Spirng知识点

目录

[1. spring（由Rod Johnson创建的一个开源框架） 1](#_Toc22782)

[1.1. 背景 1](#_Toc28311)

[1.2. 框架特征 2](#_Toc9094)

[1.3. 特性 3](#_Toc23031)

[2. 特点 3](#_Toc21398)

[2.1. 1.方便解耦，简化开发 3](#_Toc31187)

[2.2. 2.AOP编程的支持 4](#_Toc20478)

[2.3. 3.声明式事务的支持 4](#_Toc11341)

[2.4. 5.方便集成各种优秀框架 4](#_Toc3740)

[2.5. 6.降低Java EE API的使用难度 4](#_Toc1771)

[2.6. 好处 4](#_Toc29054)

[3. 作用 各个主要方面。 6](#_Toc25522)

[3.1. 基本框架 7](#_Toc12796)

[4. 步骤 8](#_Toc26724)

[4.1. 容器 10](#_Toc29065)

[2104](https://baike.baidu.com/item/spring/javascript:void(0);)

284

# spring（由Rod Johnson创建的一个开源框架）

[编辑](https://baike.baidu.com/item/spring/javascript:;)

Spring是一个开放源代码的设计层面框架，它解决的是业务逻辑层和其他各层的松耦合问题，因此它将面向接口的编程思想贯穿整个系统应用。Spring是于2003 年兴起的一个轻量级的Java 开发框架，由Rod Johnson创建。简单来说，Spring是一个分层的JavaSE/EE **full-stack(一站式)** [轻量级](https://baike.baidu.com/item/%E8%BD%BB%E9%87%8F%E7%BA%A7/10002835" \t "https://baike.baidu.com/item/spring/_blank)开源框架。

## 背景

[编辑](https://baike.baidu.com/item/spring/javascript:;)

[Rod Johnson](https://baike.baidu.com/item/Rod Johnson" \t "https://baike.baidu.com/item/spring/_blank)在2

[](https://baike.baidu.com/pic/spring/85061/0/bba1cd11728b471042fe77d4cecec3fdfd0323d1?fr=lemma%26ct=single)

002年编著的《Expert one on one J2EE design and development》一书中，对Java EE 系统框架臃肿、低效、脱离现实的种种现状提出了质疑，并积极寻求探索革新之道。以此书为指导思想，他编写了interface21框架，这是一个力图冲破J2EE传统开发的困境，从实际需求出发，着眼于轻便、灵巧，易于开发、测试和部署的轻量级开发框架。Spring框架即以interface21框架为基础，经过重新设计，并不断丰富其内涵，于2004年3月24日，发布了1.0正式版。同年他又推出了一部堪称经典的力作《Expert one-on-one J2EE Development without EJB》，该书在Java世界掀起了轩然大波，不断改变着Java开发者程序设计和开发的思考方式。在该书中，作者根据自己多年丰富的实践经验，对[EJB](https://baike.baidu.com/item/EJB" \t "https://baike.baidu.com/item/spring/_blank)的各种笨重臃肿的结构进行了逐一的分析和否定，并分别以简洁实用的方式替换之。至此一战功成，Rod Johnson成为一个改变Java世界的大师级人物。

## 框架特征

[编辑](https://baike.baidu.com/item/spring/javascript:;)

**轻量**——从大小与开销两方面而言Spring都是轻量的

。完整的Spring框架可以在一个大小只有1MB多的JAR文件里发布。并且Spring所需的处理开销也是微不足道的。此外，Spring是非侵入式的：典型地，Spring应用中的对象不依赖于Spring的特定类。

[控制反转](https://baike.baidu.com/item/%E6%8E%A7%E5%88%B6%E5%8F%8D%E8%BD%AC" \t "https://baike.baidu.com/item/spring/_blank)——Spring通过一种称作控制反转（[IoC](https://baike.baidu.com/item/IoC/4853" \t "https://baike.baidu.com/item/spring/_blank)）的技术促进了低耦合。

当应用了IoC，一个对象依赖的其它对象会通过被动的方式传递进来，而不是这个对象自己创建或者查找依赖对象。你可以认为IoC与JNDI相反——不是对象从容器中查找依赖，而是容器在对象初始化时不等对象请求就主动将依赖传递给它。

