

Spring核心容器（Core Spring container）

容器是Spring框架最核心的部分，它管理着Spring应用中bean的创建、配置和管理。在该模块中，包括了Spring bean工厂，它为Spring提供了DI的功能。基于bean工厂，我们还会发现有多种Spring应用上下文的实现。还提供了许多企业服务。所有的Spring模块都构建于核心容器之上。

面向切面编程（AOP）

Spring对面向切面编程提供了丰富的支持。这个模块是Spring应用系统中开发切面的基础，与DI一样，AOP可以帮助应用对象解耦。

数据访问与集成（Data access & integration）

Spring的JDBC和DAO模块封装了大量样板代码，这样可以使得数据库代码变得简洁，也可以更专注于我们的业务，还可以避免数据库资源释放失败而引起的问题。

另外，Spring AOP为数据访问提供了事务管理服务，同时Spring还对ORM（object-relational mapping）进行了集成，如Hibernate、MyBatis等。spring ORM模块建立在对DAO的支持之上

Web与远程调用（Web and remoting）

集成了多种流行MVC框架，自带强大的MVC框架。支持面向对象的传统Web应用基于Servlet的框架和面向使用Java Portlet API的基于Portlet应用。

除了面向用户的web应用，还提供构建了与其他应用交互的几种远程调用选择，spring远程调用服务集成了RMI Hessian Burlap JAX-WS,同时自带远程调用框架HTTP invoker

Instrumentation

该模块提供了为JVM添加代理(agent)的功能。具体来讲，就是为Tomcat提供了一个织入代理，能够为Tomcat传递类文件，就像这些类文件是被类加载器加载的一样。

测试（Test）

为了使得开发者能够很方便的进行测试，Spring提供了测试模块以致力于Spring应用的测试。

通过该模块，Spring为使用Servlet、JNDI等编写单元测试提供了一系列的mock对象实现。

我们主要来看看Spring的骨骼架构和核心组件～

（一）Spring的骨骼架构

---Spring框架

|---Spring的特性功能

|---Web

|---Transaction

|---AOP

|---JDBC

|---JMX

|---Orm

|---......

|---Spring骨骼架构

|---核心组件Context

|---核心组件Bean

|---核心组件Core

1

2

3

4

5

6

7

8

9

10

11

12

13

Core、Context、Bean构建起了整个Spring的骨骼架构，没有它们就不可能有AOP、Web等上层的特性功能。

Spring的设计理念

如果要在三个核心组件中选出核心，那就是Bean组件，其实Spring就是面向Bean的编程（BOP，Bean Oriented Programming）。

我们用Spring的原因是什么？

Spring解决了一个非常关键的问题，它可以让我们把对象之间的依赖关系转而用配置文件来管理，也就是它的依赖注入机制。而这个注入关系在一个叫IOC的容器中进行管理。

IOC又是什么？

IOC是控制反转（Inversion of Control），是被Bean包裹的对象。Spring正是通过把对象包装在Bean中从而达到管理这些对象及做一系列额外操作的目的。

这种设计策略完全类似于Java实现OOP的设计理念，构建一个数据结构，然后根据这个数据结构设计它的生存环境，并让它在这个环境中按照一定的规律不停地运动， 在它们的不停运动中设计一个系列与环境或者与其他个体完成信息交换。

核心组件是如何进行协同工作的呢？

前面说Bean是Spring中的最核心，那么把Bean比作一场演出中的演员，Context就是舞台背景，Core就是演出的道具。只有三者一起才能具备演一场好戏的条件。当然，想要演出足够精彩，还要有Spring提供的特色功能。

Bean包装的是Object，Object必然有数据，如何给这些数据提供生存环境就是Context要解决的问题。Context要发现每个Bean之间的关系，为它们建立这种关系并且维护好这种关系。所以Context就是一个Bean关系的集合，这个集合又叫做IOC容器，建立起IOC容器，Spring就可以开始为我们工作了。

Core是发现、建立和维护每个Bean之间的关系所需要的一系列工具，也可以叫它Util（多用途的、工具性质的包）。

（二）核心组件详解

Bean组件

Bean组件在Spring的org.springframework.beans包下。这个包下的所有类主要解决3件事情：Bean的定义、Bean的创建及对Bean的解析。我们只需要关心它的创建就好了，其他两个由Spring在内部帮我们完成，我们是看不见的。

Spring Bean的创建是典型的工厂模式，它的顶级接口是BeanFactory。

Bean的定义主要由BeanDefinition描述。它的定义完整地描述了在Spring的配置文件中我们定义的< bean/>节点中所有的信息，包括各种子节点。当Spring成功解析我们定义的一个< bean/>节点后，在Spring的内部它就被转化成BeanDefinition对象，以后所有的操作都是对这个对象进行的。

Bean的解析主要就是对Spring配置文件的解析。

Context组件

Context组件在Spring的org.springframework.context包下。

ApplicationContext是Context的顶级父类。

ApplicationContext继承了BeanFactory，这也说明Spring容器中国运行的主体对象是Bean。

ApplicationContext继承了ResourseLoader接口，使其可以访问到任何外部资源。

ApplicationContext必须要完成的几件事情：

标识一个应用环境

利用BeanFactory创建Bean对象

保存对象关系表

能够捕获各种事件

Context作为Spring的IOC容器，基本上整合了Spring的大部分功能。

Core组件

Core组件包含了很多关键类，一个重要的组成部分就是定义了资源的访问方式。

Resource接口封装了各种可能的资源类型，也就是对使用者来说屏蔽了文件类型的不同。

Resource接口继承了InputStreamSource接口，这样所有的资源都可以通过InputStream类来获取，所以也屏蔽了资源的提供者。

加载资源的任务是由ResourceLoader接口完成的，它屏蔽了所有的资源加载者的差异。

Context把资源的加载、解析和描述工作委托给了ResourcePatternResolver类来完成，它相当于一个接头人，把资源的加载、解析和资源的定义整合在一起便于其他组件使用。

IOC容器是如何工作的？

IOC容器实际上是Context组件结合其他两个组件共同构建了一个Bean关系网。

把IOC容器比作一个箱子，在这个箱子里有若干个球的模子，可以用这些模子来制造很多种不同的球，还有一个制造这些球模的机器，这个机器可以产生球模。那么它们的对应关系就是BeanFactory，即那个制造球模的机器，球模就是Bean，制造出来的球就是Bean的实例。

BeanFactoryPostProcessor对应到当造球模被造出来时，有机会对其做出适当的修正，即它可以帮我们修改球模。InitializingBean和DisposableBean是在球模造球的开始和结束阶段，我们可以完成一些预备和扫尾工作。BeanPostProcessor可以让我们对球模造出来的球做出适当的修正。

最后还有一个FactoryBean，它可是一个神奇的球模。这个球模不是预先就定型的，而是由你来确定它的形状。这样在这个箱子里可以发现所有我们想要的球。