**Spring学习**

[一.](#_Toc3958) **[Spring概述.](#_Toc3958)** [1](#_Toc3958)

[1.Spring概述. 1](#_Toc20396)

[(1).Spring 框架的好处 2](#_Toc20008)

[(2).依赖注入(DI) 2](#_Toc24139)

[(3) .面向方面的程序设计（AOP） 2](#_Toc28746)

[2. Spring体系结构. 3](#_Toc897)

[(1).体系结构 3](#_Toc31985)

[(2).核心容器 4](#_Toc2901)

[(3).数据访问/集成 4](#_Toc29758)

[(4).Web 4](#_Toc12466)

[(5).其他 4](#_Toc15230)

[3.Spring导入及helloWorld输出 5](#_Toc10644)

[1.导入Spring包 5](#_Toc1809)

[2.创建源文件. 6](#_Toc15820)

[3.创建 bean 的配置文件. 7](#_Toc6321)

**[二.Spring IoC 容器.](#_Toc1201)** [7](#_Toc1201)

**[三.Spring 依赖注入](#_Toc7381)** [7](#_Toc7381)

**[四.Spring Beans 自动装配](#_Toc2416)** [8](#_Toc2416)

**[五.Spring 基于注解的配置](#_Toc2110)** [8](#_Toc2110)

**[六.Spring 框架的 AOP](#_Toc21218)** [8](#_Toc21218)

**[七.Spring JDBC 框架](#_Toc7933)** [8](#_Toc7933)

**[八.Spring 事务管理](#_Toc29454)** [8](#_Toc29454)

**[九.Spring Web MVC 框架](#_Toc22624)** [8](#_Toc22624)

# **Spring概述.**

## 1.Spring概述.

Spring是一个开源的控制反转(Inversion of Control, Ioc和面向切面编程(AOP)的轻量级容器框架,其主要目的是简化软件开发.Spring 框架是一个开源的 Java 平台,Spring 是轻量级的。 Spring 框架的基础版本是在 2 MB 左右的。Spring 框架的核心特性可以用于开发任何 Java 应用程序，但是在 Java EE 平台上构建 web 应用程序是需要扩展的。 Spring 框架的目标是使 J2EE 开发变得更容易使用，通过启用基于 POJO 编程模型来促进良好的编程实践。

### (1).Spring 框架的好处

下面列出的是使用 Spring 框架主要的好处：

1.Spring 可以使开发人员使用 POJOs 开发企业级的应用程序。只使用 POJOs 的好处是你不需要一个 EJB 容器产品，比如一个应用程序服务器，但是你可以选择使用一个健壮的 servlet 容器，比如 Tomcat 或者一些商业产品。

2.Spring 在一个单元模式中是有组织的。即使包和类的数量非常大，你必须并且只需要但是你需要的，而忽略剩余的那部分。

3.Spring 不会让你白费力气做重复工作，它真正的利用了一些现有的技术，像几个 ORM 框架、日志框架、JEE、Quartz 和 JDK 计时器，其他视图技术。

4.测试一个用 Spring 编写的应用程序很容易，因为 environment-dependent 代码被放进了这个框架中。此外，通过使用 JavaBean-style POJOs，它在使用依赖注入注入测试数据时变得更容易。

5.Spring 的 web 框架是一个设计良好的 web MVC 框架，它为 web 框架，比如 Structs

或者其他工程上的或者很少受欢迎的 web 框架，提供了一个很好的供替代的选择。

6.为将特定技术的异常（例如，由 JDBC、Hibernate，或者 JDO 抛出的异常）翻译成一致的， Spring 提供了一个方便的 API，而这些都是未经检验的异常。

7.轻量级的 IOC 容器往往是轻量级的，例如，特别是当与 EJB 容器相比的时候。这有利于在内存和 CPU 资源有限的计算机上开发和部署应用程序。

8.Spring 提供了一个一致的事务管理界面，该界面可以缩小成一个本地事务（例如，使用一个单一的数据库）和扩展成一个全局事务（例如，使用 JTA）.

### (2).依赖注入(DI)

Spring 最认同的技术是控制反转的****依赖注入（DI）****模式。控制反转（IoC）是一个通用的概念，它可以用许多不同的方式去表达，依赖注入仅仅是控制反转的一个具体的例子。

依赖注入是什么,这里将依赖关系部分转化为两个类之间的关联。例如，类 A 依赖于类 B。现在，让我们看一看第二部分，注入。所有这一切都意味着类 B 将通过 IoC 被注入到类 A 中。依赖注入可以以向构造函数传递参数的方式发生，或者通过使用 setter 方法 post-construction。由于依赖注入是 Spring 框架的核心部分，所以我将在一个单独的章节中利用很好的例子去解释这一概念。