**面向切面**——Spring提供了[面向切面编程](https://baike.baidu.com/item/%E9%9D%A2%E5%90%91%E5%88%87%E9%9D%A2%E7%BC%96%E7%A8%8B" \t "https://baike.baidu.com/item/spring/_blank)的丰富支持

，允许通过分离应用的业务逻辑与系统级服务（例如审计（auditing）和[事务](https://baike.baidu.com/item/%E4%BA%8B%E5%8A%A1" \t "https://baike.baidu.com/item/spring/_blank)（[transaction](https://baike.baidu.com/item/transaction" \t "https://baike.baidu.com/item/spring/_blank)）管理）进行[内聚性](https://baike.baidu.com/item/%E5%86%85%E8%81%9A%E6%80%A7" \t "https://baike.baidu.com/item/spring/_blank)的开发。[应用对象](https://baike.baidu.com/item/%E5%BA%94%E7%94%A8%E5%AF%B9%E8%B1%A1" \t "https://baike.baidu.com/item/spring/_blank)只实现它们应该做的——完成业务逻辑——仅此而已。它们并不负责（甚至是意识）其它的系统级关注点，例如日志或事务支持。

**容器**——Spring包含并管理应用对象的配置和生命周期，在这个意义上它是一种容器，

你可以配置你的每个bean如何被创建——基于一个可配置原型（[prototype](https://baike.baidu.com/item/prototype/14335188" \t "https://baike.baidu.com/item/spring/_blank)），你的bean可以创建一个单独的实例或者每次需要时都生成一个新的实例——以及它们是如何相互关联的。然而，Spring不应该被混同于传统的重量级的EJB容器，它们经常是庞大与笨重的，难以使用。

**框架**——Spring可以将简单的[组件](https://baike.baidu.com/item/%E7%BB%84%E4%BB%B6" \t "https://baike.baidu.com/item/spring/_blank)配置、组合成为复杂的应用。在Spring中，[应用对象](https://baike.baidu.com/item/%E5%BA%94%E7%94%A8%E5%AF%B9%E8%B1%A1" \t "https://baike.baidu.com/item/spring/_blank)被声明式地组合，典型地是在一个XML文件里。Spring也提供了很多基础功能（[事务管理](https://baike.baidu.com/item/%E4%BA%8B%E5%8A%A1%E7%AE%A1%E7%90%86" \t "https://baike.baidu.com/item/spring/_blank)、持久化框架集成等等），将应用逻辑的开发留给了你。

**MVC**——Spring的作用是整合，但不仅仅限于整合，Spring 框架可以被看做是一个企业解决方案级别的框架。客户端发送请求，服务器控制器（由DispatcherServlet实现的)完成请求的转发，控制器调用一个用于映射的类HandlerMapping，该类用于将请求映射到对应的处理器来处理请求。HandlerMapping 将请求映射到对应的处理器Controller（相当于Action）在Spring 当中如果写一些处理器组件，一般实现Controller 接口，在Controller 中就可以调用一些Service 或DAO 来进行数据操作 ModelAndView 用于存放从DAO 中取出的数据，还可以存放响应视图的一些数据。 如果想将处理结果返回给用户，那么在Spring 框架中还提供一个视图组件ViewResolver，该组件根据Controller 返回的标示，找到对应的视图，将响应response 返回给用户。

所有Spring的这些特征使你能够编写更干净、更可管理、并且更易于测试的代码。它们也为Spring中的各种模块提供了基础支持。

## 特性

[编辑](https://baike.baidu.com/item/spring/javascript:;)

强大的基于 [JavaBeans](https://baike.baidu.com/item/JavaBeans" \t "https://baike.baidu.com/item/spring/_blank)的采用控制反转（Inversion of Control，IoC）原则的配置管理，使得应用程序的组件更加快捷简易。

# 特点

[编辑](https://baike.baidu.com/item/spring/javascript:;)

