

39 专栏总结

更新时间：2019-08-02 13:42:21



“学习这件事不在乎有没有人教你，最重要的是在于你自己有没有觉悟和恒心。

—— 法布尔”

随着互联网不断快速发展，为了适应业务的快速变化，架构中的服务切分粒度越来越小，微服务架构便成一种必然趋势。**Spring Cloud** 的诞生极大的促进了微服务的发展，让广大的中小企业用很低的技术成本来实施他们的微服务架构。同时 **Spring Cloud** 体系内组件众多，实际生产中需根据企业特点灵活选择使用。

在本系列课程的最后一节，我们做两个回顾：第一个回顾是，回顾专栏课程，根据专栏模块整体回顾一下 **Spring Cloud** 体系；第二个回顾是，我们以组件角色的维度，回顾 **Spring Cloud** 生态体系内的产品，让大家对 **Spring Cloud** 全局性的认识。

课程回顾

本系列课程主要分了五大部分讲述 **Spring Cloud**，五部分内容层层递进，由浅入深的了解 **Spring Cloud** 技术。

第一部分，微服务架构和 **Spring Cloud**

Spring Cloud 是微服务架构落地的最佳技术，那么微服务架构是如何诞生的呢？第一部分给大家介绍了微服务相关的事情，让大家从技术发展、行业趋势的角度来理解，为什么需要 **Spring Cloud**。

Spring Cloud 本身具有很多的故事，比如早期为了快速构建起整个生态，大量采用了 **Netflix** 公司的开源产品，当然任何选择都是有代价的，随着 **Netflix** 出现一系列停更事件，**Spring Cloud** 加大了自研力度。

Spring Cloud 依赖于 **Spring Boot** 进行开发，于是我们需要了解 **Spring Boot** 的基本特性，以及使用 **Spring Boot** 开发 **Spring Cloud** 会带来哪些便利，最后详细的阐述了 **Spring Boot** 和 **Spring Cloud** 之间的关系。

本专栏的第一部分，主要让大家站在一个更高的维度来了解 **Spring Cloud**，这样在我们学习 **Spring Cloud** 产品时，才会对它的使用场景有更深的认识。

第二部分，Spring Cloud 核心组件

第二部分是本专栏最核心的内容，讲解 Spring Cloud 核心组件的使用。从最核心的组件注册中心讲起，介绍了注册中心的由来和基本原理。其中重点介绍了 Eureka 和 Consul 两个产品，分别介绍了他们的特性、工作原理，以及如何使用。

注册中心的使用，让服务提供者和服务调用者进行了解耦，这样服务端才可根据情况动态扩容。解决了微服务架构中服务注册与发现后，如何让客户端高效调用服务端，变成了下一个关注点。Spring Cloud 体系内提供了两款产品 RestTemplate 和 Spring Cloud OpenFeign。

RestTemplate 是从 Spring3.0 开始支持的一个 HTTP 请求工具，它提供了常见的 REST 请求方案的模版。Feign 是一个声明式的 REST 客户端，它的目的就是让 REST 调用更加简单。两者各有特点，专栏给大家详细介绍了两个组件的使用。

在微服务架构中，服务网关是一个非常重要的角色，可以有效隔离内外部请求。本专栏给大家介绍了 Spring 官方出品的网关产品 Spring Cloud Gateway，经过近两年的孵化，如今 Spring Cloud Gateway 已经成为体系内的默认网关技术。

服务容错保护可以保障系统局部出现故障时，避免出现雪崩效应，保障架构整体稳定性。Resilience4j 是最新一代的熔断技术，它非常轻量级，但功能很全面，包含了请求熔断、限流、限时、缓存、隔离以及重试等，Resilience4j 也是 Greenwich 版本中推荐的容错保护技术。

除过上面这些产品，还有三个组件非常重要：Spring Cloud Config 解决了微服务架构下的分布式配置问题；Spring Cloud Bus 整合了消息中间件的使用；Spring Cloud Stream 用于构建高度可扩展的基于事件驱动的微服务，其目的是为了简化消息在 Spring Cloud 应用程序中的开发。三个组件都是在特定领域简化相关产品的使用。

第三部分，Spring Cloud 辅助组件

Spring Cloud 辅助组件是指，可以提供特定功能，帮助 Spring Cloud 监控微服务架构的运行情况，在本次专栏中主要介绍了三款产品：Spring Cloud Sleuth、Zipkin 以及 Spring Boot Admin。

Spring Cloud Sleuth 是为了解决在微服务架构中，分布式链路跟踪的问题，Sleuth 负责收集数据、Zipkin 对收集好的数据进行展示。Spring Boot Admin 在 Spring Boot Actuator 的基础上对数据进行了整合，方便我们以图形化的方式查看集群应用的状态。

