# SpringBoot常用技术整个

## SpringBoot构造并且返回一个json对象

*/\*\*  
 \* 该类必须在com.imook包下，否则访问不成功  
 \*/* **@RestController** *//@Controller+@ResponseBody  
 // @Controller* **@RequestMapping**(**"/user"**)  
**public class** UserController {  
  
 **@RequestMapping**(**"/getUser"**)  
 *// @ResponseBody//代表返回的是JS字符串，或者是js对象* **public** User hello(){  
 User u=**new** User();  
 u.setName(**"torey"**);  
 u.setAge(30);  
 u.setBirthday(**new** Date());  
 u.setPassword(**"123456"**);  
 **return** u;  
 }

}

## Jackson的基本演绎法

1 不返回某些属性数据

**@JsonIgnore***//前端不返回该数据，忽略***private** String **password**;

**@JsonIgnore***//前端不返回该数据，忽略***private** String **password**;  
**@JsonFormat**(pattern=**"yyyy-MM-dd hh:mm:ss a"**,locale=**"zh"**,timezone=**"GMT+8"**)*// 格式***private** Date **birthday**;  
**@JsonInclude**(**JsonInclude**.Include.***NON\_NULL***)*//如果为null，则不返回*

## SpringBoot热部署

### Spring Tool Suite热部署

SpringBoot使用devtools进行热部署

Pom.xml 文件

*<!--热部署-->  
<!--devtools可以实现页面热部署（既页面修改后会立即生效，  
这个可以直接在application.properties文件中配置spring.thymeleaf.cache=false来实现-->  
<!--实现类文件热部署（类文件修改后不会立即生效），实现对属性文件的热部署-->  
<!--既devtools会监听classpath下的文件变动，并且会立即重启应用（发生在保存时机），  
注意：因为其采用的虚拟机制，该项重启是很快的-->  
<!--(1)base clasloader(Base类加载器）：加载不改变的class，例如：第三方提供的jar包。-->  
<!--(2)restart classloader(Restart类加载器）：加载正在开发的Class.-->  
<!--为什么重启很快，因为重启的时候只是加载了在开发的Class,没有重新加载第三方的jar包-->*<**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-devtools</**artifactId**>  
 *<!--optional=true,依赖不会传递，该项目依赖devtools；之后依赖  
 boot项目的项目如果想要使用devtools，需要重新引入-->* <**optional**>true</**optional**>  
</**dependency**>

Application.properties文件

*#关闭缓存, 即时刷新  
#spring.freemarker.cache=false  
#spring.thymeleaf.cache=true  
  
#热部署生效  
#spring.devtools.restart.enabled=true  
#设置重启的目录,添加那个目录的文件需要restart  
#spring.devtools.restart.additional-paths=src/main/java  
# 为mybatis设置，生产环境可删除  
#restart.include.mapper=/mapper-[\\w-\\.]+jar  
#restart.include.pagehelper=/pagehelper-[\\w-\\.]+jar  
#排除那个目录的文件不需要restart  
#spring.devtools.restart.exclude=static/\*\*,public/\*\*  
#classpath目录下的WEB-INF文件夹内容修改不重启  
#spring.devtools.restart.exclude=WEB-INF/\*\**

## Intellij IDEA实现热部署

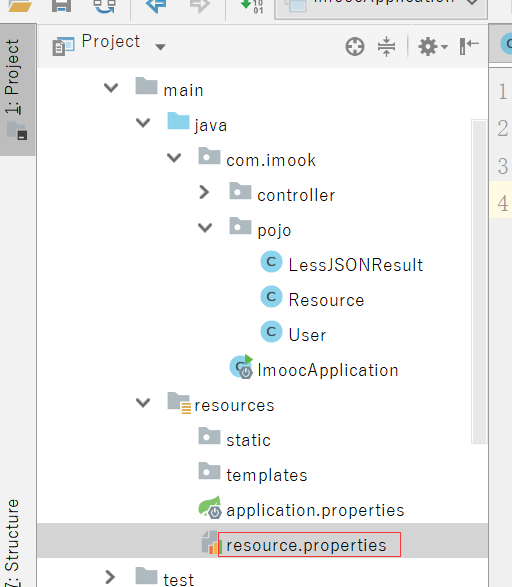
Pom.xml 文件加入：

<**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-devtools</**artifactId**>  
 *<!--optional=true,依赖不会传递，该项目依赖devtools；之后依赖  
 boot项目的项目如果想要使用devtools，需要重新引入-->* <**optional**>true</**optional**>  
</**dependency**>