## 1.方便解耦，简化开发

通过Spring提供的IoC容器，我们可以将对象之间的依赖关系交由Spring进行控制，避免硬编码所造成的过度程序耦合。有了Spring，用户不必再为单实例模式类、属性文件解析等这些很底层的需求编写代码，可以更专注于上层的应用。

## **2.AOP[编程](https://baike.baidu.com/item/%E7%BC%96%E7%A8%8B" \t "https://baike.baidu.com/item/spring/_blank)的支持**

通过Spring提供的[AOP](https://baike.baidu.com/item/AOP" \t "https://baike.baidu.com/item/spring/_blank)功能，方便进行面向切面的编程，许多不容易用传统OOP实现的功能可以通过AOP轻松应付。

## 3.**[声明式事务](https://baike.baidu.com/item/%E5%A3%B0%E6%98%8E%E5%BC%8F%E4%BA%8B%E5%8A%A1" \t "https://baike.baidu.com/item/spring/_blank)的支持**

在Spring中，我们可以从单调烦闷的事务管理代码中解脱出来，通过声明式方式灵活地进行事务的管理，提高开发效率和质量。

**4.方便程序的测试**

可以用非容器依赖的编程方式进行几乎所有的测试工作，在Spring里，测试不再是昂贵的操作，而是随手可做的事情。例如：Spring对Junit4支持，可以通过注解方便的测试Spring程序。

## 5.方便集成各种优秀框架

Spring不排斥各种优秀的开源框架，相反，Spring可以降低各种框架的使用难度，Spring提供了对各种优秀框架（如Struts,Hibernate、Hessian、Quartz）等的直接支持。

## 6.降低Java EE API的使用难度

Spring对很多难用的Java EE API（如JDBC，JavaMail，远程调用等）提供了一个薄薄的封装层，通过Spring的简易封装，这些Java EE API的使用难度大为降低。

**7.Java 源码是经典学习范例**

Spring的源码设计精妙、结构清晰、匠心独运，处处体现着大师对[Java设计模式](https://baike.baidu.com/item/Java%E8%AE%BE%E8%AE%A1%E6%A8%A1%E5%BC%8F" \t "https://baike.baidu.com/item/spring/_blank)灵活运用以及对Java技术的高深造诣。Spring框架源码无疑是Java技术的最佳实践范例。如果想在短时间内迅速提高自己的Java技术水平和应用开发水平，学习和研究Spring源码将会使你收到意想不到的效果。

## 好处

[编辑](https://baike.baidu.com/item/spring/javascript:;)

在我们进入细节以前，让我们看一下Spring可以给一个工程带来的一些好处：

Spring能有效地组织你的中间层对象，无论你是否选择使用了EJB。如果你仅仅使用了Struts或其他的包含了J2EE特有API的framework，你会发现Spring关注了遗留下的问题。Spring能消除在许多工程上对Singleton的过多使用。根据我的经验，这是一个主要的问题，它减少了系统的[可测试性](https://baike.baidu.com/item/%E5%8F%AF%E6%B5%8B%E8%AF%95%E6%80%A7" \t "https://baike.baidu.com/item/spring/_blank)和[面向对象](https://baike.baidu.com/item/%E9%9D%A2%E5%90%91%E5%AF%B9%E8%B1%A1" \t "https://baike.baidu.com/item/spring/_blank)特性。

Spring能消除使用各种各样格式的属性定制文件的需要，在整个应用和工程中，可通过一种一致的方法来进行配置。曾经我们可能对类是否有某个属性会感到迷惑，因此我们不得不去看它的javadoc或源码。自从有了Spring，你可很简单地看到类的JavaBean属性。反转控制的使用（在下面讨论）帮助完成这种简化。

Spring能通过接口而不是类促进好的编程习惯，减少编程代价到几乎为零。

Spring被设计为让使用它创建的应用尽可能少的依赖于他的APIs。在Spring应用中的大多数[业务对象](https://baike.baidu.com/item/%E4%B8%9A%E5%8A%A1%E5%AF%B9%E8%B1%A1" \t "https://baike.baidu.com/item/spring/_blank)没有依赖于Spring。

使用Spring构建的应用程序易于[单元测试](https://baike.baidu.com/item/%E5%8D%95%E5%85%83%E6%B5%8B%E8%AF%95" \t "https://baike.baidu.com/item/spring/_blank)。

