Spring Framework 概述

> 版本：5.3.23

Spring使创建Java企业级应用变得容易。它提供了在企业环境中使用Java语言所需的一切，支持Groovy和Kotlin作为JVM的替代语言，并可根据应用的需求灵活创建各种架构。从Spring Framework 5.1开始，Spring需要JDK 8+（Java SE 8+），并为JDK 11 LTS提供开箱即用的支持。建议将Java SE 8 update 60 作为Java 8的最低补丁版，但通常推荐使用最新的补丁版。

Spring支持广泛的应用场景。在大型企业中，应用程序通常存在很长时间，必须运行在升级周期超出程序员控制的JDK和应用服务器上。其他的可能是在云环境下通过单独的jar包来运行，jar包中嵌入了服务器。还有一些可能是不需要服务器的独立应用程序（如批处理或集成工作负载）。

Spring是开源的。它有一个庞大且活跃的社区。社区根据各种现实案例提供了持续不断的反馈。这使得Spring在很长的一段时间里成功发展。

1.我们所说的Spring是什么

“Spring”在不同的环境下有着不同的意思。它可以指Spring Framework项目本身，它是Spring开始的地方。Spring的其他项目都是在Spring Framework的基础上构建的。大多数情况下，人们所说的“Spring”代表整个Spring家族。这份文档专注于基础：Spring Framework本身。

Spring Framework被划分成多个模块。应用程序可以选择需要的模块。最重要的模块是核心容器，它包括配置模型和依赖注入机制。除此之外，Spring Framework 还为不同的应用架构提供了基础的支持，包括消息，事务数据和持久化以及web。它还包括基于Servlet的web框架Spring MVC以及响应式的web框架Spring WebFlux。

关于模块的注释：Spring的jar包可以部署到JDK 9的模块路径（Jigsaw）。为了支持在Jigsaw应用程序中使用，Spring Framework 5的jar包附带了“自动模块名称”清单，这个清单里定义了独立于jar包artifact名称（如“spring-core”，“spring-context”等）的稳定的语言级别的模块名称（如“spring.core”，“spring.context”等）。当然，Spring Framework的jar也可以在JDK 8 和JDK 9的类路径上工作的很好。

2.Spring和Spring Framework的历史

为了应对早期J2EE规范的复杂性，Spring于2003年应运而生。虽然有些人认为JavaEE和Spring是竞争对手，实际上Spring是对JavaEE的补充。Spring编程模型并没有包括整个JavaEE规范。相反，它从JavaEE规范中精挑细选出一个个规范：

- Servlet API(JSR 340)
- WebSocket API(JSR 356)
- Concurrency Utilities(JSR 236)
- JSON Binding API(JSR 367)
- Bean Validation(JSR 303)
- JPA(JSR 338)
- JMS(JSR 914)
- 用于协调事务的JTA/JCA(如有必要)

Spring Framework还支持依赖注入（JSR 330）和常用注解（JSR 250）规范。应用开发人员可以选择使用这些规范而不是Spring提供的特定于Spring的机制。

从Spring Framework 5.0开始，Spring需要至少Java EE 7（如Servlet 3.1，JPA 2.1），同时如果在运行时遇到Java EE 8（如Servlet 4.0，JSON Binding API），也提供开箱即用的集成。这使得Spring可以与Tomcat 8，Tomcat 9，WebSphere 9 以及JBoss EAP 7 完全兼容。

随着时间的发展，Java EE在应用程序开发上的角色已经发生改变。在Java EE和Spring的早期，创建好的应用程序是要部署在应用服务器上的。如今，有了Spring Boot的帮助，应用程序可以以devops和云友好的方式创建，其中内嵌了Servlet容器，并且很容易更改。从Spring Framework 5开始，WebFlux应用甚至可以直接不使用Servlet API，它能够运行在非Servlet容器上（如 Netty）。

Spring还在继续创新和发展。除了Spring Framework之外，还有些其他项目，如Spring Boot、Spring Security、Spring Data、Spring Cloud、Spring Batch等。要注意每个项目都有它自己的源码库，问题追踪，发布节奏。

3.设计哲学

当你学习一个框架时，了解它做了什么的同时更重要的是了解它遵循的原则。以下是Spring的指导原则：

- 在每个级别都提供选择。Spring让你能够尽可能得推迟做设计决策。例如可以通过配置切换持久层而不需要修改代码。许多其他基础设施以及和第三方API集成也是如此。
- 适应各种应用场景。Spring是灵活的，它并不关心具体如何做事。它支持不同应用场景里广泛的应用需求。
- 保持很强的向后兼容性。Spring的版本更新是被精心管理的，版本之间不会有巨大的变化。Spring支持精心选择的JDK版本和第三方库，以便于维护依赖Spring的应用程序和库。
- 关注API设计。Spring团队投入大量的思考和时间来制作直观，可以在很多版本，很多年里使用的API。

- 对代码质量设立高标准。Spring团队强调有意义的，最新的，准确的Java文档。它是少数几个可以宣称代码结构干净，包之间没有循环依赖的项目之一。