[一、快速创建：](#4162-1549962167747)

[二、idea 下的启动方式：](#5690-1549957515443)

[1、启动类启动项目](#4114-1549957523515)

[2、插件spring-boot:run启动](#1061-1549957549920)

[3、打成war包，外部tomcat启动](#8653-1549957634228)

[4、idea tomcat 启动](#6841-1549961277075)

[三、如何做单元测试](#4055-1549962208339)

[四、开发环境下的调试](#8231-1549962569187)

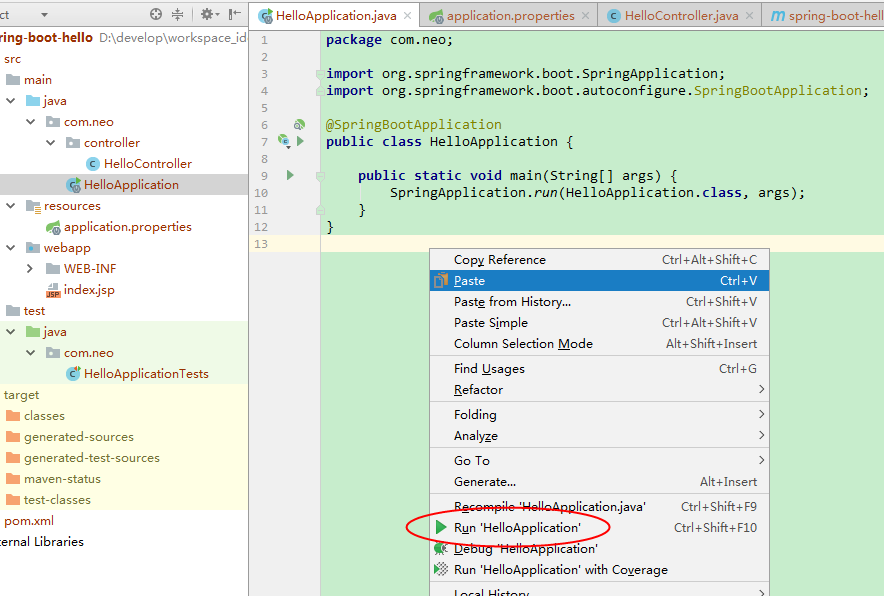
**一、快速创建：**

<http://www.ityouknow.com/springboot/2016/01/06/spring-boot-quick-start.html>

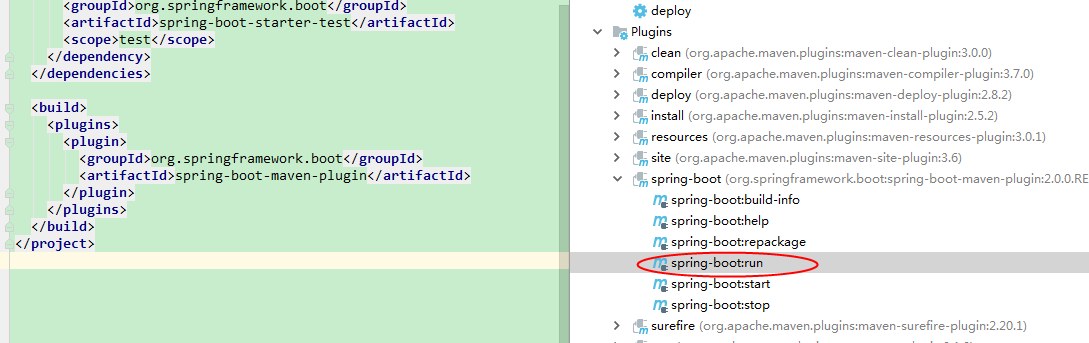
**二、idea 下的启动方式：**

<https://www.cnblogs.com/sxdcgaq8080/p/7727249.html>

**1、启动类启动项目**



**2、插件spring-boot:run启动**

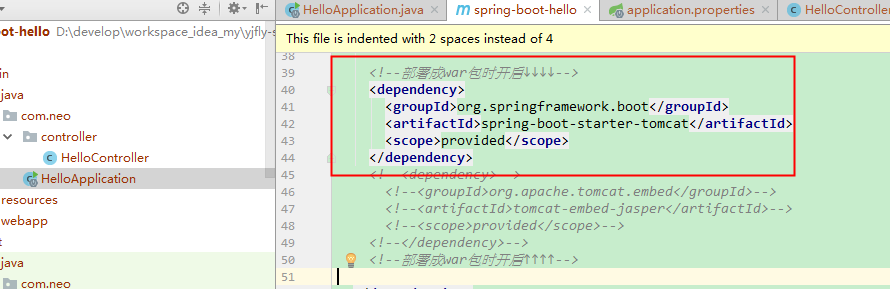


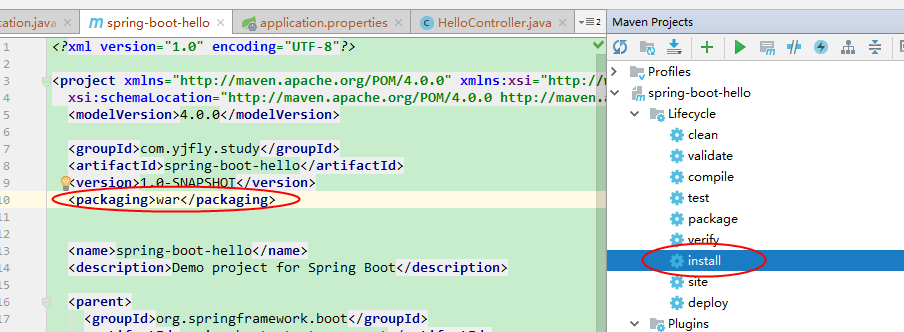
**3、打成war包，外部tomcat启动**

[springboot发布外部tomcat步骤和异常处理 java.lang.NoClassDefFoundError javax el ELManager - BenJi的博客 - CSDN博客](note://wcp154996101579095)