快捷键：Ctrl+F9

## Springboot资源属性配置

一



**com.imooc.opensource.name**=**imooc**

**com.imooc.opensource.website**=**www.imooc.com**

**com.imooc.opensource.language**=**java**

二 pom.xml文件

*<!-- 用于将资源文件映射到实体类 -->*

<**dependency**>

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

<**artifactId**>spring-boot-configuration-processor</**artifactId**>

<**optional**>true</**optional**>

</**dependency**>

三 映射类

**package** com.imook.pojo;

**import** org.springframework.boot.context.properties.**ConfigurationProperties**;

**import** org.springframework.context.annotation.**Configuration**;

**import** org.springframework.context.annotation.**PropertySource**;

**@Configuration**

**@ConfigurationProperties**(prefix = **"com.imooc.opensource"**)

**@PropertySource**(value = **"classpath:resource.properties"**)

**public class** Resource {

**private** String **name**;

**private** String **website**;

**private** String **language**;

**public** String getName() {

**return name**;

}

**public void** setName(String name) {

**this**.**name** = name;

}

**public** String getWebsite() {

**return website**;

}

**public void** setWebsite(String website) {

**this**.**website** = website;

}

**public** String getLanguage() {

**return language**;

}

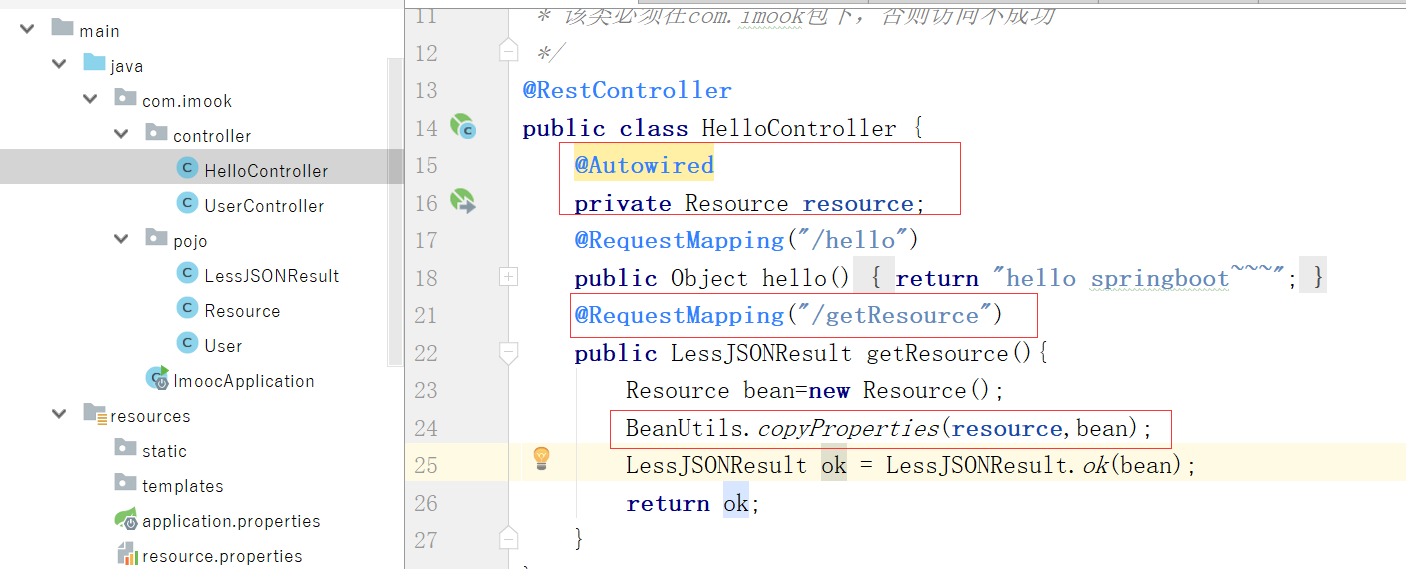
**public void** setLanguage(String language) {

**this**.**language** = language;

}

}

四、



**@RestController**

**public class** HelloController {

**@Autowired**

**private** Resource **resource**;

**@RequestMapping**(**"/hello"**)

**public** Object hello(){

**return "hello springboot~~~"**;

}

**@RequestMapping**(**"/getResource"**)

**public** LessJSONResult getResource(){

Resource bean=**new** Resource();

BeanUtils.*copyProperties*(**resource**,bean);

LessJSONResult ok = LessJSONResult.*ok*(bean);

**return** ok;

}

}

## 资源文件中配置tomcat相关属性

## SpringBoot整合freemarket

Dd

## SpringBoot整合thymeleaf(主推thymeleaf)