# 注:

By default, Spring Boot 1.4.1.RELEASE requires [Java 7](http://www.java.com/) and Spring Framework 4.3.3.RELEASE or above.

自己的开发环境: Jdk:1.79 ide:idea2016.1 maven:使用idea自带的3.0.5

Springboot项目的运行的两种方式

1. 在项目中创建一个main函数,使用下面方法来运行Application的配置类

SpringApplication.*run*(Application.class, args)

1. 生成可运行的jar包在生产环境运行,这种方式必须在maven配置中加入spring-boot-maven-plugin.

然后使用maven生命周期中的packge来打包项目,必须注意项目只能有一个main函数.

使用java命令jar tvf跟java –jar,可以查看包目录和运行包

# SpringBoot主要的目的:

简化xml配置,简化应用的打包部署流程,提供一种开包即用的服务.

官方Features:

* Create stand-alone Spring applications
* Embed Tomcat, Jetty or Undertow directly (no need to deploy WAR files)
* Provide opinionated 'starter' POMs to simplify your Maven configuration
* Automatically configure Spring whenever possible
* Provide production-ready features such as metrics, health checks and externalized configuration
* Absolutely **no code generation** and **no requirement for XML** configuration

# Maven配置:

<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
 <groupId>com.study</groupId>  
 <artifactId>springboot</artifactId>  
 <version>0.0.1</version>  
 <packaging>jar</packaging>  
 <name>springBoot</name>  
 <description>Study Spring Boot</description>  
 <!--继承springBoot父项目-->  
 <parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>1.4.1.RELEASE</version>  
 <relativePath/>  
 </parent>  
 <properties>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>  
 <java.version>1.7</java.version>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-devtools</artifactId>  
 <optional>true</optional>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-test</artifactId>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
 <build>  
 <plugins>  
 <plugin>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-maven-plugin</artifactId>  
 </plugin>  
 </plugins>  
 </build>  
</project>

如果不继承spring-boot-starter-parent项目,需要在maven加入如下依赖

<dependency>

<groupId>org.springframework.data</groupId>

<artifactId>spring-data-releasetrain</artifactId>

<version>Fowler-SR2</version>

<scope>import</scope>

<type>pom</type>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-dependencies</artifactId>

<version>1.4.1.RELEASE</version>

<type>pom</type>

<scope>import</scope>

</dependency>

# SpringBoot运行原理:

1. SpringBoot的核心功能就是由@EnableAutoConfiguration注解来提供,然后该注解导入了EnableAutoConfigurationImportSelector这个类中的SpringFactoriesLoader.loadFactoryName方法来扫描spring-boot-autoconfiguration.jar包下的spring.factories文件,此文件中声明了有哪些自动配置.

# annotation总结分析

基于类的注解:  
--初始装载  
@SpringBootApplication              spring-boot程序入口标志类  
@Configuration       自动配置，类似于加载spring加载xml 装配所有的bean事务等 所标识的类里面可以使用

@Bean 并且启动的时候会初始化bean

@ImportResource 如果需要使用xml文件作为基础配置,可以使用这个注解来引入配置文件  
@EnableAutoConfiguration       Spring-Boot 根据应用所声明的依赖来对Spring框架进行自动配置  
@ComponentScan                     规定扫描包的范围  
@PropertySources                  property扫描加载

@PropertySource 指定这个类读取配置信息的配置文件  
--业务区分  
@Component 定义该bean为一个普通组件  
@Repository 定义该bean是一个仓储，用于数据库、mq、redis以及其它一些远程访问的资源  
@Service  定义该bean是一个业务逻辑  
@Controller  定义该bean是一个控制页面访问层  
--加载条件  
@Order  配置加载顺序  
@ConditionalOnClass  该注解的参数对应的类必须存在，否则不解析该注解修饰的配置类；  
@ConditionalOnMissingBean  该注解表示，如果存在它修饰的类的bean，则不需要再创建这个bean；可以给该注解传入参数例如@ConditionOnMissingBean(name = "example")，这个表示如果name为“example”的bean存在，这该注解修饰的代码块不执行。  
@AutoConfigureAfter 在摸个自动装载类之后装载  
  
