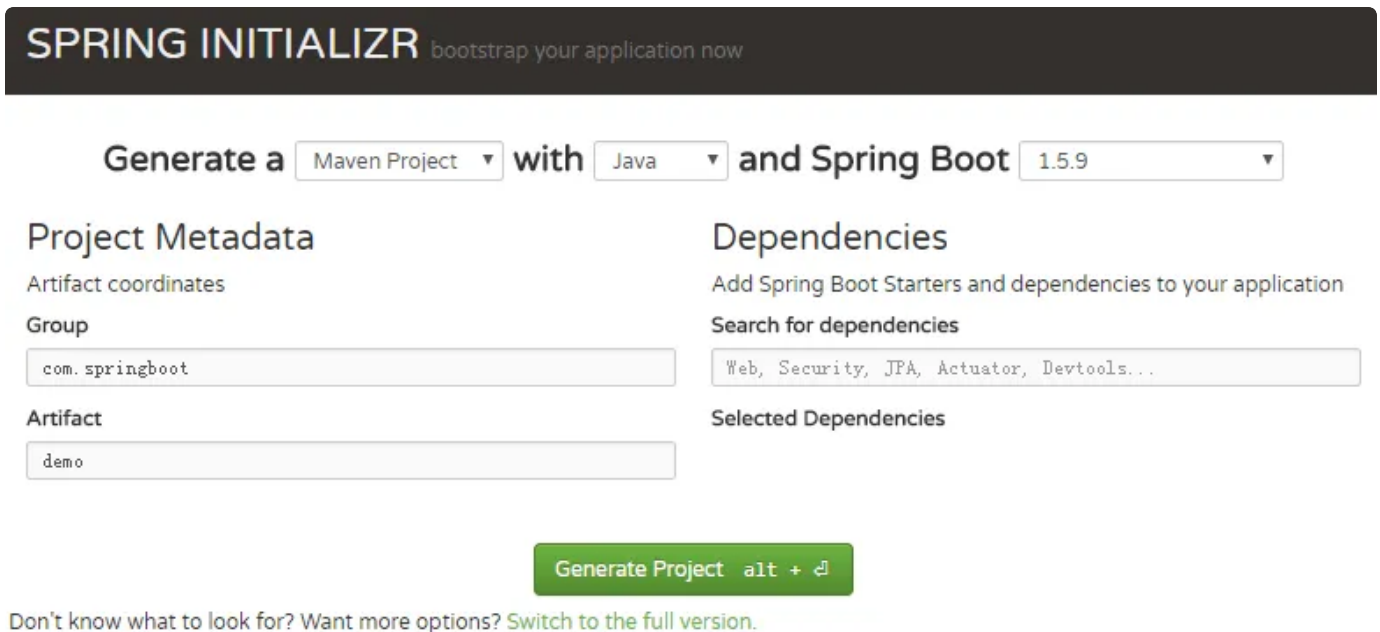


1. 开启Spring Boot

Spring Boot是在Spring框架上创建的一个全新的框架，其设计目的是简化Spring应用的搭建和开发过程。开启Spring Boot有许多种方法可供选择，这里仅介绍使用<http://start.spring.io/>来构建一个简单的Spring Boot项目。

生成项目文件

访问<http://start.spring.io/>，页面显示如下：



The screenshot shows the Spring Initializr web application interface. At the top, there is a dark header with the text "SPRING INITIALIZR" and a subtitle "bootstrap your application now". Below the header, there is a form to generate a project. The form has three main sections: "Generate a", "with", and "and Spring Boot". The "Generate a" section has a dropdown menu set to "Maven Project". The "with" section has a dropdown menu set to "Java". The "and Spring Boot" section has a dropdown menu set to "1.5.9". Below these sections, there are two columns. The left column is titled "Project Metadata" and contains two input fields: "Group" with the value "com.springboot" and "Artifact" with the value "demo". The right column is titled "Dependencies" and contains a search bar with the text "Web, Security, JPA, Actuator, Devtools...". Below the search bar, there is a section titled "Selected Dependencies". At the bottom of the form, there is a green button labeled "Generate Project" with a keyboard shortcut "alt + ⌘". Below the button, there is a link that says "Don't know what to look for? Want more options? [Switch to the full version.](#)"

