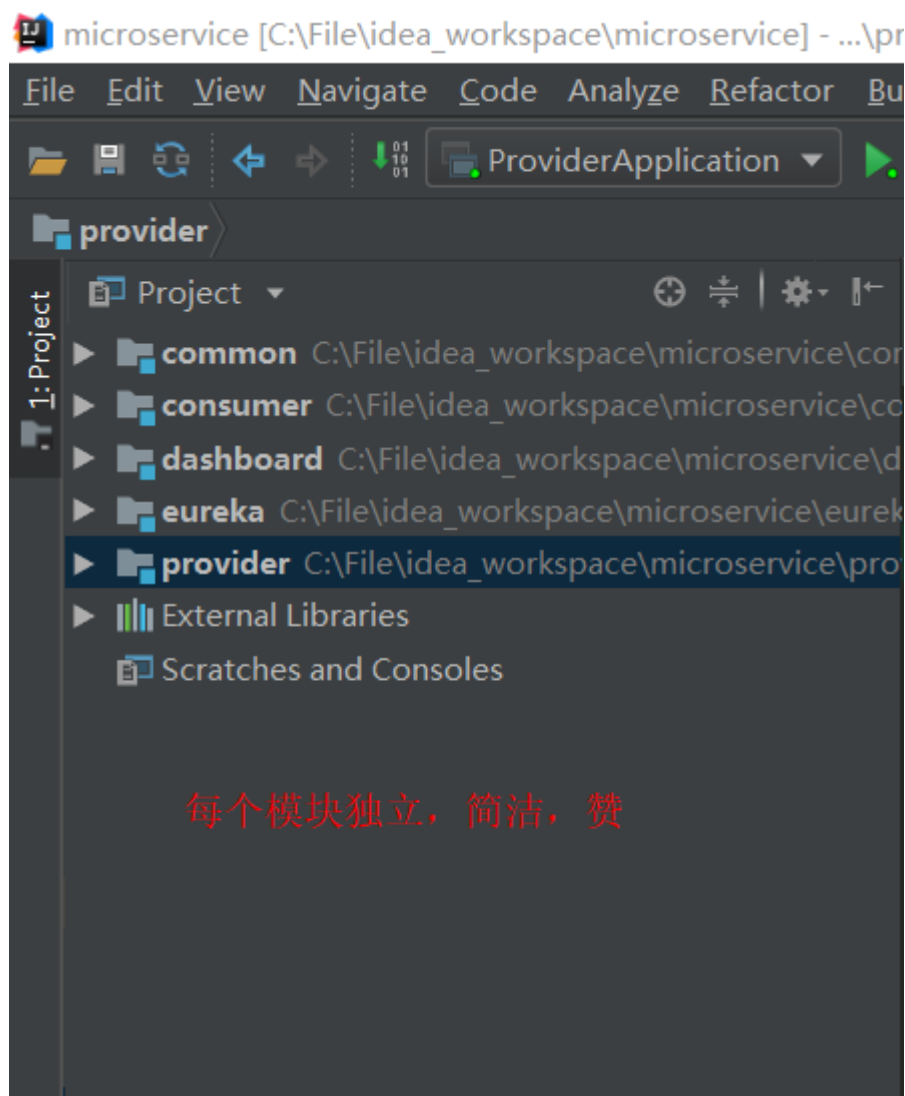
至此，我们完成了公共模块（feign代理接口 + hystrix集中处理）+ 服务模块（hystrix标记）+ eureka server（集群）+ 消费模块（注入feign代理接口）+ hystrix dashboard 监控模块的整体搭建；踩的坑如实记录下来，方便以后查看。

多说一句，最开始使用IDEA 搭建这个项目的时候，本来是按照eclipse 上的模式（Maven 多模块构建的方式）进行搭建，但是始终有问题，不是配置文件读取失败，就是ClassNotFoundException。。。测试了好久，最后使用如下方式进行SP 的项目搭建

