## 一

新建一个maven工程

Pom.xml

|  |
| --- |
| <**parent**>  <**groupId**>org.springframework.boot</**groupId**>  <**artifactId**>spring-boot-starter-parent</**artifactId**>  <**version**>1.5.3.RELEASE</**version**> </**parent**>  <**properties**>  <**project.build.sourceEncoding**>UTF-8</**project.build.sourceEncoding**>  <**java.version**>1.8</**java.version**> </**properties**>  <**dependencies**>   <**dependency**>  <**groupId**>org.springframework.boot</**groupId**>  <**artifactId**>spring-boot-starter-web</**artifactId**>  </**dependency**> </**dependencies**>  <**build**>  <**plugins**>  <**plugin**>  <**groupId**>org.springframework.boot</**groupId**>  <**artifactId**>spring-boot-maven-plugin</**artifactId**>  </**plugin**>  </**plugins**> </**build**> |

新建启动类Application.java

|  |
| --- |
| **package** com.example.spring;  **import** org.springframework.boot.SpringApplication; **import** org.springframework.boot.autoconfigure.SpringBootApplication; **import** org.springframework.web.bind.annotation.PathVariable; **import** org.springframework.web.bind.annotation.RequestMapping; **import** org.springframework.web.bind.annotation.RestController;  */\*\*  \* Created by ASUS on 2017/5/23.  \*/* @SpringBootApplication @RestController **public class** Application {   @RequestMapping(**"/{name}"**)  **public** String test1(@PathVariable(**"name"**) String name){  System.***out***.println(name);  **return "Hello "** + name;  }   **public static void** main(String[] args) {  SpringApplication.*run*(Application.**class**, args);  } } |

启动main方法，浏览器地址localhost:8080/potter，控制台打印potter，并且浏览器显示Hello potter.

## 二 listener

实现监听步骤：

1.监听类实现ApplicationListener接口

2.将监听类添加到SpringApplication实例

ApplicationStartingEvent

ApplicationEnvironmentPreparedEvent

ApplicationPreparedEvent

ApplicationReadyEvent

ApplicationFailedEvent

|  |
| --- |
| **package** com.example.spring.listener;  **import** org.slf4j.Logger; **import** org.slf4j.LoggerFactory; **import** org.springframework.boot.SpringApplication; **import** org.springframework.boot.context.event.ApplicationStartingEvent; **import** org.springframework.context.ApplicationListener;  */\*\*  \* spring-boot 启动前  \* Created by ASUS on 2017/5/23.  \*/* **public class** MyApplicationStartingEventListener **implements** ApplicationListener<ApplicationStartingEvent> {  **private** Logger **logger** = LoggerFactory.*getLogger*(MyApplicationStartingEventListener.**class**);   @Override  **public void** onApplicationEvent(ApplicationStartingEvent applicationStartingEvent) {  SpringApplication springApplication = applicationStartingEvent.getSpringApplication();  springApplication.setHeadless(**false**);  System.***out***.println(**"启动前111111111111"**);  **logger**.info(**"1111111111111111111"**);  } } |

其他同理

启动类中添加

|  |
| --- |
| **public static void** main(String[] args) {  SpringApplication springApplication = **new** SpringApplication(Application.**class**);  springApplication.addListeners(**new** MyApplicationStartingEventListener());  springApplication.addListeners(**new** MyApplicationEnvironmentPreparedEventListener());  springApplication.addListeners(**new** MyApplicationFailedEventListener());  springApplication.addListeners(**new** MyApplicationPreparedEventListener());  springApplication.addListeners(**new** MyApplicationReadyEventListener());  springApplication.run(args); } |

