第1-1课：Spring Boot 产生的背景和它的设计理念

要了解 Spring Boot 产生的背景，我们就必须要先了解一下 Spring 发展史，不仅因为 Spring Boot 来源于 Spring 体系，而且 Spring Boot 的诞生和 Spring 框架的发展息息相关。

Spring 发展历史

时间回到 2002 年，当时正是 Java EE 和 EJB 大行其道的时候，很多知名公司都是采用此技术方案进行项目开发。这时候有一个美国的小伙子认为 EJB 太过臃肿，并不是所有的项目都需要使用 EJB 这种大型框架，应该会有一种更好的方案来解决这个问题。

为了证明他的想法是正确的，于 2002 年 10 月甚至写了一本书《Expert One-on-One J2EE》，介绍了当时 Java 企业应用程序开发的情况，并指出了 Java EE 和 EJB 组件框架中存在的一些主要缺陷。在这本书中，他提出了一个基于普通 Java 类和依赖注入的更简单的解决方案。

在书中，他展示了如何在不使用 EJB 的情况下构建高质量，可扩展的在线座位预留系统。为了构建应用程序，他编写了超过 30,000 行的基础结构代码，项目中的根包命名为 com.interface21，所以人们最初称这套开源框架为 interface21，也就是 Spring 的前身。

他是谁呢，他就是大名鼎鼎的 Rod Johnson（下图），Rod Johnson 在悉尼大学不仅获得了计算机学位，同时还获得了音乐学位，更令人吃惊的是在回到软件开发领域之前，他还获得了音乐学的博士学位。现在 Rod Johnson 已经离开了 Spring，成为了一个天使投资人，同时也是多个公司的董事，早已走上人生巅峰。

在这本书发布后，一对一的 J2EE 设计和开发一炮而红。这本书免费提供的大部分基础架构代码都是高度可重用的。2003 年 Rod Johnson 和同伴在此框架的基础上开发了一个全新的框架命名为 Spring，据 Rod Johnson 介绍 Spring 是传统 J2EE 新的开始，随后 Spring 发展进入快车道。

- 2004 年 03 月，1.0 版发布。
- 2006 年 10 月，2.0 版发布。
- 2007 年 11 月，更名为 SpringSource，同时发布了 Spring 2.5。
- 2009 年 12 月，Spring 3.0 发布。
- 2013 年 12 月，Pivotal 宣布发布 Spring 框架 4.0。
- 2017 年 09 月，Spring 5.0 发布。

Spring Boot 的诞生

多年以来，Spring 平台饱受非议的一点就是大量的 XML 配置以及复杂的依赖管理。

随着使用 Spring 进行开发的个人和企业越来越多，Spring 也慢慢从一个单一简洁的小框架变成一个大而全的开源软件，Spring 的边界不断的进行扩充，到了后来 Spring 几乎可以做任何事情了，市面上主流的开源软件、中间件都有 Spring 对应组件支持，人们在享用 Spring 的便利之后，也遇到了一些问题。

Spring 每集成一个开源软件，就需要增加一些基础配置，慢慢的随着人们开发的项目越来越庞大，往往需要集成很多开源软件，后期使用 Spring 开发大型项目需要引入很多配置文件，太多的配置非常难以理解，并容易配置出错，到了后来人们甚至称 Spring 为配置地狱。

在 2013 年的 SpringOne 2GX 会议上，Pivotal 的 CTO Adrian Colyer 回应了这些批评，并且特别提到该平台将来的目标之一就是实现免 XML 配置的开发体验。Spring Boot 所实现的功能超出了这个任务的描述，开发人员不仅不再需要编写 XML，而且在一些场景中甚至不需要编写繁琐的 import 语句。

2013 年，微服务的概念也慢慢兴起，快速开发微小独立的应用变得更为急迫，Spring 刚好处在这么一个交叉点上，于 2013 年初开始的 Spring Boot 项目的研发，2014 年，Spring Boot 伴随着 Spring 4.0 诞生发布了第一个正式版本。

