

05 Spring 5.x特性、设计理念及架构

更新时间：2020-07-30 10:00:33



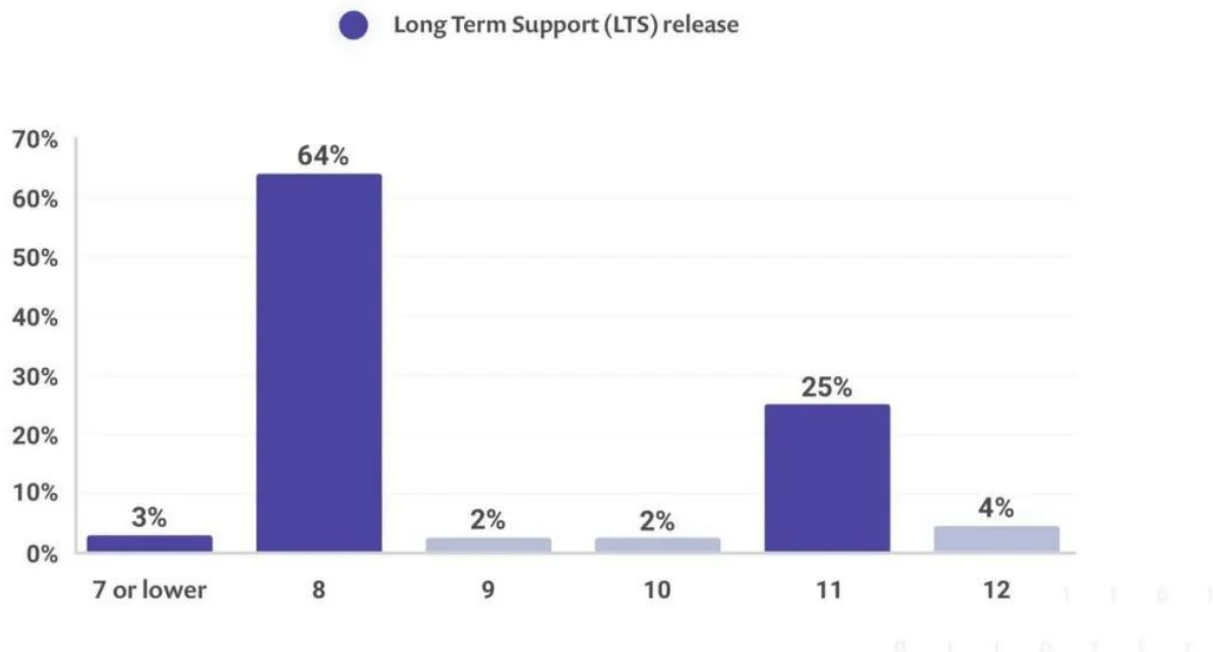
“学习这件事不在乎有没有人教你，最重要的是在于你自己有没有觉悟和恒心。”——法布尔

更多资源请+q:311861754
+v: Andvac1u

spring 5.x 出现的背景

Java 8 占据统治地位

据最新的调查：64% 的 Java 开发者使用 Java 8 作为生产环境的主要平台。情况似乎是这样，Java 开发者正在最终放弃 Java 7，但接纳 Java 9 的步伐相对缓慢。有意思的是，尽管有采纳 Java 11 的趋势，但这是一个缓慢的开始。



反应式编程的发展

反应式编程（reactive programming）又叫响应式编程，是一种基于数据流（data stream）和变化传递（propagation of change）的声明式（declarative）的编程范式。它带来了更短的处理时间和更高的硬件利用率，从而降低了运营成本。现在运行的很多大型系统都是基于响应式宣言及其原则打造的。LinkedIn、Twitter、Facebook 等很多企业使用的系统都是基于非同步和非堵塞 I/O 技术架构，因此他们的应用程序得以优化，能够最大化地利用硬件资源。这是打造可扩展型应用程序的新方法，而且正在迅速发展。

谷歌宣布 Kotlin 成为 Android 开发首选语言

自从 2017 年 Google 宣布 Kotlin 成为 Android 官方开发语言之后，Kotlin 受到广大 Android 开发者的追捧。其强大的安全性，简洁性和与 Java 的互操作性，为开发者带来了耳目一新的开发体验，也极大提升了 Android 原生代码的开发效率。