Spring能使EJB的使用成为一个实现选择，而不是应用架构的必然选择。你能选择用POJOs或local EJBs来实现业务接口，却不会影响调用代码。

Spring帮助你解决许多问题而无需使用EJB。Spring能提供一种EJB的替换物，它们适于许多web应用。例如，Spring能使用AOP提供声明性[事务](https://baike.baidu.com/item/%E4%BA%8B%E5%8A%A1" \t "https://baike.baidu.com/item/spring/_blank)而不通过使用EJB容器，如果你仅仅需要与单个的数据库打交道，甚至不需要JTA实现。

Spring为数据存取提供了一致的框架，不论是使用JDBC或O/R mapping产品（如Hibernate）。

Spring确实使你能通过最简单可行的解决办法解决你的问题。这些特性是有很大价值的。

总结起来，Spring有如下优点：

1.低侵入式设计，代码污染极低

2.独立于各种[应用服务器](https://baike.baidu.com/item/%E5%BA%94%E7%94%A8%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/spring/_blank)，基于Spring框架的应用，可以真正实现Write Once,Run Anywhere的承诺

3.Spring的DI机制降低了[业务对象](https://baike.baidu.com/item/%E4%B8%9A%E5%8A%A1%E5%AF%B9%E8%B1%A1" \t "https://baike.baidu.com/item/spring/_blank)替换的复杂性，提高了组件之间的解耦

4.Spring的AOP支持允许将一些通用任务如安全、事务、日志等进行集中式管理，从而提供了更好的复用

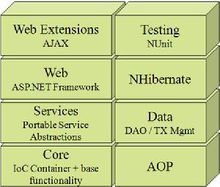
5.Spring的ORM和DAO提供了与第三方持久层框架的良好整合，并简化了底层的数据库访问

6.Spring并不强制应用完全依赖于Spring，开发者可自由选用Spring框架的部分或全部

# 作用 各个主要方面。

[编辑](https://baike.baidu.com/item/spring/javascript:;)

Spring提供许多功能，在此我将快速地依次展示其各个主要方面。

[](https://baike.baidu.com/pic/spring/85061/0/5d212aa8317294f0ca130c4b?fr=lemma%26ct=single)spring

首先，让我们明确Spring范围。尽管Spring覆盖了许多方面，但我们已经有清楚的概念，它什么应该涉及和什么不应该涉及。

**Spring的主要目的是使JavaEE易用和促进好编程习惯。**

Spring不重新开发已有的东西。因此，在Spring中你将发现没有日志记录的包，没有[连接池](https://baike.baidu.com/item/%E8%BF%9E%E6%8E%A5%E6%B1%A0" \t "https://baike.baidu.com/item/spring/_blank)，没有分布事务调度。这些均有[开源项目](https://baike.baidu.com/item/%E5%BC%80%E6%BA%90%E9%A1%B9%E7%9B%AE" \t "https://baike.baidu.com/item/spring/_blank)提供（例如Commons Logging 用来做所有的日志输出，或Commons DBCP用来作数据连接池），或由你的应用程序[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/spring/_blank)提供。因为同样的的原因，我们没有提供O/R mapping层，对此，已有友好的解决办法如Hibernate和JDO。**Spring的目标是使已存在的技术更加易用**。

例如，尽管我们没有底层[事务](https://baike.baidu.com/item/%E4%BA%8B%E5%8A%A1" \t "https://baike.baidu.com/item/spring/_blank)协调处理，但我们提供了一个抽象层覆盖了JTA或任何其他的事务策略。

Spring没有直接和其他的开源项目竞争，除非我们感到我们能提供新的一些东西。例如，像许多开发人员，我们从来没有为Struts高兴过，并且感到在MVC web framework中还有改进的余地。在某些领域，例如轻量级的IoC容器和AOP框架，Spring有直接的竞争，但是在这些领域还没有已经较为流行的解决方案。（Spring在这些区域是开路先锋。）