第四部分，介绍阿里生态和 Spring Cloud 的关系

2018年10月31日，Spring Cloud Alibaba 正式入驻了 Spring Cloud 官方孵化器，在入驻后不久，阿里就将相关中间件移植到了 Spring Cloud 生态中，比较重要的组件有 Naco、Sentinel、RocketMQ、Dubbo、Seata 等，本专栏为大家介绍了其中最重要的两个组件 Naco 和 Sentinel。

Naco 是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，Nacos 是注册中心的一个产品实现；Sentinel 是一个熔断保护产品，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。从这两个组件也可以看出 Spring Cloud 相当于 Spring Cloud Alibaba 的一个分支。

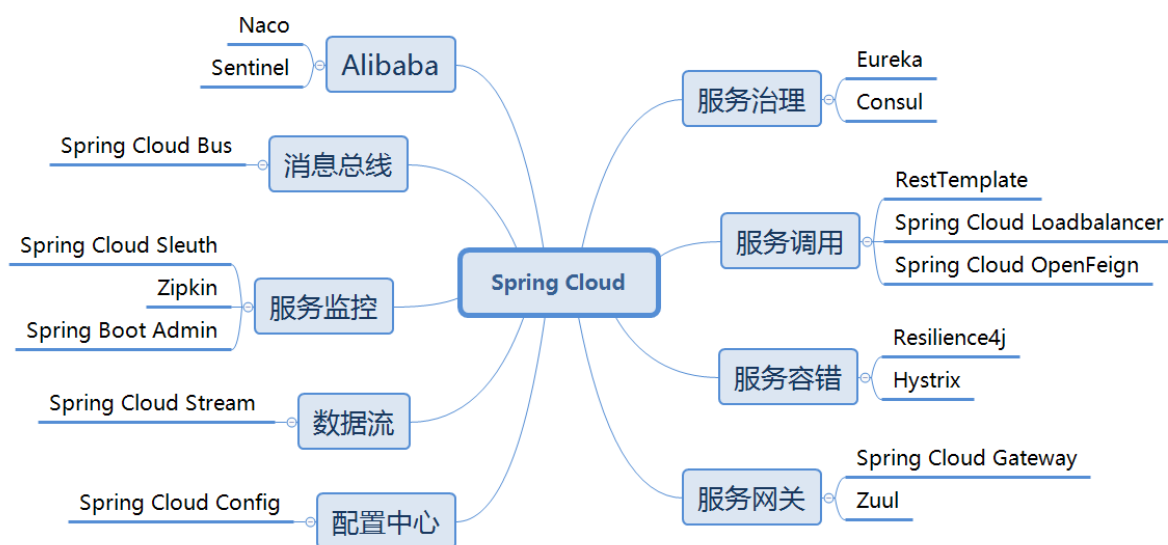
目前 Spring Cloud Alibaba 还没有发布第一个正式版本，却一直在 Spring Cloud Alibaba 体系中添加阿里的相关中间件，相信未来 Spring Cloud Alibaba 也会有更好的发展。

第五部分，专栏总结回顾

第五部分比较简单，分析微服务常见面试题，带着大家回顾一下 Spring Cloud 生态中关键技术点，最后总体回顾一下整个专栏的内容，总结 Spring Cloud 体系生态技术。