外部容器部署的话，就不能依赖于Application的main函数了，而是要以类似于web.xml文件配置的方式来启动Spring应用上下文，此时我们需要在启动类中继承SpringBootServletInitializer并实现configure方法

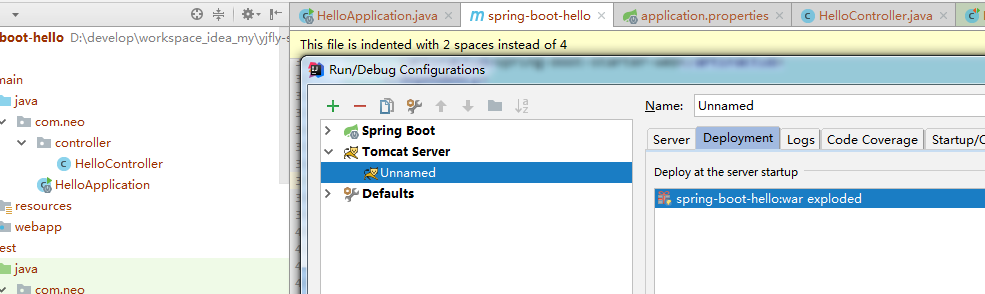






**异常处理:java.lang.NoClassDefFoundError: javax/el/ELManager (包冲突导致)**

**4、idea tomcat 启动**



**三、如何做单元测试**

打开的src/test/下的测试入口，编写简单的http请求来测试；使用mockmvc进行，利用MockMvcResultHandlers.print()打印出执行结果。

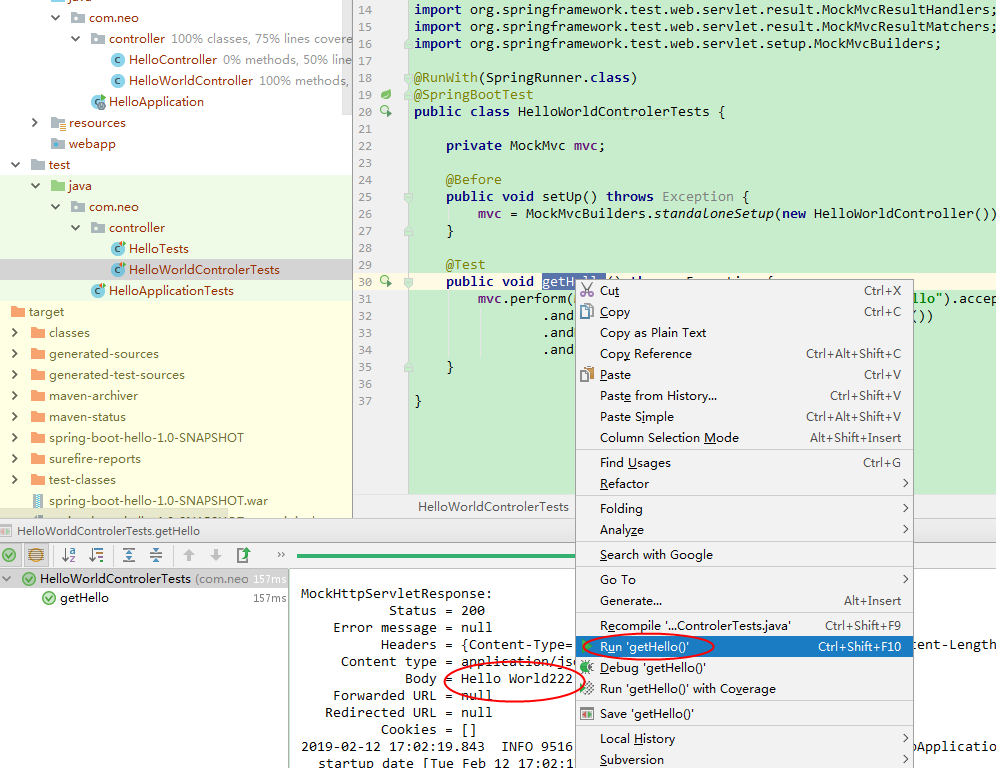
**@RunWith(**SpringRunner**.**class**)**
**@SpringBootTest**
**public** **class** **HelloTests** **{**
**private** MockMvc mvc**;**
**@Before**
**public** **void** **setUp()** **throws** Exception **{**
mvc **=** MockMvcBuilders**.**standaloneSetup**(new** HelloWorldController**()).**build**();**
**}**
**@Test**
**public** **void** **getHello()** **throws** Exception **{**
mvc**.**perform**(**MockMvcRequestBuilders**.**get**(**"/hello"**).**accept**(**MediaType**.**APPLICATION\_JSON**))**
**.**andExpect**(**status**().**isOk**())**
**.**andExpect**(**content**().**string**(**equalTo**(**"Hello World"**)));**

mvc.perform(MockMvcRequestBuilders.get("/hello").accept(MediaType.APPLICATION\_JSON))

.andExpect(MockMvcResultMatchers.status().isOk())

.andDo(MockMvcResultHandlers.print())

.andReturn();
**}**
**}**



**四、开发环境下的调试**

idea的 tomcat下

