1.首先，建立一个maven工程，引入如下依赖

*<!-- Inherit defaults from Spring Boot -->*<**parent**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-parent</**artifactId**>  
 <**version**>2.0.6.RELEASE</**version**>  
</**parent**>  
  
*<!-- Add typical dependencies for a web application -->*<**dependencies**>  
 <**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-web</**artifactId**>  
 </**dependency**>  
</**dependencies**>

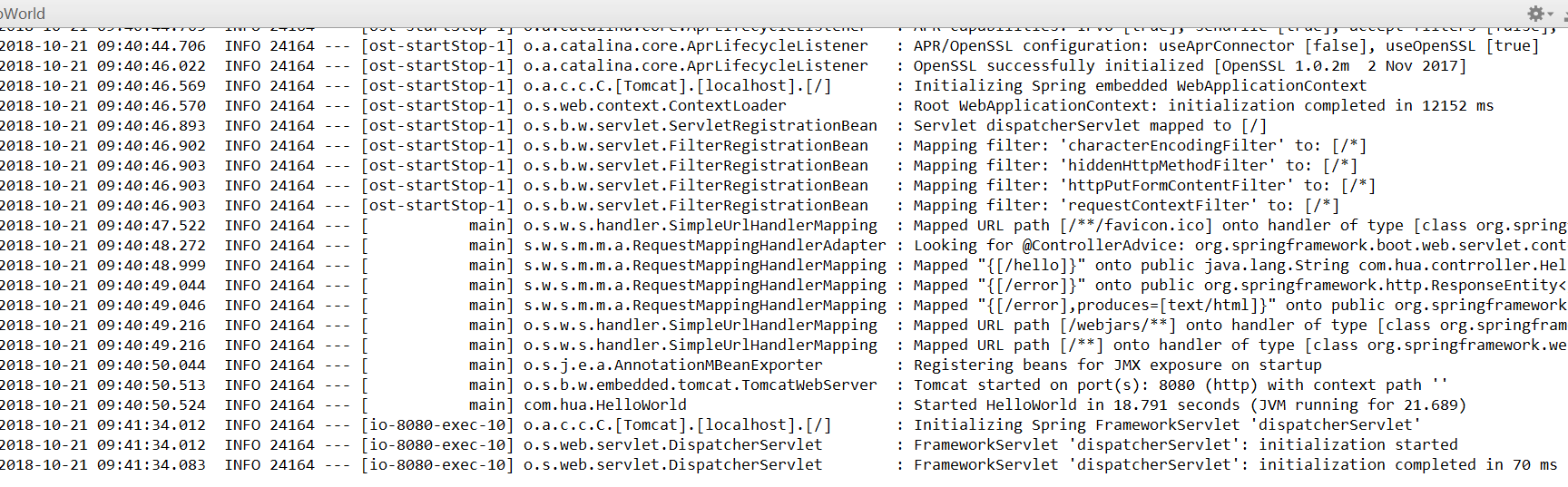
2.编写一个含有main方法的类，用@SpringBootApplication注解标识，然后main方法里面，使用SpringApplication.run()方法，传入main方法所在类和参数

**package** com.hua;  
  
**import** org.springframework.boot.SpringApplication;  
**import** org.springframework.boot.autoconfigure.SpringBootApplication;  
  
*/\*\*  
 \** ***@Author*** *lenovo  
 \** ***@Version*** *2018/10/21  
 \*/  
//@SpringBootApplication来标注一个主程序类，说明这是一个springboot应用*@SpringBootApplication  
**public class** HelloWorld {  
  
 **public static void** main(String[] args) {  
 *//spring应用启动起来* SpringApplication.*run*(HelloWorld.**class**, args);  
 }  
}

3.编写业务逻辑，此处编写一个接受/hello请求的controller，直接在页面上打印hello world

**package** com.hua.contrroller;  
  
**import** org.springframework.stereotype.Controller;  
**import** org.springframework.web.bind.annotation.RequestMapping;  
**import** org.springframework.web.bind.annotation.ResponseBody;  
  
*/\*\*  
 \** ***@Author*** *lenovo  
 \** ***@Version*** *2018/10/21  
 \*/*@Controller  
**public class** HelloController {  
  
 @ResponseBody  
 @RequestMapping(**"/hello"**)  
 **public** String hello(){  
 **return "hello world!"**;  
 }  
  