I. 新建一个空工程（顶级工程）

II. 添加模块（全部使用spring initializr插件）

III. 公共组件等放在common 模块，其他模块如果需要可以引用这个common 模块（消费模块或者服务模块等），每个模块各自独立，而引用jar包全部放在空工程下，特别方便开发

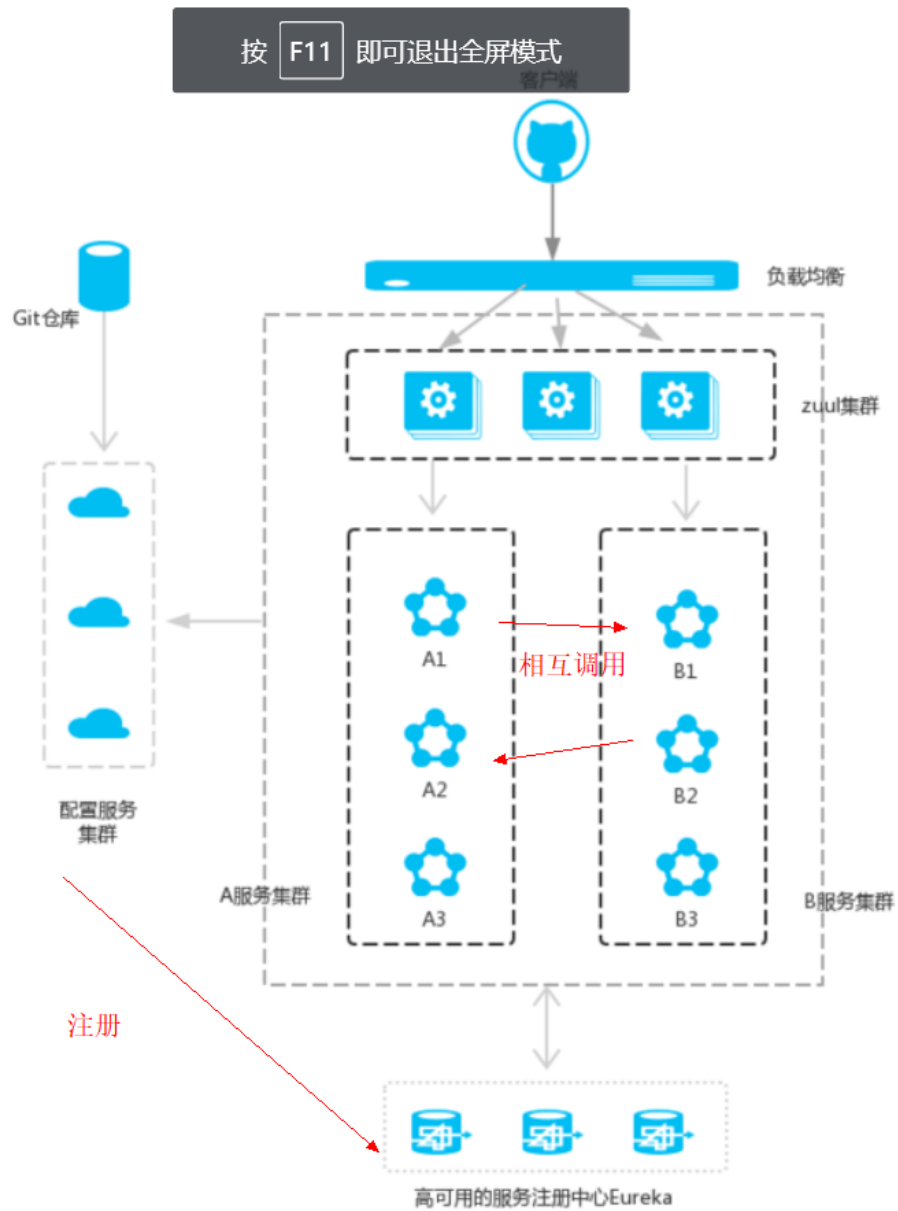


最后，本节的第三点没有，因为排版失误 -

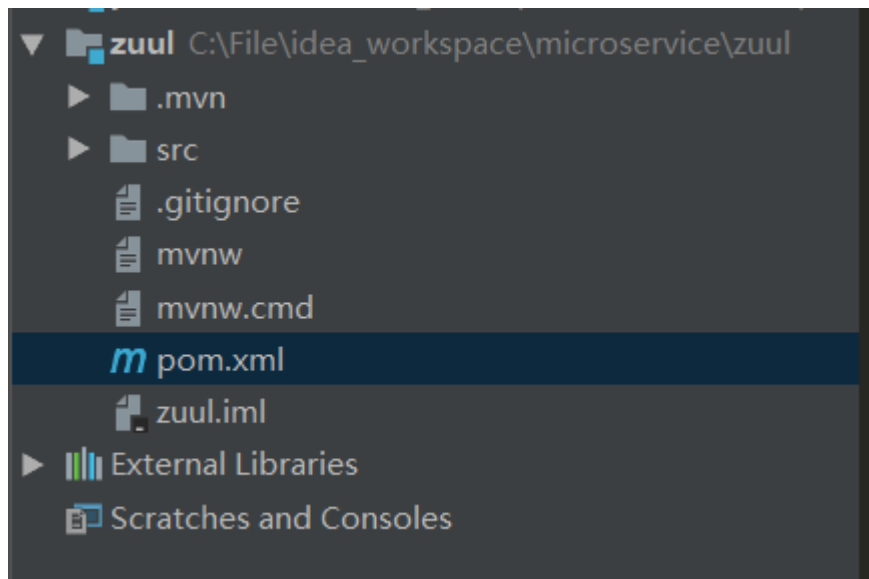
## 三、其他组件添加和完善

---

### 1. Zuul（网关组件）



### 1.1 新建一个Zuu1 模块 (引入web, zuul, eureka discovery 依赖)



1.2 pom.xml(亮点: 添加actuator 依赖和maven-resource 处理插件, 完成该模块的具体信息展示)

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
5   http://maven.apache.org/xsd/maven-4.0.0.xsd">
6   <modelVersion>4.0.0</modelVersion>
7
8   <groupId>com.cris</groupId>
9   <artifactId>zuul</artifactId>
10  <version>0.0.1-SNAPSHOT</version>
11  <packaging>jar</packaging>
12
13  <name>zuul</name>
14  <description>Demo project for Spring Boot</description>
15
16  <parent>
17    <groupId>org.springframework.boot</groupId>
18    <artifactId>spring-boot-starter-parent</artifactId>
19    <version>2.0.2.RELEASE</version>
20    <relativePath/> <!-- lookup parent from repository -->
21  </parent>
22
23  <properties>
24    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
25    <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
26    <java.version>1.8</java.version>
27    <spring-cloud.version>Finchley.BUILD-SNAPSHOT</spring-cloud.version>
28  </properties>
29
30  <dependencies>
31    <dependency>
32      <groupId>org.springframework.boot</groupId>
33      <artifactId>spring-boot-starter-web</artifactId>
```