这里选择以Maven构建，语言选择Java，Spring Boot版本为1.5.9。然后点击Switch to the full version，可看到更多的配置以及依赖选择：

Project Metadata

Artifact coordinates

Group

com.springboot

Artifact

demo

Name

demo

Description

Demo project for Spring Boot

Package Name

com.springboot.demo

Packaging

Jar

Java Version

7

Too many options? [Switch back to the simple version.](#)

在项目信息里选择以jar包的方式部署，Java版本为7。在页面的下方还可以选择诸多的依赖，这里仅选择web进行演示：

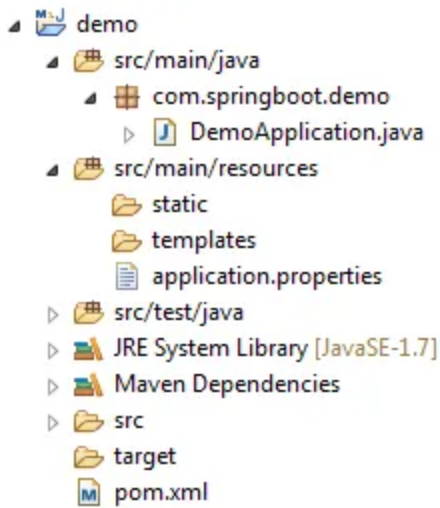
Core

- ☐ Security
Secure your application via spring-security
- ☐ Aspects
Create your own Aspects using Spring AOP and AspectJ
- ☐ Atomikos (JTA)
JTA distributed transactions via Atomikos
- ☐ Bitronix (JTA)

Web

- ☒ Web
Full-stack web development with Tomcat and Spring MVC
- ☐ Reactive Web
Reactive web development with Netty and Spring WebFlux
requires Spring Boot >=2.0.0.M1
- ☐ Websocket
Websocket development with SockJS and STOMP

最后点击页面的generate project按钮生成项目文件。文件下载后是一个压缩包，进行解压然后使用eclipse以Maven项目的形式导入。导入后eclipse会自动编译项目并下载相应的依赖，项目目录如下所示：

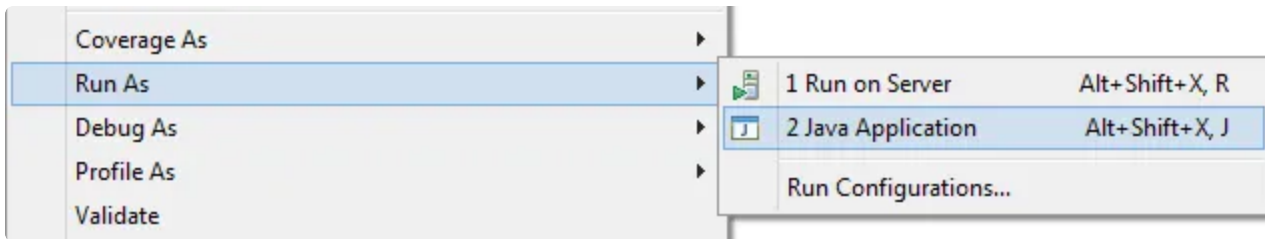


简单演示

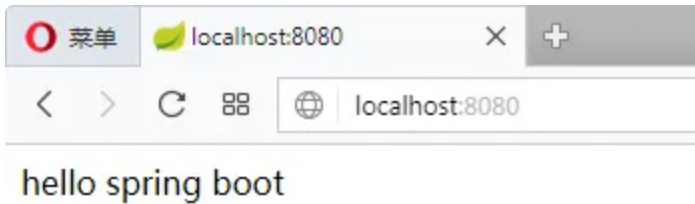
项目根目录下生成了一个artifactId+Application命名规则的入口类，为了演示简单，不再新建控制器，直接在入口类中编写代码：

```
1  package com.springboot.demo;
2
3  import org.springframework.boot.SpringApplication;
4  import org.springframework.boot.autoconfigure.SpringBootApplication;
5  import org.springframework.web.bind.annotation.RequestMapping;
6  import org.springframework.web.bind.annotation.RestController;
7
8  @RestController
9  @SpringBootApplication
10 public class DemoApplication {
11
12     @RequestMapping("/")
13     String index() {
14         return "hello spring boot";
15     }
16
17     public static void main(String[] args) {
18         SpringApplication.run(DemoApplication.class, args);
19     }
20 }
```