基于属性的注解:  
@Value 加载配置属性的值  
@Autowired 自动注入bean  
@Qualifier 当存在多个bean注入时，需要通过name进行过滤  
@Resource 获取当前jvm的resource 也类似依赖注入  
@Inject 字段注入bean

@Scope 描述Spring容器如何创建bean的实例.Scope有以下几种:

1. Singleton: 单例;
2. Prototype:每次调用都创建一个新的实例;
3. Request:Web项目中,给每一个http request新建一个Bean实例:
4. Session: Web项目中,给每一个http request新建一个Bean实例:
5. GlobalSession:这个只在portal应用中有用,给每一个global http session 新建一个Bean实例

基于方法的注解:  
@Bean  发布一个返回object类型的bean，类似配置xml发布一个bean  
@PostConstruct 指定当类加载完成的时候就会执行该方法

# SpringBootStarters介绍

SpringBoot拥有很多starter组件,如果项目中需要添加对应的技术就需要添加对应的Starter.下面是SpringBoot 1.4.1 官方给出的所有Starter.

| Name | Description | Pom |
| --- | --- | --- |
| spring-boot-starter-thymeleaf | Starter for building MVC web applications using Thymeleaf views | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-thymeleaf/pom.xml) |
| spring-boot-starter-ws | Starter for using Spring Web Services. Deprecated as of 1.4 in favor of[spring-boot-starter-web-services](http://docs.spring.io/spring-boot/docs/1.4.1.RELEASE/reference/htmlsingle/#spring-boot-starter-web-services) | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-ws/pom.xml) |
| spring-boot-starter-data-couchbase | Starter for using Couchbase document-oriented database and Spring Data Couchbase | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-data-couchbase/pom.xml) |
| spring-boot-starter-artemis | Starter for JMS messaging using Apache Artemis | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-artemis/pom.xml) |
| spring-boot-starter-web-services | Starter for using Spring Web Services | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-web-services/pom.xml) |
| spring-boot-starter-mail | Starter for using Java Mail and Spring Framework’s email sending support | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-mail/pom.xml) |
| spring-boot-starter-data-redis | Starter for using Redis key-value data store with Spring Data Redis and the Jedis client | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-data-redis/pom.xml) |
| spring-boot-starter-web | Starter for building web, including RESTful, applications using Spring MVC. Uses Tomcat as the default embedded container | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-web/pom.xml) |
| spring-boot-starter-data-gemfire | Starter for using GemFire distributed data store and Spring Data GemFire | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-data-gemfire/pom.xml) |
| spring-boot-starter-activemq | Starter for JMS messaging using Apache ActiveMQ | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-activemq/pom.xml) |
| spring-boot-starter-data-elasticsearch | Starter for using Elasticsearch search and analytics engine and Spring Data Elasticsearch | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-data-elasticsearch/pom.xml) |
| spring-boot-starter-integration | Starter for using Spring Integration | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-integration/pom.xml) |
| spring-boot-starter-test | Starter for testing Spring Boot applications with libraries including JUnit, Hamcrest and Mockito | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-test/pom.xml) |
| spring-boot-starter-hornetq | Starter for JMS messaging using HornetQ. Deprecated as of 1.4 in favor of[spring-boot-starter-artemis](http://docs.spring.io/spring-boot/docs/1.4.1.RELEASE/reference/htmlsingle/#spring-boot-starter-artemis) | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-hornetq/pom.xml) |
| spring-boot-starter-jdbc | Starter for using JDBC with the Tomcat JDBC connection pool | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-jdbc/pom.xml) |
| spring-boot-starter-mobile | Starter for building web applications using Spring Mobile | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-mobile/pom.xml) |
| spring-boot-starter-validation | Starter for using Java Bean Validation with Hibernate Validator | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-validation/pom.xml) |
| spring-boot-starter-hateoas | Starter for building hypermedia-based RESTful web application with Spring MVC and Spring HATEOAS | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-hateoas/pom.xml) |
| spring-boot-starter-jersey | Starter for building RESTful web applications using JAX-RS and Jersey. An alternative to[spring-boot-starter-web](http://docs.spring.io/spring-boot/docs/1.4.1.RELEASE/reference/htmlsingle/#spring-boot-starter-web) | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-jersey/pom.xml) |
| spring-boot-starter-data-neo4j | Starter for using Neo4j graph database and Spring Data Neo4j | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-data-neo4j/pom.xml) |
