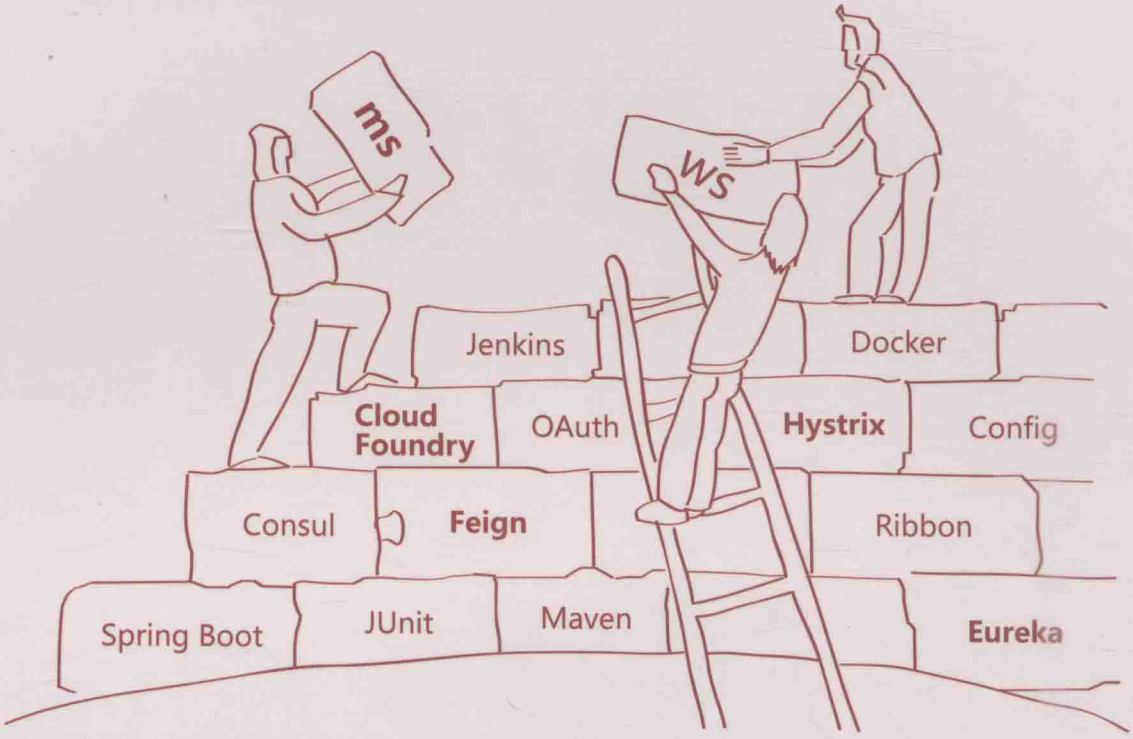
10年IT老兵亲述Spring Cloud实战经验

有思路 / 有案例 / 能落地!



Spring Cloud开发

从入门到实战

王 勇 编著

从Spring Boot微框架搭建、Spring Cloud常用组件应用，到微服务案例实战，层层剥茧式分析，全流程讲解Spring Cloud开发应用，实战案例拿来就用，快速上手。

中国水利水电出版社
www.waterpub.com.cn

Spring Cloud 开发从入门到实战

王 勇 编著

中国水利水电出版社
www.waterpub.com.cn
·北京·

内 容 提 要

《Spring Cloud开发从入门到实战》以Spring Cloud微服务架构为中心，全面系统地介绍了Spring Cloud常用组件的应用，以及微服务涉及的相关技术。本书内容包括：微服务介绍、微框架Spring Boot、服务注册与发现、服务的提供者与消费者、模板引擎、服务的雪崩与熔断、分布式配置中心、API网关、Cloud Foundry、消息驱动、单点登录、Activity工作流、ElasticSearch、ELK Stack、多线程、Redis缓存技术、微服务监控、API文档、持续集成和金丝雀部署，最后以Spring Cloud实战案例来进一步演练Spring Cloud的微服务解决方案。

《Spring Cloud开发从入门到实战》语言简练，内容通俗易懂，实用性强，结构清晰，层层剥茧式分析、全流程实例讲解Spring Cloud核心组件应用与微服务开发。实战案例可以拿来就用，帮助初学者快速上手。本书内容全面，读者不但可以系统地学习Spring Cloud的相关知识，而且还可以全面掌握微服务架构应用的设计、开发、部署和运维等知识。

