# 开始玩转Spring

本篇文档是提供关于Spring的详细信息。文档提供了综合性的Spring的框架信息，括了其原有的属性、以及潜在的原理如（“依赖注入”）。  
相较于本文档，我更加推崇[Getting Started](https://spring.io/)，它更容易被你理解。  
最后，当你想要理解本篇文章的同时，我建议你先设定一个短期目标来开始你的学习或许会更好，通过学习[Getting Started Building a RESTful Web Service](https://spring.io/guides/gs/rest-service/)让你脚先湿了（隐喻一口气吃不了大胖子，脚踏实地。）

# 介绍下Spring

Spring框架提供具有综合性能力的基础架构，构建应用是基于POJO，同样也支持JAVA SE。  
它的胜任力：

* 能够处理数据库事务性
* 能够处理远程协作任务
* 不通过JMX处理JVM管理操作
* 不同过JMS处理消息操作

## 依赖注入&控制反转 Dependency Injection and Inversion of Control

早在2004年，Martin Fowler就提出了“哪些方面的控制被反转了？”这个问题。他总结出是依赖对象的获得被反转了。基于这个结论，他为控制反转创造了一个更好的名字：依赖注入。许多非凡的应用（比HelloWorld.java更加优美，更加复杂）都是由两个或是更多的类通过彼此的合作来实现业务逻辑，这使得每个对象都需要，与其合作的对象（也就是它所依赖的对象）的引用。如果这个获取过程要靠自身实现，那么如你所见，这将导致代码高度耦合并且难以测试。  
IoC 亦称为 “依赖倒置原理”(“Dependency Inversion Principle”)。差不多所有框架都使用了“倒置注入(Fowler 2004)技巧，这可说是IoC原理的一项应用。SmallTalk，C++, Java 或各种.NET 语言等面向对象程序语言的程序员已使用了这些原理。控制反转是Spring框架的核心。应用控制反转，对象在被创建的时候，由一个调控系统内所有对象的外界实体，将其所依赖的对象的引用，传递给它。也可以说，依赖被注入到对象中。所以，控制反转是，关于一个对象如何获取他所依赖的对象的引用，这个责任的反转。

## 2.2 模块化Modules

Spring提供了具有20个模块，分别归类为Core Container, Data Access/Integration, Web, AOP (Aspect Oriented Programming), Instrumentation, Messaging, and Test等，详情见下面的图

## 2.3 Spring内核容器（Core Container）

Core container由spring-core, spring-beans, spring-context, spring-expression(SPL)模块组成Spring Framework 4.x的新特性

# 新特性和提高在Spring Framework 4.x

Spring Framework起源于2004年，Spring Framework 2.0提供了xml和AspectJ；Spring Framework 2.5捆绑了annotation；Spring Framework 3.0是基于JAVA5代码库，提供了@Configuration模型。  
Spring Framework 4.x是基于JAVA8的新特性而完成的，你仍然能够使用Spring Framework 4.x，不过最低要求是JAVA6,为了更好的Spring Framework 4.x发展，我们需要剔除些废除的类和方法。  
请查阅基于的[Spring Framework GitHub Wiki](https://github.com/spring-projects/spring-framework/wiki)的迁移方案[migration guide for upgrading to Spring 4.0](https://github.com/spring-projects/spring-framework/wiki/Migrating-from-earlier-versions-of-the-spring-framework)，以免出现异常。

## 3.1 如何提高学习Spring Framework

更好的学习Spring Framework，我个人建议是通过查看，[spring.io](https://spring.io/)

## 3.2 去除废除的包和方法Removed Deprecated Packages and Methods

前提条件，在你想升级到Spring Framework 4.x时，请确保老版本的API都不再被使用到。  
如果要查看备忘录，请翻阅[API Differences Report.](http://docs.spring.io/spring-framework/docs/3.2.4.RELEASE_to_4.0.0.RELEASE/)  
基于新版本，所以其它的第三方的开源框架也相应被升级，如： Hibernate 3.6+, EhCache 2.1+, Quartz 1.8+, Groovy 1.8+, Joda-Time 2.0+,Jackson 2.0+（Jackson 1.8/1.9依旧支持Spring Framework 3.2）

## 3.3 JAVA8

支持lambda表达式和方法回调接口

## 3.4 JAVA6和JAVA7

## 3.5 Groovy Bean 定义 DSL

通过Groovy语言去定义DSL，相较于XML，Groovy定义更为方便  
def reader = new GroovyBeanDefinitionReader(myApplicationContext)  
reader.beans {  
dataSource(BasicDataSource) {  
driverClassName = "org.hsqldb.jdbcDriver"  
url = "jdbc:hsqldb:mem:grailsDB"  
username = "sa"  
password = ""  
settings = [mynew:"setting"]  
}  
sessionFactory(SessionFactory) {  
dataSource = dataSource  
}  
myService(MyService) {  
nestedBean = { AnotherBean bean ->  
dataSource = dataSource  
}  
}  
}  
想要查看更多信息，请查阅[GroovyBeanDefinitionReader](http://docs.spring.io/spring/docs/current/javadoc-api/org/springframework/beans/factory/groovy/GroovyBeanDefinitionReader.html)

## 3.6 内核容器提高Core Container Improvements

有许多提高的性能在Spring Framework 4.x版本中。

* Spring支持范型注入Bean，例如你想spring data的注解@Repository，可以@Autowired Repository customerRepository
* 如果用meta-annotation，你可以用自定义annotation方式
* 用@Order annotation and Ordered interface的方式对于注解按顺序自动加载进list
* @Lazy annotation能够按照@Bean定义那样注入到point中
* @Description annotation人如其名，介绍下JAVA的配置信息
* 通过@Conditional annotation能够像@Profile过滤掉不符合条件的beans
* CGLIB-based proxy classes不再需要构造函数申明，是由作为Spring框架一部门的[objenesis library](http://code.google.com/p/objenesis/" \o "objenesis library)来完成内敛和分布式重新组转完成
* 通过LocaleContext来管理时间组件

## 3.7 普遍的web方面的提高General Web Improvements

Spring Framework 4.0主攻点是Servlet 3.0+环境，当然Servlet 2.5依旧支持，如果你用 Spring MVC Test Framework就必须包含Servlet 3.0

* @RestController annotation代替掉@ResponseBody和@RequestMapping
* 通过AsyncRestTemplate你能够异步处理REST请求
* Spring提供强大的时间组件支持

## 3.8 WebSocket, SockJS, and STOMP Messaging

spring-messaging包含STOMP协议，@Controller能够处理来自于HTTP请求的@RequestMapping和来自于客户端消息请求的@MessageMapping

# 新特性和提高New Features and Enhancements in Spring Framework 4.1