Spring Boot 并不是要成为 Spring 平台里面众多“Foundation”层项目的替代者。Spring Boot 的目标不在于为已解决的问题域提供新的解决方案，而是为平台带来另一种开发体验，从而简化对这些已有技术的使用。对于已经熟悉 Spring 生态系统的开发人员来说，Spring Boot 是一个很理想的选择；对于采用 Spring 技术的新人来说，Spring Boot 提供一种更简洁的方式来使用这些技术。

Spring Boot 开发团队

我们经常会看到在介绍 Spring Boot 的时候有这么一句：Spring Boot 是由 Pivotal 团队提供的全新框架。由此我们得知 Spring Boot 是由 Pivotal 团队所研发，那么 Pivotal 团队到底是一个什么样的团队呢？其实这里的 Pivotal 团队是指 Pivotal 公司。

Pivotal 公司介绍：致力于“改变世界构造软件的方式（We are transforming how the world builds software）”，提供云原生应用开发 PaaS 平台及服务，帮助企业客户采用敏捷软件开发方法论，从而提高软

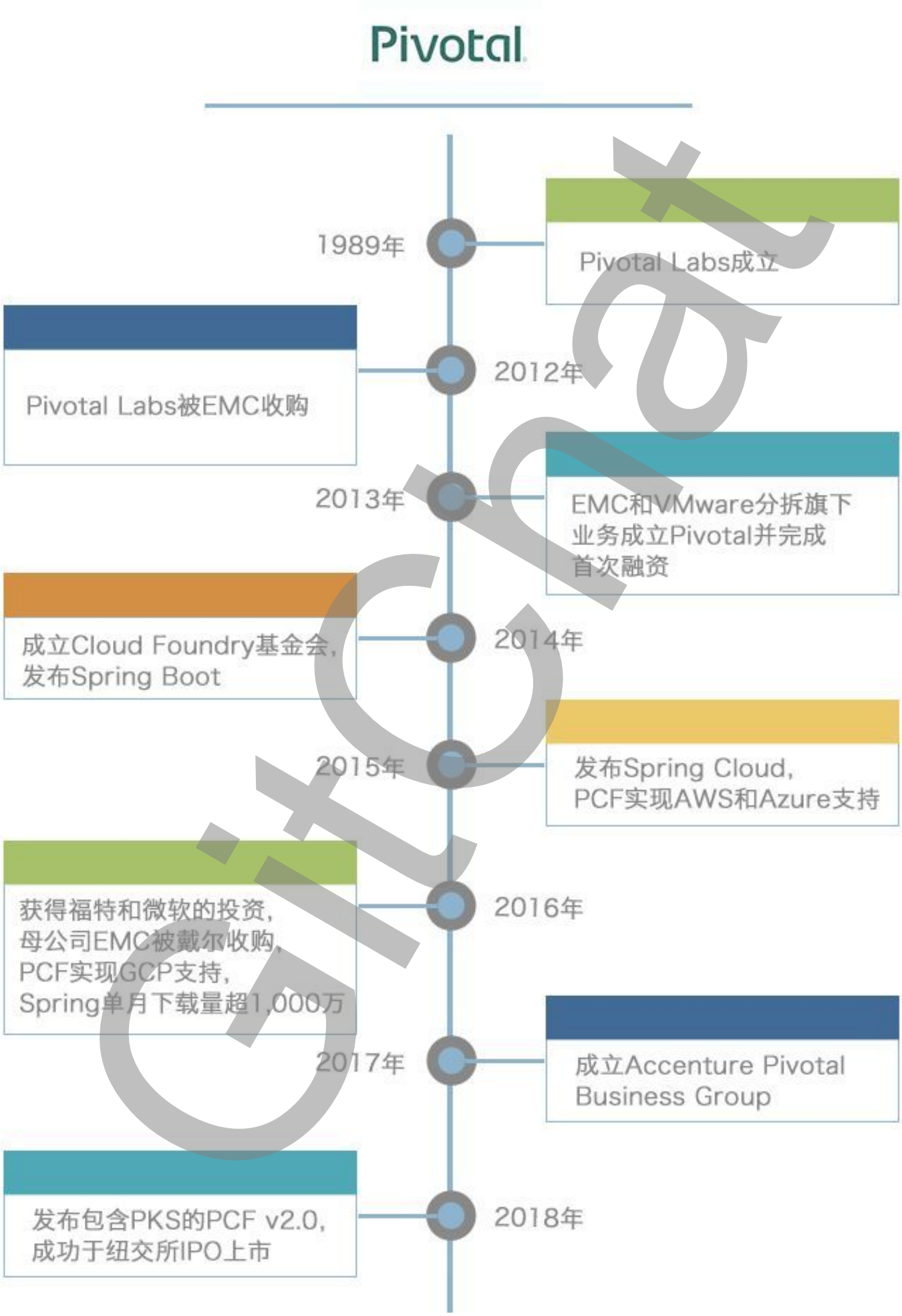
件开发人员工作效率、减少运维成本，实现数字化转型、IT 创新，并最终实现业务创新。