《Spring Cloud开发从入门到实战》适合Spring Cloud的入门读者阅读，也适合致力于互联网开发和Java编程开发的进阶读者阅读。对微服务架构有兴趣的运维人员及数据库管理人员亦可选择此书阅读。本书也可以作为相关培训机构的教材使用。

图书在版编目（CIP）数据

Spring Cloud 开发从入门到实战 / 王勇编著 . — 北京：
中国水利水电出版社 , 2020.6

ISBN 978-7-5170-8439-6

Ⅰ . ① S… Ⅱ . ①王… Ⅲ . ①互联网络—网络服务器
Ⅳ . ① TP368.5

中国版本图书馆 CIP 数据核字 (2020) 第 035063 号

书　　名	Spring Cloud 开发从入门到实战 Spring Cloud KAIFA CONG RUMEN DAO SHIZHAN
作　　者	王勇　编著
出版发行	中国水利水电出版社 （北京市海淀区玉渊潭南路 1 号 D 座 100038） 网址：www.waterpub.com.cn E-mail：zhiboshangshu@163.com 电话：（010）62572966-2205/2266/2201（营销中心）
经　　售	北京科水图书销售中心（零售） 电话：（010）88383994、63202643、68545874 全国各地新华书店和相关出版物销售网点
排　　版	北京智博尚书文化传媒有限公司
印　　刷	三河市鑫焱淼印装有限公司
规　　格	190mm×235mm　16 开本　17.5 印张　416 千字
版　　次	2020 年 6 月第 1 版　2020 年 6 月第 1 次印刷
印　　数	0001—5000 册
定　　价	69.80 元

前　言

Spring 全家桶在 Java 世界的地位很重要，它不仅为 Java 开发者证明了基于注解开发、AOP（面向切面编程）开发以及面向接口开发能够给程序带来极大的灵活性，而且带来了依赖注入、声明式事务、统一的异常处理、模块自动化加载、更简单的 Maven 管理、更简单的单元测试等优秀的开发实践。

本书采用大量的代码与案例分析，行文深入浅出、图文并茂，将枯燥生硬的理论知识用诙谐幽默、浅显直白的口语娓娓道来。本书抛开深奥的理论化条文，除了必备的基础理论知识介绍外，绝不贪多求全，特别强调实务操作、快速上手，绝不囿于示意与演示，更注重实战展示——从如何创建Spring Boot、如何注册服务，到调用服务、服务熔断、案例分析。随着本书的介绍，您的Spring Cloud学习之旅一定会成为一种难忘的体验。

因作者水平有限，本书难免存有疏漏和不当之处，敬请指正。

本书特色

● 本书内容实用、详略得当，讲解符合初学者的认知规律，示例通俗易懂，且易于构建、运行和测试，能够让读者在学习微服务架构时快速进入实战。

● 本书内容丰富，不仅涵盖Spring Cloud的核心组件，还介绍了如何通过Spring Boot搭建微服务，并介绍了Kafka、Docker和Redis等流行技术。

本书内容及体系结构

本书共21章，首先从微服务基础框架Spring Boot讲起；其次重点讲述了Spring Cloud中的核心组件；最后介绍了微服务涉及的相关技术。

第1章　什么是微服务：从面向服务的架构（SOA）讲到微服务原则与优势，最后以Spring Cloud与Dubbo对比的方式，阐明微服务Spring Cloud的优势。

弟2章　微框架Spring Boot：Spring Boot是一个Spring框架模块，它为Spring框架提供RAD（快速应用程序开发）功能，它高度依赖启动器模板功能，该功能非常强大且完美无缺。Spring Boot同样也是Spring Cloud的重要组成部分。

第3章　从服务注册与发现说起：在微服务中，消费者为了完成一次服务请求，需要知道具体服务的详细地址（IP和端口）。传统应用都运行在物理服务器上，服务实例的网络位置都是相对固

定的。怎样从一个经常变更的配置中读取网络位置显得尤为重要。

第4章 服务提供者与服务消费者的关系：什么是服务提供者和服务消费者？服务提供者是指服务的被调用方，即为其他服务提供服务的服务；服务消费者是指服务的调用方，即依赖其他服务的服务。