然后右键点击DemoApplication，选择run as → Java Application：

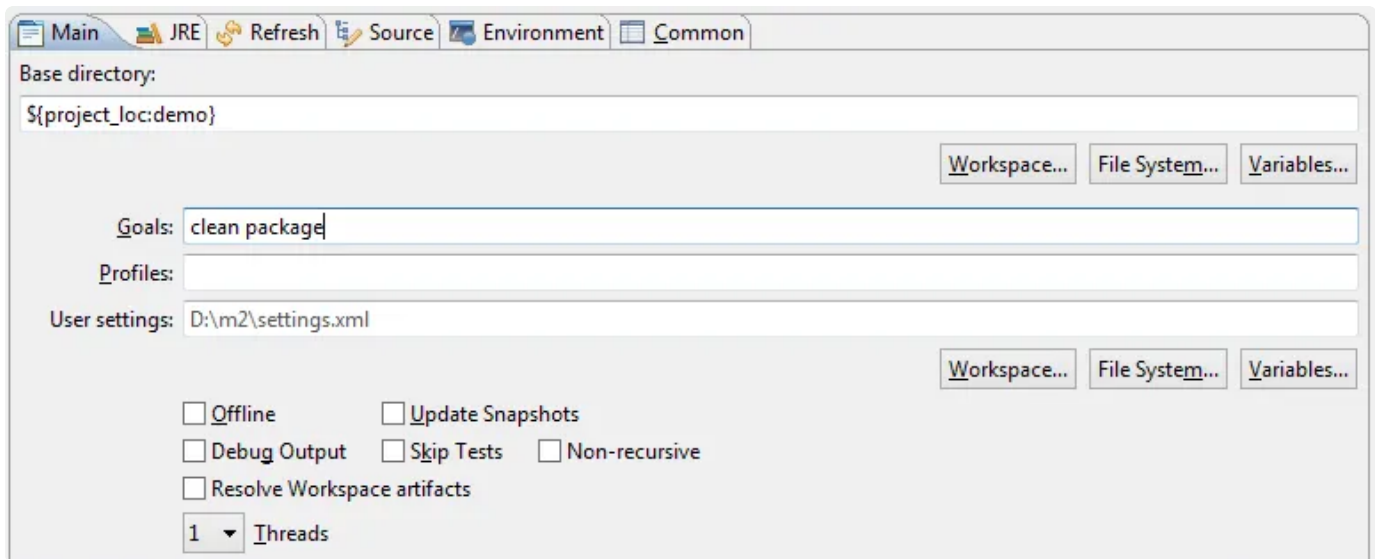


访问<http://localhost:8080>，页面显示如下：



打包发布

在eclipse中右击项目，选择run as → Maven build...，如下图所示：



在Goals中输入 `clean package` 命令，然后点击下方的run就将项目打包成jar包（初次打包会自动下载一些依赖）。

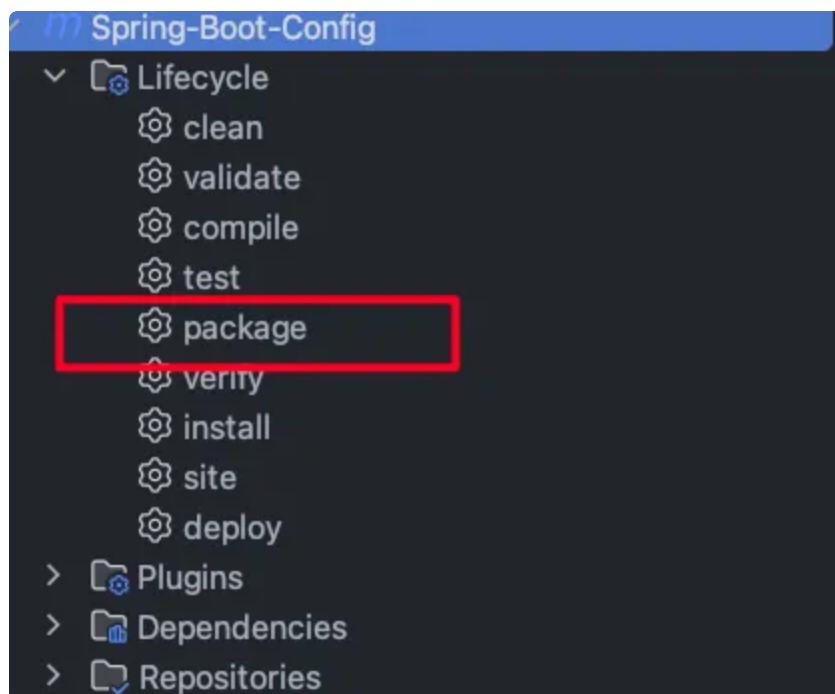
打包完毕后可看到项目目录target文件夹下生成了一个jar文件：



访问<http://localhost:8080>，效果如上。

打包发布2

通过maven自带的插件进行打包



聊聊pom.xml

打开pom.xml可看到配置如下：

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w
3.org/2001/XMLSchema-instance"
4   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apach
5e.org/xsd/maven-4.0.0.xsd">
6   <modelVersion>4.0.0</modelVersion>
7
8   <groupId>com.springboot</groupId>
9   <artifactId>demo</artifactId>
10   <version>0.0.1-SNAPSHOT</version>
11   <packaging>jar</packaging>
12
13   <name>Start-Spring-Boot</name>
14   <description>Demo project for Spring Boot</description>
15
16   <parent>
17     <groupId>org.springframework.boot</groupId>
18     <artifactId>spring-boot-starter-parent</artifactId>
19     <version>2.7.17</version>
20     <relativePath/> <!-- lookup parent from repository -->
21   </parent>
22
23   <properties>
24     <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
25     <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncod
26ing>
27     <java.version>1.8</java.version>
28   </properties>
29
30   <dependencies>
31     <dependency>
32       <groupId>org.springframework.boot</groupId>
33       <artifactId>spring-boot-starter-web</artifactId>
34     </dependency>
35
36     <dependency>
37       <groupId>org.springframework.boot</groupId>
38       <artifactId>spring-boot-starter-test</artifactId>
39       <scope>test</scope>
40     </dependency>
41   </dependencies>
42
43   <build>
44     <plugins>
45       <plugin>
```

```
43         <groupId>org.springframework.boot</groupId>
44         <artifactId>spring-boot-maven-plugin</artifactId>
