05 Spring Cloud 和 Spring Boot 之间的密切关系

更新时间：2019-06-19 17:54:01

> 人不可有傲气，但不可无傲骨。
>
> ——徐悲鸿

前面两节给大家介绍了什么是 Spring Cloud ， Spring Boot 的设计原理，同时我们也了解到 Spring Cloud 是用 Spring Boot 构建开发的，因此具备了 Spring Boot 快速开发的一些特性。那么 Spring Cloud 和 Spring Boot 具体是如何配合的呢？

想要了解 Spring Cloud 和 Spring Boot 是如何配合使用的，就必须先要了解 Spring Boot 的 Starter 机制。

Spring Boot Starter

Spring Boot Starter 是 Spirng Boot **约定优于配置**理念的最佳实现。Spring Boot Starter 有两个核心组件：自动配置代码和提供自动配置模块及其它有用的依赖。也就意味着，当项目中引入某个组件的 Starter ，项目启动时就会针对此组件进行默认配置，从而达到“开箱即用”，一般情况下仅需要少量的配置或者不配置即可使用组件对应的功能。

Spring Boot 由众多 Starter 组成，随着版本的推移 Starter 家族成员也与日俱增。在传统 Maven 项目中通常将一些层、组件拆分为模块来管理，以便相互依赖复用，在 Spring Boot 项目中我们则可以创建自定义 Spring Boot Starter 来达成该目的。

Spring Boot 拥有强大的融合社区开源软件的能力。在没有使用 Spring Boot 之前，我们需要按照每个开源软件的特性，将对应的组件包集成到我们的开发项目中，因为每个组件的设计理念和开发团队都不一致，因此会有很多不同的调用风格在我们的项目中。

Spring Boot 整合了主流的开源软件形成了一系列的 Starter，让我们有了一致的编程体验来集成各种软件，Spring Boot 在集成的时候做了大量的优化，让我们在集成的时候往往只需要很少的配置和代码就可以完成。可以说各种 Starters 就是 Spring Boot 最大的优势之一。

因此 Spring Cloud 在构建自己的组件产品时，也充分地吸收了 Spring Boot Starter 的设计理念，每一个 Spring Cloud 组件都包装成 Starter ，在项目需要使用 Spring Cloud 相关组件时，只需要引入对应的 Spring Cloud Starter 即可。

也就是说 Spring Cloud 在构建注册中心时，将 Eureka 、Consul 等框架用 Spring Boot 技术包装成对应的 Starter 组件，微服务体系中的其它产品也是这个思路。 这样 Spring Cloud 就完全将整个体系构建在了 Spring Boot 之上，充分利用了 Spring Boot 约定优于配置的理念，重构了生态内的产品，让我们使用 Spring Cloud 产品更加快速便捷。

Spring Cloud 体系介绍

按照 Starter 的设计理念，Spring Cloud 共集成了 25 个子项目，每个项目包含一个或者多个第三方的组件或者框架：

1、Spring Cloud Config 配置中心，利用 Git 集中管理程序的配置
2、Spring Cloud Netflix 集成众多 Netflix 的开源软件
3、Spring Cloud Bus 消息总线，利用分布式消息将服务和服务实例连接在一起，用于在一个集群中传播状态的变化
4、Spring Cloud Cloudfoundry 利用 Pivotal Cloudfoundry 集成你的应用程序
5、Spring Cloud Open Service Broker 为建立管理云托管服务的服务代理提供了一个起点
6、Spring Cloud Cluster 基于 Zookeeper， Redis， Hazelcast， Consul 实现的领导选举和平民状态模式的抽象和实现
7、Spring Cloud Consul 基于 Hashicorp Consul 实现的服务发现和配置管理
8、Spring Cloud Security 在 Zuul 代理中为 OAuth2 rest 客户端和认证头转发提供负载均衡
9、Spring Cloud Sleuth SpringCloud应用的分布式追踪系统，和 Zipkin，HTrace，ELK 兼容
10、Spring Cloud Data Flow 一个云本地程序和操作模型，组成数据微服务在一个结构化的平台上
11、Spring Cloud Stream 基于 Redis，Rabbit，Kafka 实现的消息微服务，简单声明模型用以在 Spring Cloud 应用中收发消息
12、Spring Cloud Stream App Starters 基于 Spring Boot 为外部系统提供 Spring 的集成
13、Spring Cloud Task 短生命周期的微服务，为 Spring Booot 应用简单声明添加功能和非功能特性
14、Spring Cloud Task App Starters
15、Spring Cloud Zookeeper 服务发现和配置管理基于 Apache Zookeeper
16、Spring Cloud AWS 快速和亚马逊网络服务集成
17、Spring Cloud Connectors 便于 PaaS 应用在各种平台上连接到后端像数据库和消息经纪服务
18、Spring Cloud Starters （项目已经终止并且在 Angel.SR2 后的版本和其他项目合并）
19、Spring Cloud CLI 插件用 Groovy 快速创建 Spring Cloud 组件应用
20、Spring Cloud Contract，一个消费者驱动的、面向 Java 的契约框架
21、Spring Cloud Gateway，Spring Cloud 官方推出的网关服务
22、Spring Cloud OpenFeign，提供 OpenFeign 集成到 Spring Boot 应用中的方式，为微服务架构下服务之间的调用提供解决方案
23、Spring Cloud Pipelines，实现了一个基于 Spring Cloud 架构的研发自动化流程
24、Spring Cloud Function，促进函数作为主要的开发单元。该项目提供了一个通用的模型，用于在各种平台上部署基于函数的软件，包括像 Amazon AWS Lambda 这样的 FaaS（函数即服务，function as a service）平台
25、Spring Cloud Kubernetes，Spring Cloud 提供对 Kubernetes 的支持

> 这个数量还在一直增加...