第5章 模板引擎：为了使用户页面和业务数据相互分离而产生，它将从后台返回的数据生成特定格式的文档。用于网站的模板引擎就是生成HTML文档。

第6章 服务的雪崩与熔断：典型的分布式系统由许多协作在一起的服务组成，这些服务容易出现故障或延迟响应。如果服务失败，可能会影响性能的其他服务，并可能使应用程序的其他部分无法访问，或者在最坏情况下会导致整个应用程序崩溃。

第7章 分布式配置中心：随着服务/业务的越来越多，配置文件更是眼花缭乱，每次不知道因为部署/安装问题浪费多少时间，更不知道因为配置问题出过多少问题。如果采用分布式的开发模式，需要的配置文件将会随着服务增加而不断增多。某一个基础服务信息变更，都会引起一系列的更新和重启，导致运维人员苦不堪言，并且也容易出错。配置中心便是解决此类问题的灵丹妙药。

第8章 API网关:API网关是微服务架构中很重要的一个部分，是发起每个请求的入口，也可以在网关上做协议转换、权限控制、请求统计和限流等其他工作。

第9章 Cloud Foundry：Cloud Foundry是一个开源平台即服务（PaaS），提供云、开发人员框架和应用程序服务的选择。它是开源的，由Cloud Foundry Foundation管理。最初的Cloud Foundry由VMware开发，目前由GE、EMC和VMware的合资公司Pivotal管理。

第10章 消息驱动：Spring Cloud Stream是一个用来为微服务应用构建消息驱动能力的架构，为一些供应商的消息中间件产品提供个性化的自动化配置实现，并且引入了发布—订阅、消费组以及分区三个核心概念。

第11章 单点登录：单点登录 (Single Sign On，SSO)就是把多个系统的登录验证整合在一起，这样，无论用户登录任何一个应用，都可以直接以登录过的身份访问其他应用，不必每次都访问其他系统再登录。

第12章 Activity工作流：Activity实现了工作流程的自动化，提高了企业运营的效率、改善了企业资源的利用、提高了企业运作的灵活性和适应性、提高了量化考核业务处理的效率、减少了浪费。流程图就像流水线一样，张三请完假，相应地李四就会收到张三的任务审批申请，若通过，则流程结束；若不通过，就会通知张三，张三可以再次发起申请。

第13章 ElasticSearch：ElasticSearch是一个基于Lucene的搜索服务器。它提供了一个分布式多用户能力的全文搜索引擎，基于RESTful Web接口。ElasticSearch是用Java开发的，并作为Apache许可条款下的开放源码发布，是当前流行的企业级搜索引擎。其设计用于云计算中，能够实现实时搜索，不仅稳定、可靠、快速，而且安装使用方便。

第14章 ELK Stack：通过使用微服务，我们已经能够解决许多遗留问题，并且它允许创建稳定的分布式应用程序，并对代码、团队规模、维护、发布周期、云计算等进行所需的控制。但它也引入了一些挑战，如分布式日志管理、查看在许多服务中分布的完整事务的日志与一般的分布式调试的能力。ElasticSearch、Logstash和Kibana一起称为ELK Stack，它们用于实时搜索、分析

和可视化日志数据。

第15章 多线程：多线程是指从软件或者硬件上实现多线程并发执行的技术。具有多线程能力的计算机因有硬件支持而能够在同一时间执行多于一个线程，进而提升整体处理性能。线程可以获得更大的吞吐量，但是开销很大，如线程栈空间的大小开销、切换线程需要的时间开销，所以通过线程池进行重复利用，当线程使用完毕之后，就放回线程池，避免创建与销毁的开销。

第16章 Redis缓存技术：Redis基于内存，也可以基于磁盘持久化NoSQL数据库，使用C语言开发。Redis开创了一种新的数据存储思路，使用Redis，不用在面对功能单调的数据库时把精力放在处理如何将大象放进冰箱这样的问题上，而是利用Redis灵活多变的数据结构和数据操作为不同的大象构建不同的冰箱。

第17章 微服务监控：由于在微服务体系下，各种服务众多，仅靠人力维护服务不现实，成本极高，因此微服务监控很有必要。

第18章 API文档：随着微服务架构的日益普及，服务与服务直接对接也变得日益密切起来，REST风格变得大势所趋。Swagger是为了描述一套标准的而且是和语言无关的REST API的规范。对于外部调用者来说，只通过Swagger文档即可清楚Server端提供的服务，而不需要阅读源码或接口文档说明。官网上有关于Swagger的丰富的资源，包括Swagger Editor、Swagger UI以及Swagger为各种开发语言提供的SDK。这些资源为REST API的提供者以及调用者提供了极大的便利。

