《Spring Cloud微服务实战》读书笔记

撰写人：Relph Hu

# 说明

原书采用的是Spring Cloud 1.3.7版本，笔者采用了Spring Cloud 2.1.1版本，试图用最新的稳定版进行阅读学习。

读书笔记中的内容，只会将2.1.1版本中的不同列出来，其他未列出的内容与书中一致。

参考链接：<http://blog.didispace.com/spring-cloud-learning/>

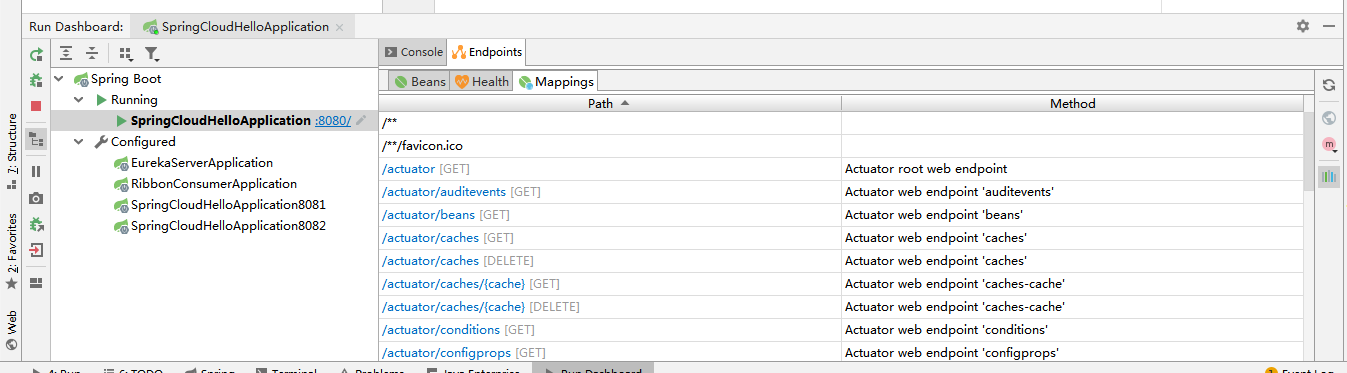
# 微服务构建：Spring Boot

## 初识actuator

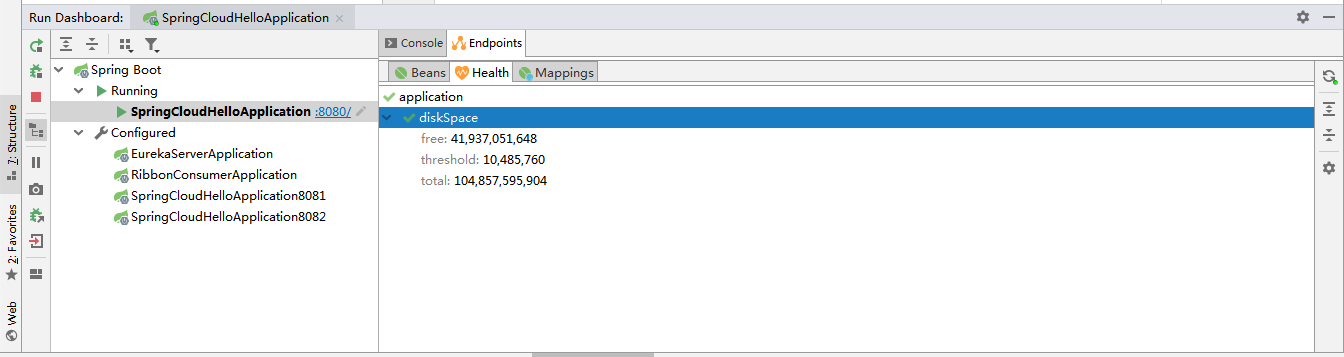
在application.properties文件中配置actuator的相关配置，其中info开头的属性，就是访问info端点中显示的相关内容，在Spring Boot 2.x中，默认只开放了info、health两个端点，剩余的需要自己通过配置management.endpoints.web.exposure.include属性来加载。如果想单独操作某个端点可以使用management.endpoint.端点.enabled属性进行启用或禁用。

|  |
| --- |
| **management.endpoints.web.exposure.include**=**\***  **server.port**=**8080**  **spring.application.name** = **hello-service** |

