



Spring基础 - Spring和Spring框架组成

Spring是什么？它是如何诞生的？有哪些主要的组件和核心功能呢？本文通过这几个问题帮助你构筑Spring和Spring Framework的整体认知。@pdai

- Spring基础 - Spring和Spring框架组成
 - 什么是Spring?
 - Spring的起源
 - Spring的特性和优势
 - Spring有哪些组件?
 - Core Container（Spring的核心容器）
 - Data Access/Integration（数据访问 / 集成）
 - Web模块
 - AOP、Aspects、Instrumentation和Messaging
 - Test模块
 - 为什么用Spring?
 - 学习Spring时参考哪些资料呢?
 - Spring 的官方项目和教程
 - Spring 的归档文档
 - Spring 的官方Github

什么是Spring?

首先，Spring是什么？它是如何诞生的？它的诞生是为了解决什么问题？@pdai

Spring的起源

百度百科中关于Spring的起源介绍如下：

要谈Spring的历史，就要先谈J2EE。J2EE应用程序的广泛实现是在1999年和2000年开始的，它的出现带来了诸如事务管理之类的**核心中间层概念的标准化**，但是在实践中并没有获得绝对的成功，因为**开发效率**，开发难度和实际的**性能**都令人失望。

曾经使用过EJB开发JAVA EE应用的人，一定知道，在EJB开始的学习和应用非常的艰苦，很多东西都不能一下子就很容易的理解。EJB要严格地实现各种不同类型的接口，类似的或者重复的代码大量存在。而配置也是复杂和单调，同样使用JNDI进行对象查找的代码也是单调而枯燥。虽然有一些开发工作随着xdoclet的出现，而有所缓解，但是学习EJB的高昂代价，和极低的开发效率，极高的资源消耗，都造成了EJB的使用困难。而Spring出现的初衷就是为了解决类似的这些问题。

Spring的一个最大的目的就是使JAVA EE开发更加容易。同时，Spring之所以与Struts、Hibernate等单层框架不同，是因为Spring致力于提供一个以统一的、高效的方式构造整个应用，并且可以将单层框架以最佳的组合揉和在一起建立一个连贯的体系。可以说Spring是一个提供了更完善开发环境的一个框架，可以为POJO(Plain Ordinary Java Object)对象提供企业级的服务。

Spring的形成，最初来自Rod Jahnson所著的一本很有影响力的书籍《Expert One-on-One J2EE Design and Development》^[1]，就是在这本书中第一次出现了Spring的一些核心思想，该书出版于2002年。

Spring的特性和优势

从Spring 框架的**特性**来看：

- 非侵入式：基于Spring开发的应用中的对象可以不依赖于Spring的API
- 控制反转：IOC——Inversion of Control，指的是将对象的创建权交给 Spring 去创建。使用 Spring 之前，对象的创建都是由我们自己在代码中new创建。而使用 Spring 之后，对象的创建都是给了 Spring 框架。
- 依赖注入：DI——Dependency Injection，是指依赖的对象不需要手动调用 setXX 方法去设置，而是通过配置赋值。
- 面向切面编程：Aspect Oriented Programming——AOP
- 容器：Spring 是一个容器，因为它包含并且管理应用对象的生命周期
- 组件化：Spring 实现了使用简单的组件配置组合成一个复杂的应用。在 Spring 中可以使用XML和Java注解组合这些对象。
- 一站式：在 IOC 和 AOP 的基础上可以整合各种企业应用的开源框架和优秀的第三方类库（实际上 Spring 自身也提供了表现层的 Spring MVC 和持久层的 Spring JDBC）