45     </plugin>
46 </plugins>
47 </build>
48
49 </project>
```

spring-boot-starter-parent

spring-boot-starter-parent指定了当前项目为一个Spring Boot项目，它提供了诸多的默认Maven依赖，具体可查看目录D:\m2\repository\org\springframework\boot\spring-boot-dependencies\1.5.9.RELEASE下的spring-boot-dependencies-1.5.9.RELEASE.pom文件，这里仅截取一小部分：


```

1 <properties>
2   ...
3   <spring-security.version>4.2.3.RELEASE</spring-security.version>
4   <spring-security-jwt.version>1.0.8.RELEASE</spring-security-jwt.version>
5   <spring-security-oauth.version>2.0.14.RELEASE</spring-security-oauth.ver
6   <spring-session.version>1.3.1.RELEASE</spring-session.version>
7   <spring-social.version>1.1.4.RELEASE</spring-social.version>
8   <spring-social-facebook.version>2.0.3.RELEASE</spring-social-facebook.ve
9   <spring-social-linkedin.version>1.0.2.RELEASE</spring-social-linkedin.ve
10  <spring-social-twitter.version>1.1.2.RELEASE</spring-social-twitter.vers
11  <spring-ws.version>2.4.2.RELEASE</spring-ws.version>
12  <sqlite-jdbc.version>3.15.1</sqlite-jdbc.version>
13  <statsd-client.version>3.1.0</statsd-client.version>
14  <sun-mail.version>${javax-mail.version}</sun-mail.version>
15  <thymeleaf.version>2.1.6.RELEASE</thymeleaf.version>
16  <thymeleaf-extras-springsecurity4.version>2.1.3.RELEASE</thymeleaf-extra
17  <thymeleaf-extras-conditionalcomments.version>2.1.2.RELEASE</thymeleaf-e
18  <thymeleaf-layout-dialect.version>1.4.0</thymeleaf-layout-dialect.versio
19  <thymeleaf-extras-data-attribute.version>1.3</thymeleaf-extras-data-attr
20  <thymeleaf-extras-java8time.version>2.1.0.RELEASE</thymeleaf-extras-java
21  <tomcat.version>8.5.23</tomcat.version>
22  ...
23 </properties>

```

需要说明的是，并非所有在 `<properties>` 标签中配置了版本号依赖都有被启用，其启用与否取决于您是否配置了相应的starter。比如tomcat这个依赖就是spring-boot-starter-web的传递性依赖（下面将会描述到）。

当然，我们可以手动改变这些依赖的版本。比如我想把thymeleaf的版本改为3.0.0.RELEASE，我们可以在pom.xml中进行如下配置：