| spring-boot-starter-websocket | Starter for building WebSocket applications using Spring Framework’s WebSocket support | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-websocket/pom.xml) |
| spring-boot-starter-aop | Starter for aspect-oriented programming with Spring AOP and AspectJ | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-aop/pom.xml) |
| spring-boot-starter-amqp | Starter for using Spring AMQP and Rabbit MQ | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-amqp/pom.xml) |
| spring-boot-starter-data-cassandra | Starter for using Cassandra distributed database and Spring Data Cassandra | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-data-cassandra/pom.xml) |
| spring-boot-starter-social-facebook | Starter for using Spring Social Facebook | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-social-facebook/pom.xml) |
| spring-boot-starter-jta-atomikos | Starter for JTA transactions using Atomikos | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-jta-atomikos/pom.xml) |
| spring-boot-starter-security | Starter for using Spring Security | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-security/pom.xml) |
| spring-boot-starter-mustache | Starter for building MVC web applications using Mustache views | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-mustache/pom.xml) |
| spring-boot-starter-data-jpa | Starter for using Spring Data JPA with Hibernate | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-data-jpa/pom.xml) |
| spring-boot-starter | Core starter, including auto-configuration support, logging and YAML | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter/pom.xml) |
| spring-boot-starter-velocity | Starter for building MVC web applications using Velocity views. Deprecated since 1.4 | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-velocity/pom.xml) |
| spring-boot-starter-groovy-templates | Starter for building MVC web applications using Groovy Templates views | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-groovy-templates/pom.xml) |
| spring-boot-starter-freemarker | Starter for building MVC web applications using FreeMarker views | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-freemarker/pom.xml) |
| spring-boot-starter-batch | Starter for using Spring Batch | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-batch/pom.xml) |
| spring-boot-starter-redis | Starter for using Redis key-value data store with Spring Data Redis and the Jedis client. Deprecated as of 1.4 in favor of [spring-boot-starter-data-redis](http://docs.spring.io/spring-boot/docs/1.4.1.RELEASE/reference/htmlsingle/#spring-boot-starter-data-redis) | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-redis/pom.xml) |
| spring-boot-starter-social-linkedin | Stater for using Spring Social LinkedIn | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-social-linkedin/pom.xml) |
| spring-boot-starter-cache | Starter for using Spring Framework’s caching support | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-cache/pom.xml) |
| spring-boot-starter-data-solr | Starter for using the Apache Solr search platform with Spring Data Solr | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-data-solr/pom.xml) |
| spring-boot-starter-data-mongodb | Starter for using MongoDB document-oriented database and Spring Data MongoDB | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-data-mongodb/pom.xml) |
| spring-boot-starter-jooq | Starter for using jOOQ to access SQL databases. An alternative to[spring-boot-starter-data-jpa](http://docs.spring.io/spring-boot/docs/1.4.1.RELEASE/reference/htmlsingle/#spring-boot-starter-data-jpa) or [spring-boot-starter-jdbc](http://docs.spring.io/spring-boot/docs/1.4.1.RELEASE/reference/htmlsingle/#spring-boot-starter-jdbc) | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-jooq/pom.xml) |
| spring-boot-starter-jta-narayana | Spring Boot Narayana JTA Starter | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-jta-narayana/pom.xml) |
| spring-boot-starter-cloud-connectors | Starter for using Spring Cloud Connectors which simplifies connecting to services in cloud platforms like Cloud Foundry and Heroku | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-cloud-connectors/pom.xml) |
| spring-boot-starter-jta-bitronix | Starter for JTA transactions using Bitronix | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-jta-bitronix/pom.xml) |
| spring-boot-starter-social-twitter | Starter for using Spring Social Twitter | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-social-twitter/pom.xml) |
| spring-boot-starter-data-rest | Starter for exposing Spring Data repositories over REST using Spring Data REST | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-data-rest/pom.xml) |

