Maven轻松搭建Spring Boot项目 不用去官网生成 非常精简易懂!

关于框架的搭建是比较头疼的，因为一般首先要找到回忆，然后构建、打包、调试、测试等。网上很多对于Spring Boot项目的介绍比较繁杂，初学者可能很容易摸不着头脑。

其实Spring Boot出现的目的就是尽量的简化配置，一听到公司要用这个你第一反应应该是舒适，因为它的确要比Spring MVC简洁很多，比三大框架的搭建也要简洁很多，简洁到你几乎立马就可以开始写业务代码。到底如何简洁，我们来看看。

1. 首先创建一个maven项目，创建那种最简单的即可，不需要示例文件。

groupId的填写格式一般为com.{公司名}.{项目名}

（*请根据公司的实际情况填写，如果是个人就更随意了，org.{你的名字}.{xxoo}*）

artifactId的填写很随意，就是模块名，比如三层架构的项目，有这些模块：dao, business, web

模块名之间要分割的话用-（中划线）不要用小数点哦

版本就默认的，一般开发的版本肯定是：0.0.1-SNAPSHOT

2. 配置POM.xml文件

直接贴了，请看下面的注释部分：

*<?***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**>org.cd.dp</**groupId**>  
 <**artifactId**>design-patterns</**artifactId**>  
 <**version**>0.0.1-SNAPSHOT</**version**>  
  
 *<!--这是spring boot的父依赖项目，只要加入这个，就可以省去很多配置，比如maven打包需要的那些插件、构建的过程等。另外，加上这个后，spring boot的扩展很容易，必须要再加上版本号了，比如后面的spring-boot-starter-web可以不用指定版本号，原理很简单，spring-boot-starter-web-parent里面有spring boot相关的dependency management  
 -->* <**parent**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-parent</**artifactId**>  
 <**version**>1.5.4.RELEASE</**version**>  
 <**relativePath** />  
 </**parent**>  
  
 <**properties**>  
 <**project.build.sourceEncoding**>UTF-8</**project.build.sourceEncoding**>  
 <**project.reporting.outputEncoding**>UTF-8</**project.reporting.outputEncoding**>  
 <**java.version**>1.8</**java.version**>  
 </**properties**>  
  
 *<!--上面已经加过spring boot parent的依赖并指定版本号了，这里加入spring-boot-maven-plugin依赖即可（不需要填写版本号）。为什么要加这个插件呢？加入这个插件后可以将所有必要的dependency打入到jar包中，打包是不会包含dependency  
 -->* <**build**>  
 <**plugins**>  
 <**plugin**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-maven-plugin</**artifactId**>  
 </**plugin**>  
 </**plugins**>  
 </**build**>  
 <**name**>design-patterns</**name**>  
  
 *<!--加入spring boot项目的web模块依赖，别小瞧这一句哦，加上这个之后几乎就包含了构建一个web项目需要的所有依赖，比如Spring MVC的相关依赖，Spring core, context, aop... 还有log相关的依赖，jackson的依赖以及一些基础的工具包  
 -->* <**dependencies**>  
 <**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-web</**artifactId**>  
 </**dependency**>  
 </**dependencies**>  
  
</**project**>

3. 新建启动类。XXXApplication.java

在src/main/java下新建一个启动类（包名你自己定义），上面加入SpringBoot的注解@SpringBootApplication

我还在这里面加了一个测试接口，一会启动项目可以看看是否启动成功。

@SpringBootApplication  
@RestController  
**public class** StartApplication {  
  
 **public static void** main(String[] args) {  
 **new** SpringApplicationBuilder(StartApplication.**class**).run(args);  
 }  
  
 @RequestMapping(**"/test"**)  
 **public** String test() {  
 System.***out***.println(**"Test Success"**);  
 **return "Test Success"**;  
 }  
  
}

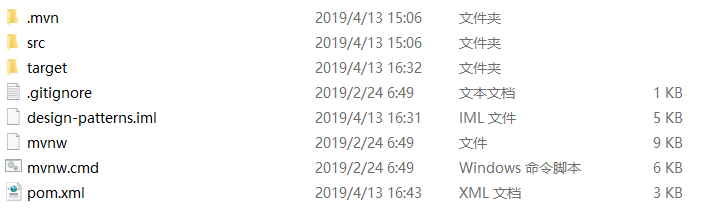
4. application.properties 或者 application.yml文件：