```

1 <properties>
2   <thymeleaf.version>3.0.0.RELEASE</thymeleaf.version>
3 </properties>

```

spring-boot-starter-web

Spring Boot提供了许多开箱即用的依赖模块，这些模块都是以spring-boot-starter-XX命名的。比如要开启Spring Boot的web功能，只需要在pom.xml中配置spring-boot-starter-web即可：

```
XML |
1 <dependency>
2   <groupId>org.springframework.boot</groupId>
3   <artifactId>spring-boot-starter-web</artifactId>
4 </dependency>
```

因为其依赖于spring-boot-starter-parent，所以这里可以不用配置version。保存后Maven会自动帮我们下载spring-boot-starter-web模块所包含的jar文件。如果需要具体查看spring-boot-starter-web包含了哪些依赖，我们可以右键项目选择run as → Maven Build...，在Goals中输入命令 `dependency:tree`，然后点击run即可在eclipse控制台查看到如下信息：

```

1 ▾ [INFO] +- org.springframework.boot:spring-boot-starter-web:jar:1.5.9.RELEASE:compile
2 ▾ [INFO] | +- org.springframework.boot:spring-boot-starter:jar:1.5.9.RELEASE:compile
3 ▾ [INFO] | | +- org.springframework.boot:spring-boot:jar:1.5.9.RELEASE:compile
4 ▾ [INFO] | | +- org.springframework.boot:spring-boot-autoconfigure:jar:1.5.9.RELEASE:compile
5 ▾ [INFO] | | +- org.springframework.boot:spring-boot-starter-logging:jar:1.5.9.RELEASE:compile
6 ▾ [INFO] | | | +- ch.qos.logback:logback-classic:jar:1.1.11:compile
7 ▾ [INFO] | | | | \- ch.qos.logback:logback-core:jar:1.1.11:compile
8 ▾ [INFO] | | | +- org.slf4j:jcl-over-slf4j:jar:1.7.25:compile
9 ▾ [INFO] | | | +- org.slf4j:jul-to-slf4j:jar:1.7.25:compile
10 ▾ [INFO] | | | \- org.slf4j:log4j-over-slf4j:jar:1.7.25:compile
11 ▾ [INFO] | | \- org.yaml:snakeyaml:jar:1.17:runtime
12 ▾ [INFO] | +- org.springframework.boot:spring-boot-starter-tomcat:jar:1.5.9.RELEASE:compile
13 ▾ [INFO] | | +- org.apache.tomcat.embed:tomcat-embed-core:jar:8.5.23:compile
14 ▾ [INFO] | | | \- org.apache.tomcat:tomcat-annotations-api:jar:8.5.23:compile
15 ▾ [INFO] | | +- org.apache.tomcat.embed:tomcat-embed-el:jar:8.5.23:compile
16 ▾ [INFO] | | \- org.apache.tomcat.embed:tomcat-embed-websocket:jar:8.5.23:compile
17 ▾ [INFO] | +- org.hibernate:hibernate-validator:jar:5.3.6.Final:compile
18 ▾ [INFO] | | +- javax.validation:validation-api:jar:1.1.0.Final:compile
19 ▾ [INFO] | | +- org.jboss.logging:jboss-logging:jar:3.3.1.Final:compile
20 ▾ [INFO] | | \- com.fasterxml:classmate:jar:1.3.4:compile
21 ▾ [INFO] | +- com.fasterxml.jackson.core:jackson-databind:jar:2.8.10:compile
22 ▾ [INFO] | | +- com.fasterxml.jackson.core:jackson-annotations:jar:2.8.0:compile
23 ▾ [INFO] | | | \- com.fasterxml.jackson.core:jackson-core:jar:2.8.10:compile
24 ▾ [INFO] | +- org.springframework:spring-web:jar:4.3.13.RELEASE:compile
25 ▾ [INFO] | | +- org.springframework:spring-aop:jar:4.3.13.RELEASE:compile
26 ▾ [INFO] | | +- org.springframework:spring-beans:jar:4.3.13.RELEASE:compile
27 ▾ [INFO] | | | \- org.springframework:spring-context:jar:4.3.13.RELEASE:compile
28 ▾ [INFO] | | \- org.springframework:spring-webmvc:jar:4.3.13.RELEASE:compile
29 ▾ [INFO] | \- org.springframework:spring-expression:jar:4.3.13.RELEASE:compile

```

上述这些依赖都是隐式依赖于spring-boot-starter-web，我们也可以手动排除一些我们不需要的依赖。

比如spring-boot-starter-web默认集成了tomcat，假如我们想把它换为jetty，可以在pom.xml中spring-boot-starter-web下排除tomcat依赖，然后手动引入jetty依赖：

XML

