|  |
| --- |
| SpringCloud |
| **微服务架构演变** |
|  |
| **2022-10-24 20:53:10** |



1、Spring Cloud Alibaba介绍

[1.1、主要功能](#_1.1、主要功能)

[1.2、组件](#_1.2、组件)

1、Spring Cloud Alibaba简介



#### 1.1、主要功能

1. 服务限流降级：默认支持 WebServlet、WebFlux、OpenFeign、RestTemplate、Spring Cloud Gateway、Dubbo 和 RocketMQ 限流降级功能的接入，可以在运行时通过控制台实时修改限流降级规则，还支持查看限流降级 Metrics 监控。

2. 服务注册与发现：适配 Spring Cloud 服务注册与发现标准，默认集成对应 Spring Cloud 版本所支持的负载均衡组件的适配。

3. 分布式配置管理：支持分布式系统中的外部化配置，配置更改时自动刷新。

消息驱动能力：基于 Spring Cloud Stream 为微服务应用构建消息驱动能力。

4. 分布式事务：使用 @GlobalTransactional 注解， 高效并且对业务零侵入地解决分布式事务问题。

5. 阿里云对象存储：阿里云提供的海量、安全、低成本、高可靠的云存储服务。支持在任何应用、任何时间、任何地点存储和访问任意类型的数据。

6. 分布式任务调度：提供秒级、精准、高可靠、高可用的定时（基于 Cron 表达式）任务调度服务。同时提供分布式的任务执行模型，如网格任务。网格任务支持海量子任务均匀分配到所有 Worker（schedulerx-client）上执行。

7. 阿里云短信服务：覆盖全球的短信服务，友好、高效、智能的互联化通讯能力，帮助企业迅速搭建客户触达通道。

#### 1.2、组件

[**Sentinel**](https://github.com/alibaba/Sentinel)：把流量作为切入点，从流量控制、熔断降级、系统负载保护等多个维度保护服务的稳定性。

[**Nacos**](https://github.com/alibaba/Nacos)：一个更易于构建云原生应用的动态服务发现、配置管理和服务管理平台。

[**RocketMQ**](https://rocketmq.apache.org/)：一款开源的分布式消息系统，基于高可用分布式集群技术，提供低延时的、高可靠的消息发布与订阅服务。

[**Seata**](https://github.com/seata/seata)：阿里巴巴开源产品，一个易于使用的高性能微服务分布式事务解决方案。

[**Alibaba Cloud OSS**](https://www.aliyun.com/product/oss): 阿里云对象存储服务（Object Storage Service，简称 OSS），是阿里云提供的海量、安全、低成本、高可靠的云存储服务。您可以在任何应用、任何时间、任何地点存储和访问任意类型的数据。

[**Alibaba Cloud SchedulerX**](https://cn.aliyun.com/aliware/schedulerx): 阿里中间件团队开发的一款分布式任务调度产品，提供秒级、精准、高可靠、高可用的定时（基于 Cron 表达式）任务调度服务。

[**Alibaba Cloud SMS**](https://www.aliyun.com/product/sms): 覆盖全球的短信服务，友好、高效、智能的互联化通讯能力，帮助企业迅速搭建客户触达通道。

2、nacos

#### 2.1、Dubbo、Feign、RestTemplate

Dubbo、Feign 和 RestTemplate 都是 Java 中用于实现远程调用的工具。它们的区别在于：

1. Dubbo 是一种 RPC 框架，它提供了服务注册、发现、负载均衡、容错等功能，可以让开发者更方便地实现分布式系统的开发。
2. Feign 是一种基于接口的 HTTP 客户端，它可以通过注解的方式定义 HTTP 请求，并且可以与 Spring Cloud 中的服务注册中心集成，实现服务的自动发现和负载均衡。
3. RestTemplate 是 Spring 框架中的一个 HTTP 客户端，它可以发送 HTTP 请求，并且可以通过配置 RestTemplate 的拦截器来实现负载均衡等功能。

总的来说，Dubbo 更适合于大规模的分布式系统，Feign 更适合于微服务架构，而 RestTemplate 则更适合于简单的 HTTP 请求。

#### 2.2、@EnableDiscoveryClient

@EnableDiscoveryClient 的作用都是能够让注册中心发现、并扫描到该服务。

Eureka使用@EnableEurekaClient注解，只对Eureka起作用，@EnableDiscoveryClient 对 Eureka、Zookeeper、Consul 等注册中心都有效。

从 SpringCloud Edgware(Nov 22, 2017) 版本开始, @EnableEurekaClient 和 @EnableDiscoveryClient 注解都可以省略了,只需要在 pom.xml 中引入依赖、在application.yml 上进行相关配置,就可以将微服务注册到注册中心上。但是为了起标识作用,最好还是加上 @EnableDiscoveryClient 注解。