第19章 持续集成：介绍微服务为什么会谈到自动化部署？“互联网+”的需要。在信息越来越繁杂的互联网时代，公司运行的项目越来越多，项目相关服务繁多，服务之间存在复杂的依赖关系，运维与管理任务越来越繁重，手工交付需要花费很多的人力与时间，且安全性和时效性均无法保证。随着企业对版本上线质量和速度的要求越来越高，敏捷开发、Devops的接受度越来越高。传统的交付方式因为项目之间缺少依赖、环境不一致、版本不一致、人为操作失误等情况，使得项目交付过程中问题不断，而互联网企业发展节奏快、版本发布频率高，上线出故障影响面广、影响度高，因而企业对敏捷开发、持续集成、自动发布都有强烈的需求。

第20章 金丝雀部署：每次部署到生产环境时，我们都会担心更改将会影响用户体验。无论使用什么技术或策略进行部署，可能出错的事情都会出错，这是墨菲定律。

第21章 Spring Cloud实战：项目选用Spring Cloud微服务解决方案，框架的搭建基于Spring Boot，使用到的技术有Feign、Hystrix、Ribbon、Eureka、Cloud-Config、OAuth2.0、ES。

本书读者对象

本书适用于所有Java语言的编程开发人员，所有对Spring Boot感兴趣并希望使用Spring Boot开发框架进行开发的人员，已经使用过Spring Boot框架但希望更好地使用Spring Boot的开发人员，以及系统设计师、架构师等设计人员均可选择本书学习。同时，本书对运维人员和DBA（数据库管理者）等也具有一定的参考价值。

本书资源的获取及联系方式

使用手机微信扫描下面的二维码，获取本书配套资源（本书配套资源包）、与作者交流本书的技术问题（在线答疑），或者付费与作者进行技术拓展问答（付费技术拓展）。

致谢

本书能够顺利出版，是作者、编辑和所有审校人员共同努力的结果，在此表示深深的感谢。同时，祝福所有读者在职场一帆风顺。

编　　者

目　录

第1章　什么是微服务

微服务是业界最新的流行语，每个人似乎都在以这种或那种方式谈论它或者使用它。

1.1　面向服务的架构

面向服务的架构（SOA）是一种软件体系结构，应用程序的不同组件通过网络上的通信协议向其他组件提供服务。通信可以是简单的数据传递，也可以是两个或多个服务彼此协调连接。

SOA架构中主要有两个角色：服务提供者（Provider）和服务使用者（Consumer）。软件代理可以扮演这两个角色。Consumer层是用户（人、应用程序或第三方的其他组件）与SOA交互的点，Provider层由SOA架构内的所有服务构成。

1.2　微　服　务

微服务是SOA之后越来越流行的体系结构模式之一。如果您关注行业趋势，就会发现，如今商业机构不再像几年前那样，开发大型应用程序，来管理端到端之间的业务功能，而是选择快速灵活的微服务。

微服务有助于打破大型应用程序的界限，并在系统内部构建逻辑上独立的小型系统。例如，使用Amazon AWS，可以轻松构建云应用程序，这是微服务一个很好的例子，如图1.1所示。