在Idea中的Run Dashboard->Endpoints->Mappings下面可以看到所有的controller的访问链接。



也可以看到Health的端点信息

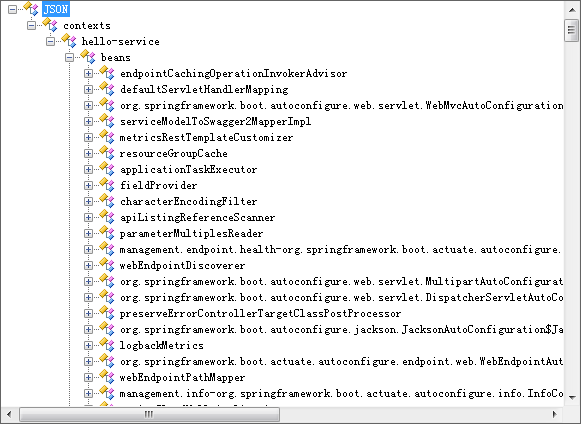


## 初生端口

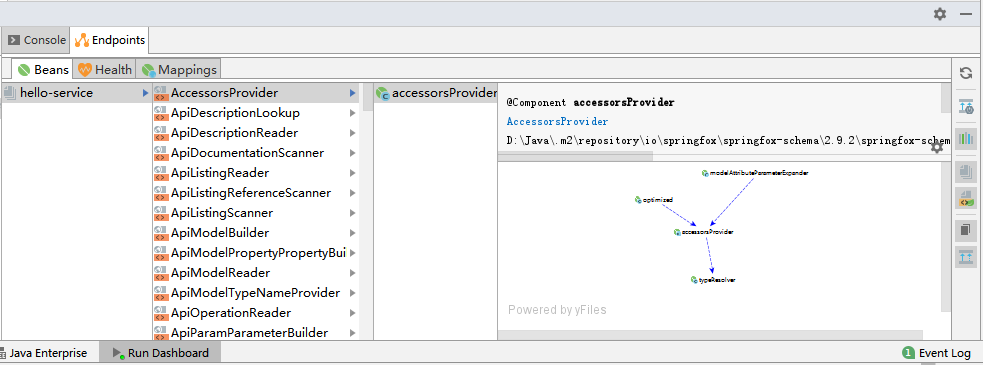
在Spring Cloud 2.X中，/autoconfig端点被替换为/actuator/conditions