**Spring Boot production starters**

|  |  |  |
| --- | --- | --- |
| spring-boot-starter-actuator | Starter for using Spring Boot’s Actuator which provides production ready features to help you monitor and manage your application | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-actuator/pom.xml) |
| spring-boot-starter-remote-shell | Starter for using the CRaSH remote shell to monitor and manage your application over SSH | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-remote-shell/pom.xml) |

**Spring Boot technical starters**

|  |  |  |
| --- | --- | --- |
| spring-boot-starter-undertow | Starter for using Undertow as the embedded servlet container. An alternative to [spring-boot-starter-tomcat](http://docs.spring.io/spring-boot/docs/1.4.1.RELEASE/reference/htmlsingle/#spring-boot-starter-tomcat) | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-undertow/pom.xml) |
| spring-boot-starter-jetty | Starter for using Jetty as the embedded servlet container. An alternative to [spring-boot-starter-tomcat](http://docs.spring.io/spring-boot/docs/1.4.1.RELEASE/reference/htmlsingle/#spring-boot-starter-tomcat) | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-jetty/pom.xml) |
| spring-boot-starter-logging | Starter for logging using Logback. Default logging starter | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-logging/pom.xml) |
| spring-boot-starter-tomcat | Starter for using Tomcat as the embedded servlet container. Default servlet container starter used by[spring-boot-starter-web](http://docs.spring.io/spring-boot/docs/1.4.1.RELEASE/reference/htmlsingle/#spring-boot-starter-web) | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-tomcat/pom.xml) |
| spring-boot-starter-log4j2 | Starter for using Log4j2 for logging. An alternative to [spring-boot-starter-logging](http://docs.spring.io/spring-boot/docs/1.4.1.RELEASE/reference/htmlsingle/#spring-boot-starter-logging) | [Pom](https://github.com/spring-projects/spring-boot/tree/v1.4.1.RELEASE/spring-boot-starters/spring-boot-starter-log4j2/pom.xml) |

# SpringBootStarters和工具使用

Developer tools are automatically disabled when running a fully packaged application.

主要功能:提供Reload和Restart ,远端连接,远端调试等功能

# mybatis集成

## 添加依赖:

<!-- mybatis支持 -->  
<dependency>  
 <groupId>org.mybatis.spring.boot</groupId>  
 <artifactId>mybatis-spring-boot-starter</artifactId>  
 <version>1.3.0</version>  
</dependency>

## 在springboot application.yml配置文件上配置mybatis

**mybatis:** *#配置mybatis的类型命名,扫描下面包下的Alias注解的类* **type-aliases-package:** com.aygx.myblog.entity  
 *#引入mybatis的mapper文件* **mapper-locations:** classpath:mapper/\*Mapper.xml  
 *#引入mybatis配置文件* **config-location:** classpath:mybatis-conf.xml

## 配置全局datasource

**spring:  
 datasource:  
 url:** jdbc:mysql://localhost:3306/myblog  
 **username:** root  
 **password:** root  
 **driver-class-name:** com.mysql.jdbc.Driver

# mybatis集成pagehelper实现分页查询

## 引入依赖

<dependency>  
 <groupId>com.github.pagehelper</groupId>  
 <artifactId>pagehelper</artifactId>  
 <version>3.7.5</version>  
</dependency>

