<http://blog.csdn.net/yingxiake/article/details/51288727>

版权声明：本文为博主原创文章，未经博主允许不得转载。

[**spring**](http://lib.csdn.net/base/javaee) boot 在springmvc的视图解析器方面就默认集成了ContentNegotiatingViewResolver和BeanNameViewResolver，在视图引擎上就已经集成自动配置的模版引擎，如下：   
1. FreeMarker   
2. Groovy   
3. Thymeleaf   
4. Velocity (deprecated in 1.4)   
6. Mustache

JSP技术spring boot 官方是不推荐的，原因有三：   
1. 在tomcat上，jsp不能在嵌套的tomcat容器解析即不能在打包成可执行的jar的情况下解析   
2. Jetty 嵌套的容器不支持jsp   
3. Undertow

而其他的模版引擎spring boot 都支持，并默认会到classpath的templates里面查找模版引擎，这里假如我们使用freemarker模版引擎

1.现在pom.xml启动freemarker模版引擎视图

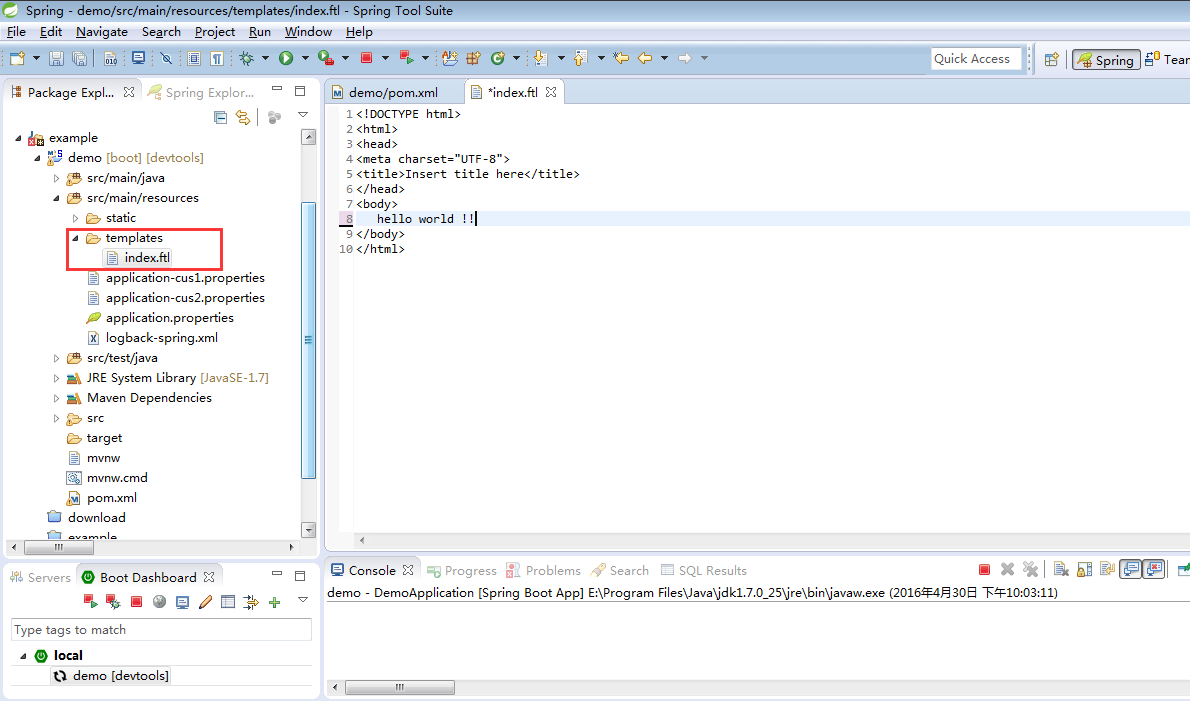
<dependency>

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

<artifactId>spring-boot-starter-freemarker</artifactId>

</dependency>

2.定义一个模版后缀是ftp,注意是在classpath的templates目录下



3.在controller上返回视图路径

@RestController

@RequestMapping("/task")

public class TaskController {

@RequestMapping("/mvc1")

public ModelAndView mvc1() {

return new ModelAndView("index");