```
32     </dependency>
33     <dependency>
34         <groupId>org.springframework.cloud</groupId>
35         <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
36     </dependency>
37     <dependency>
38         <groupId>org.springframework.cloud</groupId>
39         <artifactId>spring-cloud-starter-netflix-zuul</artifactId>
40     </dependency>
41
42     <dependency>
43         <groupId>org.springframework.boot</groupId>
44         <artifactId>spring-boot-starter-test</artifactId>
45         <scope>test</scope>
46     </dependency>
47     <!-- actuator 监控信息完善 -->
48     <dependency>
49         <groupId>org.springframework.boot</groupId>
50         <artifactId>spring-boot-starter-actuator</artifactId>
51     </dependency>
52 </dependencies>
53
54 <dependencyManagement>
55     <dependencies>
56         <dependency>
57             <groupId>org.springframework.cloud</groupId>
58             <artifactId>spring-cloud-dependencies</artifactId>
59             <version>${spring-cloud.version}</version>
60             <type>pom</type>
61             <scope>import</scope>
62         </dependency>
63     </dependencies>
64 </dependencyManagement>
65
66 <build>
67     <finalName>zuul</finalName>
68     <resources>
69         <resource>
70             <directory>src/main/resources</directory>
71             <filtering>true</filtering>
72         </resource>
73     </resources>
74     <plugins>
75         <plugin>
76             <groupId>org.apache.maven.plugins</groupId>
77             <artifactId>maven-resources-plugin</artifactId>
78             <configuration>
79                 <delimiters>
80                     <delimiter>${</delimiter>
81                 </delimiters>
82             </configuration>
83         </plugin>
84
```

```

85         <plugin>
86             <groupId>org.springframework.boot</groupId>
87             <artifactId>spring-boot-maven-plugin</artifactId>
88         </plugin>
89     </plugins>
90 </build>
91
92     <repositories>
93         <repository>
94             <id>spring-snapshots</id>
95             <name>Spring Snapshots</name>
96             <url>https://repo.spring.io/snapshot</url>
97             <snapshots>
98                 <enabled>true</enabled>
99             </snapshots>
100         </repository>
101         <repository>
102             <id>spring-milestones</id>
103             <name>Spring Milestones</name>
104             <url>https://repo.spring.io/milestone</url>
105             <snapshots>
106                 <enabled>false</enabled>
107             </snapshots>
108         </repository>
109     </repositories>
110
111 </project>

```

### 1.3 yml文件

```

1  server:
2      port: 8766      # 端口设置
3
4  spring:
5      application:
6          name: microservicecloud-zuul-gateway      # 注册到Eureka 以及对外暴露的微服务名字
              (重要)
7
8  eureka:
9      client:
10         serviceUrl:
11             defaultZone: http://localhost:8761/eureka/      # 将我们的zuul服务注册到eureka server 的地
              址
12         instance:
13             prefer-ip-address: true      #注册到服务中心使用ip进行注册（服务名称显示的ip详
              情)
14
15
16  info:
17      app.name: cris-microservicecloud
18      company.name: www.cris.com
19
20  app.programmer: zc-cris

```



```

20 build.artifactId: $project.artifactId$
21 build.version: $project.version$

```

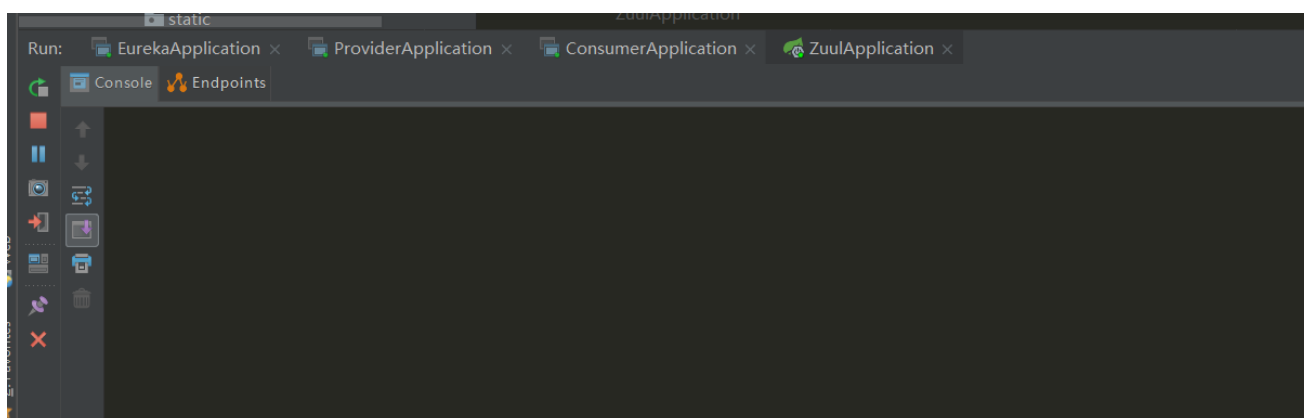
## 1.4 主启动类

```

1 @SpringBootApplication
2 @EnableZuulProxy           // 开启zuul 代理
3 @EnableEurekaClient       // 将服务注册到eureka server
4 public class ZuulApplication {
5
6     public static void main(String[] args) {
7         SpringApplication.run(ZuulApplication.class, args);
8     }
9 }

```

## 1.5 启动测试



localhost:8761

spring Eureka

HOME LAST 1000 SINCE STARTUP

### System Status

Environment	test	Current time	2018-05-25T11:40:06 +0800
Data center	default	Uptime	00:24
		Lease expiration enabled	false
		Renews threshold	5
		Renews (last min)	3

EMERGENCY! EUREKA MAY BE INCORRECTLY CLAIMING INSTANCES ARE UP WHEN THEY'RE NOT. RENEWALS ARE LESSER THAN THRESHOLD  
INSTANCES ARE NOT BEING EXPIRED JUST TO BE SAFE.

### DS Replicas

### Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
MICROSERVICECLOUD-ZUUL-GATEWAY	n/a (1)	(1)	UP (1) - localhost:microservicecloud-zuul-gateway:8766
SERVICE-PROVIDER	n/a (1)	(1)	UP (1) - localhost:service-provider:8762

网关模块已经注册到eureka server

服务模块

MicroService-SpringCloud/... Eureka 192.168.46.1:8766/actuator/...

192.168.46.1:8766/actuator/info

zc-cris (zc-cris) 面试过阿里等互联网... java学习 java电子书... Docker - 在任何地方... Spring

JSON 原始数据 头

保存 复制