}

4.直接运行该main方法测试



5.部署到服务器

首先加入一个打包成jar的springboot的插件

在pom.xml引入如下代码

<build>

<plugins>

<plugin>

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

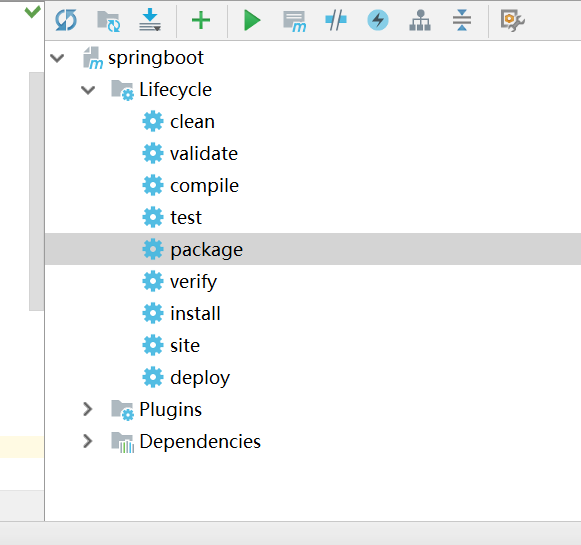
<artifactId>spring-boot-maven-plugin</artifactId>

</plugin>

</plugins>

</build>

然后用maven的package来打包



再然后将用java -jar的命令来运行此jar包



探究helloworld

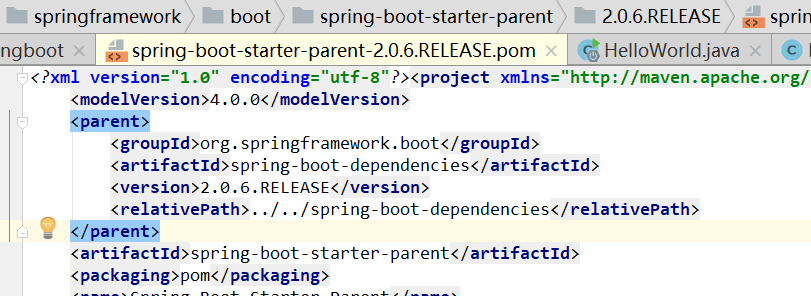
1.pom.xml文件

刚刚在加入了两个小配置

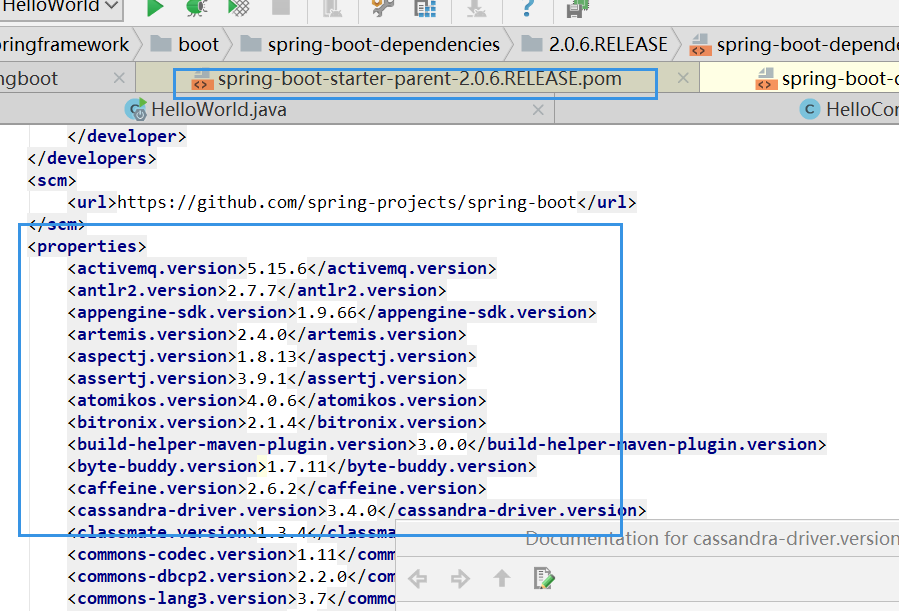
1）第一个，引入了一个父项目

*<!-- Inherit defaults from Spring Boot -->*<**parent**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-parent</**artifactId**>  
 <**version**>2.0.6.RELEASE</**version**>  
</**parent**>