**Spring也得益于内在的一致性。**

所有的开发者都在唱同样的的赞歌，基础想法依然是Expert One-on-One J2EE设计与开发的那些。

并且我们已经能够使用一些主要的概念，例如倒置控制，来处理多个领域。

**Spring在**[应用服务器](https://baike.baidu.com/item/%E5%BA%94%E7%94%A8%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/spring/_blank)之间是可移植的。

当然保证可移植性总是一次挑战，但是我们避免任何特定平台或非标准化，并且支持在WebLogic，Tomcat，Resin，JBoss，WebSphere和其他的应用服务器上的用户。

## 基本框架

[编辑](https://baike.baidu.com/item/spring/javascript:;)

Spring 框架是一个分层架构，由 7 个定义良好的模块组成。Spring

[](https://baike.baidu.com/pic/spring/85061/0/d1571724cc23f467d40742ac?fr=lemma%26ct=single)Spring

模块构建在核心容器之上，核心容器定义了创建、配置和管理 bean 的方式，如图所示：

组成Spring框架的每个模块（或组件）都可以单独存在，或者与其他一个或多个模块联合实现。每个模块的功能如下：1、核心容器：核心容器提供 Spring 框架的基本功能(Spring Core)。核心容器的主要组件是 BeanFactory，它是[工厂模式](https://baike.baidu.com/item/%E5%B7%A5%E5%8E%82%E6%A8%A1%E5%BC%8F" \t "https://baike.baidu.com/item/spring/_blank)的实现。BeanFactory 使用[控制反转](https://baike.baidu.com/item/%E6%8E%A7%E5%88%B6%E5%8F%8D%E8%BD%AC" \t "https://baike.baidu.com/item/spring/_blank)（IOC） 模式将应用程序的配置和依赖性规范与实际的应用程序代码分开。

2、Spring 上下文：Spring 上下文是一个[配置文件](https://baike.baidu.com/item/%E9%85%8D%E7%BD%AE%E6%96%87%E4%BB%B6" \t "https://baike.baidu.com/item/spring/_blank)，向 Spring框架提供上下文信息。Spring 上下文包括企业服务，例如[JNDI](https://baike.baidu.com/item/JNDI" \t "https://baike.baidu.com/item/spring/_blank)、EJB、电子邮件、国际化、校验和调度功能。

3、Spring AOP：通过配置管理特性，Spring AOP 模块直接将面向切面的编程功能集成到了 Spring 框架中。所以，可以很容易地使 Spring 框架管理的任何对象支持[AOP](https://baike.baidu.com/item/AOP" \t "https://baike.baidu.com/item/spring/_blank)。Spring AOP 模块为基于 Spring 的应用程序中的对象提供了事务管理服务。通过使用 Spring AOP，不用依赖 EJB 组件，就可以将声明性事务管理集成到应用程序中。

4、Spring DAO：JDBC[DAO](https://baike.baidu.com/item/DAO" \t "https://baike.baidu.com/item/spring/_blank)抽象层提供了有意义的异常层次结构，可用该结构来管理[异常处理](https://baike.baidu.com/item/%E5%BC%82%E5%B8%B8%E5%A4%84%E7%90%86" \t "https://baike.baidu.com/item/spring/_blank)和不同数据库供应商抛出的错误消息。异常层次结构简化了错误处理，并且极大地降低了需要编写的异常代码数量（例如打开和关闭连接）。Spring DAO 的面向 JDBC 的异常遵从通用的 DAO 异常层次结构。

5、Spring ORM：Spring 框架插入了若干个[ORM](https://baike.baidu.com/item/ORM" \t "https://baike.baidu.com/item/spring/_blank)框架，从而提供了 ORM 的对象关系工具，其中包括[JDO](https://baike.baidu.com/item/JDO" \t "https://baike.baidu.com/item/spring/_blank)、[Hibernate](https://baike.baidu.com/item/Hibernate" \t "https://baike.baidu.com/item/spring/_blank)和[iBatis](https://baike.baidu.com/item/iBatis" \t "https://baike.baidu.com/item/spring/_blank)SQL Map。所有这些都遵从 Spring 的通用[事务](https://baike.baidu.com/item/%E4%BA%8B%E5%8A%A1" \t "https://baike.baidu.com/item/spring/_blank)和 DAO 异常层次结构。