```
▼ app:
  name: "cris-microservicecloud"
  programmer: "zc-cris"
▼ company:
  name: "www.cris.com"
▼ build:
  artifactId: "zuul"
  version: "0.0.1-SNAPSHOT"
```

点击eureka server 的网关服务连接，  
可以获取到详细信息

## 1.6 zuul 的路由功能

- 修改yml文件，定义服务访问规则

```
1 # zuul代理指定服务模块，将其对外暴露的服务名隐藏换个马甲
2 zuul:
3   prefix: /cris # 设置公共的访问前缀
4   ignored-services: "*" # 隐藏所有服务路径
5   routes:
6     user.serviceId: service-provider # 指定注册在eureka server 上的服务名
7     user.path: /user/** # 必须通过以下路径访问
```

zc-cris/MicroService-S... Eureka localhost:8766/service localhost:8766/cris/us... localhost:8762/provi

localhost:8766/service-provider/provider/get

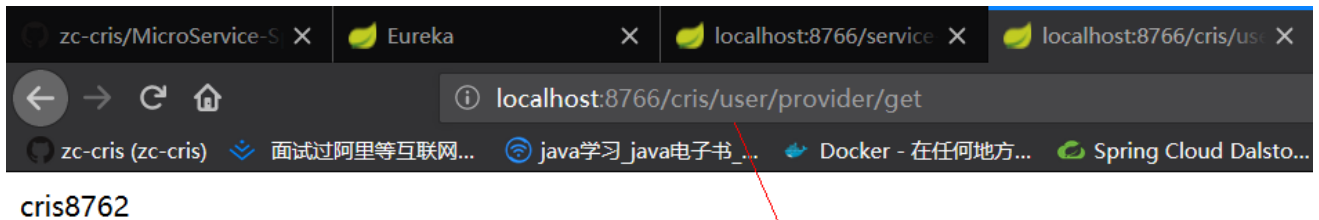
zc-cris (zc-cris) 面试过阿里等互联网... java学习 java电子书... Docker - 在任何地方... Spring Cloud Dalsto... Maven Repository: S

# Whitelabel Error Page

This application has no explicit mapping for /error, so you are seeing this as a fallback.

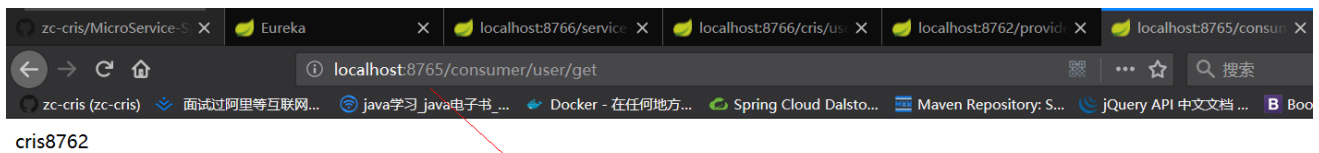
Fri May 25 14:23:05 CST 2018  
There was an unexpected error (type=Not Found, status=404).  
No message available

网关ip: 网关端口/服务模块注册名/服务映射路径  
无法正常调用服务模块的服务  
说明zuul 网关起作用了

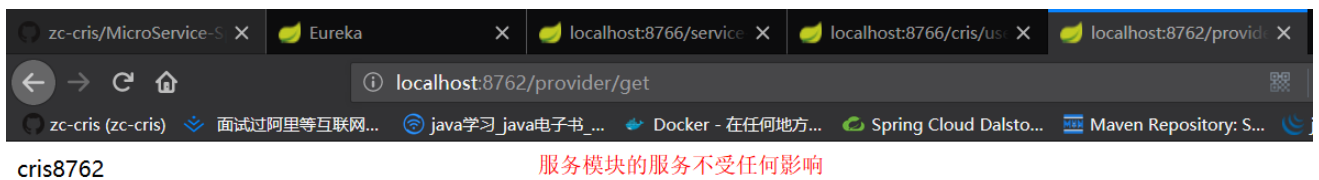


网关ip: 网关端口/前缀名/路由规则/服务模块调用路径可以发现, zuul 定义的访问规则生效了

注意: zuul主要针对的是外界访问我们的微服务架构, 及其进行访问控制和过滤, 但是我们自己微服务架构内的各个模块之间的调用不受影响



消费模块调用服务模块不受zuul的影响, 照常调用



服务模块的服务不受任何影响

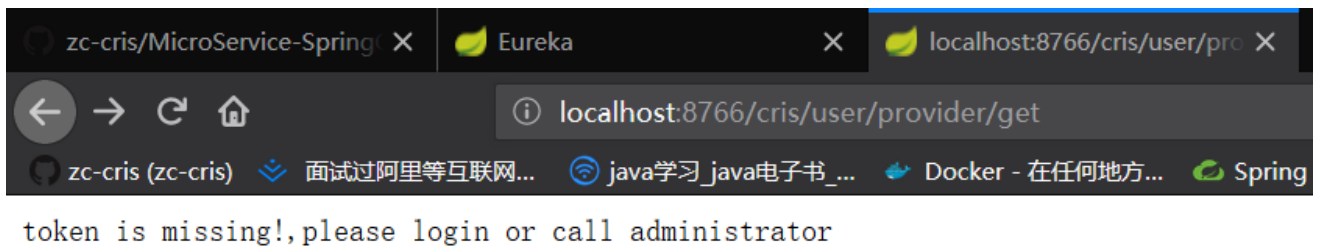
## 1.7 zuul 的过滤功能

针对用户的请求, 我们还可以使用zuul 做权限认证

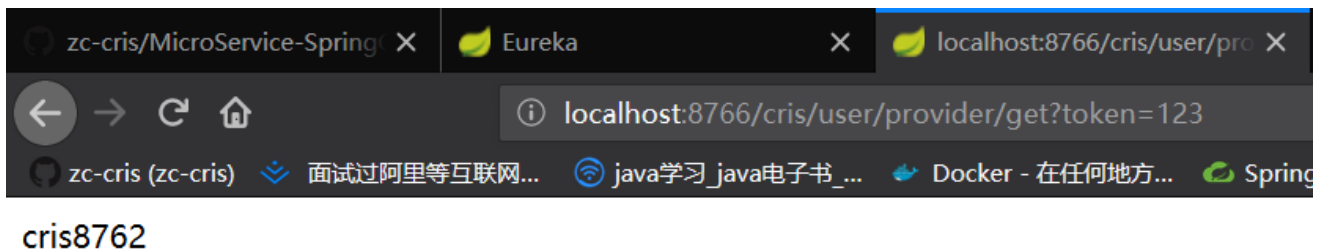
- 新建一个过滤器做认证

