

SpringCloud之二

Spring Cloud Eureka学习

看很多网上文档都要使用Eureka来进行服务注册发现，所以先选择Eureka来进行学习。

Spring Cloud Eureka

一、Eureka介绍

Spring Cloud Eureka 是 Spring Cloud Netflix 微服务套件的一部分，基于 Netflix Eureka 做了二次封装，主要负责完成微服务架构中的服务治理功能。除了用Eureka来做注册中心，我们还可以使用Consul,Etcd,Zookeeper等等来作为服务的注册中心。

有用过dubbo的同学应该清楚，dubbo中也有几种注册中心，有基于 Zookeeper的，有基于redis的等等，用的最多的还是Zookeeper方式。

至于使用哪种方式，其实都是可以的，**注册中心无非就是管理所有服务的信息和状态。**

用我们生活中的例子来说明的话，我觉得12306比较合适。

首先12306就好比一个注册中心，N量火车都注册在了12306上面，我们顾客就好比调用的客户端，当我们需要坐火车时，我们会去12306上看有没有票，有票就可以购买，然后拿到火车的班次，时间等等，最后出发。

程序也是一样，当你需要调用某一个服务的时候，你会先去Eureka中去拉取服务列表，查看你调用的服务在不在其中，在的话就拿到服务地址，端口，等等信息，然后调用。

注册中心带来的好处就是你不需要知道有多少提供方，你只需要关注注册中心即可，你不必关系有多少火车在运行，你只需要去12306上看有没有票可以买就可以。



二、Eureka Server

1、创建项目

- 首先创建一个maven工程(或者用<http://start.spring.io/>来创建一个spring cloud项目)
- 在pom.xml增加依赖(如果下载包特别慢可以考虑使用阿里云的maven镜像服务器<http://cxytiandi.com/blog/detail/5321>)

代码示例如下，pom.xml：

```
1 <!-- Spring Boot -->
2 <parent>
3   <groupId>org.springframework.boot</groupId>
4   <artifactId>spring-boot-starter-parent</artifactId>
5   <version>1.5.4.RELEASE</version>
6   <relativePath />
7 </parent>
8 <dependencies>
9   <!-- eureka -->
10  <dependency>
11    <groupId>org.springframework.cloud</groupId>
12    <artifactId>spring-cloud-starter-eureka-server</artifactId>
13  </dependency>
14 </dependencies>
```

```

15 <!-- Spring Cloud -->
16 <dependencyManagement>
17   <dependencies>
18     <dependency>
19       <groupId>org.springframework.cloud</groupId>
20       <artifactId>spring-cloud-dependencies</artifactId>
21       <version>Dalston.SR1</version>
22       <type>pom</type>
23       <scope>import</scope>
24     </dependency>
25   </dependencies>
26 </dependencyManagement>

```

无需怀疑，这个肯定的使用了springboot来进行构建。

2、创建启动类

EurekaServerApplication.java

```

1  package xyz.jiangnanke.eureka;
2
3  import org.springframework.boot.SpringApplication;
4  import org.springframework.boot.autoconfigure.SpringBootApplication;
5  import
org.springframework.cloud.netflix.eureka.server.EnableEurekaServer;
6
7  /**
8   * @Auther: zhengfeng
9   * @Date: 2018\12\18 0018 16:20
10   * @Description: 服务注册中心
11   */
12  @EnableEurekaServer
13  @SpringBootApplication
14  public class EurekaServerApplication {
15
16      public static void main(String[] args) {
17          SpringApplication.run(EurekaServerApplication.class, args);
18      }
19
20  }

```

3、设置Eureka监听类

EurekaStateChangeListener.java

```
1 package xyz.jiangnanke.eureka;
2
3 import com.netflix.appinfo.InstanceInfo;
4 import org.springframework.cloud.netflix.eureka.server.event.EurekaInstanceCanceledEvent;
5 import org.springframework.cloud.netflix.eureka.server.event.EurekaInstanceRegisteredEvent;
6 import org.springframework.cloud.netflix.eureka.server.event.EurekaInstanceRenewedEvent;
7 import org.springframework.cloud.netflix.eureka.server.event.EurekaRegistryAvailableEvent;
8 import org.springframework.cloud.netflix.eureka.server.event.EurekaServerStartedEvent;
9 import org.springframework.context.event.EventListener;
10 import org.springframework.stereotype.Component;
11
12 /**
13  * @Auther: zhengfeng
14  * @Date: 2018\12\18 00:18 16:21
15  * @Description: Eureka事件监听
16  */
17 @Component
18 public class EurekaStateChangeListener {
19
20     @EventListener
21     public void listen(EurekaInstanceCanceledEvent event) {
22         System.err.println(event.getServerId() + "\t" + event.getAppName() + "服务下线");
23     }
24
25     @EventListener
26     public void listen(EurekaInstanceRegisteredEvent event) {
27         InstanceInfo instanceInfo = event.getInstanceInfo();
28         System.err.println(instanceInfo.getAppName() + "进行注册");
29     }
30
31     @EventListener
32     public void listen(EurekaInstanceRenewedEvent event) {
33         System.err.println(event.getServerId() + "\t" + event.getAppName() + "服务进行续约");
34     }
35 }
```

```

35
36  @EventListener
37  public void listen(EurekaRegistryAvailableEvent event) {
38      System.err.println("注册中心 启动");
39  }
40
41  @EventListener
42  public void listen(EurekaServerStartedEvent event) {
43      System.err.println("Eureka Server 启动");
44  }
45  }