```
1 <dependencies>
2   <dependency>
3     <groupId>org.springframework.boot</groupId>
4     <artifactId>spring-boot-starter-web</artifactId>
5     <exclusions>
6       <exclusion>
7         <groupId>org.springframework.boot</groupId>
8         <artifactId>spring-boot-starter-tomcat</artifactId>
9       </exclusion>
10    </exclusions>
11  </dependency>
12
13  <dependency>
14    <groupId>org.springframework.boot</groupId>
15    <artifactId>spring-boot-starter-jetty</artifactId>
16  </dependency>
17 </dependencies>
```

tips：依赖的坐标可以到上述的spring-boot-dependencies-1.5.9.RELEASE.pom文件里查找。再次运行 `dependency:tree`：

```

1 ▾ [INFO] +- org.springframework.boot:spring-boot-starter-web:jar:1.5.9.RELEASE:compile
2   ...
3 ▾ [INFO] +- org.springframework.boot:spring-boot-starter-jetty:jar:1.5.9.RELEASE:compile
4 ▾ [INFO] | +- org.eclipse.jetty:jetty-servlets:jar:9.4.7.v20170914:compile
5 ▾ [INFO] | | +- org.eclipse.jetty:jetty-continuation:jar:9.4.7.v20170914:compile
6 ▾ [INFO] | | +- org.eclipse.jetty:jetty-http:jar:9.4.7.v20170914:compile
7 ▾ [INFO] | | +- org.eclipse.jetty:jetty-util:jar:9.4.7.v20170914:compile
8 ▾ [INFO] | | \- org.eclipse.jetty:jetty-io:jar:9.4.7.v20170914:compile
9 ▾ [INFO] | +- org.eclipse.jetty:jetty-webapp:jar:9.4.7.v20170914:compile
10 ▾ [INFO] | | +- org.eclipse.jetty:jetty-xml:jar:9.4.7.v20170914:compile
11 ▾ [INFO] | | \- org.eclipse.jetty:jetty-servlet:jar:9.4.7.v20170914:compile
12 ▾ [INFO] | | \- org.eclipse.jetty:jetty-security:jar:9.4.7.v20170914:compile
13 ▾ [INFO] | | \- org.eclipse.jetty:jetty-server:jar:9.4.7.v20170914:compile
14 ▾ [INFO] | +- org.eclipse.jetty.websocket:websocket-server:jar:9.4.7.v20170914:compile
15 ▾ [INFO] | | +- org.eclipse.jetty.websocket:websocket-common:jar:9.4.7.v20170914:compile
16 ▾ [INFO] | | | \- org.eclipse.jetty.websocket:websocket-api:jar:9.4.7.v20170914:compile
17 ▾ [INFO] | | +- org.eclipse.jetty.websocket:websocket-client:jar:9.4.7.v20170914:compile
18 ▾ [INFO] | | | \- org.eclipse.jetty:jetty-client:jar:9.4.7.v20170914:compile
19 ▾ [INFO] | | \- org.eclipse.jetty.websocket:websocket-servlet:jar:9.4.7.v20170914:compile
20 ▾ [INFO] | | \- javax.servlet:javax.servlet-api:jar:3.1.0:compile