```
1 /**
2  * @ClassName MyFilter
3  * @Description TODO
4  * @Author zc-cris
5  * @Version 1.0
6  */
7 @Component
```

```
8 public class MyFilter extends ZuulFilter {
9
10     // 过滤规则
11     @Override
12     public String filterType() {
13         return "pre";
14     }
15
16     // 过滤顺序
17     @Override
18     public int filterOrder() {
19         return 0;
20     }
21
22     // 是否需要过滤, 一般开启
23     @Override
24     public boolean shouldFilter() {
25         return true;
26     }
27
28     // 认证规则
29     @Override
30     public Object run() throws ZuulException {
31         RequestContext currentContext = RequestContext.getCurrentContext();
32         HttpServletRequest request = currentContext.getRequest();
33         String token = request.getParameter("token");
34         if (token == null){
35             currentContext.setSendZuulResponse(false);
36             currentContext.setResponseStatusCode(401);
37             try {
38                 currentContext.getResponse().getWriter().write("token is missing!,please
login or call administrator");
39             } catch (IOException e) {
40                 e.printStackTrace();
41             }
42         }
43         return null;
44     }
45 }
```



外界访问就必须提供凭证了



拥有凭证，外界才可以访问我们的整个微服务体系

- 注意：
  - `filterType`: 返回一个字符串代表过滤器的类型，在zuul中定义了四种不同生命周期的过滤器类型，具体如下：
    - `pre`: 路由之前

```
1 - routing: 路由之时
```

```
1 - post: 路由之后
```

```
1 - error: 发送错误调用
```

- `filterOrder`: 过滤的顺序，越小优先级越高

- shouldFilter: 这里可以写逻辑判断, 是否要过滤
- run: 过滤器的具体逻辑。可用很复杂, 包括查sql, nosql去判断该请求到底有没有权限访问。

## 2. config 分布式配置组件 (结合git/github)

### 2.1 pom.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-
  4.0.0.xsd">
4   <modelVersion>4.0.0</modelVersion>
5
6   <groupId>com.cris</groupId>
7   <artifactId>config-server</artifactId>
8   <version>0.0.1-SNAPSHOT</version>
9   <packaging>jar</packaging>
10
11   <name>config-server</name>
12   <description>Demo project for Spring Boot</description>
13
14   <parent>
15     <groupId>org.springframework.boot</groupId>
16     <artifactId>spring-boot-starter-parent</artifactId>
17     <version>2.0.2.RELEASE</version>
18     <relativePath/> <!-- lookup parent from repository -->
19   </parent>
20
21   <properties>
22     <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
23     <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
24     <java.version>1.8</java.version>
25     <spring-cloud.version>Finchley.BUILD-SNAPSHOT</spring-cloud.version>
26   </properties>
27
28   <dependencies>
29     <dependency>
30       <groupId>org.springframework.boot</groupId>
31       <artifactId>spring-boot-starter-web</artifactId>
32     </dependency>
33     <dependency>
34       <groupId>org.springframework.cloud</groupId>
35       <artifactId>spring-cloud-config-server</artifactId>
36     </dependency>
37     <dependency>
38       <groupId>org.springframework.cloud</groupId>
39       <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
40     </dependency>
41
42     <dependency>
```

```

43     <groupId>org.springframework.boot</groupId>
44     <artifactId>spring-boot-starter-test</artifactId>
45     <scope>test</scope>
46 </dependency>
47 </dependencies>
48
49 <dependencyManagement>
50   <dependencies>
51     <dependency>
52       <groupId>org.springframework.cloud</groupId>
53       <artifactId>spring-cloud-dependencies</artifactId>
54       <version>${spring-cloud.version}</version>
55       <type>pom</type>
56       <scope>import</scope>
57     </dependency>
58   </dependencies>
59 </dependencyManagement>
60
61 <build>
62   <plugins>
63     <plugin>
64       <groupId>org.springframework.boot</groupId>
65       <artifactId>spring-boot-maven-plugin</artifactId>
66     </plugin>
67   </plugins>
68 </build>
69
70 <repositories>
71   <repository>
72     <id>spring-snapshots</id>
73     <name>Spring Snapshots</name>
74     <url>https://repo.spring.io/snapshot</url>
75     <snapshots>
76       <enabled>true</enabled>
77     </snapshots>
78   </repository>
79   <repository>
80     <id>spring-milestones</id>
81     <name>Spring Milestones</name>
82     <url>https://repo.spring.io/milestone</url>
83     <snapshots>
84       <enabled>false</enabled>
85     </snapshots>
86   </repository>
87 </repositories>
88 </project>

```

## 2.2 yml

```

1  server:
2    port: 8767
3

```

```

4  eureka:
5      client:
6          serviceUrl:
7              defaultZone: http://localhost:8761/eureka/    # 将我们的zuul服务注册到eureka server 的地址
8      instance:
9          prefer-ip-address: true                        #注册到服务中心使用ip进行注册（服务名称显示的ip详情)
10
11  spring:
12      application:
13          name: microservicecloud-config
14      cloud:
15          config:
16              server:
17                  git:
18                      uri: git@github.com:zc-cris/springcloud_server_config_demo.git    #github上的仓库名字

```

## 2.3 主启动类

```

1  @SpringBootApplication
2  @EnableConfigServer
3  public class ConfigServerApplication {
4
5      public static void main(String[] args) {
6          SpringApplication.run(ConfigServerApplication.class, args);
7      }
8  }

```

## 2.4 启动并测试



zc-cris/springcloud\_server\_c X Eureka X localhost:8766/cris/user/pro X localhost:8762/provider/get X localhost:8765/consumer/us X +

localhost:8761 搜索

zc-cris (zc-cris) 面试过阿里等互联网... java学习 java电子书... Docker - 在任何地方... Spring Cloud Dalsto... Maven Repository: S... jQuery API 中文文档... Bootstrap 中文文档...