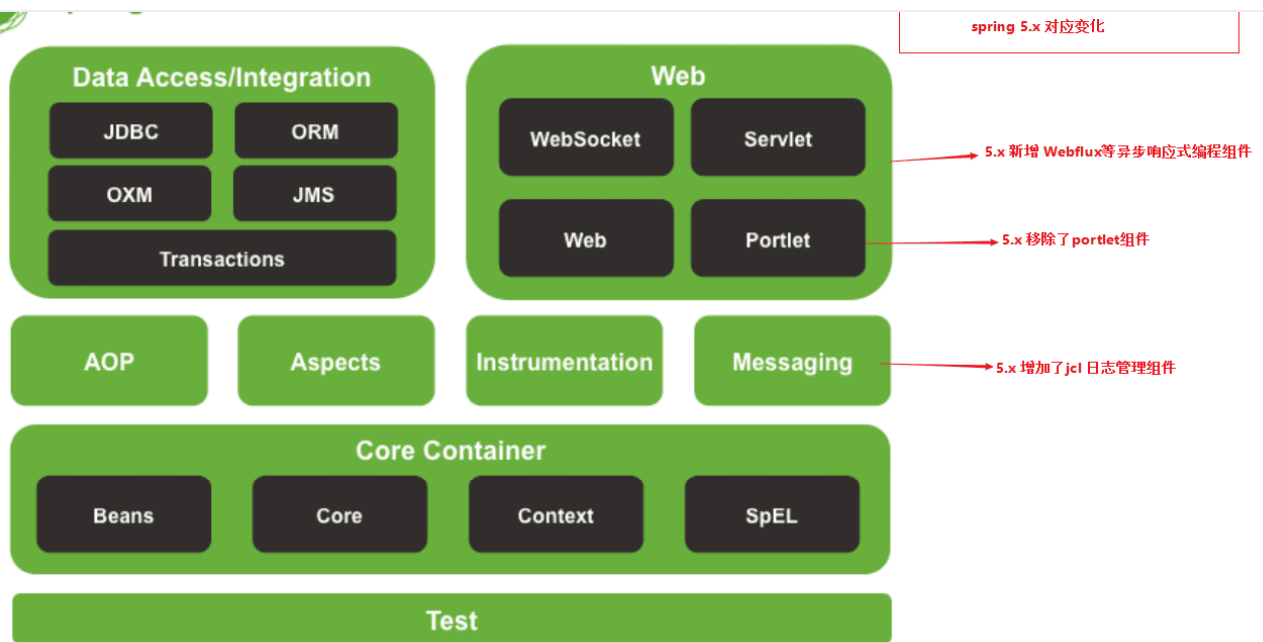
从使用Spring 框架的**好处**看：

- 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。
- 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，你只要担心你需要的，而其它的就可以忽略了。
- 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 ORM 框架、日志框架、JEE、Quartz 和 JDK 计时器，其他视图技术。
- 测试一个用 Spring 编写的应用程序很容易，因为环境相关的代码被移动到这个框架中。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。
- Spring 的 web 框架是一个设计良好的 web MVC 框架，它为比如 Struts 或者其他工程上的或者不怎么受欢迎的 web 框架提供了一个很好的供替代的选择。MVC 模式导致应用程序的不同方面(输入逻辑，业务逻辑和UI逻辑)分离，同时提供这些元素之间的松散耦合。模型(Model)封装了应用程序数据，通常它们将由 POJO 类组成。视图(View)负责渲染模型数据，一般来说它生成客户端浏览器可以解释 HTML 输出。控制器(Controller)负责处理用户请求并构建适当的模型，并将其传递给视图进行渲染。
- Spring 对 JavaEE 开发中非常难用的一些 API（JDBC、JavaMail、远程调用等），都提供了封装，使这些API应用难度大大降低。
- 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。
- Spring 提供了一致的事务管理接口，可向下扩展到（使用一个单一的数据库，例如）本地事务并扩展到全局事务（例如，使用 JTA）

Spring有哪些组件？

Spring Framework有哪些组件呢？

下图来自，[官方文档 Spring-framework 5.0](#)；需要注意的是，虽然这个图来源于Spring Framework 5.0 M4 版本，但是它依然是V4版本的图，比如Spring 5版本中的web模块已经去掉了Portlet模块，新增了WebFlux模块等。