```

4、设置环境信息

application.properties

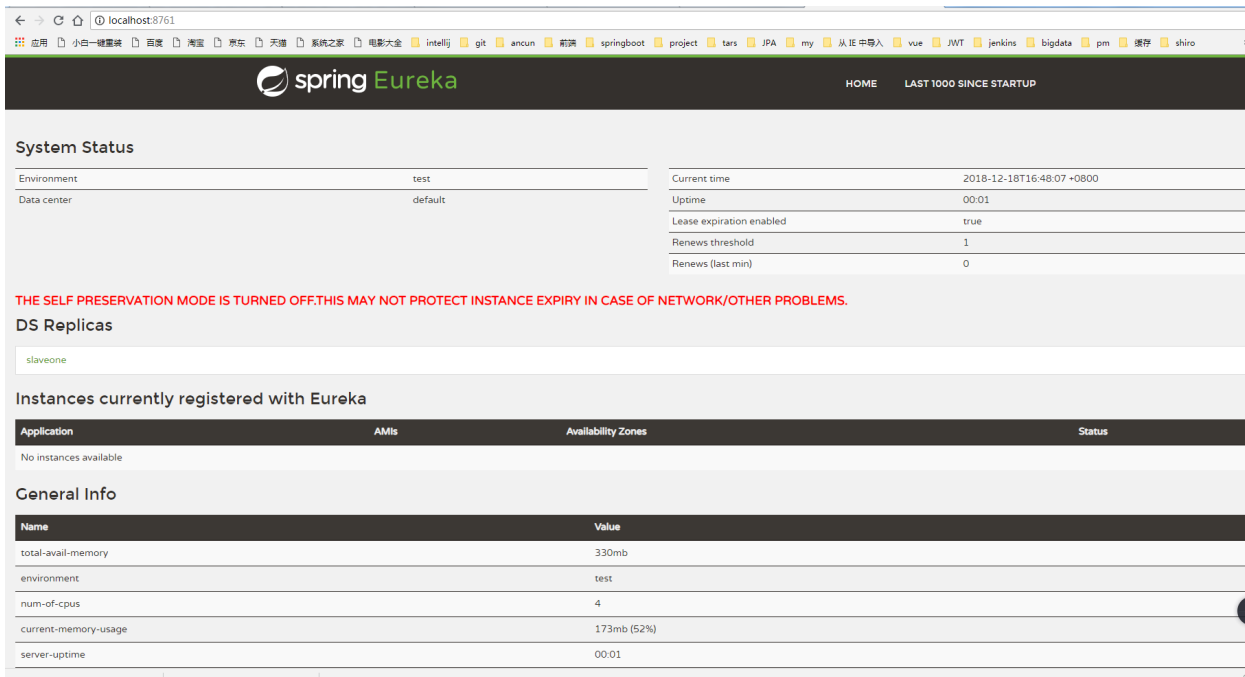
```

1  #服务端口
2  server.port=8761
3  #服务名称
4  spring.application.name=my-eureka
5  #服务地址绑定
6  eureka.instance.hostname=localhost
7  # 由于该应用为注册中心，所以设置为false，代表不向注册中心注册自己
8  eureka.client.register-with-eureka=false
9  # 由于注册中心的职责就是维护服务实例，他并不需要去检索服务，所以也设置为false
10 eureka.client.fetch-registry=false
11 # 关闭自我保护
12 eureka.server.enableSelfPreservation=false
13 eureka.server.eviction-interval-timer-in-ms=5000
14 security.basic.enabled=true
15 security.user.name=jiangnanke
16 security.user.password=123456
17
18 spring.profiles.active=master

```

5、启动

最后启动EurekaServerApplication，访问<http://localhost:8761/>就可以打开管理页面了。



(今天手痒，所以抽出了一个pom的父模块，导致代码有一点点改动)