25 个框架中有一部分（主要是 Netflix 相关）都已经处于维护状态，有些刚刚加入 Spring Cloud 生态中。本系列专栏将覆盖 Spring Cloud 体系的绝大部分产品，重点介绍日常项目使用最多的技术热点，方便大家学完专栏后快速实践。

Spring Cloud 和 Spring Boot 版本对应关系

我们知道 Spring Cloud 是依赖 Spring Boot 构建的，因此两者必须配合来使用，如果相互版本没有匹配好，在使用的过程中会报错。

Spring Cloud	Spring Boot
Greenwich	2.1.x
Finchley	2.0.x
Edgware	1.5.x
Dalston	1.5.x
Camden	1.4.x
Brixton	1.3.x
Angel	1.2.x

Spring Cloud 仍然在快速发展中，在使用的过程中偶尔可能会遇到一些 bug ，在出现这些问题的时候，首先确认此问题只出现在特定的版本上。如果是，优先选择升级小版本解决类似的问题。

Spring Boot 1.x 和 Spirng Boot 2.x 系列之间升级很多 API，新老版本并不兼容，考虑从低版本升级到高版本的同学，应注意版本之间的差异性。

Spring 、Spring Boot 和 Spring Cloud 的关系

我们再来重新梳理一下 Spring 、Spring Boot 和 Spring Cloud 三者之间的关系。

Spring 最初的两大核心功能 Spring Ioc 和 Spring Aop 成就了 Spring，Spring 在这两大核心功能上不断地发展，才有了 Spring 事务、Spring MVC 等一系列伟大的产品，最终成就了 Spring 帝国，到了后期 Spring 几乎可以解决企业开发中的所有问题。

Spring Boot 是在强大的 Spring 帝国生态基础上面发展而来，发明 Spring Boot 不是为了取代 Spring ，是为了让人们更容易地使用 Spring 。所以说没有 Spring 强大的功能和生态，就不会有后期的 Spring Boot 火热。 Spring Boot 使用约定优于配置的理念，重新重构了 Spring 的使用，让 Spring 后续的发展更有生命力。

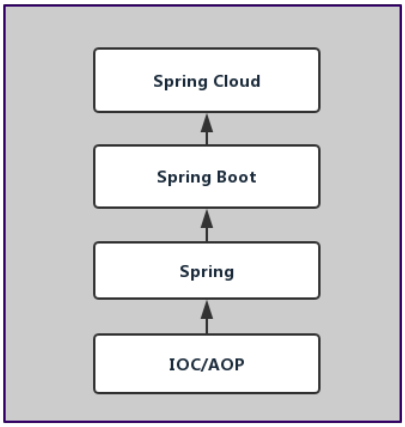
Spring 并没有重复制造轮子，它只是将目前各家公司开发的比较成熟、经得起实际考验的服务框架组合起来，通过 Spring Boot 风格进行再封装，屏蔽掉了复杂的配置和实现原理，最终给开发者留出了一套简单易懂、易部署和易维护的分布式系统开发工具包。

Spring Cloud 是一系列框架的有序集合。它利用 Spring Boot 的开发便利性巧妙地简化了分布式系统基础设施的开发，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，都可以用 Spring Boot 的开发风格做到一键启动和部署。

Spring Cloud 是为了解决微服务架构中服务治理而提供的一系列功能的开发框架，并且 Spring Cloud 是完全基于 Spring Boot 而开发。Spring Cloud 利用 Spring Boot 特性整合了开源行业中优秀的组件，整体对外提供了一套在微服务架构中服务治理的解决方案。

综上我们可以这样来理解，正是由于 Spring Ioc 和 Spring Aop 两个强大的功能才有了 Spring ，Spring 生态不断的发展才有了 Spring Boot ，使用 Spring Boot 让 Spring 更易用更有生命力，Spring Cloud 是基于 Spring Boot 开发的一套微服务架构下的服务治理方案。

以下为它们之间的关系。



Spring Boot 和 Spring Cloud 的区别

Spring Boot 专注于快速方便地开发单个个体微服务。Spring Cloud 关注全局的微服务协调整理治理框架，它将 Spring Boot 开发的一个个单体微服务整合并管理起来， 为各个服务之间提供配置管理、服务发现、断路器、路由、微代理、事件总线、全局锁、精选决策、分布式会话等集成服务。

Spring Boot 可以离开 Spring Cloud 独立开发项目，但是 Spring Cloud 离不开 Spring Boot，属于依赖关系。Spring Boot 专注于快速、方便地开发单个微服务个体，Spring Cloud 关注全局的服务治理框架。

小结

Spring Cloud 深度融合了 Spring Boot ，定制了很多 Spirng Cloud Starter 组件，当我们需要使用 Spring Cloud 对应的组件时，只需要引入对应的 `spring-cloud-starter-xxx` 即可，极大地方便了 Spring Cloud 子项目的部署和使用。

Spring Cloud 深度使用了 Spirng Boot ，因此 Spring Boot 具备的特性，Spring Cloud 也几乎都具有，Spring Boot 的这些特性以及快速开发的模式，极大地促进了 Spring Cloud 的发展。

本专栏所有代码git地址：https://github.com/justdojava/spring-cloud-learning

本文作者：纯洁的微笑、江南一点雨

04 Spring Boot 的设计理念和简单实践

06 Eureka 服务注册中心的搭建

精选留言 0

欢迎在这里发表留言，作者筛选后可公开显示

目前暂无任何讨论