上图中包含了 Spring 框架的所有模块，这些模块可以满足一切企业级应用开发的需求，在开发过程中可以根据需求有选择性地使用所需要的模块。下面分别对这些模块的作用进行简单介绍（并且结合SpringFramework5.x源码模块帮助你对应好各模块关系）。

Core Container (Spring的核心容器)

Spring 的核心容器是其他模块建立的基础，由 Beans 模块、Core 核心模块、Context 上下文模块和 SpEL 表达式语言模块组成，没有这些核心容器，也不可能有 AOP、Web 等上层的功能。具体介绍如下。

- **Beans 模块**：提供了框架的基础部分，包括控制反转和依赖注入。
- **Core 核心模块**：封装了 Spring 框架的底层部分，包括资源访问、类型转换及一些常用工具类。
- **Context 上下文模块**：建立在 Core 和 Beans 模块的基础之上，集成 Beans 模块功能并添加资源绑定、数据验证、国际化、Java EE 支持、容器生命周期、事件传播等。ApplicationContext 接口是上下文模块的焦点。
- **SpEL 模块**：提供了强大的表达式语言支持，支持访问和修改属性值，方法调用，支持访问及修改数组、容器和索引器，命名变量，支持算数和逻辑运算，支持从 Spring 容器获取 Bean，它也支持列表投影、选择和一般的列表聚合等。

对应的源码模块如下：

spring-aspects	Polishing	9 months ago
spring-beans	Polish contribution	8 days ago
spring-context-indexer	Polish Javadoc in spring-context-indexer	3 months ago
spring-context-support	Use StringBuilder.append(char) where possible	2 months ago
spring-context	Polish tests	9 days ago
spring-core	Polish contribution	8 days ago
spring-expression	Polish tests	9 days ago
spring-instrument	Delete obsolete log4j config	17 months ago
spring-jcl	Polishing	2 years ago
spring-jdbc	DataClassRowMapper suppresses setter method ...	last month
spring-jms	Polish contribution	8 days ago
spring-messaging	ObjectMapper.configure(MapperFeature, boolea...	24 days ago

Data Access/Integration (数据访问 / 集成)

数据访问 / 集成层包括 JDBC、ORM、OXM、JMS 和 Transactions 模块，具体介绍如下。

- **JDBC 模块**：提供了一个 JDBC 的样例模板，使用这些模板能消除传统冗长的 JDBC 编码还有必须的事务控制，而且能享受到 Spring 管理事务的好处。
- **ORM 模块**：提供与流行的“对象-关系”映射框架无缝集成的 API，包括 JPA、JDO、Hibernate 和 MyBatis 等。而且还可以使用 Spring 事务管理，无需额外控制事务。
- **OXM 模块**：提供了一个支持 Object /XML 映射的抽象层实现，如 JAXB、Castor、XMLBeans、JiBX 和 XStream。将 Java 对象映射成 XML 数据，或者将XML 数据映射成 Java 对象。
- **JMS 模块**：指 Java 消息服务，提供一套“消息生产者、消息消费者”模板用于更加简单的使用 JMS，JMS 用于用于在两个应用程序之间，或分布式系统中发送消息，进行异步通信。
- **Transactions 事务模块**：支持编程和声明式事务管理。

对应的源码模块如下：

spring-instrument	Delete obsolete log4j config	17 months ago
spring-jcl	Polishing	2 years ago
spring-jdbc	DataClassRowMapper suppresses setter method ...	last month
spring-jms	Polish contribution	8 days ago
spring-messaging	ObjectMapper.configure(MapperFeature, boolea...	24 days ago
spring-orm	Polish tests	4 months ago
spring-oxm	Polishing	last month
spring-r2dbc	Polishing	3 months ago
spring-test	Polish printMvcResultsToWriterWithFailingGlobal...	12 days ago
spring-tx	Lazy initialization of transaction UUID (with depr...	last month
spring-web	Add doOnDiscard hook for streaming mode	3 days ago
spring-webflux	Polishing contribution	12 days ago
spring-webmvc	Polish tests	9 days ago
spring-websocket	Avoid StringIndexOutOfBoundsException in Web...	22 days ago
src	Improve @Cacheable documentation regarding ...	20 days ago