21 ▾ [INFO] | +- org.eclipse.jetty.websocket:javax-websocket-server-impl:jar:9.4.7.v20170914:compile
22 ▾ [INFO] | | +- org.eclipse.jetty:jetty-annotations:jar:9.4.7.v20170914:compile
23 ▾ [INFO] | | | +- org.eclipse.jetty:jetty-plus:jar:9.4.7.v20170914:compile
24 ▾ [INFO] | | | +- javax.annotation:javax.annotation-api:jar:1.2:compile
25 ▾ [INFO] | | | +- org.ow2.asm:asm:jar:5.1:compile
26 ▾ [INFO] | | | \- org.ow2.asm:asm-commons:jar:5.1:compile
27 ▾ [INFO] | | | \- org.ow2.asm:asm-tree:jar:5.1:compile
28 ▾ [INFO] | | +- org.eclipse.jetty.websocket:javax-websocket-client-impl:jar:9.4.7.v20170914:compile
29 ▾ [INFO] | | \- javax.websocket:javax.websocket-api:jar:1.0:compile

```

可看到tomcat已被替换为了jetty。

spring-boot-maven-plugin

spring-boot-maven-plugin为Spring Boot Maven插件，提供了：

1. 把项目打包成一个可执行的超级JAR（uber-JAR），包括把应用程序的所有依赖打入JAR文件内，并为JAR添加一个描述文件，其中的内容能让你用 `java -jar` 来运行应用程序。
2. 搜索 `public static void main()` 方法来标记为可运行类。

补充

@RestController

来源

org.springframework.web.bind.annotation.RestController

作用

- 标识该类是一个 **Spring MVC Controller**，并且返回的内容会直接写入 HTTP 响应体，而不是返回一个视图（如模板引擎页面）。
- 等同于 @Controller + @ResponseBody 的组合，适用于构建 RESTful Web 服务。

@SpringBootApplication

来源

org.springframework.boot.autoconfigure.SpringBootApplication

作用

- 用来标识一个主配置类，Spring Boot 会自动完成以下功能：
 - **@Configuration**：将该类标识为 Spring 的配置类，允许定义 @Bean 方法。

- **@EnableAutoConfiguration**: 启用 Spring Boot 的自动配置机制，自动加载相关的配置和 Bean。
- **@ComponentScan**: 扫描当前包及其子包中的组件（如 @Component、@Service、@Repository、@Controller 等）。

作用范围:

是 Spring Boot 应用的入口，通常用于主类上。

@RequestMapping()

来源

org.springframework.web.bind.annotation.RequestMapping

作用

- 用于定义 URL 映射关系。
- 这里将路径 "/" 映射到 index() 方法，当客户端访问根路径（如 http://localhost:8080/）时，会调用该方法。

默认行为

如果没有显式定义 HTTP 方法类型（如 GET、POST），则支持所有类型。

细节

也可以用于类上（类级别映射）和方法上（方法级别映射）。

整体流程说明

1. 启动类:

- @SpringBootApplication 告诉 Spring Boot 启动自动配置并扫描相关组件。
- 执行SpringApplication.run(DemoApplication.class, args)，启动嵌入式 Web 容器（如

Tomcat)，并启动应用。

2. REST Controller:

- @RestController 将 DemoApplication 标识为一个控制器类。
- 定义了一个映射到根路径 "/" 的方法 index()，该方法返回字符串 "hello spring boot"。

3. 返回数据:

由于 @RestController 的作用，index() 返回的字符串会被直接写入 HTTP 响应体并返回给客户端。