### .面向方面的程序设计（AOP）

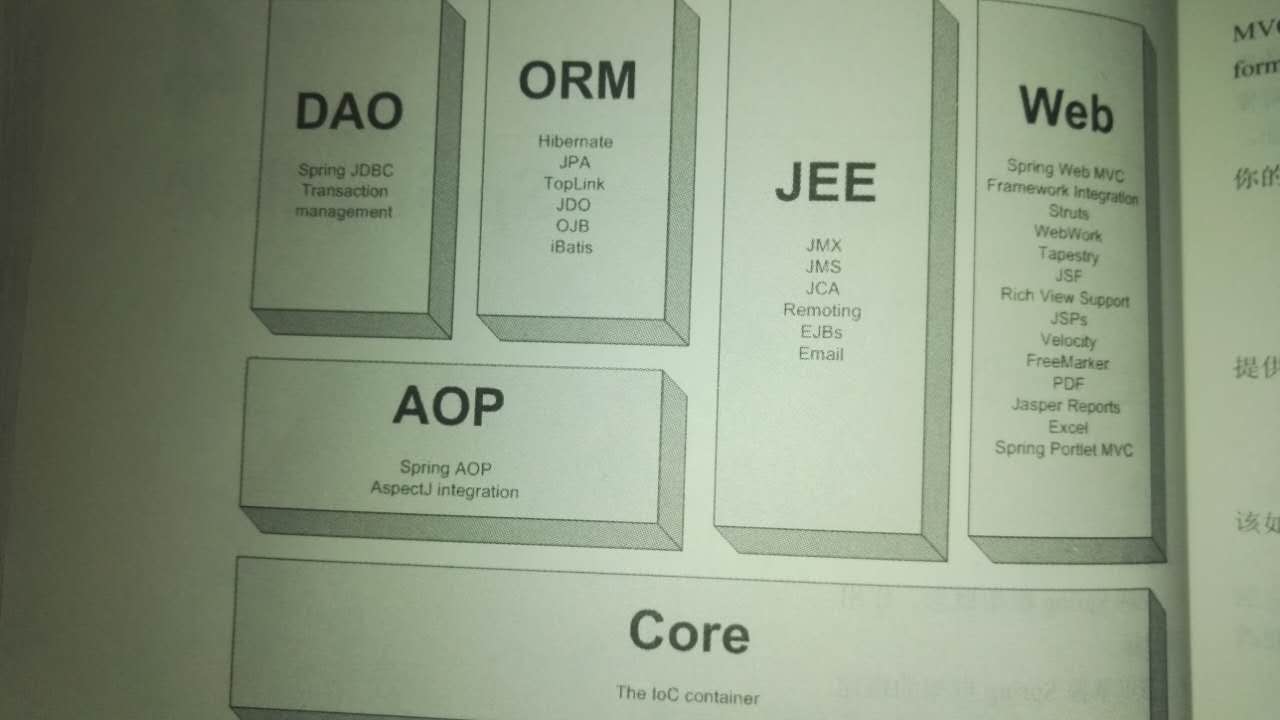
Spring 框架的一个关键组件是****面向方面的程序设计（AOP）****框架。一个程序中跨越多个点的功能被称为****横切关注点****，这些横切关注点在概念上独立于应用程序的业务逻辑。有各种各样常见的很好的关于方面的例子，比如日志记录、声明性事务、安全性，和缓存等等。

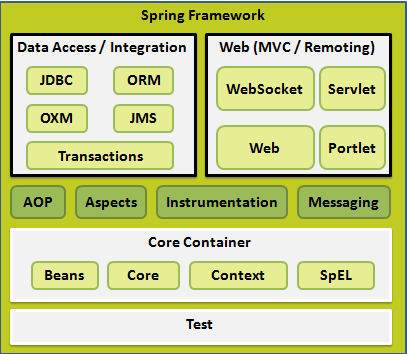
在 OOP 中模块化的关键单元是类，而在 AOP 中模块化的关键单元是方面。AOP 帮助你将横切关注点从它们所影响的对象中分离出来，然而依赖注入帮助你将你的应用程序对象从彼此中分离出来。

Spring 框架的 AOP 模块提供了面向方面的程序设计实现，允许你定义拦截器方法和切入点，可以实现将应该被分开的代码干净的分开功能。我将在一个独立的章节中讨论更多关于 Spring AOP 的概念

## Spring体系结构.

### (1).体系结构





Spring 有可能成为所有企业应用程序的一站式服务点，然而，Spring 是模块化的，允许你挑选和选择适用于你的模块，不必要把剩余部分也引入。下面的部分对在 Spring 框架中所有可用的模块给出了详细的介绍。