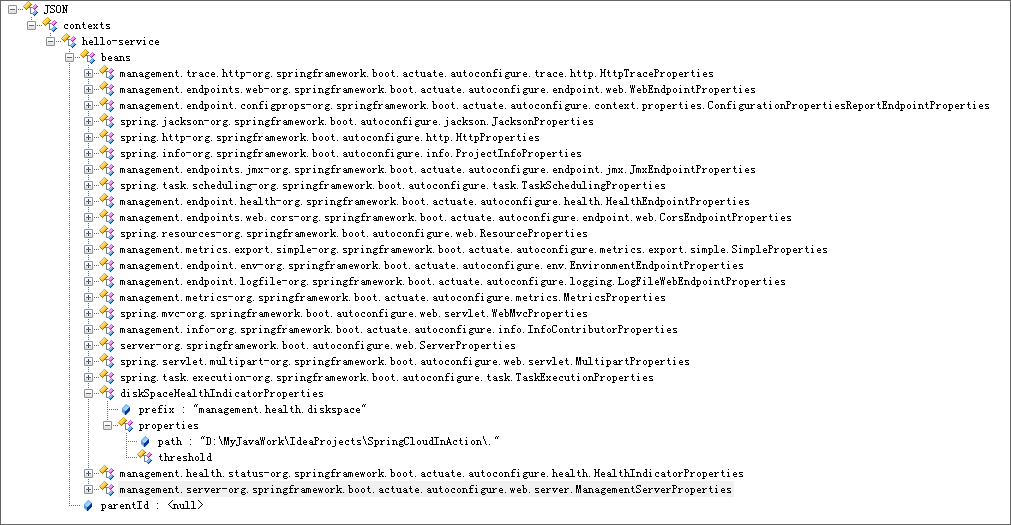
/beans端点被替换为/actuator/beans



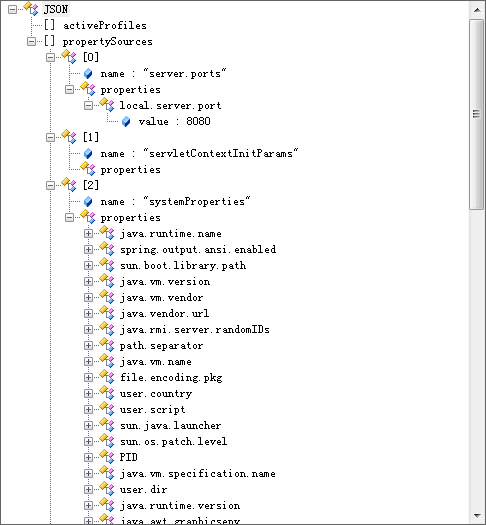
在Idea中也能看到所有启动的beans



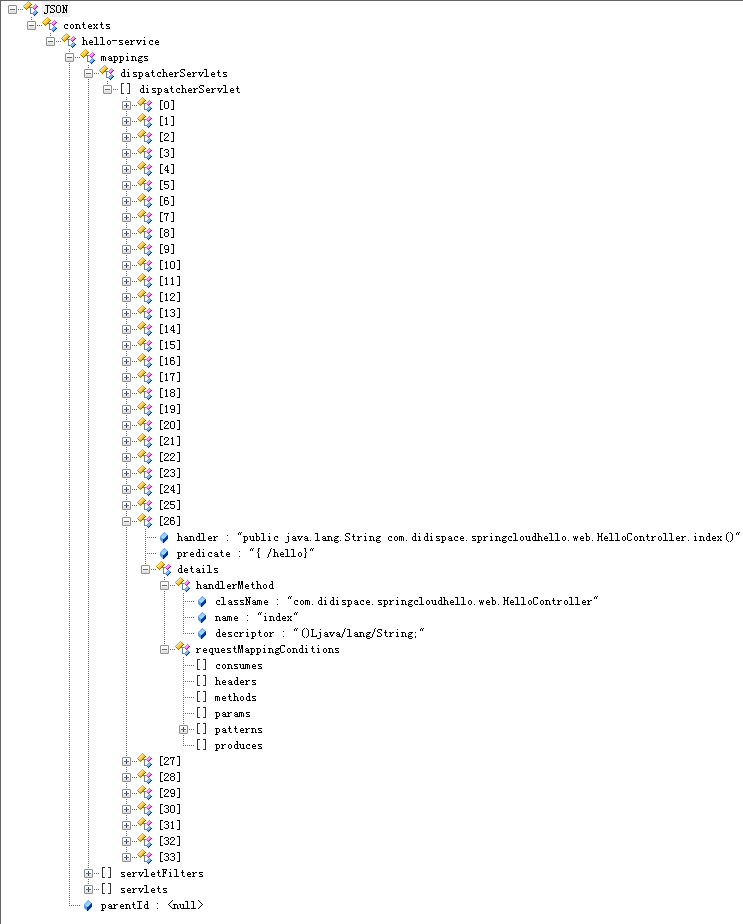
/configprops端点被替换为/actuator/configprops



/env端点被替换为/actuator/env



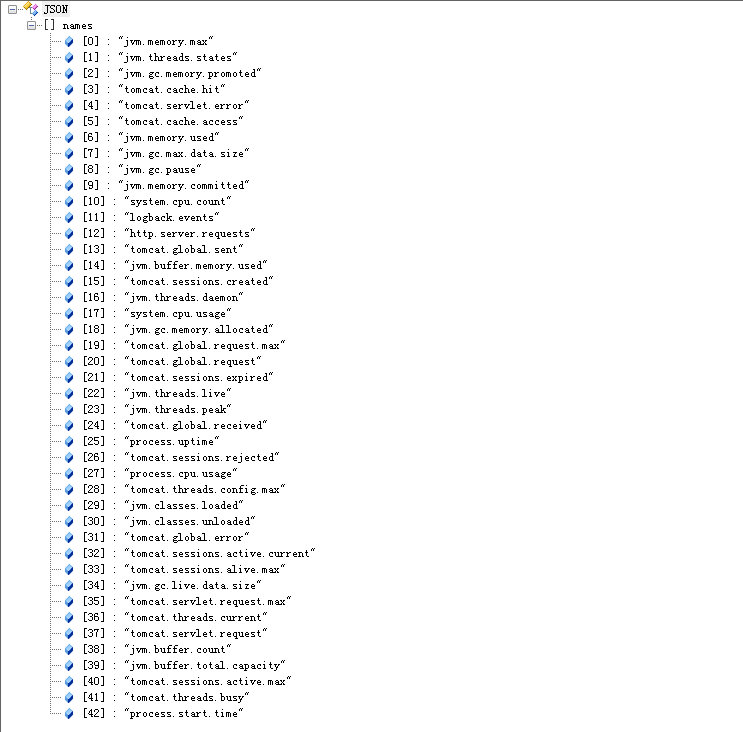
/mappings端点被替换为/actuator/mappings



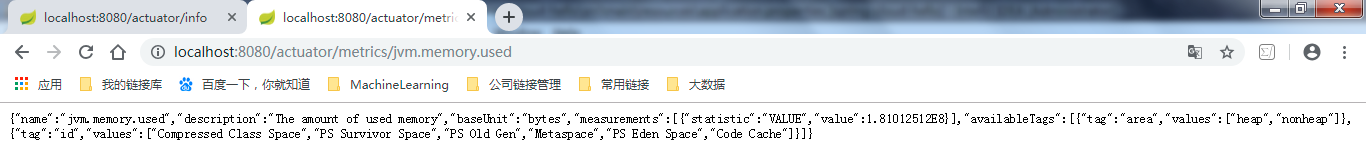
/info端点被替换为/actuator/info



/metrics端点被替换为/actuator/metrics



然后再通过访问各个name，可以单独得到度量指标的值



|  |
| --- |
| **{**  "name"**:** "jvm.memory.max"**,**  "description"**:** "The maximum amount of memory in bytes that can be used for memory management"**,**  "baseUnit"**:** "bytes"**,**  "measurements"**:** **[**  **{**  "statistic"**:** "VALUE"**,**  "value"**:** 5519179775.0  **}**  **],**  "availableTags"**:** **[**  **{**  "tag"**:** "area"**,**  "values"**:** **[**  "heap"**,**  "nonheap"  **]**  **},**  **{**  "tag"**:** "id"**,**  "values"**:** **[**  "Compressed Class Space"**,**  "PS Survivor Space"**,**  "PS Old Gen"**,**  "Metaspace"**,**  "PS Eden Space"**,**  "Code Cache"  **]**  **}**  **]**  **}** |

# 服务治理：Spring Cloud Eureka

## 搭建服务注册中心

Application.properties中的配置修改

|  |
| --- |
| **server.port**=**1111  eureka.instance.hostname**=**localhost** *# 由于该应用为注册中心，设置为false，代表不向注册中心注册自己* **eureka.client.register-with-eureka**=**false** *# 由于注册中心的职责是维护服务实例，它并不需要去检索服务，设置为false* **eureka.client.fetch-registry**=**false** *# 开发时不统计上线率* **eureka.server.enable-self-preservation**=**false eureka.client.service-url.defaultZone**=**http://${eureka.instance.hostname}:${server.port}/eureka/** |

## 注册服务提供者

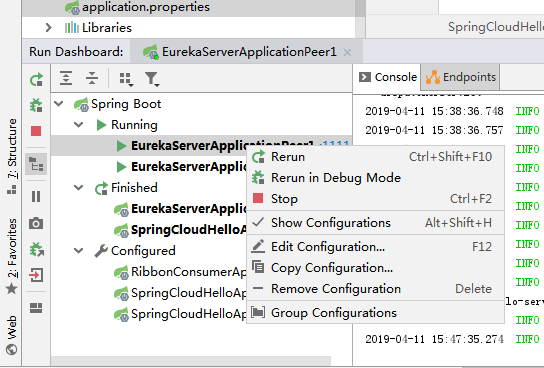
由于2.1.1版本中已经没有了client.getLocalServiceInstance()方法，需要用org.springframework.cloud.client.serviceregistry.Registration取得ServiceId，获取到ServiceInstance。