Spring 5.x 新特性

Spring 与时俱进，充分吸收新的营养，加入自己的体系。Spring 的新特性主要如下：



总体来说，spring 5.x 的特性可以分为以下几类：

支持更高版本的 **JDK (8及以上)**： 将不支持 jdk8 以下版本；

核心框架的修订： 由于 jdk8 反射的增强，Spring 5 支持有效获取 Method 的参数支持 @Nullable 和 @NotNull 注解提供了基于 Java 8 默认方法构建的选择性声明；

核心容器更新： 支持候选组件索引，可以替代类路径扫描。该支持已被添加到类路径扫描器的候选组件标识步骤的快捷方式中。GenericApplicationContext 和 AnnotationConfigApplicationContext 中实现函数式编程风格。对接口方法上的事务、缓存和异步注释的一致检测。XML 配置名称空间简化为无版本模式；

使用 **Kotlin** 进行函数式编程： 引入了对 JetBrains Kotlin 语言的支持；

反应式编程模型。

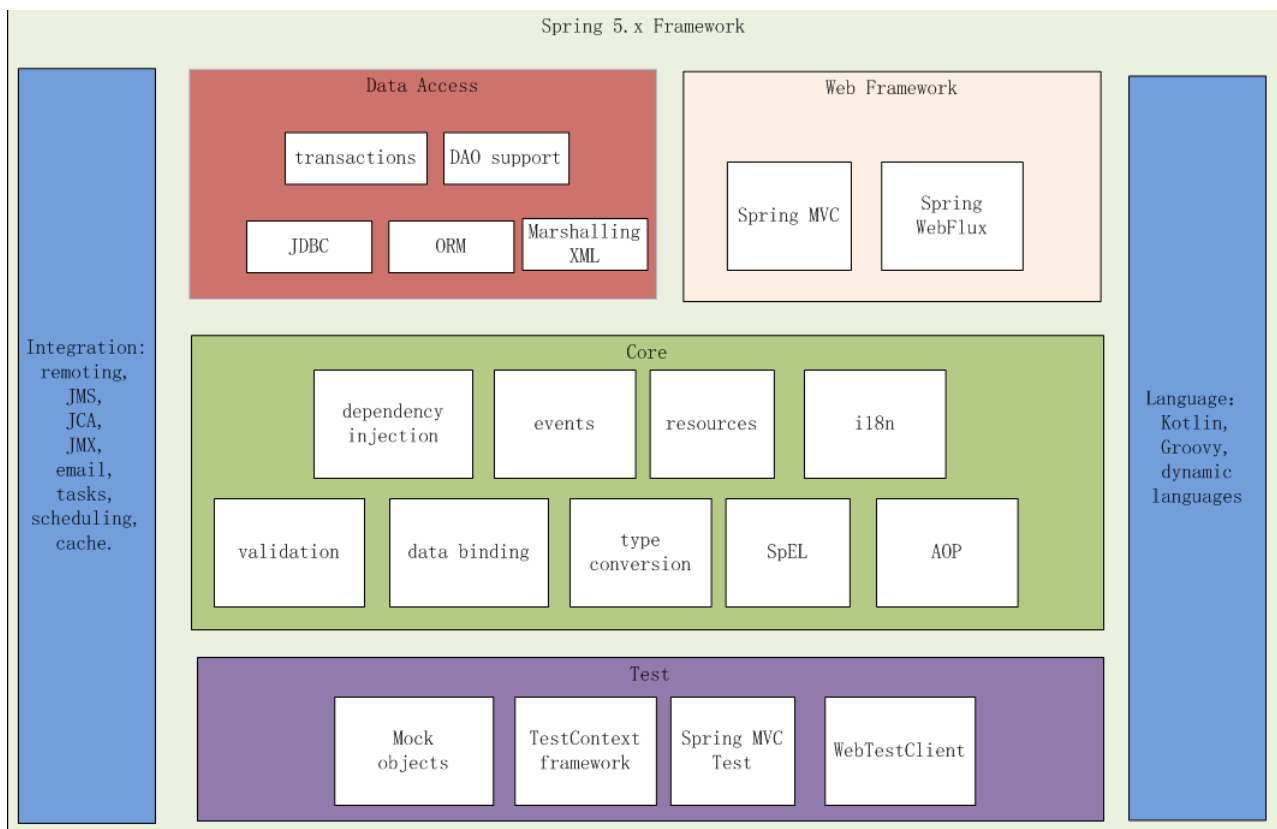
Spring 设计理念

- **可扩展性**： Spring 为你在每个层次提供选择权。Spring 允许您尽可能推迟设计决策。例如，您可以通过配置切换持久化提供方，而无需更改代码。Spring 提供灵活的方式和许多其他基础设施问题和与第三方API 的集成；
- **兼容性**： 一种体系容纳不同的其它框架。Spring 并不干预其它框架如何完成工作，而是通过适配器的方式兼容其它框架。进而支持各种不同角度的应用程序需求；
- **维护性**： 保持强大的向后兼容性。Spring 的版本之间尽量控制会对升级造成负担的改变。另外，Spring 支持精心选择的 JDK 版本和第三方库，以方便维护依赖于 Spring 的应用程序和库；
- **用户友好的 API 设计**： 强调 API 设计，Spring 团队投入了大量的时间和精力来开发易用的 API，以便这些 API 可以适用于许多版本、许多年；
- **高质量代码**： 制定高标准的代码质量。Spring 框架非常强调有意义的、保持同步的和精确的 javadoc。它是少数几个可以声明代码结构干净的项目之一，Spring 的包之间没有循环依赖关系。

Spring 5.x 总体架构

在上面的文章里，已经花费了大笔墨来描述 Spring 的总体架构和各个组件的依赖关系，但还是认为这远远不够，重要的事情要说三遍。

这就是 Spring5.x 的总体架构：



核心架构（Core）

- **IoC Container:** 包括 BeanFactory, ApplicationContext 容器
- **Events:** 事件
- **Resources:** 资源使用
- **i18n:** 国际化
- **Validation:** 数据校验
- **Data Binding:** 数据绑定
- **Type Conversion:** 类型转换
- **SpEL:** 表达式计算
- **AOP:** 面向切面编程

更多资源请+q:311861754
+v: Andvac1u