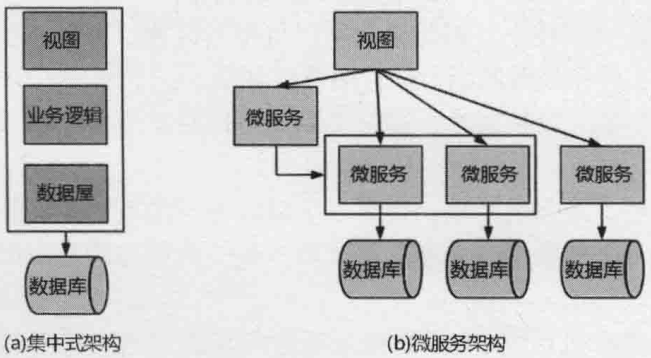


图1.1　微服务架构

每个微服务都有自己的业务层和数据库。在生产环境中，任意微服务的崩溃都不会影响其他微服务正常使用。

1.3 微服务的原则

微服务的原则如下。

1. 单一服务原则

一个单元(一个类、函数或者微服务)应该有且只有一个职责。无论如何，一个微服务不应该包含多于一个职责。

2. 围绕业务建模

根据业务定义的接口，比根据技术框架定义的接口更稳定，不仅可以帮助我们形成更稳定的接口，也能确保我们能够更好地反映业务流程的变化。

3. 自动化集成

想做到“微”，只能把服务拆到细，怎么有效管理大量的服务，给运维带来了挑战。持续集成、自动化测试、持续交付、容器技术等自动化元素也就势在必行。

4. 去中心化

构建微服务不仅要从技术的角度去思考，更得从团队的角度入手。去中心化使得参与设计研发工作的所有人都可以进行分布式决策。

5. 独立性

保障每个微服务都可以独立开发、独立发布，一旦某个微服务出现服务阻塞，不会影响全盘服务。完全独立的微服务组件有助于实现自由装配的服务设计。

1.4 微服务的优势

通过微服务，架构师和开发人员可以选择适合每个微服务(多语言架构)的目的架构和技术，这样可以灵活地以更具成本效益的方式设计更合适的解决方案。

由于服务相当简单且规模较小，因此企业可以负担新流程、算法、业务逻辑等的实验。它使企业能够通过提供快速实验和失败的能力进行颠覆性创新。

微服务能够实现选择性、可扩展性，即每个服务可以独立地按比例放大或缩小，并且缩放成本相对小于单片方法。

微服务是独立的，独立地部署模块，当第二个微服务不能按照我们的需要执行时，能够用另一个类似的微服务替换一个微服务。它有助于采取正确的购买与构建决策，这通常是许多企业面临的挑战。

微服务帮助我们构建有机系统(有机系统是通过向其添加越来越多的功能，而在一段时间内横向增长的系统)。由于微服务都是关于可独立管理的服务，它可以在需要时添加越来越多的服务，同时对现有服务的影响最小。

技术变革是软件开发的障碍之一。使用微服务，可以单独更改或升级每项服务的技术，而不

是升级整个应用程序。

由于微服务将服务运行时环境与服务本身打包在一起，因此可以使多个版本的服务在同一环境中共存。

最后，微服务还使小型、专注的敏捷团队能够进行开发。团队将根据微服务的界限进行组织。

1.5　Dubbo与Spring Cloud

Dubbo是阿里开源的一个 SOA 服务治理解决方案，文档丰富，在国内的使用度非常高。作为新一代的服务框架，Spring Cloud 提出的口号是开发面向云环境的应用程序，它为微服务架构提供了一站式的配套技术。

Dubbo与Spring Cloud对比见表1.1。

表1.1　Dubbo与Spring Cloud对比

性　能	Dubbo	Spring Cloud
服务注册与发现	ZooKeeper	Eureka
服务调用方式	RPC	REST API
服务监控	Dubbo-moitor	Admin
服务熔断	不完善	Hystrix
服务网关	—	Zuul
服务配置	—	Config
服务跟踪	—	Sleuth
服务总线	—	Bus
数据流	—	Stream
批量任务	—	Task

服务调用方式:Spring Cloud抛弃了Dubbo的RPC通信，采用的是基于HTTP的REST方式。严格来说，这两种方式各有优劣。虽然一定程度上来说，REST牺牲了服务调用的性能，但却比RPC更灵活，服务提供方和调用方的依赖只依靠一纸契约，不存在代码级别的强依赖，这在强调快速演化的微服务环境下显得更加合适。

第2章　微框架Spring Boot

Spring Boot是一个Spring框架模块，它为Spring框架提供RAD（快速应用开发）功能。它高度依赖于启动器模板功能，该功能非常强大且完美无缺。Spring Boot同样也是Spring Cloud的重要组成部分。

2.1　Spring Boot概述

Spring Boot是由Pivotal团队提供的全新框架，其设计目的是用来简化新Spring应用的初始搭建以及开发过程。该框架使用了特定的方式进行配置，从而使开发人员不再需要定义样板化的配置。通过这种方式，Spring Boot致力于在蓬勃发展的快速应用开发领域(rapid application development)成为领导者。