三、Eureka Client

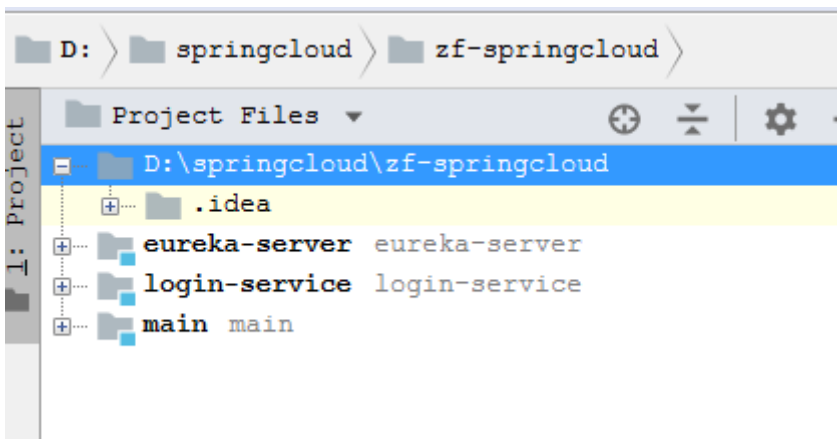
首先说明一下我的项目结构。

main

eureka-server

login-service (eureka-client)

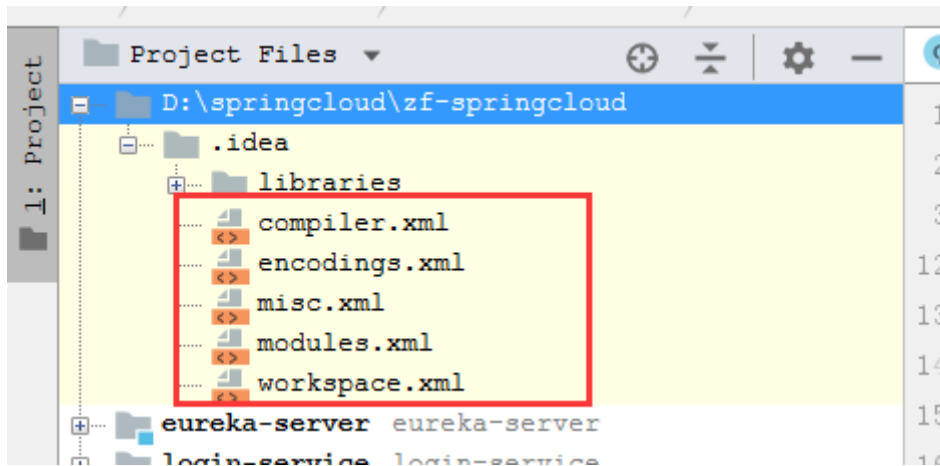
如图：



需要注意的地方

1、创建问题

如果发现删除模块之后再重新创建同名的模块，则需要删除project里面的一些配置信息，不然会发现一些很奇怪的问题，可以自己看看。



2、父子模块pom的配置

main模块的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.apach
  e.org/xsd/maven-4.0.0.xsd">
4   <modelVersion>4.0.0</modelVersion>
5
6   <groupId>xyz.jiangnanke</groupId>
7   <artifactId>main</artifactId>
8   <version>0.0.1-SNAPSHOT</version>
9   <packaging>pom</packaging>
10
11   <name>main</name>
12   <description>main project for Spring Boot</description>
13
14   <parent>
15     <groupId>org.springframework.boot</groupId>
16     <artifactId>spring-boot-starter-parent</artifactId>
17     <version>2.1.1.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.outputEncodi
  ng>
24     <java.version>1.8</java.version>
25     <spring-cloud.version>Finchley.RELEASE</spring-cloud.version>
```