数据访问（Data Access）

- **Transactions:** 事务管理
- **DAO Support:** 数据访问对象
- **JDBC:** jdbcTemplate 数据库访问模板
- **O/R Mapping:** jpa, Hibernate, 对象关系映射
- **XML Marhalling:** xml 读取

Web 应用（Web Framework）

阻塞式 Web 应用

- **Spring MVC:** 传统 Servlet, 阻塞式响应;
- **WebSocket:** WebSocket 使得客户端和服务端之间的数据交换变得更加简单, 允许服务端主动向客户端推送数据;
- **SockJS:** SockJS 是一个浏览器上运行的 JavaScript 库, 如果浏览器不支持 WebSocket, 该库可以模拟对

WebSocket 的支持，实现浏览器和 Web 服务器之间低延迟、全双工、跨域的通讯通道；

- **STOMP Messaging:** 专为消息中间件设计。

非阻塞式 Web 应用

- **Spring WebFlux:** Spring WebFlux 是一套全新的 Reactive Web 栈技术，实现完全非阻塞，支持 Reactive Streams 背压等特性，并且运行环境不限于 Servlet 容器（Tomcat、Jetty、Undertow），如 Netty 等；
- **WebClient:** WebClient 是从 Spring WebFlux 5.0 版本开始提供的一个非阻塞的基于响应式编程的进行 Http 请求的客户端工具。它的响应式编程的基于 Reactor 的；
- **WebSocket:** 所谓 WebSocket，类似于 Socket，它的作用是可以让 Web 应用中的客户端和服务端建立全双工通信。

测试框架（Test）

- **Mock Objects:** 单元测试
- **TestContext Framework:** 上下文集成测试
- **Spring MVC Test:** Spring MVC集成测试
- **WebTestClient:** Spring WebFlux 集成测试

Spring 和其它的集成（integration）

- **Remoting:** 远程调用
- **JMS:** 消息
- **JCA:** 认证
- **JMX:** 监控
- **Email:** 邮件
- **Tasks:** 任务
- **Scheduling:** 调度
- **Caching:** 缓存

更多资源请+q:311861754
+v: Andvac1u

Spring 支持的语言（Language）

- **Kotlin:** Google 力推的语言
- **Groovy:** 动态 JVM 上运行的语言
- **Dynamic Languages:** 其它动态语言

小结

本文是进入 Spring 源码实战的序，理论为主，从 Spring5.x 产生的背景到 Spring5 的新特征，直至 Spring 本身的架构，重点是 Spring 框架本身的构成及内部组件的应用场景，接下来篇章就需要小伙伴们动手的时候了！

}