# Spring Boot核心-企业级开发-分布武系统开发

## 微服务、原生云应用

微服务（ Microservice ）是近两年来非常火的概念，它的含义是：使用定义好边界的小的独立组件来做好一件事情。微服务是相对于传统单块式架构而言的。

单块式架构是一份代码，部署和伸缩都是基于单个单元进行的。它的优点是易于部署，但是面临着可用性低、可伸缩性差、集中发布的生命周期以及违反单一功能原则（ Single Responsibility Principle ）。微服务的出现解决了这个问题，它以单个独立的服务来做一个功能，且要做好这个功能。但使用微服务不可避免地将功能按照边界拆分为单个服务，体现出分布式的特征，这时每个微服务之间的通信将是我们要解决的问题。

Spring Cloud 的出现为我们解决分布式开发常用到的问题给出了完整的解决方案。Spring Cloud 基于Spring Boot，为我们提供了配置管理、服务发现、断路器、代理服务等我们在做分布式开发时常用问题的解决方案。

基于Spring Cloud 开发的程序特别适合在Docker 或者其他专业PaaS（平台即服务，如Cloud Foundry ）部署，所以又称作原生云应用（ Cloud Native Application ）。

## Spring Cloud 快速入门

### 配置服务

Spring Cloud 提供了Config Server，它有在分布式系统开发中外部配置的功能。通过Config Server，我们可以集中存储所有应用的配置文件。Config Server 支持在git 或者在文件系统中放置配置文件。可以使用以下格式来区分不同应用的不同配置文件：

|  |
| --- |
| /{application}/{profile}[/{label}]  /{application}-{profile}.yaml  /{lable}/{application}-{profile}.yaml  /{application}-{profile}.properties  /{lable}/{application}-{profile}.properties |

Spring Cloud 提供了注解@EnableConfigServer 来启用配置服务。

### 服务发现

Spring Cloud 通过Netflix OSS 的Eureka 来实现服务发现，服务发现的主要目的是为了让每个服务之间可以互相通信。Eureka Server 为微服务注册中心。

Spring Cloud 使用注解的方式提供了Eureka 服务端（ @EnableEurekaServer ）和客户端(@EnableEurekaClient ）。

### 路由网关

路由网关的主要目的是为了让所有的微服务对外只有一个接口，我们只需访问一个网关地址，即可由网关将我们的请求代理到不同的服务中。

Spring Cloud 是通过Zuul 来实现的，支持自动路由映射到在Eureka Server 上注册的服务。Spring Cloud 提供了注解@EnableZuulProxy 来启用路由代理。

### 负载均衡

Spring Cloud 提供了Ribbon 和Feign 作为客户端的负载均衡。在Spring Cloud 下，使用Ribbon 直接注入一个RestTemplate 对象即可，此RestTemplate 巳做好负载均衡的配置：而使用Feign 只简定义个注解，有＠FeignClient 注解的接口，然后使用@RequestMapping 注解在方法上映射远程的REST 服务，此方法也是做好负载均衡配的。

### 断路器

断路器（ Circuit Breaker ），主要是为了解决当某个方法调用失败的时候，调用后备方法来替代失败的方法，以达到容铺、阻止级联错误等功能。

Spring Cloud 使用@EnableCircuitBreaker 来启用断路器支持，使用@HystrixCommand 的fallbackMethod 来指定后备方法。

Spring Cloud 还给我们提供了一个控制台来监控断赂器的运行情况。通过@EnableHystrixDashboard 注解开启。