点进去他的父项目发现这个还引入了一个父项目



然后再点进去，发现其中有很多属性，我们的依赖就是从这里面找的，这里来管理springboot里面的所有依赖



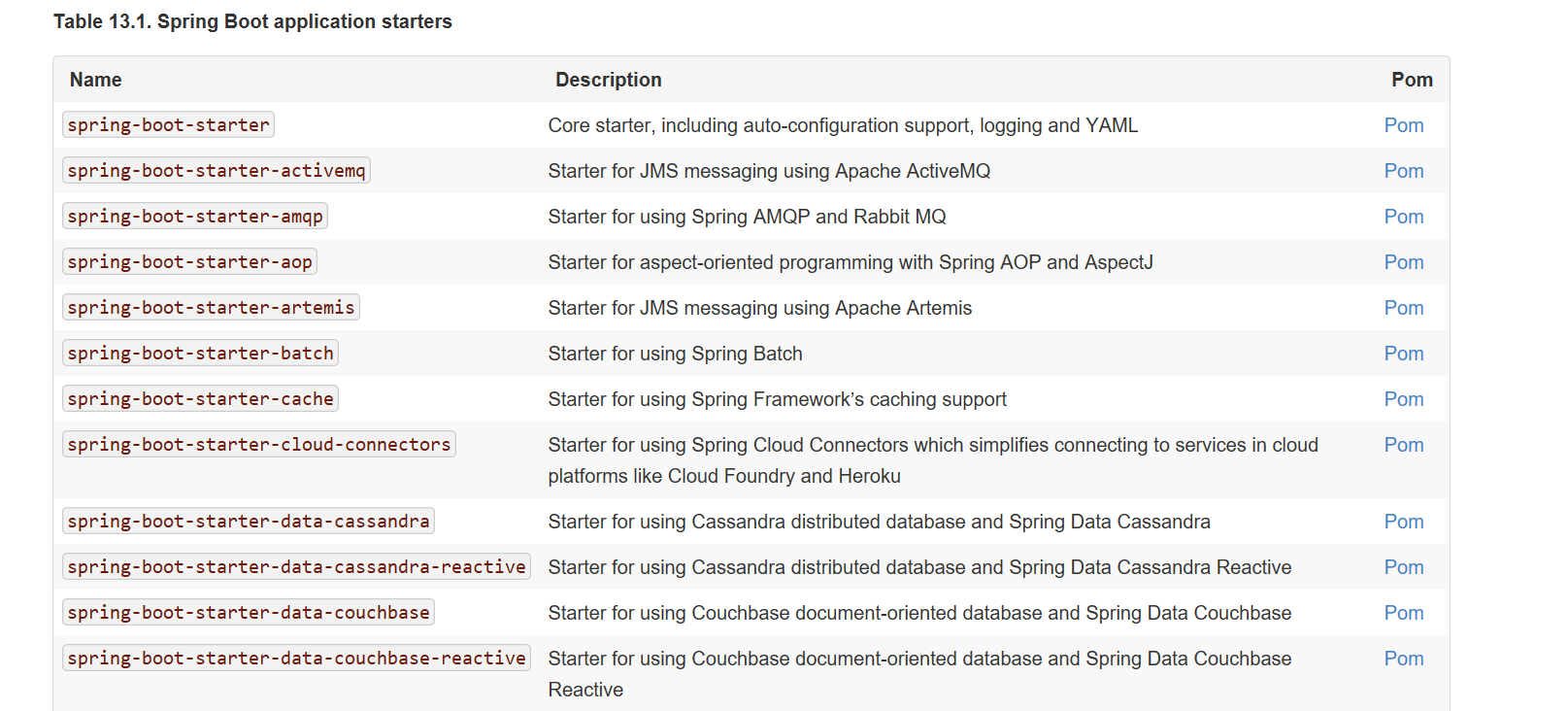
Springboot版本仲裁中心

所以说，导入依赖一般不用写版本，如果引入的依赖是这里面没有的需要写版本号

2）启动器，第二个小配置

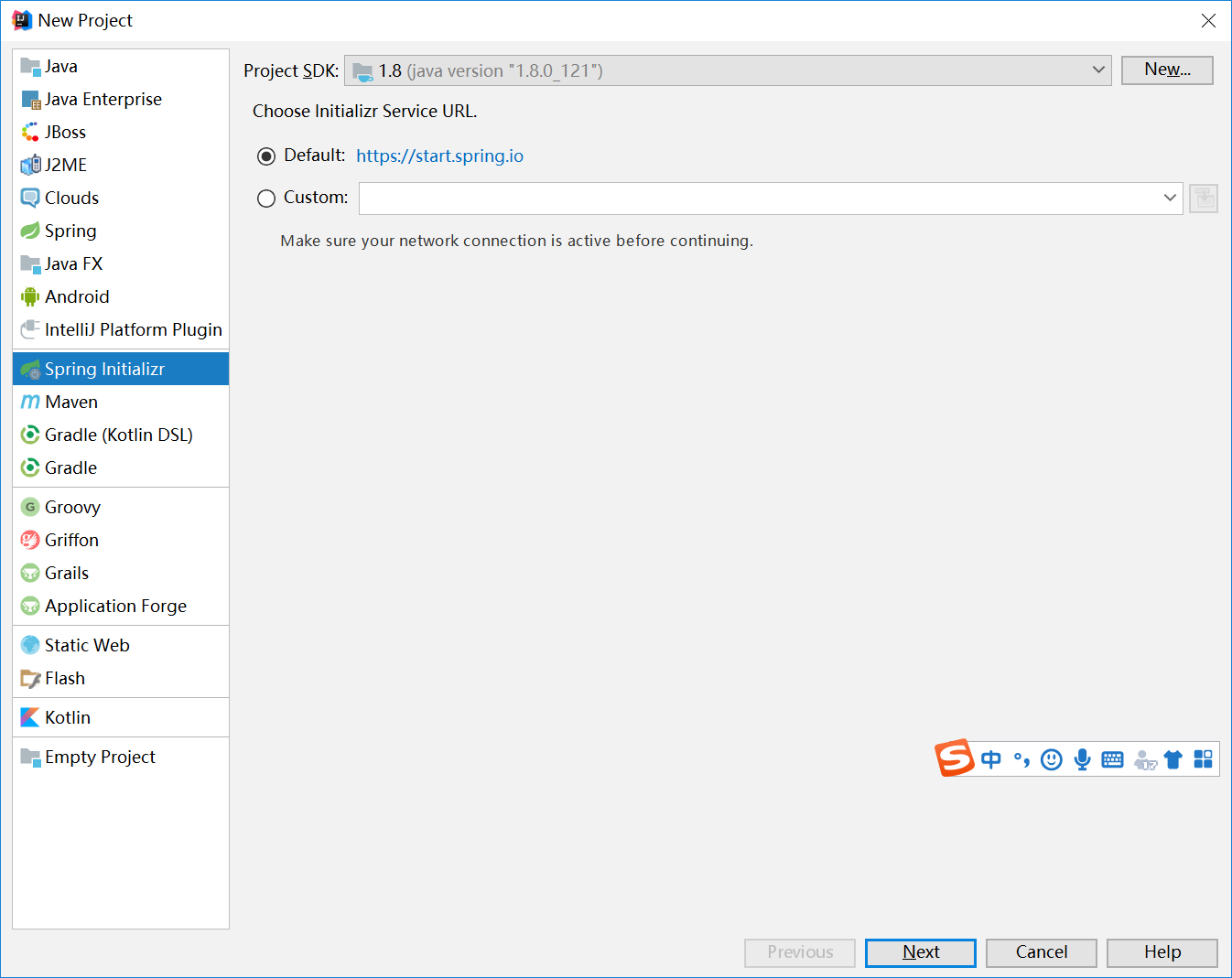
*<!-- Add typical dependencies for a web application -->*<**dependencies**>  
 <**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-web</**artifactId**>  
 </**dependency**>  
</**dependencies**>

这个东西是spring-boot-starter中的其中之一，因为我们要做web项目，所以说导入的web的启动器，这里面为我们引入了一些依赖信息