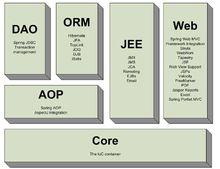
6、Spring Web 模块：Web 上下文模块建立在应用程序上下文模块之上，为基于 Web 的应用程序提供了上下文。所以，Spring框架支持与 Jakarta Struts 的集成。Web 模块还简化了处理多部分请求以及将请求参数绑定到[域对象](https://baike.baidu.com/item/%E5%9F%9F%E5%AF%B9%E8%B1%A1" \t "https://baike.baidu.com/item/spring/_blank)的工作。

7、Spring MVC 框架：[MVC](https://baike.baidu.com/item/MVC" \t "https://baike.baidu.com/item/spring/_blank)框架是一个全功能的构建 Web应用程序的 MVC 实现。通过策略接口，MVC框架变成为高度可配置的，MVC 容纳了大量视图技术，其中包括 JSP、[Velocity](https://baike.baidu.com/item/Velocity" \t "https://baike.baidu.com/item/spring/_blank)、[Tiles](https://baike.baidu.com/item/Tiles" \t "https://baike.baidu.com/item/spring/_blank)、iText 和 POI。模型由javabean构成，存放于Map；视图是一个接口，负责显示模型；控制器表示逻辑代码，是Controller的实现。Spring框架的功能可以用在任何[J2EE](https://baike.baidu.com/item/J2EE" \t "https://baike.baidu.com/item/spring/_blank)[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/spring/_blank)中，大多数功能也适用于不受管理的环境。Spring 的核心要点是：支持不绑定到特定 J2EE服务的可重用业务和[数据访问对象](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E8%AE%BF%E9%97%AE%E5%AF%B9%E8%B1%A1" \t "https://baike.baidu.com/item/spring/_blank)。毫无疑问，这样的对象可以在不同J2EE 环境（Web 或EJB）、独立应用程序、[测试环境](https://baike.baidu.com/item/%E6%B5%8B%E8%AF%95%E7%8E%AF%E5%A2%83" \t "https://baike.baidu.com/item/spring/_blank)之间重用。

# 步骤

[编辑](https://baike.baidu.com/item/spring/javascript:;)

Spring确实使你能通过最简单可行的解决办法来解决你的问题。而这是有有很大价值的。同时他的[源代码](https://baike.baidu.com/item/%E6%BA%90%E4%BB%A3%E7%A0%81" \t "https://baike.baidu.com/item/spring/_blank)的设计理念也受到很多程序员的追捧，简洁，易用.可是从哪着手研究Spring却是很多新手头疼的地方，下面的参考资料将带领大家慢慢的深入解析Spring

[](https://baike.baidu.com/pic/spring/85061/0/a8014c086e061d95593cc22076f40ad162d9cab4?fr=lemma%26ct=single)spring

1 Spring中的事务处理

2 IoC容器在Web容器中的启动

3 Spring JDBC

4 Spring MVC

5 Spring AOP获取Proxy

6 Spring声明式事务处理

7 Spring AOP中对拦截器调用的实现

8 Spring驱动Hibernate的实现

9 Spring Acegi框架鉴权的实现

**IOC和AOP**

[控制反转](https://baike.baidu.com/item/%E6%8E%A7%E5%88%B6%E5%8F%8D%E8%BD%AC" \t "https://baike.baidu.com/item/spring/_blank)模式（也称作依赖性注入）的基本概念是：不创建对象，但是描述创建它们的方式。在代码中不直接与对象和服务连接，但在[配置文件](https://baike.baidu.com/item/%E9%85%8D%E7%BD%AE%E6%96%87%E4%BB%B6" \t "https://baike.baidu.com/item/spring/_blank)中描述哪一个组件需要哪一项服务。容器 （在 Spring 框架中是 IOC 容器） 负责将这些联系在一起。在典型的 IOC 场景中，容器创建了所有对象，并设置必要的属性将它们连接在一起，决定什么时间调用方法。下表列出了 IOC 的一个实现模式。

