Springboot：整个spring技术栈的大整合。

微服务：一个应用应该是一组小型服务，可以通过HTTP的方式进行互通。每一个功能元素最终都是一个可独立替换和独立升级的软件单元。

@SpringBootApplication 标注在某个类上说明这个类是Springboot的主配置类

@Configuration 标注在某个类上，表示这是一个Springboot的配置类

@EnableAutoConfiguration 开启自动配置功能

@RestController

public class HelloController {

@RequestMapping(value="hello")

public String hello(){

return "Hello Word 之 Spring boot";

}

}

SpringMVC常用注解@Controller,@Service,@repository,@Component

@Controller 用于标记在一个类上，使用它标记的类就是一个SpringMVC Controller 对象。分发处理器将会扫描使用了该注解的类的方法。通俗来说，被Controller标记的类就是一个控制器，这个类中的方法，就是相应的动作。  
@RequestMapping是一个用来处理请求地址映射的注解，可用于类或方法上。用于类上，表示类中的所有响应请求的方法都是以该地址作为父路径。比如图一中，跳转到登录页面的路径就是localhost:8080/xxx-war/user/toLogin



配置文件：application.properties和application.yml

YAML：以数据为中心，更适合做配置文件

YAML基本语法：

K:(空格) V :表示一对键值对

以空格缩进来控制层级关系

Serve:

Port: 8081

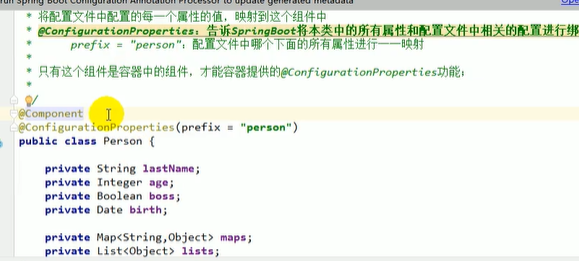
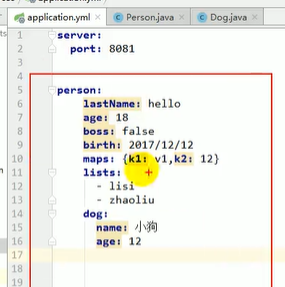
Path: /hello

属性和值也是大小写敏感





实例：



将配置文件中配置的每一个属性的值映射到组件，利用@ConfigurationProperties

需要@Component加入容器中

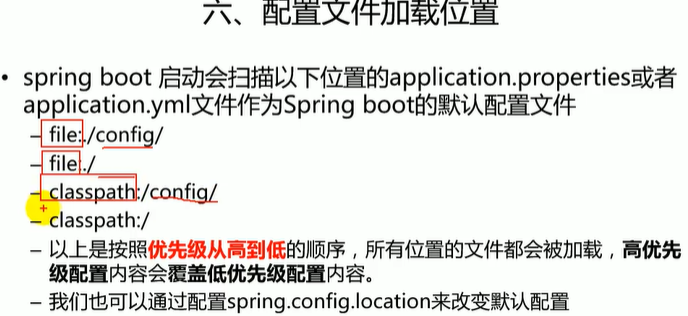


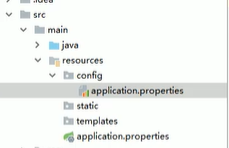




Profile







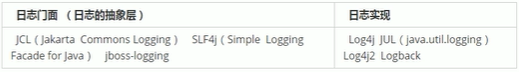


**自动配置原理**

HEEPencoding：



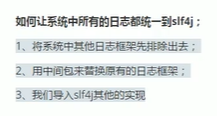
日志：



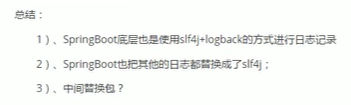
Springboot选用SLF4J和logback

SLF4J使用：



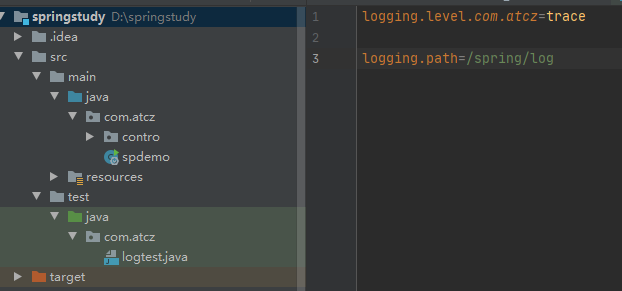


Spring中日志总结



1. 如果我们要引入其他框架，一定要把这个框架的默认日志依赖移除掉
2. Spring框架使用的是**commons-logging**；springboot能自动适配所有的日志，而且底层使用**slf4j+logback**的方式记录日志，引入其他框架的时候，只需要把这个框架依赖的日志框架排除掉。

日志文件 路径设置





@ResponseBody：将java对象转为json格式的数据。将controller的方法返回的对象通过适当的转换器转换为指定的格式之后，写入到response对象的body区，通常用来返回JSON数据或者是XML数据。