## Mybatis配置表中添加拦截器插件

<plugins>  
 <!-- com.github.pagehelper为PageHelper类所在包名 -->  
 <plugin interceptor="com.github.pagehelper.PageHelper">  
 <property name="dialect" value="mysql"/>  
 <!-- 该参数默认为false -->  
 <!-- 设置为true时，会将RowBounds第一个参数offset当成pageNum页码使用 -->  
 <!-- 和startPage中的pageNum效果一样-->  
 <property name="offsetAsPageNum" value="true"/>  
 <!-- 该参数默认为false -->  
 <!-- 设置为true时，使用RowBounds分页会进行count查询 -->  
 <property name="rowBoundsWithCount" value="true"/>  
 <!-- 设置为true时，如果pageSize=0或者RowBounds.limit = 0就会查询出全部的结果 -->  
 <!-- （相当于没有执行分页查询，但是返回结果仍然是Page类型）-->  
 <property name="pageSizeZero" value="true"/>  
 <!-- 3.3.0版本可用 - 分页参数合理化，默认false禁用 -->  
 <!-- 启用合理化时，如果pageNum<1会查询第一页，如果pageNum>pages会查询最后一页 -->  
 <!-- 禁用合理化时，如果pageNum<1或pageNum>pages会返回空数据 -->  
 <property name="reasonable" value="false"/>  
 <!-- 3.5.0版本可用 - 为了支持startPage(Object params)方法 -->  
 <!-- 增加了一个`params`参数来配置参数映射，用于从Map或ServletRequest中取值 -->  
 <!-- 可以配置pageNum,pageSize,count,pageSizeZero,reasonable,不配置映射的用默认值 -->  
 <!-- 不理解该含义的前提下，不要随便复制该配置 -->  
 <property name="params" value="pageNum=start;pageSize=limit;"/>  
 <!-- always总是返回PageInfo类型,check检查返回类型是否为PageInfo,none返回Page -->  
 <property name="returnPageInfo" value="check"/>  
 </plugin>  
</plugins>

## 测试样例

//第一个参数是页码,第二个参数是页面大小  
Page<Article> page = PageHelper.*startPage*(1,1);  
List<Article> articles = articleMapper.getArticles();  
System.*out*.println(page);  
PageInfo<Article> info = new PageInfo<Article>(page);  
System.*out*.println(info);  
for (Article article : articles){  
 System.*out*.println(article);  
}

# 自行配置Servlet、Filter、Listener

## 代码注册通过

ServletRegistrationBean、 FilterRegistrationBean 和 ServletListenerRegistrationBean 获得控制。 也可以通过实现 ServletContextInitializer 接口直接注册。

## 注解扫描方式

在 SpringBootApplication 上使用@ServletComponentScan注解后，Servlet、Filter、Listener 可以直接通过 @WebServlet、@WebFilter、@WebListener 注解自动注册，无需其他代码。

# 整合CXF

@Configuration  
public class CxfConfig {  
 @Bean  
 public ServletRegistrationBean dispatcherServlet() {  
 return new ServletRegistrationBean(new CXFServlet(), "/services/\*");  
 }  
 @Bean(name = Bus.DEFAULT\_BUS\_ID)  
 public SpringBus springBus() {  
 return new SpringBus();  
 }  
 @Bean  
 public UserService userService() {  
 return new UserServiceImpl();  
 }  
 @Bean  
 public Endpoint endpoint() {  
 EndpointImpl endpoint = new EndpointImpl(springBus(), userService());  
 endpoint.publish("/user");  
 return endpoint;  
 }  
}

# Spring问题汇总

|  |  |
| --- | --- |
| 问题: | 解决办法 |
| 使用maven打包的时候报错显示读取了配置信息的类无法创建,并且提示无法找到对应的配置信息. | 这种情况下在该类指定配置信息: @PropertySource("classpath:config/application-custom.properties") |
| Spring Security4 对http post操作方法进行了防跨域请求拦截.  在跨域的场景下，客户端访问服务端会首先发起get请求，这个get请求在到达服务端的时候，服务端的Spring security会有一个过滤 器 CsrfFilter去检查这个请求，如果这个request请求的http header里面的X-CSRF-COOKIE的token值为空的时候，服务端就好自动生成一个 token值放进这个X-CSRF-COOKIE值里面，客户端在get请求的header里面获取到这个值，如果客户端有表单提交的post请求，则要求客户端要 携带这个token值给服务端，在post请求的header里面设置\_csrf属性的token值，提交的方式可以是ajax也可以是放在form里面设置hidden 属性的标签里面提交给服务端，服务端就会根据post请求里面携带的token值进行校验，如果跟服务端发送给合法客户端的token值是一样的，那么 这个post请求就可以受理和处理，如果不一样或者为空，就会被拦截。由于恶意第三方可以劫持session id，而很难获取token值，所以起到了 安全的防护作用。 | 1. 可以通过配置个人拦截器   http.addFilterBefore(customizeFilterSecurityInterceptor, FilterSecurityInterceptor.class);   1. 关闭spring security的scrf的拦截   http.csrf().disable(); |
|  |  |
|  |  |