# spring Eureka

HOME LAST 1000 SINCE S

## System Status

Environment	test	Current time	2018-05-25T15:42:...
Data center	default	Uptime	04:26
		Lease expiration enabled	false
		Renews threshold	6
		Renews (last min)	6

EMERGENCY! EUREKA MAY BE INCORRECTLY CLAIMING INSTANCES ARE UP WHEN THEY'RE NOT. RENEWALS ARE LESSER THAN THREE INSTANCES ARE NOT BEING EXPIRED JUST TO BE SAFE.

## DS Replicas

### Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
MICROSERVICECLOUD-CONFIG → config server 模块	n/a (1)	(1)	UP (1) - localhost:microservicecloud-config:8767
MICROSERVICECLOUD-ZUUL-GATEWAY	n/a (1)	(1)	UP (1) - localhost:microservicecloud-zuul-gateway:8766
SERVICE-PROVIDER	n/a (1)	(1)	UP (1) - localhost:service-provider:8762

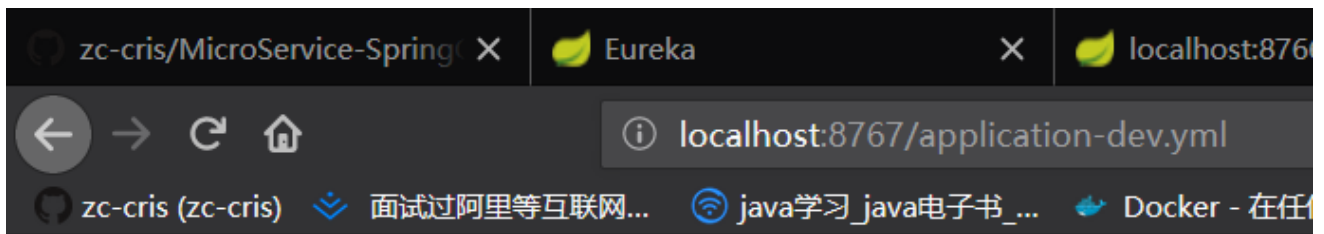
zc-cris/MicroService-Spring X Eureka X localhost:8766/cris/user/pro X localhost:8762/prov

localhost:8767/application-test.yml

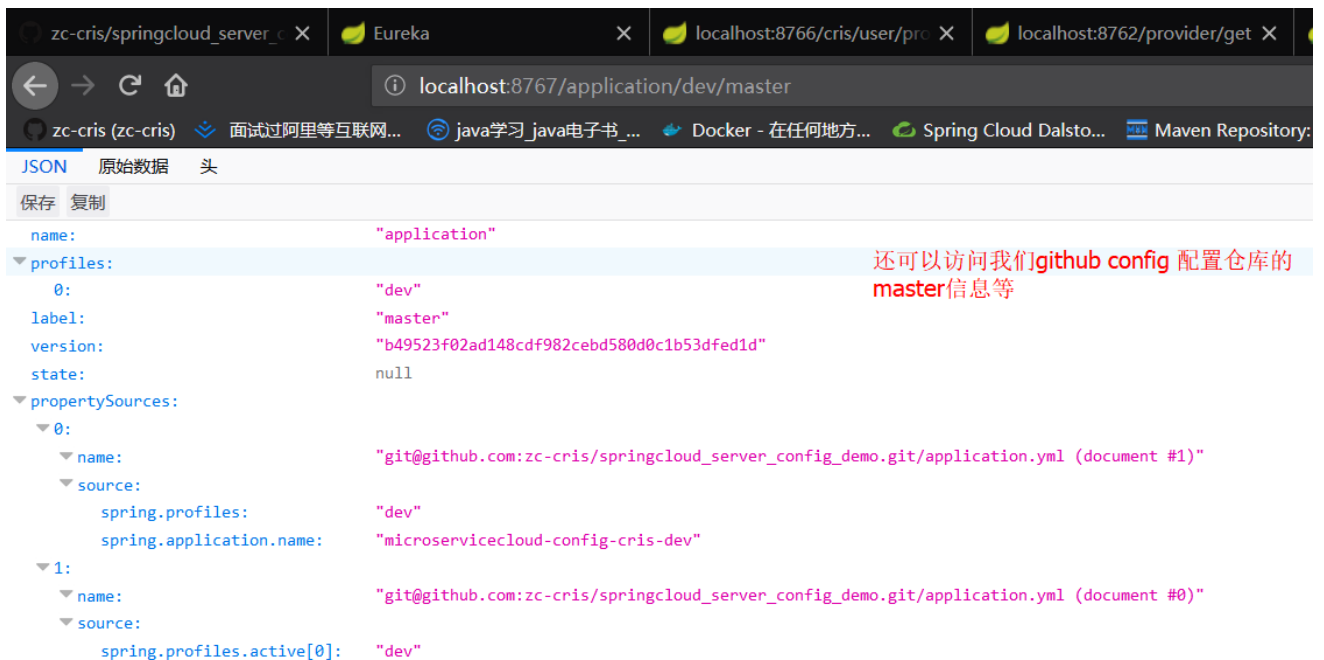
zc-cris (zc-cris) 面试过阿里等互联网... java学习 java电子书... Docker - 在任何地方... Spring Cloud Dalsto... Mav

```
spring:
  application:
    name: microservicecloud-config-cris-test
  profiles:
    active:
      - dev
```