```
26 </properties>
27
28 <modules>
29 <module>../eureka-server</module>
30 <module>../login-service</module>
31 </modules>
32
33 <dependencies>
34 <dependency>
35 <groupId>org.springframework.boot</groupId>
36 <artifactId>spring-boot-starter</artifactId>
37 </dependency>
38
39 <dependency>
40 <groupId>org.springframework.boot</groupId>
41 <artifactId>spring-boot-starter-test</artifactId>
42 <scope>test</scope>
43 </dependency>
44 </dependencies>
45
46 <dependencyManagement>
47 <dependencies>
48 <dependency>
49 <groupId>org.springframework.cloud</groupId>
50 <artifactId>spring-cloud-dependencies</artifactId>
51 <version>${spring-cloud.version}</version>
52 <type>pom</type>
53 <scope>import</scope>
54 </dependency>
55 </dependencies>
56 </dependencyManagement>
57
58 <build>
59 <plugins>
60 <plugin>
61 <groupId>org.springframework.boot</groupId>
62 <artifactId>spring-boot-maven-plugin</artifactId>
63 </plugin>
64 </plugins>
65 </build>
66
```


eureka-server的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.apach
e.org/xsd/maven-4.0.0.xsd">
4   <modelVersion>4.0.0</modelVersion>
5
6   <groupId>xyz.jiangnanke</groupId>
7   <artifactId>eureka-server</artifactId>
8   <version>0.0.1-SNAPSHOT</version>
9   <name>eureka-server</name>
10  <description>Demo project for Spring Boot</description>
11
12  <parent>
13    <groupId>xyz.jiangnanke</groupId>
14    <artifactId>main</artifactId>
15    <version>0.0.1-SNAPSHOT</version>
16  </parent>
17
18
19  <dependencies>
20    <dependency>
21      <groupId>org.springframework.cloud</groupId>
22      <artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>
23    </dependency>
24
25  </dependencies>
26
27
28 </project>
```