|  |  |
| --- | --- |
| 类型 1 | 服务需要实现专门的接口，通过接口，由对象提供这些服务，可以从对象查询[依赖性](https://baike.baidu.com/item/%E4%BE%9D%E8%B5%96%E6%80%A7" \t "https://baike.baidu.com/item/spring/_blank)（例如，需要的附加服务）【接口注入】。 |
| 类型 2 | 通过[JavaBean](https://baike.baidu.com/item/JavaBean" \t "https://baike.baidu.com/item/spring/_blank)的属性（例如 setter 方法）分配依赖性【setter方法注入】。 |
| 类型 3 | 依赖性以[构造函数](https://baike.baidu.com/item/%E6%9E%84%E9%80%A0%E5%87%BD%E6%95%B0" \t "https://baike.baidu.com/item/spring/_blank)的形式提供，不以 JavaBean 属性的形式公开【构造器注入】。 |

Spring 框架的 IOC 容器采用类型 2 和类型3 实现。

面向切面的编程，即 AOP，是一种编程技术，它允许程序员对横切关注点或横切典型的职责分界线的行为（例如日志和事务管理）进行模块化。AOP 的核心构造是方面，它将那些影响多个类的行为封装到可重用的模块中。

AOP 和 IOC 是补充性的技术，它们都运用模块化方式解决企业[应用程序开发](https://baike.baidu.com/item/%E5%BA%94%E7%94%A8%E7%A8%8B%E5%BA%8F%E5%BC%80%E5%8F%91" \t "https://baike.baidu.com/item/spring/_blank)中的复杂问题。在典型的[面向对象](https://baike.baidu.com/item/%E9%9D%A2%E5%90%91%E5%AF%B9%E8%B1%A1" \t "https://baike.baidu.com/item/spring/_blank)开发方式中，可能要将日志记录语句放在所有方法和 Java 类中才能实现日志功能。在 AOP 方式中，可以反过来将日志服务模块化，并以声明的方式将它们应用到需要日志的[组件](https://baike.baidu.com/item/%E7%BB%84%E4%BB%B6" \t "https://baike.baidu.com/item/spring/_blank)上。当然，优势就是 Java 类不需要知道日志服务的存在，也不需要考虑相关的代码。所以，用 Spring AOP 编写的应用程序代码是松散耦合的。

AOP 的功能完全集成到了 Spring[事务管理](https://baike.baidu.com/item/%E4%BA%8B%E5%8A%A1%E7%AE%A1%E7%90%86" \t "https://baike.baidu.com/item/spring/_blank)、日志和其他各种特性的上下文中。

AOP 编程方面, Spring 提供了很多特性,例如 PointCut, Advice, Advisor, 粗略来说就是事务管理、日志和其他各种特性 的切入地点。

## 容器

[编辑](https://baike.baidu.com/item/spring/javascript:;)

Spring 设计的核心是 org.springframework.beans 包，它的设计目标是与 JavaBean 组件一起使用。这个包通常不是由用户直接使用，而是由[服务器](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E5%99%A8" \t "https://baike.baidu.com/item/spring/_blank)将其用作其他多数功能的底层中介。下一个最高级抽象是BeanFactory接口，它是工厂设计模式的实现，允许通过名称创建和检索对象。BeanFactory 也可以管理对象之间的关系。

BeanFactory 支持两个对象模型。

1、[单态](https://baike.baidu.com/item/%E5%8D%95%E6%80%81" \t "https://baike.baidu.com/item/spring/_blank)模型提供了具有特定名称的对象的共享实例，可以在查询时对其进行检索。Singleton是默认的也是最常用的对象模型。对于无状态服务对象很理想。

2、原型模型确保每次检索都会创建单独的对象。在每个用户都需要自己的对象时，[原型模型](https://baike.baidu.com/item/%E5%8E%9F%E5%9E%8B%E6%A8%A1%E5%9E%8B" \t "https://baike.baidu.com/item/spring/_blank)最适合。

bean 工厂的概念是 Spring 作为 IOC 容器的基础。IOC 将处理事情的责任从应用程序代码转移到框架。

词条图册[更多图册](https://baike.baidu.com/pic/spring/85061?fr=lemma" \t "https://baike.baidu.com/item/spring/_blank)