通过我们新建的config server模块成功访问到了我的github上的配置文件信息

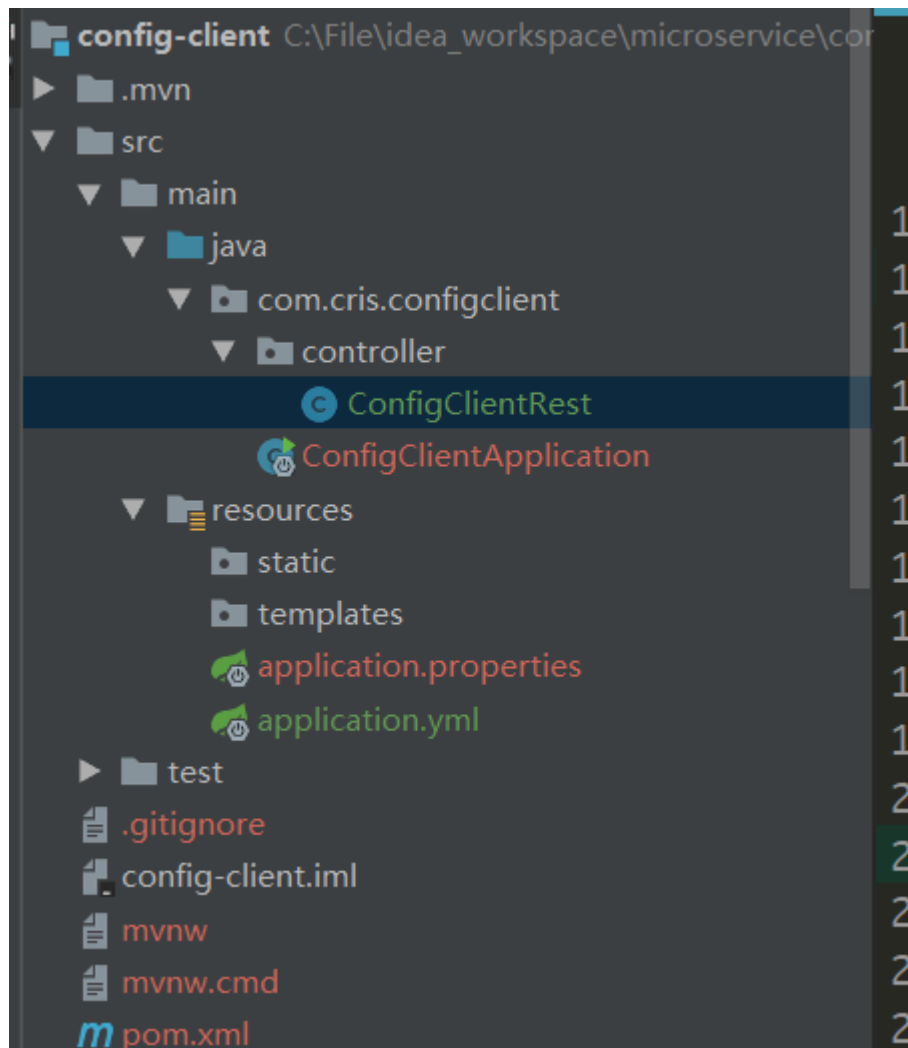


```
spring:
  application:
    name: microservicecloud-config-cris-dev
  profiles:
    active:
      - dev
```



## 2.5 测试config client 模块能否通过config server端读取到github上的配置信息

1、新建一个config 版的customer 模块 (引入eureka discovery, web, config client 依赖)



2、pom.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
  http://maven.apache.org/xsd/maven-4.0.0.xsd">
4     <modelVersion>4.0.0</modelVersion>
5
6     <groupId>com.cris</groupId>
7     <artifactId>config-client</artifactId>
8     <version>0.0.1-SNAPSHOT</version>
9     <packaging>jar</packaging>
10
11     <name>config-client</name>
12     <description>Demo project for Spring Boot</description>
13
14     <parent>
15         <groupId>org.springframework.boot</groupId>
16         <artifactId>spring-boot-starter-parent</artifactId>
17         <version>2.0.2.RELEASE</version>
18         <relativePath/> <!-- lookup parent from repository -->
19     </parent>
```

```
20
21 <properties>
22   <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
23   <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
24   <java.version>1.8</java.version>
25   <spring-cloud.version>Finchley.BUILD-SNAPSHOT</spring-cloud.version>
26 </properties>
27
28 <dependencies>
29   <dependency>
30     <groupId>org.springframework.boot</groupId>
31     <artifactId>spring-boot-starter-web</artifactId>
32   </dependency>
33   <dependency>
34     <groupId>org.springframework.cloud</groupId>
35     <artifactId>spring-cloud-starter-config</artifactId>
36   </dependency>
37   <dependency>
38     <groupId>org.springframework.cloud</groupId>
39     <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
40   </dependency>
41
42   <dependency>
43     <groupId>org.springframework.boot</groupId>
44     <artifactId>spring-boot-starter-test</artifactId>
45     <scope>test</scope>
46   </dependency>
47 </dependencies>
48
49 <dependencyManagement>
50   <dependencies>
51     <dependency>
52       <groupId>org.springframework.cloud</groupId>
53       <artifactId>spring-cloud-dependencies</artifactId>
54       <version>${spring-cloud.version}</version>
55       <type>pom</type>
56       <scope>import</scope>
57     </dependency>
58   </dependencies>
59 </dependencyManagement>
60
61 <build>
62   <plugins>
63     <plugin>
64       <groupId>org.springframework.boot</groupId>
65       <artifactId>spring-boot-maven-plugin</artifactId>
66     </plugin>
67   </plugins>
68 </build>
69
70 <repositories>
71   <repository>
72     <id>spring-snapshots</id>
```