Web模块

Spring 的 Web 层包括 Web、Servlet、WebSocket 和 Webflux 组件，具体介绍如下。

- **Web 模块**：提供了基本的 Web 开发集成特性，例如多文件上传功能、使用的 Servlet 监听器的 IOC 容器初始化以及 Web 应用上下文。
- **Servlet 模块**：提供了一个 Spring MVC Web 框架实现。Spring MVC 框架提供了基于注解的请求资源注入、更简单的数据绑定、数据验证等及一套非常易用的 JSP 标签，完全无缝与 Spring 其他技术协作。
- **WebSocket 模块**：提供了简单的接口，用户只要实现响应的接口就可以快速的搭建 WebSocket Server，从而实现双向通讯。
- **Webflux 模块**：Spring WebFlux 是 Spring Framework 5.x中引入的新的响应式web框架。与Spring MVC不同，它不需要Servlet API，是完全异步且非阻塞的，并且通过Reactor项目实现了Reactive Streams规范。Spring WebFlux 用于创建基于事件循环执行模型的完全异步且非阻塞的应用程序。

此外Spring4.x中还有Portlet 模块，在Spring 5.x中已经移除

- **Portlet 模块**：提供了在 Portlet 环境中使用 MVC 实现，类似 Web-Servlet 模块的功能。

对应的源码模块如下：

spring-tx	Lazy initialization of transaction UUID (with deprecated getter methods)	last month
spring-web	Add doOnDiscard hook for streaming mode	4 days ago
spring-webflux	Polishing contribution	12 days ago
spring-webmvc	Polish tests	9 days ago
spring-websocket	Avoid StringIndexOutOfBoundsException in WebSocketMessageBrokerSt...	22 days ago
src	Improve @Cacheable documentation regarding java.util.Optional	20 days ago
.editorconfig	Add EditorConfig	4 years ago

AOP、Aspects、Instrumentation和Messaging

在 Core Container 之上是 AOP、Aspects 等模块，具体介绍如下：

- **AOP 模块**：提供了面向切面编程实现，提供比如日志记录、权限控制、性能统计等通用功能和业务逻辑分离的技术，并且能动态的把这些功能添加到需要的代码中，这样各司其职，降低业务逻辑和通用功能的耦合。
- **Aspects 模块**：提供与 AspectJ 的集成，是一个功能强大且成熟的面向切面编程（AOP）框架。
- **Instrumentation 模块**：提供了类工具的支持和类加载器的实现，可以在特定的应用服务器中使用。
- **messaging 模块**：Spring 4.0 以后新增了消息（Spring-messaging）模块，该模块提供了对消息传递体系结构和协议的支持。
- **jcl 模块**：Spring 5.x中新增了日志框架集成的模块。

对应的源码模块如下：

framework-bom	Remove BOM workaround	16 months ago
gradle	Reintroduce left-hand side navigation in ref docs	last month
integration-tests	Drop explicit zeroing at instantiation of Atomic* ...	10 months ago
spring-aop	Polish contribution	16 days ago
spring-aspects	Polishing	9 months ago
spring-beans	Polish contribution	8 days ago
spring-context-indexer	Polish Javadoc in spring-context-indexer	3 months ago
spring-context-support	Use StringBuilder.append(char) where possible	2 months ago
spring-context	Polish tests	9 days ago
spring-core	Polish contribution	8 days ago
spring-expression	Polish tests	9 days ago
spring-instrument	Delete obsolete log4j config	17 months ago
spring-jcl	Polishing	2 years ago
spring-jdbc	DataClassRowMapper suppresses setter method ...	last month
spring-jms	Polish contribution	8 days ago
spring-messaging	ObjectMapper.configure(MapperFeature, boolea...	24 days ago

Test模块

包含Mock Objects, TestContext Framework, Spring MVC Test, WebTestClient。

对应的源码模块如下：

spring-test	Polishing	3 months ago
spring-r2dbc	Polish printMvcResultsToWriterWithFailingGlobalResultMatcher()	12 days ago
spring-tx	Lazy initialization of transaction UUID (with deprecated getter methods)	last month


为什么用Spring?