login-service的pom.xml

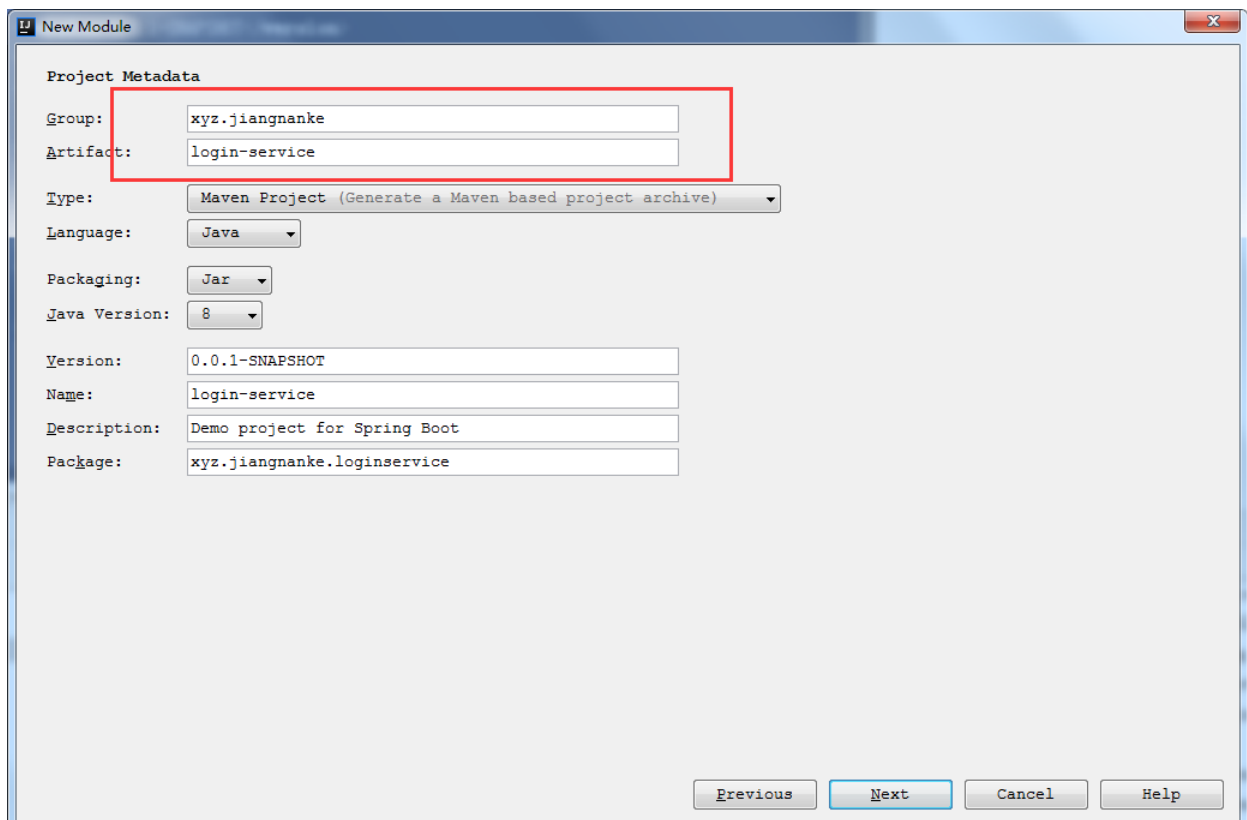
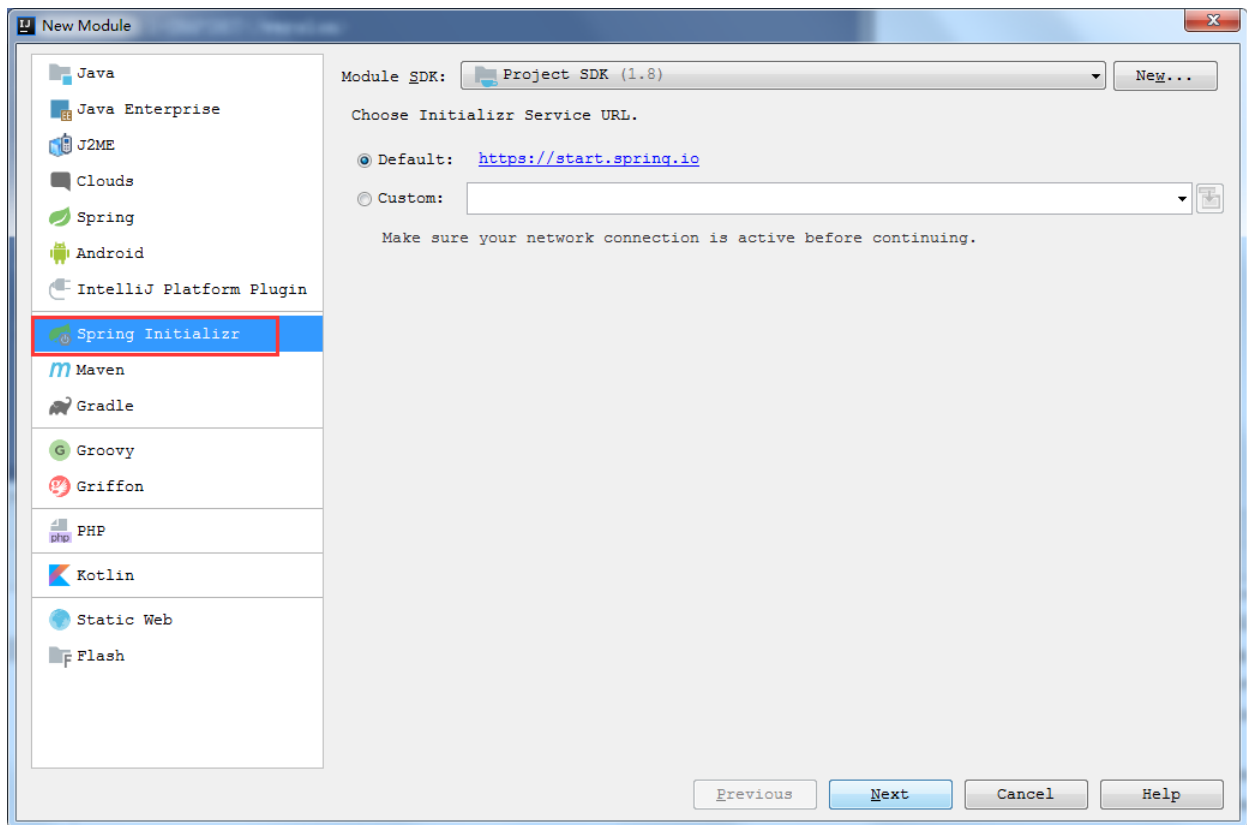
```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.
w3.org/2001/XMLSchema-instance"
3   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apach
e.org/xsd/maven-4.0.0.xsd">
4   <modelVersion>4.0.0</modelVersion>
```

```
5
6 <groupId>xyz.jiangnanke</groupId>
7 <artifactId>login-service</artifactId>
8 <version>0.0.1-SNAPSHOT</version>
9 <name>login-service</name>
10 <description>Demo project for Spring Boot</description>
11
12 <parent>
13 <groupId>xyz.jiangnanke</groupId>
14 <artifactId>main</artifactId>
15 <version>0.0.1-SNAPSHOT</version>
16 </parent>
17
18 <dependencies>
19 <dependency>
20 <groupId>org.springframework.boot</groupId>
21 <artifactId>spring-boot-starter-web</artifactId>
22 </dependency>
23 <dependency>
24 <groupId>org.springframework.cloud</groupId>
25 <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
26 </dependency>
27
28 </dependencies>
29
30
31 </project>
```