Spring 框架提供约 20 个模块，可以根据应用程序的要求来使用。

### (2).核心容器

核心容器由核心，Bean，上下文和表达式语言模块组成，它们的细节如下：

****1.核心****模块提供了框架的基本组成部分，包括 IoC 和依赖注入功能。

****2.Bean**** 模块提供 BeanFactory，它是一个工厂模式的复杂实现。

****3.上下文****模块建立在由核心和 Bean 模块提供的坚实基础上，它是访问定义和配置的任何对象的媒介。ApplicationContext 接口是上下文模块的重点。

****4.表达式语言****模块在运行时提供了查询和操作一个对象图的强大的表达式语言。

### (3).数据访问/集成

数据访问/集成层包括 JDBC，ORM，OXM，JMS 和事务处理模块，它们的细节如下：

****1.JDBC**** 模块提供了删除冗余的 JDBC 相关编码的 JDBC 抽象层。

****2.ORM**** 模块为流行的对象关系映射 API，包括 JPA，JDO，Hibernate 和 iBatis，提供了集成层。

****3.OXM**** 模块提供了抽象层，它支持对 JAXB，Castor，XMLBeans，JiBX 和 XStream 的对象/XML 映射实现。

4.Java 消息服务 ****JMS**** 模块包含生产和消费的信息的功能。

****5.事务****模块为实现特殊接口的类及所有的 POJO 支持编程式和声明式事务管理。

### (4).Web

Web 层由 Web，Web-MVC，Web-Socket 和 Web-Portlet 组成，它们的细节如下：

****1.Web**** 模块提供了基本的面向 web 的集成功能，例如多个文件上传的功能和使用 servlet 监听器和面向 web 应用程序的上下文来初始化 IoC 容器。

****2.Web-MVC**** 模块包含 Spring 的模型-视图-控制器（MVC），实现了 web 应用程序。

****3.Web-Socket**** 模块为 WebSocket-based 提供了支持，而且在 web 应用程序中提供了客户端和服务器端之间通信的两种方式。

****4.Web-Portlet**** 模块提供了在 portlet 环境中实现 MVC，并且反映了 Web-Servlet 模块的功能。

### (5).其他

还有其他一些重要的模块，像 AOP，Aspects，Instrumentation，Web 和测试模块，它们的细节如下：

****1.AOP**** 模块提供了面向方面的编程实现，允许你定义方法拦截器和切入点对代码进行干净地解耦，它实现了应该分离的功能。

****2.Aspects**** 模块提供了与 ****AspectJ**** 的集成，这是一个功能强大且成熟的面向切面编程（AOP）框架。

****3.Instrumentation**** 模块在一定的应用服务器中提供了类 instrumentation 的支持和类加载器的实现。

****4.Messaging**** 模块为 STOMP 提供了支持作为在应用程序中 WebSocket 子协议的使用。它也支持一个注解编程模型，它是为了选路和处理来自 WebSocket 客户端的 STOMP 信息。

****5.测试****模块支持对具有 JUnit 或 TestNG 框架的 Spring 组件的测试。

## 3.Spring导入及helloWorld输出

### 1.导入Spring包



### 2.创建源文件.





关于主要程序有以下两个要点需要注意：

1.第一步是我们使用框架 API ****ClassPathXmlApplicationContext()**** 来创建应用程序的上下文。这个 API 加载 beans 的配置文件并最终基于所提供的 API，它处理创建并初始化所有的对象，即在配置文件中提到的 beans。

2.第二步是使用已创建的上下文的 ****getBean()****方法来获得所需的 bean。这个方法使用 bean 的 ID 返回一个最终可以转换为实际对象的通用对象。一旦有了对象，你就可以使用这个对象调用任何类的方法。

### 3.创建 bean 的配置文件.



你必须确保这个文件在 CLASSPATH 中是可用的，并在主应用程序中使用相同的名称，而在 MainApp.java 文件中创建应用程序的上下文.

Beans.xml 用于给不同的 bean 分配唯一的 ID，并且控制不同值的对象的创建，而不会影响 Spring 的任何源文件。例如，使用下面的文件，你可以为 “message” 变量传递任何值，因此你就可以输出信息的不同值，而不会影响的 HelloWorld.java和MainApp.java 文件所示。

当 Spring 应用程序被加载到内存中时，框架利用了上面的配置文件来创建所有已经定义的 beans，并且按照 标签的定义为它们分配一个唯一的 ID。你可以使用 标签来传递在创建对象时使用不同变量的值。

# **二.[Spring IoC 容器](http://www.w3cschool.cn/wkspring/f8pc1hae.html" \o "Spring IoC 容器).**

# **三.[Spring 依赖注入](http://www.w3cschool.cn/wkspring/1h9m1h9m.html" \o "Spring 依赖注入)**

# **四.[Spring Beans 自动装配](http://www.w3cschool.cn/wkspring/o1qy1h9q.html" \o "Spring Beans 自动装配)**

# **五.[Spring 基于注解的配置](http://www.w3cschool.cn/wkspring/43851h9t.html" \o "Spring 基于注解的配置)**

# **六.[Spring 框架的 AOP](http://www.w3cschool.cn/wkspring/izae1h9w.html" \o "Spring 框架的 AOP)**

# **七.[Spring JDBC 框架](http://www.w3cschool.cn/wkspring/zjs51h9z.html" \o "Spring JDBC 框架)**

# **八.[Spring 事务管理](http://www.w3cschool.cn/wkspring/by1r1ha2.html" \o "Spring 事务管理)**

# **九.[Spring Web MVC 框架](http://www.w3cschool.cn/wkspring/9pf81ha5.html" \o "Spring Web MVC 框架)**