那么为什么用Spring呢？来看看官网对这个问题的回答👉

最重要的体现在它能做什么，这是Spring的核心所在


spring.io/why-spring

What can Spring do?




Microservices

Quickly deliver production-grade features with independently evolvable microservices.




Reactive

Spring's asynchronous, nonblocking architecture means you can get more from your computing resources.




Cloud

Your code, any cloud—we've got you covered. Connect and scale your services, whatever your platform.




Web apps

Frameworks for fast, secure, and responsive web applications connected to any data store.




Serverless

The ultimate flexibility. Scale up on demand and scale to zero when there's no demand.



Event Driven


Integrate with your enterprise. React to business events. Act on your streaming data in realtime.



Batch

Automated tasks. Offline processing of data at a time to suit you.

且官方对此专门对此做了详细介绍，感兴趣可以看下



Why Spring

Learn

Projects

Training

Support

Community

Why Spring?

Spring makes programming Java quicker, easier, and safer everybody. Spring's focus on speed, simplicity, and productivity made it the **world's most popular** Java framework.

Overview

Microservices

Reactive

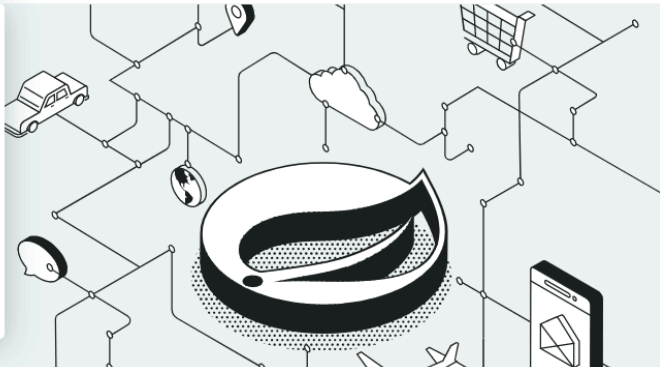
