# springboot

## Starter pom

Starter POMs是可以包含到应用中的一个方便的依赖关系描述符集合。你可以获取所有Spring及相关技术的一站式服务，而不需要翻阅示例代码，拷贝粘贴大量的依赖描述符。例如，如果你想使用Spring和JPA进行数据库访问，只需要在你的项目中包含spring-boot-starter-data-jpa依赖，然后你就可以开始了。

该starters包含很多你搭建项目，快速运行所需的依赖，并提供一致的，管理的传递依赖集。

名字有什么含义：所有的starters遵循一个相似的命名模式：spring-boot-starter-\*，在这里\*是一种特殊类型的应用程序。该命名结构旨在帮你找到需要的starter。很多IDEs集成的Maven允许你通过名称搜索依赖。例如，使用相应的Eclipse或STS插件，你可以简单地在POM编辑器中点击ctrl-space，然后输入”spring-boot-starter”可以获取一个完整列表。

| **名称** | **描述** |
| --- | --- |
| spring-boot-starter | 核心Spring Boot starter，包括自动配置支持，日志和YAML |
| spring-boot-starter-actuator | 生产准备的特性，用于帮你监控和管理应用 |
| spring-boot-starter-amqp | 对”高级消息队列协议”的支持，通过spring-rabbit实现 |
| spring-boot-starter-aop | 对面向切面编程的支持，包括spring-aop和AspectJ |
| spring-boot-starter-batch | 对Spring Batch的支持，包括HSQLDB数据库 |
| spring-boot-starter-cloud-connectors | 对Spring Cloud Connectors的支持，简化在云平台下（例如，Cloud Foundry 和Heroku）服务的连接 |
| spring-boot-starter-data-elasticsearch | 对Elasticsearch搜索和分析引擎的支持，包括spring-data-elasticsearch |
| spring-boot-starter-data-gemfire | 对GemFire分布式数据存储的支持，包括spring-data-gemfire |
| spring-boot-starter-data-jpa | 对”Java持久化API”的支持，包括spring-data-jpa，spring-orm和Hibernate |
| spring-boot-starter-data-mongodb | 对MongoDB NOSQL数据库的支持，包括spring-data-mongodb |
| spring-boot-starter-data-rest | 对通过REST暴露Spring Data仓库的支持，通过spring-data-rest-webmvc实现 |
| spring-boot-starter-data-solr | 对Apache Solr搜索平台的支持，包括spring-data-solr |
| spring-boot-starter-freemarker | 对FreeMarker模板引擎的支持 |
| spring-boot-starter-groovy-templates | 对Groovy模板引擎的支持 |
| spring-boot-starter-hateoas | 对基于HATEOAS的RESTful服务的支持，通过spring-hateoas实现 |
| spring-boot-starter-hornetq | 对”Java消息服务API”的支持，通过HornetQ实现 |
| spring-boot-starter-integration | 对普通spring-integration模块的支持 |
| spring-boot-starter-jdbc | 对JDBC数据库的支持 |
| spring-boot-starter-jersey | 对Jersey RESTful Web服务框架的支持 |
| spring-boot-starter-jta-atomikos | 对JTA分布式事务的支持，通过Atomikos实现 |
| spring-boot-starter-jta-bitronix | 对JTA分布式事务的支持，通过Bitronix实现 |
| spring-boot-starter-mail | 对javax.mail的支持 |
| spring-boot-starter-mobile | 对spring-mobile的支持 |
| spring-boot-starter-mustache | 对Mustache模板引擎的支持 |
| spring-boot-starter-redis | 对REDIS键值数据存储的支持，包括spring-redis |
| spring-boot-starter-security | 对spring-security的支持 |
| spring-boot-starter-social-facebook | 对spring-social-facebook的支持 |
| spring-boot-starter-social-linkedin | 对spring-social-linkedin的支持 |
| spring-boot-starter-social-twitter | 对spring-social-twitter的支持 |
| spring-boot-starter-test | 对常用测试依赖的支持，包括JUnit, Hamcrest和Mockito，还有spring-test模块 |
| spring-boot-starter-thymeleaf | 对Thymeleaf模板引擎的支持，包括和Spring的集成 |
| spring-boot-starter-velocity | 对Velocity模板引擎的支持 |
| spring-boot-starter-web | 对全栈web开发的支持，包括Tomcat和spring-webmvc |
| spring-boot-starter-websocket | 对WebSocket开发的支持 |
| spring-boot-starter-ws | 对Spring Web服务的支持 |

生产准备的starters

| **名称** | **描述** |
| --- | --- |
| spring-boot-starter-actuator | 添加生产准备特性，比如指标和监控 |
| spring-boot-starter-remote-shell | 添加远程ssh shell支持 |

technical starters

| **名称** | **描述** |
| --- | --- |
| spring-boot-starter-jetty | 导入Jetty HTTP引擎（作为Tomcat的替代） |
| spring-boot-starter-log4j | 对Log4J日志系统的支持 |
| spring-boot-starter-logging | 导入Spring Boot的默认日志系统（Logback） |
| spring-boot-starter-tomcat | 导入Spring Boot的默认HTTP引擎（Tomcat） |
| spring-boot-starter-undertow | 导入Undertow HTTP引擎（作为Tomcat的替代） |

### 三级标题

#### 四级标题

#### Xxx

## JavaConfig

概念：<https://blog.csdn.net/albenxie/article/details/82633775>

根据源码进行讲解：

<https://baijiahao.baidu.com/s?id=1655850697384425668&wfr=spider&for=pc>

spring3.0新增了许多java config的注解，特别是spring boot，基本都是清一色的java config。

@Configuration

在类上打上这一标签，表示这个类是配置类

@ComponentScan

相当于xml的<context:componentscan basepakage=>

@Bean

bean的定义，相当于xml的

<bean id="objectMapper" class="org.codehaus.jackson.map.ObjectMapper" />

@EnableWebMvc

相当于xml的<mvc:annotation-driven>

@ImportResource

相当于xml的 <import resource="applicationContext-cache.xml">

@PropertySource

spring 3.1开始引入，它是基于java config的注解，用于读取properties文件

@Profile

spring3.1开始引入,一般用于多环境配置，激活时可用@ActiveProfiles注解,@ActiveProfiles("dev")

等同于xml配置

<beans profile="dev">

<bean id="beanname" class="com.pz.demo.ProductRPC"/>

</beans>

激活该profile spring.profiles.active,也可设置默认值 spring.profiles.default

<context-param>

<param-name>spring.profiles.default</param-name>

<param-value>dev</param-value>

</context-param>

### 作用

防止Spring项目大量的烂用XML

XML唯一的问题是，只有在运行时环境时你才能发现各种配置及语法错误，但是如果使用Spring IDE Plugin(或者STS)的话，它会在编码时提示这些问题。

使用XML来配置Bean失去了编译时的类型安全检查。

Spring使用注释来描述Bean的配置与采用XML相比，因类注释是在一个类源代码中，可以获得类型安全检查的好处。可以良好的支持重构。

## 不同环境的应用

开发一个项目有几个阶段，开发、测试、生产。这代表着它们可能处于不同的环境中，其中针对某一阶段一些配置和配置文件等也不尽相同。

配置文件中使用spring.profiles.active来指定所处的环境或使用哪一套配置

# 一级标题

## 二级标题

### 三级标题

#### 四级标题

#### xxx