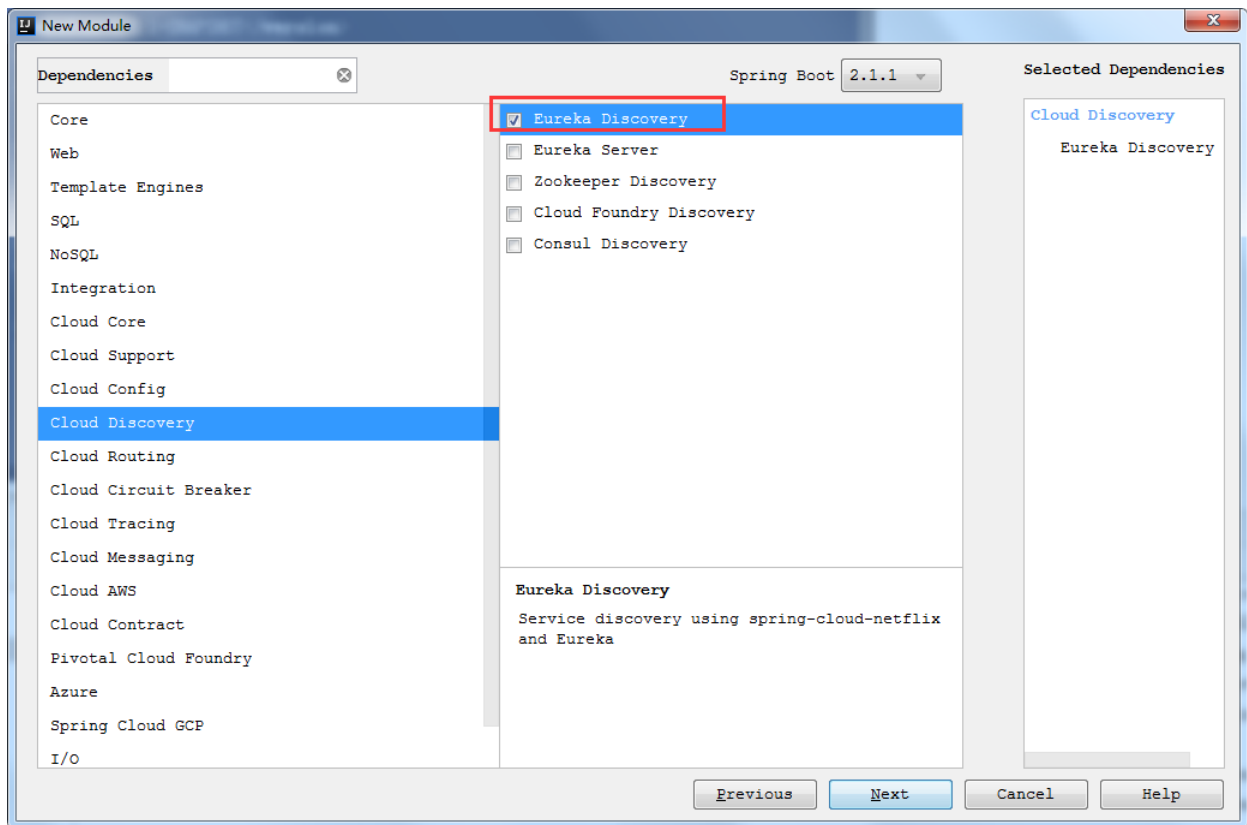
好了，接下来看login-service的搭建

1、创建项目

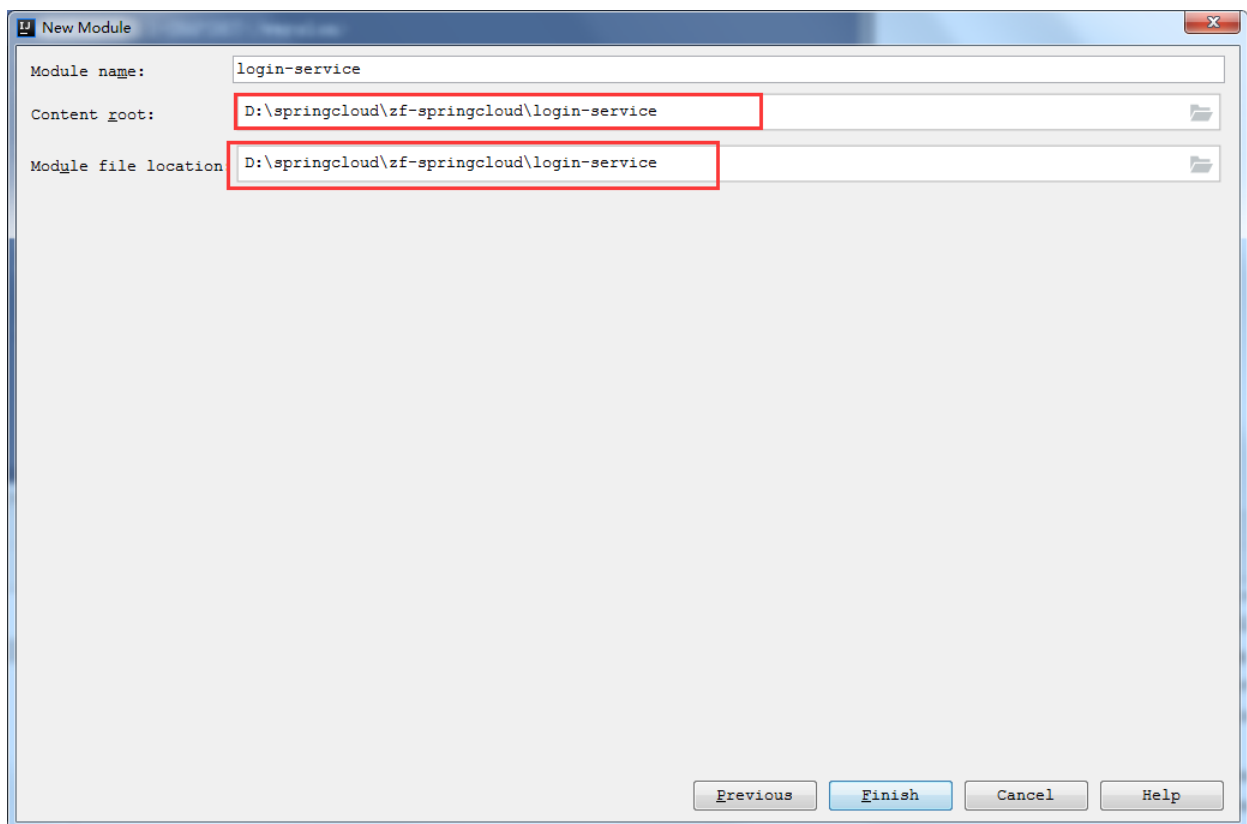
类似于eureka-server创建一个springboot项目



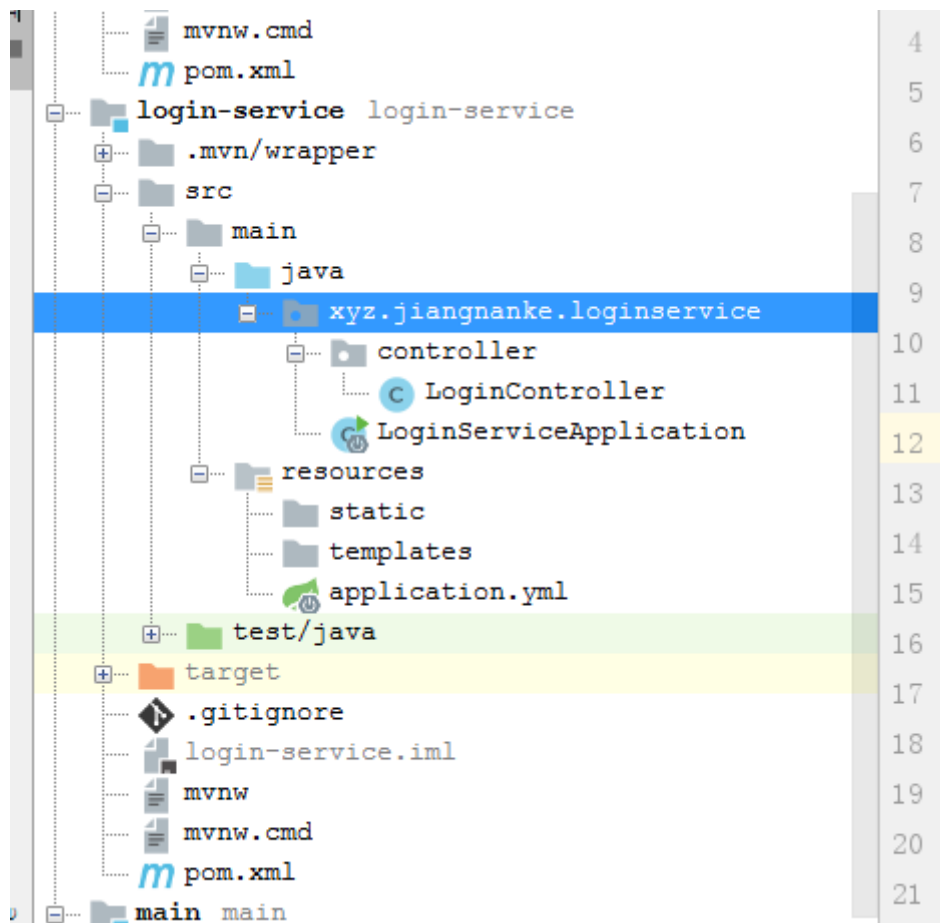
选择eureka的client依赖，当然还可以选择web依赖



一定要注意红框的路径，如果覆盖了原有的模块，那就和我一样悲剧了，暂时只能都删除了，然后重新创建



最后我的login-service的模块结构如下：



2、配置项目

关于pom.xml的配置看上面

关于application.yml的配置如下：

```
1 server:  
2   port: 8762  
3  
4 spring:  
5   application:  
6     name: login-service  
7  
8 eureka:  
9   client:  
10    serviceUrl:  
11    defaultZone: http://localhost:8761/eureka/
```