Event Driven

Cloud

Web Applications

Serverless

Batch

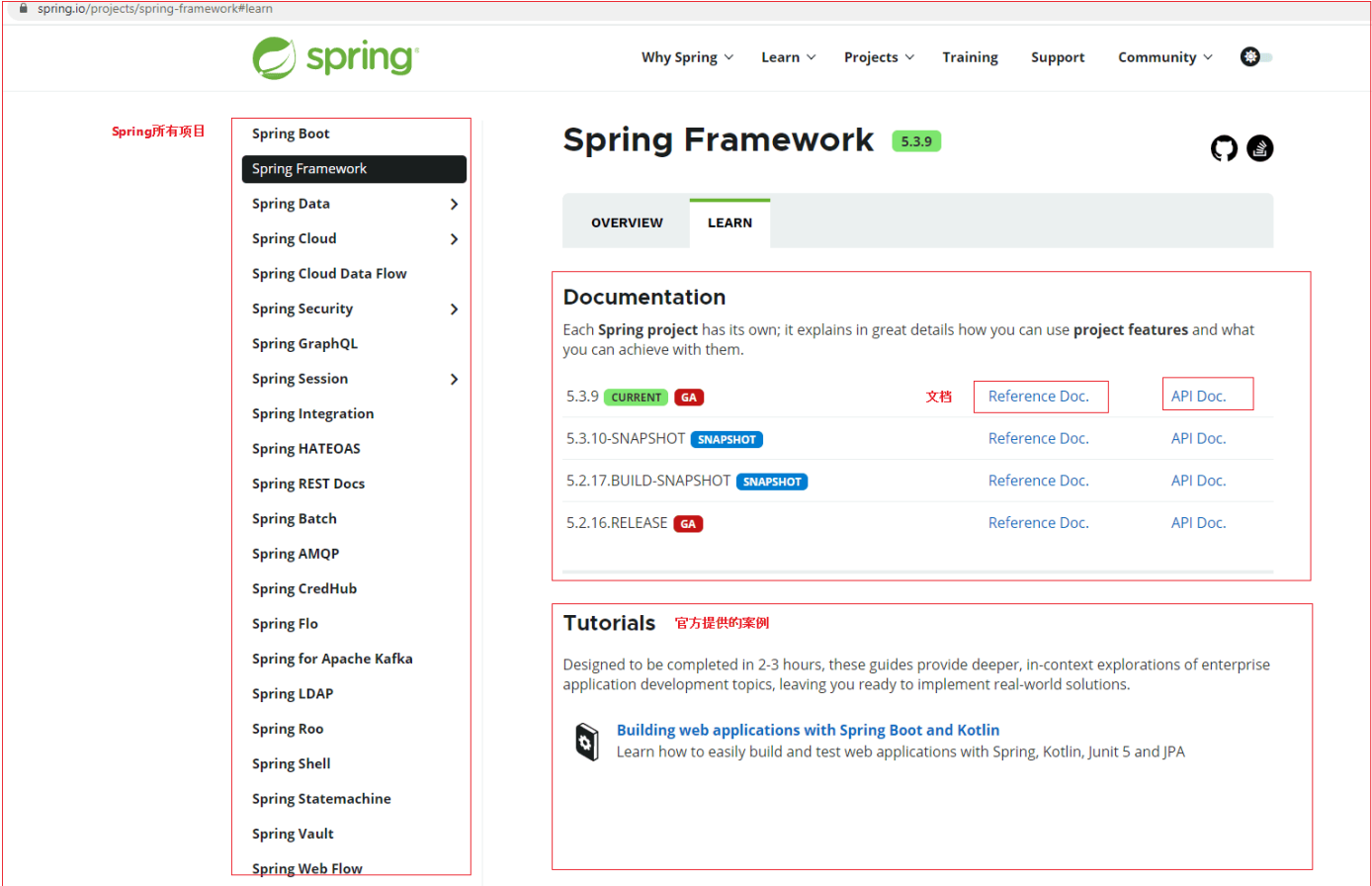


学习Spring时参考哪些资料呢？

非常负责的告诉你，最好最全的资料在Spring的官网，Spring能成为最主要的企业开发框架，文档和生态体系也做的很好；这里介绍下如何获取官方的学习资源。@pdai

Spring 的官方项目和教程

官方的项目和教程，地址[在这里](https://spring.io/projects/spring-framework#learn)，在学习Spring时，一定要把它当做生态体系，而不是一个简单的开发框架。



Spring 的归档文档

官方提供了系统性的文档的FTP，你可以在[这里](https://spring.io/projects/spring-framework#learn)找到所有历史版本的PDF/HTML版本。

	5.3.3 to 5.3.4/	2021-02-16 10:56	-
	5.3.4-SNAPSHOT/	2021-01-12 07:00	-
	5.3.4/	2021-02-16 11:00	-
	5.3.4 to 5.3.5/	2021-03-16 08:04	-
	5.3.5-SNAPSHOT/	2021-02-16 11:20	-
	5.3.5/	2021-03-16 08:20	-
	5.3.5 to 5.3.6/	2021-04-13 11:06	-
	5.3.6-SNAPSHOT/	2021-03-16 08:40	-
	5.3.6/	2021-04-13 11:20	-
	5.3.6 to 5.3.7/	2021-05-12 05:39	-
	5.3.7-SNAPSHOT/	2021-04-13 11:40	-
	5.3.7/	2021-05-12 06:00	-
	5.3.7 to 5.3.8/	2021-06-09 07:33	-
	5.3.8-SNAPSHOT/	2021-05-12 06:20	-
	5.3.8/	2021-06-09 08:00	-
	5.3.8 to 5.3.9/	2021-07-14 06:07	-
	5.3.9-SNAPSHOT/	2021-06-09 08:20	-
	5.3.9/	2021-07-14 07:00	-
	5.3.x-SNAPSHOT/	2021-07-14 07:40	-
	5.3.x/	2021-07-14 07:00	-
	current-SNAPSHOT/	2021-07-14 07:40	-
	current/	2021-07-14 07:00	-
	upgrade/	2010-03-01 16:02	-