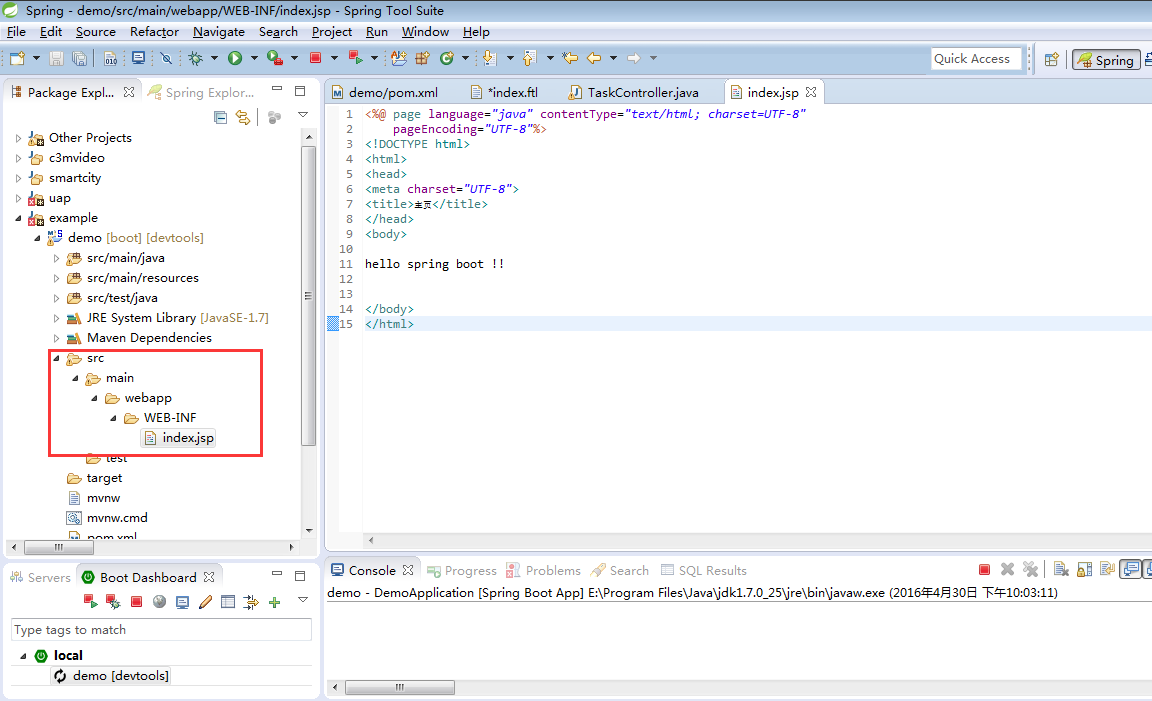
}

}

@RestController默认就会在每个方法上加上@Responsebody,方法返回值会直接被httpmessageconverter转化，如果想直接返回视图，需要直接指定modelAndView。

虽然，jsp技术，spring boot 官方不推荐，但考虑到是常用的技术，这里也来集成下jsp技术

1.首先，需要在你项目上加上webapp标准的web项目文件夹



2.在配置文件上配置，jsp的前缀和后缀

spring.mvc.view.prefix=/WEB-INF/

spring.mvc.view.suffix=.jsp

3.然后直接在controller中返回视图

@RestController

@RequestMapping("/task")

public class TaskController {

@RequestMapping("/mvc1")

public ModelAndView mvc1() {

return new ModelAndView("index");

}

}

这里要注意，只能是打成war包在非嵌套的tomcat容器才能看到效果，直接在嵌套的tomcat容器是看不到效果的，因为不支持，例如在IDE直接右键run main函数或者打成可执行的jar包都不行。

例外，如果出现freemarker模版引擎和jsp技术同时存在的话，springmvc会根据解析器的优先级来返回具体的视图，默认，FreeMarkerViewResolver的优先级大于InternalResourceViewResolver的优先级，所以同时存在的话，会返回freemarker视图