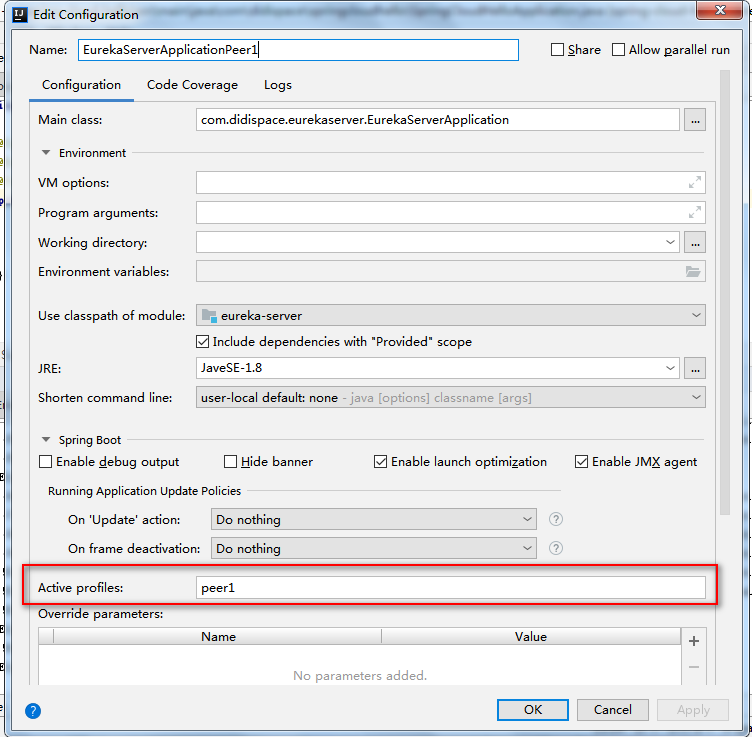
修改之后的代码如下：

|  |
| --- |
| @RestController **public class** HelloController {   **private** Logger **logger** = LoggerFactory.*getLogger*(getClass());   @Autowired  **private** Registration **registration**;   @Autowired  **private** DiscoveryClient **client**;   @RequestMapping(value = **"/hello"**, method = RequestMethod.***GET***)  **public** String index(){  List<ServiceInstance> instances = **client**.getInstances(**registration**.getServiceId());  **if**(!instances.isEmpty()){  **for**(ServiceInstance itm : instances){  **logger**.info(**"/hello, host:"** + itm.getHost() + **", service\_id:"** + itm.getServiceId());  }  }   **return "Hello World"**;  }  } |

## 高可用注册中心

如果需要启动两个注册中心，可以利用Idea中对Run Dashboard的Spring Boot启动进行配置，指定Active profiles的properties文件。





## Eureka详解

### DiscoveryClient接口

@EnableDiscoveryClient源码：该类是一个接口类，主要用来开启DiscoveryClient的实例。

|  |
| --- |
| */\*\*  \* Annotation to enable a DiscoveryClient implementation.  \** ***@author*** *Spencer Gibb  \*/* @Target(ElementType.***TYPE***) @Retention(RetentionPolicy.***RUNTIME***) @Documented @Inherited @Import(EnableDiscoveryClientImportSelector.**class**) **public** @**interface** EnableDiscoveryClient {   */\*\*  \* If true, the ServiceRegistry will automatically register the local server.  \*/* **boolean** autoRegister() **default true**; } |

org.springframework.cloud.client.discovery.DiscoveryClient代码如下：

|  |
| --- |
| */\*\*  \* Represents read operations commonly available to discovery services such as Netflix  \* Eureka or consul.io.  \*  \** ***@author*** *Spencer Gibb  \** ***@author*** *Olga Maciaszek-Sharma  \*/* **public interface** DiscoveryClient **extends** Ordered {   **int *DEFAULT\_ORDER*** = 0;   */\*\*  \* A human-readable description of the implementation, used in HealthIndicator.  \*  \** ***@return*** *The description.  \*/* String description();   */\*\*  \* Gets all ServiceInstances associated with a particular serviceId.  \*  \** ***@param serviceId*** *The serviceId to query.  \** ***@return*** *A List of ServiceInstance.  \*/* List<ServiceInstance> getInstances(String serviceId);   */\*\*  \** ***@return*** *All known service IDs.  \*/* List<String> getServices();   */\*\*  \* Default implementation for getting order of discovery clients.  \*  \** ***@return*** *order  \*/* @Override  **default int** getOrder() {  **return *DEFAULT\_ORDER***;  } } |

DiscoveryClient的注释：

Represents read operations commonly available to discovery services such as Netflix Eureka or consul.io