在java/main/resource下面新建一个配置文件，比如application.properties 或者 application.yml

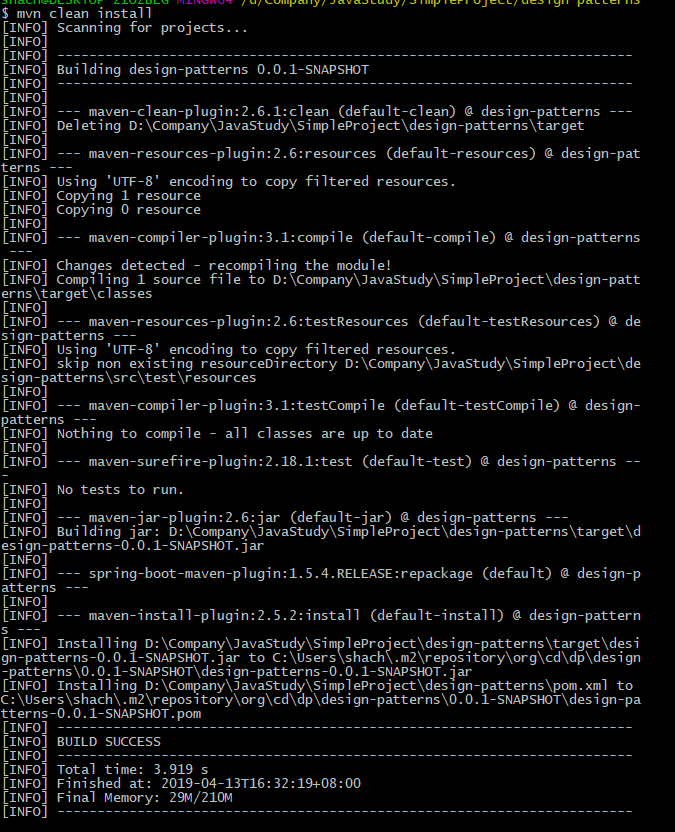
作用是一样的哈，在里面可以加入一些配置了。比如加入端口号配置：

**server.port**=**8081**

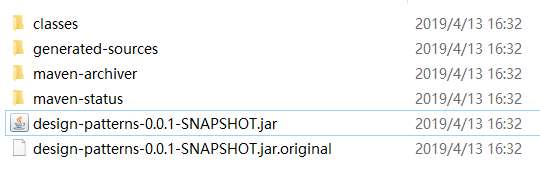
好了，一个Spring boot项目就搭建完毕了，可以去项目根目录下打包试试：



运行命令：mvn clean install



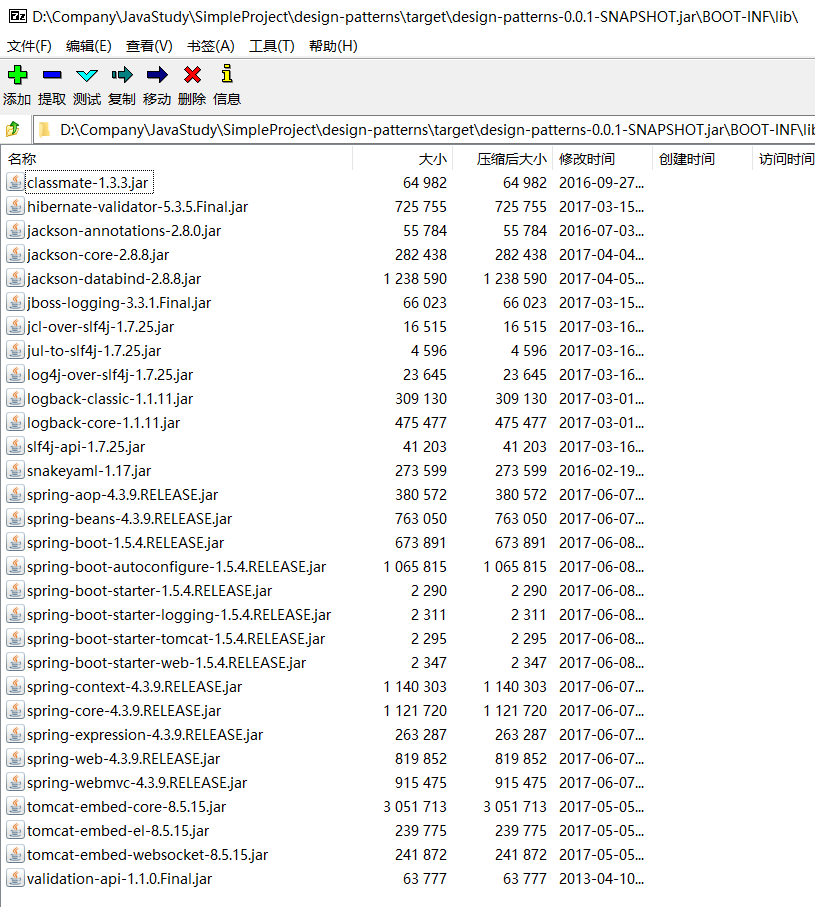
然后去target目录看看是否有一个jar包产生了，打开这个jar包，看看依赖是不是都打进去了：



OK，依赖确实都进去了，惊人吧 这么多！我们只是加了一个spring-boot-starter-web.

没错，这就是starter-web包含的所有依赖。

除了Spring 自己的（web, webmvc, core, context, beans, aop, express）还包括了一些日志第三方依赖，字段验证相关的依赖，JSON处理的依赖，甚至Tomcat服务器都有了。



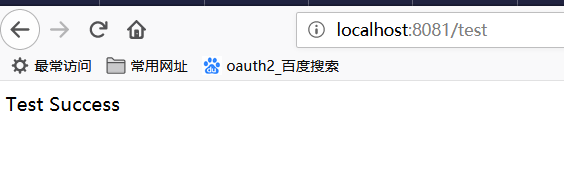
既然啥都有了，那还说什么呢，直接启动吧！

运行命令：java -jar {包名}



启动成功，还有个好看的logo呢！

在浏览器输入：<http://localhost:8081/test> 看看刚才写的那个测试接口是不是可以访问了。



欧耶！over