官方文档给了一窝窝的启动器，以后用到哪个功能，就导入相应功能的启动器即可

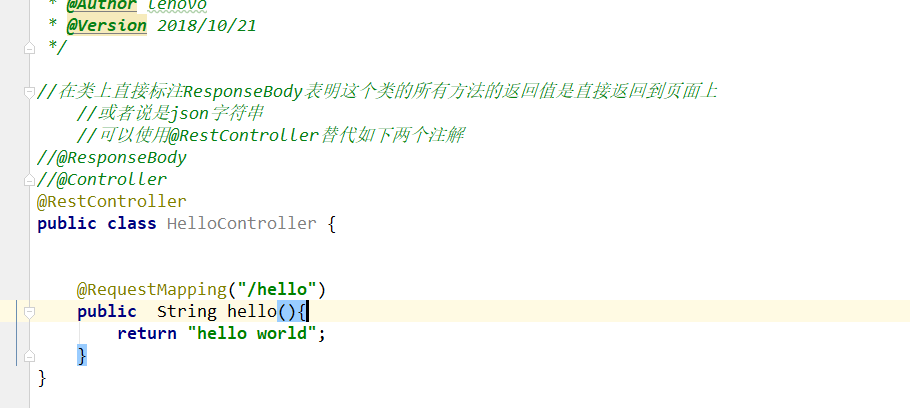
使用快速建立项目的想到建立项目



就直接创建了一个项目，，，然后直接写业务逻辑就行了，真他娘的方便啊

它帮我们引入了上面自己引入的东西，顺便帮我们引入了一个test，即springboot的单元测试启动器

编写业务逻辑，注意下面是个人没学习过得，，注意些



1.给我们生成了主程序

2.还给我们生成了resources文件夹

Static:保存所有的静态资源

Templates：保存所有的模板页面：（Spring Boot默认jar包使用嵌入式的Tomcat，默认不支持JSP）；可以使用模板引擎（freemarker，thymeleaf）

Application.properties: SpringBoot应用的配置文件，可以修改一些默认设置

Springboot

2.yaml配置文件简介