```

73     <name>Spring Snapshots</name>
74     <url>https://repo.spring.io/snapshot</url>
75     <snapshots>
76         <enabled>true</enabled>
77     </snapshots>
78 </repository>
79 <repository>
80     <id>spring-milestones</id>
81     <name>Spring Milestones</name>
82     <url>https://repo.spring.io/milestone</url>
83     <snapshots>
84         <enabled>false</enabled>
85     </snapshots>
86 </repository>
87 </repositories>
88
89
90 </project>

```

### 3、yml

```

1  spring:
2    cloud:
3      config:
4        name: microservicecloud-config-client # 需要从github上读取的资源名称, 注意没有yml后缀名
5        profile: dev # 本次访问的配置环境
6        label: master
7        uri: http://localhost:8767      #客户端微服务启动后, 先去8767端口寻找服务端, 通过服务端获取
github的repository 地址
8    application:
9      name: config-client
10
11  eureka:
12    client:
13      serviceUrl:
14        defaultZone: http://localhost:8761/eureka/    # 将我们的zuul服务注册到eureka server 的地
址
15    instance:
16      prefer-ip-address: true    #注册到服务中心使用ip进行注册 (服务名称显示的ip详情)
17  server:
18    port: 8768

```

### 4、主启动类

```

1 @SpringBootApplication
2 @EnableEurekaClient
3 public class ConfigClientApplication {
4
5     public static void main(String[] args) {
6         SpringApplication.run(ConfigClientApplication.class, args);
7     }
8 }

```

## 5、测试用的controller

```

1 /**
2  * @ClassName ConfigClientrest
3  * @Description TODO
4  * @Author zc-cris
5  * @Version 1.0
6  */
7 @RestController
8 public class ConfigClientRest {
9
10     @Value("${spring.application.name}")
11     private String applicationName;
12
13     @Value("${eureka.client.service-url.defaultZone}")
14     private String eurekaServers;
15
16     @Value("${server.port}")
17     private String port;
18
19     @GetMapping("/config")
20     public String getConfig() {
21         String string = "applicationName:" + applicationName + ",eurekaServers:" +
22         eurekaServers + ",port:" + port;
23         System.out.println(string);
24         return string;
25     }
26 }

```

## 6、往github上传配置文件

zc-cris / springcloud\_server\_config\_demo

Watch 0Star

CodeIssues 0Pull requests 0Projects 0WikiInsightsSettings

Branch: master springcloud\_server\_config\_demo / config-client.yml

zc-cris 第二个sc案例的config\_client 模块配置

1 contributor

27 lines (26 sloc) | 393 Bytes

RawBlameHist

```
1  spring:
2    profiles:
3      active:
4        - dev
5
6  ---
7  server:
8    port: 8768
9  spring:
10   profiles: dev
11   application:
12     name: config-client
13 eureka:
14   client:
15     service-url:
16       defaultZone: http://localhost:8761/eureka/
17   ---
```

7. 启动测试

zc-cris/springcloud\_server\_...Eurekalocalhost:8766/cris/user/pr...localhost:8762/provider/get...localhost:8765/consumer/u...+

localhost:8761

zc-cris (zc-cris)面试过阿里等互联网...java学习 java电子书...Docker - 在任何地方...Spring Cloud Dalsto...Maven Repository: S...jQuery API 中文文档...Bootstrap 中文文档...WebJars - Web

spring Eureka

HOME LAST 1000 SINCE STARTUP

System Status

Environment	test	Current time	2018-05-25T16:41:33 +0800
Data center	default	Uptime	05:25
		Lease expiration enabled	false
		Renews threshold	8
		Renews (last min)	6

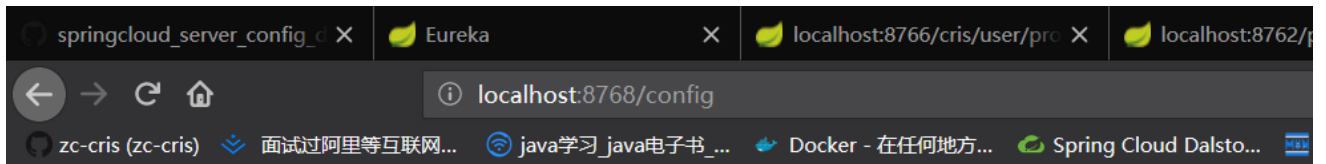
EMERGENCY! EUREKA MAY BE INCORRECTLY CLAIMING INSTANCES ARE UP WHEN THEY'RE NOT. RENEWALS ARE LESSER THAN THRESHOLD AND INSTANCES ARE NOT BEING EXPIRED JUST TO BE SAFE.

DS Replicas

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
CONFIG-CLIENT	n/a (1)	(1)	UP (1) - localhost:config-client:8768
MICROSERVICECLOUD-CONFIG	n/a (1)	(1)	UP (1) - localhost:microservicecloud-config:8767
MICROSERVICECLOUD-ZUUL-GATEWAY	n/a (1)	(1)	UP (1) - localhost:microservicecloud-zuul-gateway:8766
SERVICE-PROVIDER	n/a (1)	(1)	UP (1) - localhost:service-provider:8762

我们的config client 模块从github上读取的配置文件信息并且注册到我们本地的eureka server 上



applicationName:config-client,eurekaServers:http://localhost:8761/eureka/,port:8768

这就是我们的config版的微服务模块通过config server模块从github上读取到的配置信息

8. 如果对SpringCloud的config组件还想有个综合的实战练习,可以参考我的第一季SC整合案例,这里就不在进行重复性操作了,只要config server可以从github上读取配置,我们的微服务模块可以通过server端读取到各自的配置信息(配置集中化管理),即可