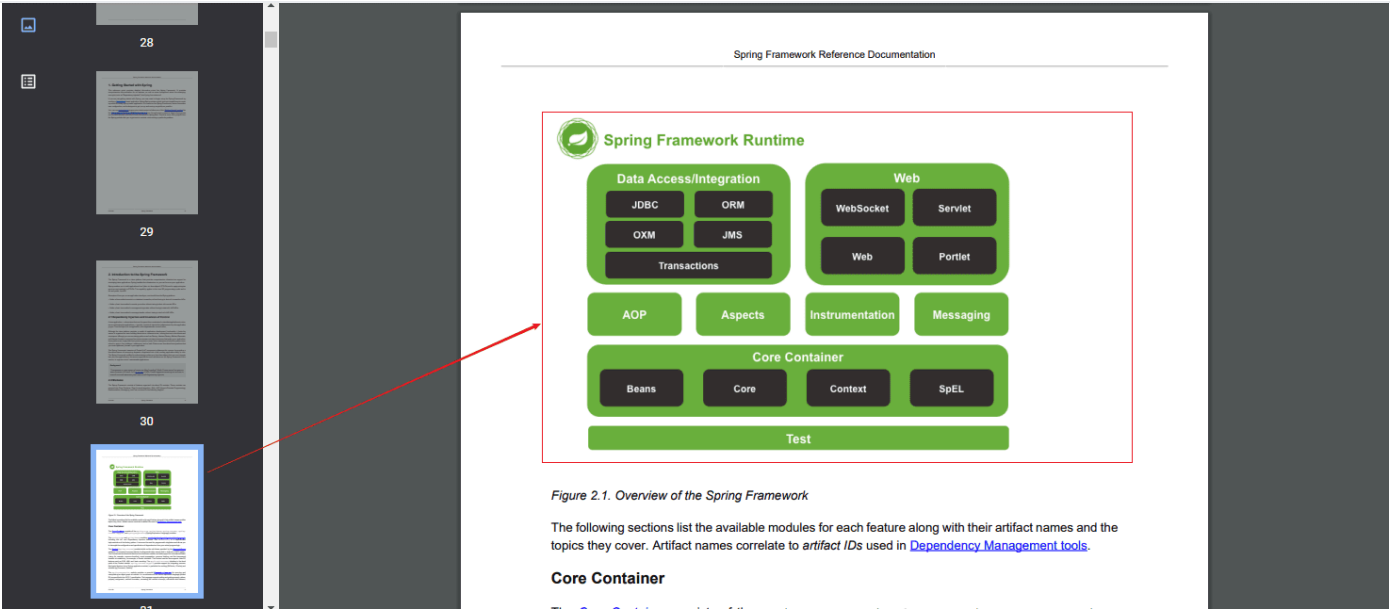
当前版本文档

可以看到很多系统性的文档，包括上面引用的图，

docs.spring.io/spring-framework/docs/5.0.0.M4/spring-framework-reference/...