3、写测试类

我这边提供了一个处理登录的路径controller

LoginController.java

```
1 package xyz.jiangnanke.loginservice.controller;
```

```

2
3 import org.springframework.beans.factory.annotation.Value;
4 import org.springframework.web.bind.annotation.RequestMapping;
5 import org.springframework.web.bind.annotation.RequestParam;
6 import org.springframework.web.bind.annotation.RestController;
7
8 /**
9  * @Auther: zhengfeng
10  * @Date: 2018\12\19 0019 15:48
11  * @Description:
12  */
13 @RestController
14 public class LoginController {
15
16     @Value("${server.port}")
17     String port;
18
19     @RequestMapping("/login")
20     public String home(@RequestParam(value = "name", defaultValue = "jiangnanke") String name) {
21         System.out.println(" this is login for name :" + name);
22         return "hi " + name + " ,login is success! the port is:" + port;
23     }
24
25 }

```

当然在Application启动类需要配置一些注解

如下：

```

1 package xyz.jiangnanke.loginservice;
2
3 import org.springframework.beans.factory.annotation.Value;
4 import org.springframework.boot.SpringApplication;
5 import org.springframework.boot.autoconfigure.EnableAutoConfiguration;
6 import org.springframework.boot.autoconfigure.SpringBootApplication;
7 import org.springframework.cloud.netflix.eureka.EnableEurekaClient;
8 import org.springframework.context.annotation.ComponentScan;
9 import org.springframework.context.annotation.Configuration;
10 import org.springframework.web.bind.annotation.RequestMapping;
11 import org.springframework.web.bind.annotation.RequestParam;
12
13 @EnableEurekaClient
14 @SpringBootApplication

```

```

15 // @Configuration
16 // @ComponentScan
17 // @EnableAutoConfiguration
18 public class LoginServiceApplication {
19
20     public static void main(String[] args) {
21         SpringApplication.run(LoginServiceApplication.class, args);
22     }
23
24 }
25

```

@Configuration , @ComponentScan , @EnableAutoConfiguration , 这三个注解用SpringBootApplication来进行替代，具体三个注解是什么作用，可以去看看springboot注解

4、启动

启动后的变化以及效果，如图：

The screenshot shows the Spring Eureka dashboard. The top navigation bar includes 'HOME' and 'LAST 1000 SINCE STARTUP'. The 'System Status' section displays the following information:

Environment	test	Current time	2018-12-19T15:51:35+0800
Data center	default	Uptime	00:14
		Lease expiration enabled	false
		Renews threshold	3
		Renews (last min)	2

Below the status section, there is a red warning message: "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."

The 'DS Replicas' section shows 'Instances currently registered with Eureka'.

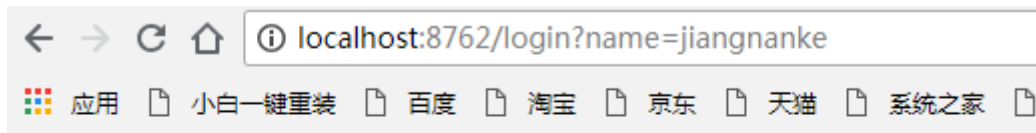
Application	AMIs	Availability Zones	Status
LOGIN-SERVICE	n/a (1)	(1)	UP (1) - zhengfeng.mshome.net:login-service:8762

The 'General Info' section displays the following information:

Name	Value
total-avail-memory	411mb
environment	test
num-of-cpus	4
current-memory-usage	177mb (43%)
server-up-time	00:14
registered-replicas	

The screenshot shows a web browser window with the address bar displaying 'zhengfeng.mshome.net:8762/login?name=jiangnanke'. The browser's address bar also shows '不安全' (Not Secure). The page content displays the message: 'hi jiangnanke ,login is success! the port is:8762'.

两个域名都可以



hi jiangnanke ,login is success! the port is:8762

参考资料

服务的注册与发现Eureka(Finchley版本)：

<https://blog.csdn.net/forezp/article/details/81040925>

SpringCloud 官网：<http://spring.io/projects/spring-cloud>

SpringCloud 开源代码：<https://github.com/spring-cloud>

Spring Cloud 中文官方文档：<https://springcloud.cc/>

<https://github.com/Netflix/eureka>