Pivotal 公司可谓是大牛云集，公司研发的产品有：Spring 以及衍生框架、缓存中间件 Redis、消息队列框架 RabbitMQ、数据引擎产品 Greenplum，还有 Tomcat、Groovy 里的一些顶级开发者，DevOps 理论的提出者都在这个公司。2016 年风靡全球的云原生理念亦是 Pivotal 公司提出，美国硅谷著名的精益化创业书籍的作者 Eric Ries 也加入了 Pivotal 公司。Spring Boot 为什么如此的优秀，正是因为背后有这些全球的顶级开发者。

回顾 Pivotal 公司的发展历史，简直就是一场商业并购大片：

- 1989 年，米创立 Pivotal Labs 公司，它的主营业务是与客户合作，帮助客户开发软件，曾给谷歌、Twitter 公司做技术支持；
- 2003 年，EMC 收购了 VMware；
- 2009 年，VMware 收购了 Spring 公司；
- 2012 年，EMC 以现金方式收购了 Pivotal Labs 公司；
- 2013 年，EMC 和 VMware 分拆出其 Cloud Foundry、Pivotal Labs、Greenplum 等云计算、大数据资源，GE 投资 1.05 亿美元，成立新公司 Pivotal；
- 2015 年，EMC 又被 DELL 所收购。

Pivotal 公司成立之后，于 2014 年发布了 Spring Boot，2015 年发布了 Spring Cloud，2018 年 Pivotal 公司在纽约上市。我们可以通过一张图来了解 Pivotal 公司的发展史。


Pivotal
1989年
Pivotal Labs成立
2012年
Pivotal Labs被EMC收购
2013年
EMC和VMware分拆旗下业务成立Pivotal并完成首次融资
2014年
成立Cloud Foundry基金会，发布Spring Boot
2015年
发布Spring Cloud，PCF实现AWS和Azure支持
2016年
获得福特和微软的投资，母公司EMC被戴尔收购，PCF实现GCP支持，Spring单月下载量超1,000万
2017年
成立Accenture Pivotal Business Group
2018年
发布包含PKS的PCF v2.0，成功于纽交所IPO上市

约定优于配置

那么什么是约定优于配置呢?

约定优于配置（Convention Over Configuration），也称作按约定编程，是一种软件设计范式，旨在减少软件开发人员需做决定的数量、获得简单的好处，而又不失灵活性。

本质是说，开发人员仅需规定应用中不符约定的部分。例如，如果模型中有个名为 User 的类，那么数据库中对应的表就会默认命名为 user。只有在偏离这一约定时，例如将该表命名为“user_info”，才需写有关这个名字的配置。

我们可以按照这个思路来设想，我们约定 Controller 层就是 Web 请求层可以省略 MVC 的配置；我们约定在 Service 结尾的类自动注入事务，就可以省略了 Spring 的切面事务配置…。

在 Spring 体系中，Spring Boot JPA 就是约定优于配置最佳实现之一，不需要关注表结构，我们约定类名即是表名，属性名即是表的字段，String 对应 varchar，long 对应 bigint，只有需要一些特殊要求的属性，我们再单独进行配置，按照这个约定我们可以将以前的工作大大的简化。

Spring Boot 体系将约定优于配置的思想展现得淋淋尽致，小到配置文件，中间件的默认配置，大到内置容器、生态中的各种 Starters 无不遵循此设计规则。Spring Boot 鼓励各软件组织方创建自己的 Starter，创建 Starter 的核心组件之一就是 autoconfigure 模块，也是 Starter 的核心功能，在启动的时候进行自动装配，属性默认化配置。