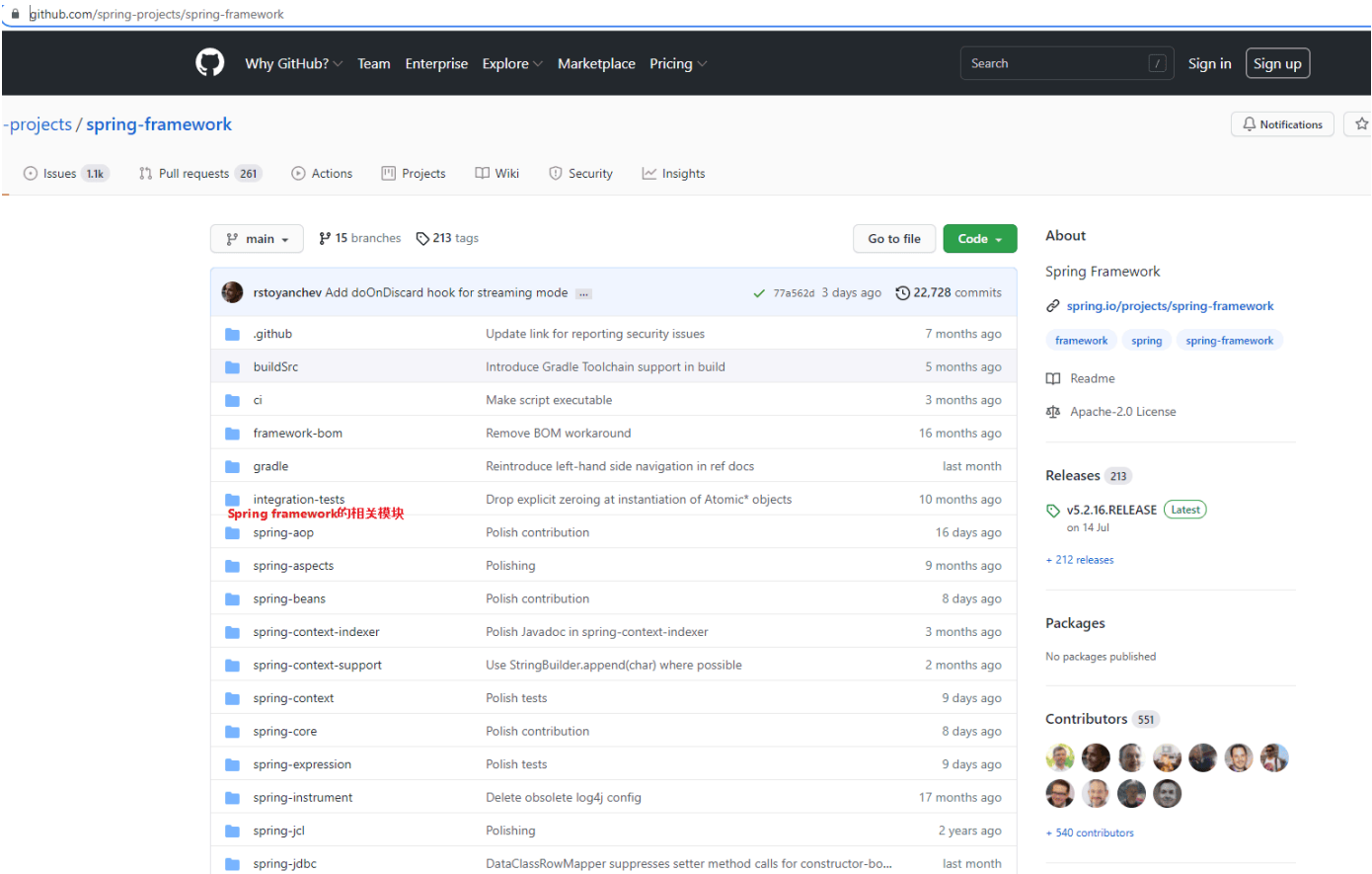
Index of /spring-framework/docs/5.0.0.M4/spring-framework-reference/pdf

Name	Last modified	Size	Description
 Parent Directory		-	
 spring-framework-ref...>	2016-12-30 13:10	5.2M	



Spring 的官方Github

Spring官方的GitHub在这里，它包含着Spring-framework的源码，如果你感兴趣，可以从这里clone代码进行阅读。



💡 我要纠错

← ♥Spring框架知识体系详解♥

Spring基础 - Spring简单例子引入Spring要点 →

苏ICP备19053722号 | pdai | copyright © 2017-present