告诉我们这个接口定义了服务通常的读取操作的抽象方法，分析这个接口下的三个方法，作用分别是：获取描述信息、根据服务id获取服务列表、获取所有的服务的id。

### Eureka Server的URL列表配置

在com.netflix.discovery.endpoint.EndpointUtils代码中可以找到关于eureka.client.serviceUrl.defaultZone加载属性的方法。

|  |
| --- |
| */\*\*  \* Get the list of all eureka service urls from properties file for the eureka client to talk to.  \* 通过客户端配置文件得到Eureka服务URL的列表  \*  \** ***@param clientConfig*** *the clientConfig to use  \** ***@param instanceZone*** *The zone in which the client resides  \** ***@param preferSameZone*** *true if we have to prefer the same zone as the client, false otherwise  \** ***@return*** *an (ordered) map of zone -> list of urls mappings, with the preferred zone first in iteration order  \*/* **public static** Map<String, List<String>> getServiceUrlsMapFromConfig(EurekaClientConfig clientConfig, String instanceZone, **boolean** preferSameZone) {  Map<String, List<String>> orderedUrls = **new** LinkedHashMap<>();  *//* *从配置文件中得到实例的Region，一个微服务应用只可以属于一个Region，默认为default可以通过eureka.client.region属性来定义*   String region = *getRegion*(clientConfig);  *// 得到可用的Zones，可以通过eureka.client.availability-zones属性来设置，availZones默认取defaultZone*  String[] availZones = clientConfig.getAvailabilityZones(clientConfig.getRegion());  **if** (availZones == **null** || availZones.**length** == 0) {  availZones = **new** String[1];  availZones[0] = ***DEFAULT\_ZONE***;  }  ***logger***.debug(**"The availability zone for the given region {} are {}"**, region, availZones);  **int** myZoneOffset = *getZoneOffset*(instanceZone, preferSameZone, availZones);   String zone = availZones[myZoneOffset];  List<String> serviceUrls = clientConfig.getEurekaServerServiceUrls(zone);  **if** (serviceUrls != **null**) {  orderedUrls.put(zone, serviceUrls);  }  **int** currentOffset = myZoneOffset == (availZones.**length** - 1) ? 0 : (myZoneOffset + 1);  **while** (currentOffset != myZoneOffset) {  zone = availZones[currentOffset];  serviceUrls = clientConfig.getEurekaServerServiceUrls(zone);  **if** (serviceUrls != **null**) {  orderedUrls.put(zone, serviceUrls);  }  **if** (currentOffset == (availZones.**length** - 1)) {  currentOffset = 0;  } **else** {  currentOffset++;  }  }   **if** (orderedUrls.size() < 1) {  **throw new** IllegalArgumentException(**"DiscoveryClient: invalid serviceUrl specified!"**);  }  **return** orderedUrls; } |

下面代码实现了得到eureka.client.region，如果没有配置，就设置为defalut。

|  |
| --- |
| */\*\*  \* Get the region that this particular instance is in.  \*  \** ***@return*** *- The region in which the particular instance belongs to.  \*/* **public static** String getRegion(EurekaClientConfig clientConfig) {  String region = clientConfig.getRegion();  **if** (region == **null**) {  region = ***DEFAULT\_REGION***;  }  region = region.trim().toLowerCase();  **return** region; } |

下面这段代码就是EurekaClientConfig的实现，该代码处于

com.netflix.discovery.DefaultEurekaClientConfig类中。获取eureka.client.region属性的值，如果没有配置，默认为us-east-1。

|  |
| --- |
| */\*  \* (non-Javadoc)  \*  \* @see com.netflix.discovery.EurekaClientConfig#getRegion()  \*/* @Override **public** String getRegion() {  DynamicStringProperty defaultEurekaRegion = **configInstance**.getStringProperty(***CLIENT\_REGION\_FALLBACK\_KEY***, Values.***DEFAULT\_CLIENT\_REGION***);  **return configInstance**.getStringProperty(**namespace** + ***CLIENT\_REGION\_KEY***, defaultEurekaRegion.get()).get(); } |

该代码同样处于com.netflix.discovery.DefaultEurekaClientConfig类中。从eureka.client.availabilityZones.defaultZone属性对应的值中去除所有逗号两边的空格，并进行字符串分割。

|  |
| --- |
| */\*  \* (non-Javadoc)  \*  \* @see com.netflix.discovery.EurekaClientConfig#getAvailabilityZones()  \*/* @Override **public** String[] getAvailabilityZones(String region) {  **return configInstance** .getStringProperty(  **namespace** + region + **"."** + ***CONFIG\_AVAILABILITY\_ZONE\_PREFIX***,  ***DEFAULT\_ZONE***).get().split(***URL\_SEPARATOR***); } |