可以说正是因为 Spring Boot 简化的配置和众多的 Starters 才让 Spring Boot 变得简单、易用、快速上手，也可以说正是约定优于配置的思想的彻底落地才让 Spring Boot 走向辉煌。Spring Boot 约定优于配置的思想让 Spring Boot 项目非常容易上手，让编程变的更简单，其实编程本该很简单，简单才是编程的美。

Starters

Spring Boot Starters 基于约定优于配置的理念来设计，Spring Boot Starter 中有两个核心组件：自动配置代码和提供自动配置模块及其它有用的依赖。也就意味着当我们项目中引入某个 Starter，即拥有了此软件的默认使用能力，除非我们需要特定的配置，一般情况下我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。

以下为常用的 Spring Boot Starter 列表。

名称	描述	Pom
spring-boot-starter	核心 Starter，包括自动配置支持，日志和 YAML	Pom
spring-boot-starter-activemq	用于使用 Apache ActiveMQ 实现 JMS 消息	Pom
spring-boot-starter-amqp	用于使用 Spring AMQP 和 Rabbit MQ	Pom
spring-boot-starter-cache	用于使用 Spring 框架的缓存支持	Pom
spring-boot-starter-data-elasticsearch	用于使用 ElasticSearch 搜索，分析引擎和 Spring Data ElasticSearch	Pom
spring-boot-starter-data-jpa	用于使用 Hibernate 实现 Spring Data JPA	Pom
spring-boot-starter-data-mongodb	用于使用基于文档的数据库 MongoDB 和 Spring Data MongoDB	Pom
spring-boot-starter-data-redis	用于使用 Spring Data Redis 和 Jedis 客户端操作键—值数据存储 Redis	Pom
spring-boot-starter-jta-atomikos	用于使用 Atomikos 实现 JTA 事务	Pom
sring-boot-starter-mail	用于使用 Java Mail 和 Spring 框架 Email 发送支持	Pom
spring-boot-starter-quartz	用于定时任务 Quartz 的支持	Pom
spring-boot-starter-security	对 Spring Security 的支持	Pom
spring-boot-starter-test	用于测试 Spring Boot 应用，支持常用测试类库，包括 JUnit、Hamcrest 和 Mockito	Pom
spring-boot-starter-thymeleaf	用于使用 Thymeleaf 模板引擎构建 MVC Web 应用	Pom
spring-boot-starter-validation	用于使用 Hibernate Validator 实现 Java Bean 校验	Pom
spring-boot-starter-web	用于使用 Spring MVC 构建 Web 应用，包括 RESTful。Tomcat 是默认的内嵌容器	Pom
spring-boot-starter-websocket	用于使用 Spring 框架的 WebSocket 支持构建 WebSocket 应用	Pom

> 这里只节选了我们最常使用的 Starter，完整的 Starter 参考这里：**Spring Boot application starters**。

因为 Spring Boot 足够的强大，很多第三方社区都进行了主动的集成比如：MyBatis、RabbitMQ（高级用法）等，第三方社区支持的列表，可以在这里查看 Community Contributions，可以看到社区贡献的其他 Starters 列表。

看完这些 Starters 会不会瞬间觉得 Spring Boot 很强大，几乎我们涉及的开源软件都做了支持，在 Spring Boot 环境下使用这些软件，仅仅只需要引入对应的 Starter 包即可。

Spring、Spring Boot 和 Spring Cloud 的关系

Spring 最初核心的两大核心功能 Spring IoC 和 Spring Aop 成就了 Spring，Spring 在这两大核心功能上不断的发展，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的使用 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 IoC/Aop > Spring > Spring Boot > Spring Cloud

总结

Spring Boot 诞生一方面是因为 Spring 自身发展所遇到的问题，另一方面在微服务思想诞生之际，急需要一款快速开发工具来实现微服务技术落地，在这样的背景下诞生了 Spring Boot。Spring Boot 整体的设计思想是：约定优于配置、依赖此设计思路，Spring Boot 进行了大刀阔斧的改革，让人们非常容易的实现开发、测试、部署。众多的 Starters 成就了 Spring Boot 的发展，让使用 Spring Boot 开发项目变的更加简单。