Spring Boot提供了一个强大的一键式Spring的集成开发环境，能够单独进行一个Spring应用的开发，其中：

（1）集中式配置（application.properties）+注解，大大简化了开发流程。

（2）内嵌的Tomcat和Jetty容器，可直接打成jar包启动，无须提供Java war包以及烦琐的Web配置。

（3）提供了Spring各个插件的基于Maven的pom模板配置，开箱即用，便利无比。

（4）可以在任何您想自动化配置的地方实现。

（5）提供更多的企业级开发特性，如系统监控、健康诊断、权限控制。

（6）无冗余代码生成和XML强制配置。

（7）提供支持强大的RESTful风格的编码，非常简洁。

2.2　Spring Boot快速搭建

集成开发环境（IDE）：IntelliJ IDEA；Spring Boot版本：1.5.2。操作步骤如下。

（1）在Eclipse中，选择File → New Project选项，并选择Maven → Maven Project选项，如图2.1所示。

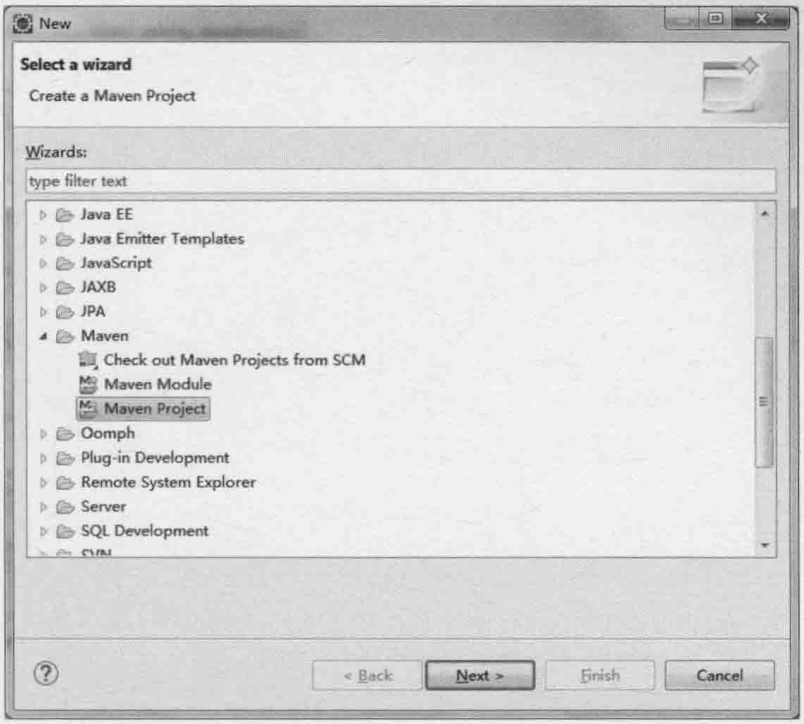


图2.1　选择一个Maven项目

（2）选择新Maven项目的架构类型，如选择maven-archetype-quickstart，如图2.2所示。

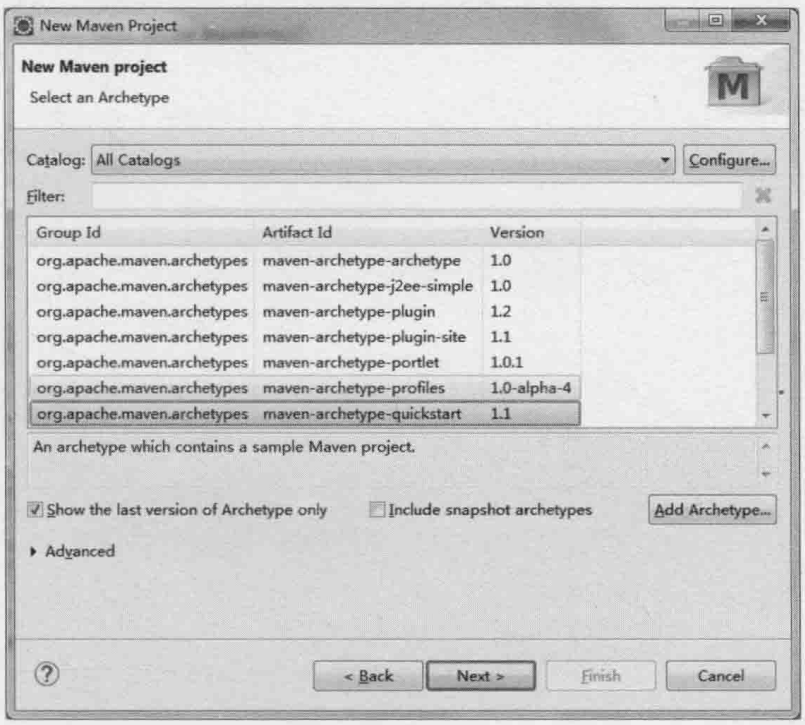


图2.2　选择maven-archetype-quickstart