Spring Cloud 技术栈

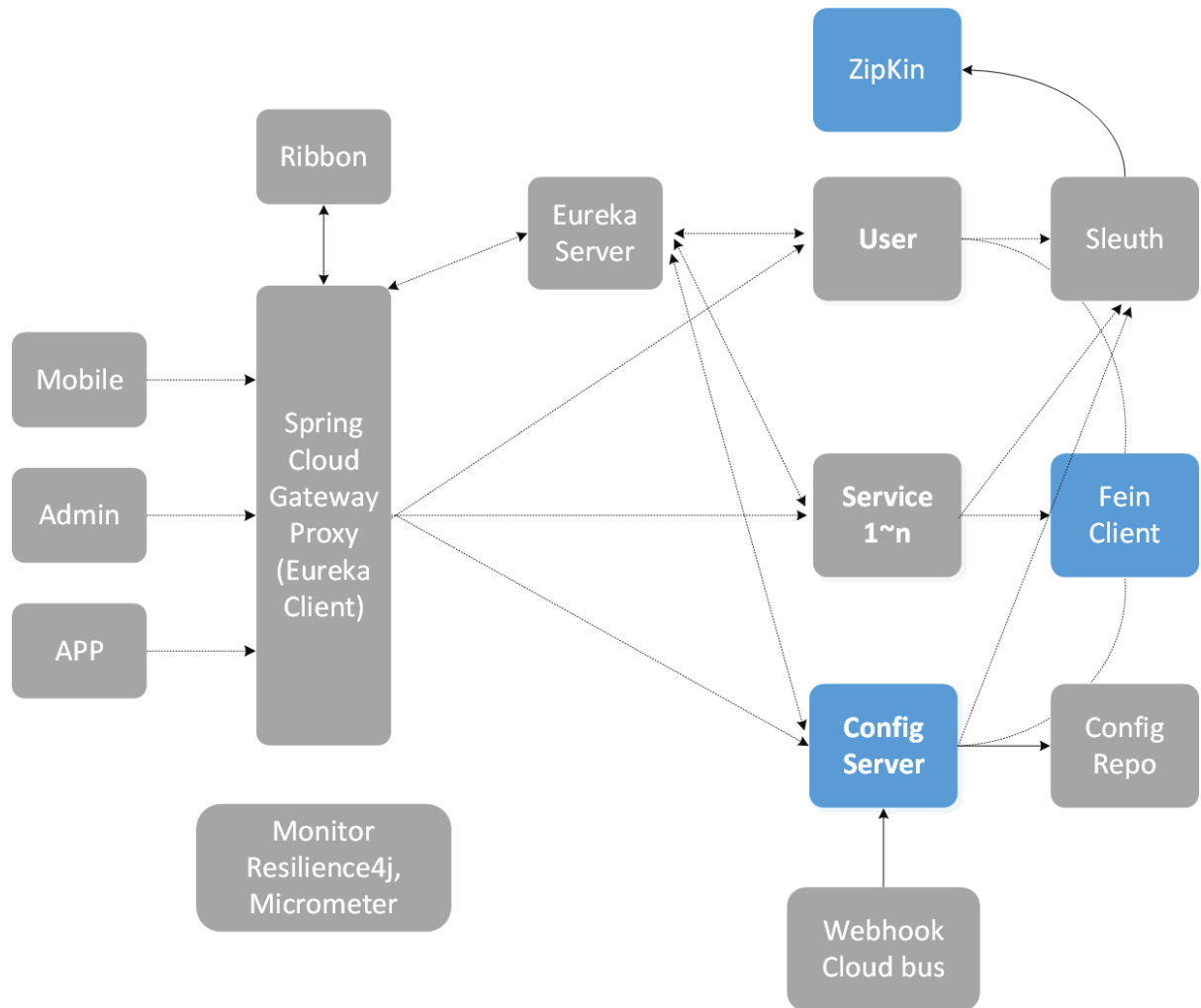
我们再从 Spring Cloud 技术栈的角度，来回顾一下本系列专栏所涉及的相关组件，如下图：



- 服务治理，主要讲解了 Eureka 和 Consul 的使用。
- 服务调用，介绍了 RestTemplate、Spring Cloud Loadbalancer、Spring Cloud OpenFeign 的调用方式。
- 服务容错，详细介绍了 Resilience4j 的使用，简单介绍和 Hystrix 的区别。
- 服务网关，介绍了 Spring Cloud Gateway 的使用，以及 Zuul 的简单介绍。
- 配置中心，配置中心产品 Spring Cloud Config 使用详解。
- 数据流，Spring Cloud Stream 的介绍与使用。
- 服务监控，分布式链接跟踪 Spring Cloud Sleuth、图形展示 Zipkin，以及 Spring Boot Admin 的使用说明。
- 消息总线，介绍了 Spring Cloud Bus 的使用。
- Spring Cloud Alibaba，介绍了阿里巴巴在 Spring Cloud 体系内发展，重点介绍了 Naco 和 Sentinel 的使用。

总结

我们从整体上来看一下 Spring Cloud 各个组件如何来配套使用：



从上图可以看出 Spring Cloud 各个组件相互配合，合作支持了一套完整的微服务架构。

- 前端请求通过服务网关 Spring Cloud Gateway，将请求转发到相关服务。
- Eureka 负责服务的注册与发现，很好将各服务连接起来，替代技术方案 Consul。
- Resilience4j 负责监控服务之间的调用情况，连续多次失败进行熔断保护。
- Micrometer 负责监控 Resilience4j 的熔断情况，并给予图形化的展示
- Spring Cloud Config 提供了统一的配置中心服务
- 当配置文件发生变化时，Spring Cloud Bus 负责通知各服务去获取最新的配置信息
- 图中没有画出，可以根据 Spring Cloud Stream 构建消息驱动的微服务
- 最后我们使用 Sleuth+Zipkin 将所有的请求数据记录下来，方便我们进行后续分析

Spring Cloud 从设计之初就考虑了绝大多数互联网公司架构演化所需的功能，如服务发现注册、配置中心、消息总线、负载均衡、断路器、数据监控等。这些功能都是以插拔的形式提供出来，方便我们系统架构演进的过程中，可以合理的选择需要的组件进行集成，从而在架构演进的过程中会更加平滑、顺利。

微服务架构是一种趋势，Spring Cloud 提供了标准化的、全站式的技术方案，有效推进服务端软件系统技术水平的进步。

使用 Spring Cloud 二年多，深感 Spring Cloud 给中小企业架构带来的改变。也希望大家通过此专栏课程的学习，能对 Spring Cloud 有更进一步的了解，在工作中有所帮助。

}