启动后log

|  |
| --- |
| 启动前111111111111  spring boot 对应Enviroment已经准备完毕，但此时上下文context还没有创建。  . \_\_\_\_ \_ \_\_ \_ \_  /\\ / \_\_\_'\_ \_\_ \_ \_(\_)\_ \_\_ \_\_ \_ \ \ \ \  ( ( )\\_\_\_ | '\_ | '\_| | '\_ \/ \_` | \ \ \ \  \\/ \_\_\_)| |\_)| | | | | || (\_| | ) ) ) )  ' |\_\_\_\_| .\_\_|\_| |\_|\_| |\_\\_\_, | / / / /  =========|\_|==============|\_\_\_/=/\_/\_/\_/  :: Spring Boot :: (v1.5.3.RELEASE)  2017-06-03 18:48:39.870 INFO 21556 --- [ main] com.example.spring.Application : Starting Application on DESKTOP-2BHUVHG with PID 21556 (E:\WORK\workspaces\IDEA-5-6\spring-boot-learn\target\classes started by ASUS in E:\WORK\workspaces\IDEA-5-6\spring-boot-learn)  2017-06-03 18:48:39.873 INFO 21556 --- [ main] com.example.spring.Application : No active profile set, falling back to default profiles: default  ApplicationPreparedEvent org.springframework.boot.context.embedded.AnnotationConfigEmbeddedWebApplicationContext@75881071  2017-06-03 18:48:39.935 INFO 21556 --- [ main] ationConfigEmbeddedWebApplicationContext : Refreshing org.springframework.boot.context.embedded.AnnotationConfigEmbeddedWebApplicationContext@75881071: startup date [Sat Jun 03 18:48:39 CST 2017]; root of context hierarchy  2017-06-03 18:48:41.294 INFO 21556 --- [ main] s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat initialized with port(s): 8080 (http)  2017-06-03 18:48:41.301 INFO 21556 --- [ main] o.apache.catalina.core.StandardService : Starting service Tomcat  2017-06-03 18:48:41.302 INFO 21556 --- [ main] org.apache.catalina.core.StandardEngine : Starting Servlet Engine: Apache Tomcat/8.5.14  2017-06-03 18:48:41.369 INFO 21556 --- [ost-startStop-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext  2017-06-03 18:48:41.369 INFO 21556 --- [ost-startStop-1] o.s.web.context.ContextLoader : Root WebApplicationContext: initialization completed in 1439 ms  2017-06-03 18:48:41.470 INFO 21556 --- [ost-startStop-1] o.s.b.w.servlet.ServletRegistrationBean : Mapping servlet: 'dispatcherServlet' to [/]  2017-06-03 18:48:41.474 INFO 21556 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'characterEncodingFilter' to: [/\*]  2017-06-03 18:48:41.474 INFO 21556 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'hiddenHttpMethodFilter' to: [/\*]  2017-06-03 18:48:41.474 INFO 21556 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'httpPutFormContentFilter' to: [/\*]  2017-06-03 18:48:41.474 INFO 21556 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'requestContextFilter' to: [/\*]  2017-06-03 18:48:41.685 INFO 21556 --- [ main] s.w.s.m.m.a.RequestMappingHandlerAdapter : Looking for @ControllerAdvice: org.springframework.boot.context.embedded.AnnotationConfigEmbeddedWebApplicationContext@75881071: startup date [Sat Jun 03 18:48:39 CST 2017]; root of context hierarchy  2017-06-03 18:48:41.726 INFO 21556 --- [ main] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped "{[/{name}]}" onto public java.lang.String com.example.spring.Application.test1(java.lang.String)  2017-06-03 18:48:41.729 INFO 21556 --- [ main] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped "{[/error]}" onto public org.springframework.http.ResponseEntity<java.util.Map<java.lang.String, java.lang.Object>> org.springframework.boot.autoconfigure.web.BasicErrorController.error(javax.servlet.http.HttpServletRequest)  2017-06-03 18:48:41.729 INFO 21556 --- [ main] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped "{[/error],produces=[text/html]}" onto public org.springframework.web.servlet.ModelAndView org.springframework.boot.autoconfigure.web.BasicErrorController.errorHtml(javax.servlet.http.HttpServletRequest,javax.servlet.http.HttpServletResponse)  2017-06-03 18:48:41.749 INFO 21556 --- [ main] o.s.w.s.handler.SimpleUrlHandlerMapping : Mapped URL path [/webjars/\*\*] onto handler of type [class org.springframework.web.servlet.resource.ResourceHttpRequestHandler]  2017-06-03 18:48:41.749 INFO 21556 --- [ main] o.s.w.s.handler.SimpleUrlHandlerMapping : Mapped URL path [/\*\*] onto handler of type [class org.springframework.web.servlet.resource.ResourceHttpRequestHandler]  2017-06-03 18:48:41.773 INFO 21556 --- [ main] o.s.w.s.handler.SimpleUrlHandlerMapping : Mapped URL path [/\*\*/favicon.ico] onto handler of type [class org.springframework.web.servlet.resource.ResourceHttpRequestHandler]  2017-06-03 18:48:41.871 INFO 21556 --- [ main] o.s.j.e.a.AnnotationMBeanExporter : Registering beans for JMX exposure on startup  2017-06-03 18:48:41.919 INFO 21556 --- [ main] s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat started on port(s): 8080 (http)  ApplicationReadyEvent StandardServletEnvironment {activeProfiles=[], defaultProfiles=[default], propertySources=[server.ports,servletConfigInitParams,servletContextInitParams,systemProperties,systemEnvironment,random,applicationConfig: [classpath:/application.properties]]}  2017-06-03 18:48:41.923 INFO 21556 --- [ main] com.example.spring.Application : Started Application in 2.321 seconds (JVM running for 2.746) |

## 三 properties / yml配置

类路径下创建application.properties配置文件，内容如下：

|  |
| --- |
| **user.userName**=**hujintao user.age**=**25** |

创建一个bean类，注意：@component申明为bean，@PropertySource找到配置文件，@ConfigurationProperties注解匹配配置文件中前缀是user的配置

|  |
| --- |
| **package** com.example.pojo;  **import** org.springframework.boot.context.properties.ConfigurationProperties; **import** org.springframework.context.annotation.PropertySource; **import** org.springframework.stereotype.Component;  */\*\*  \* Created by ASUS on 2017/6/4.  \*/* @Component @ConfigurationProperties(prefix = **"user"**) *// @PropertySource("classpath:config/user.properties")* @PropertySource(**"classpath:application.properties"**) **public class** User {  **private** String **userName**;  **private int age**;   **public int** getAge() {  **return age**;  }   **public void** setAge(**int** age) {  **this**.**age** = age;  }   **public** String getUserName() {  **return userName**;  }   **public void** setUserName(String userName) {  **this**.**userName** = userName;  }   **public** User(){   }   @Override  **public** String toString() {  **return "User{"** +  **"userName='"** + **userName** + **'\''** +  **", age="** + **age** +  **'}'**;  } } |

创建一个controller，

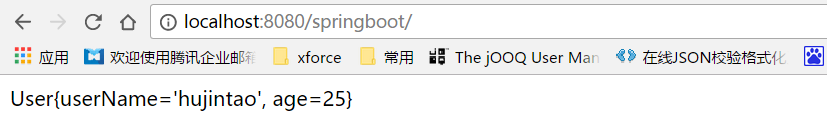
声明的User类通过@autowired自动注入

|  |
| --- |
| @RestController @RequestMapping(**"/springboot"**) **public class** HelloWorld {   @Autowired  **private** User **user**;   @RequestMapping(**"/"**)  **public** String helloWorld(){  System.***out***.println(**user**);  **return user**.toString();  } } |

启动Application.java，浏览器输入localhost:8080/springboot/：

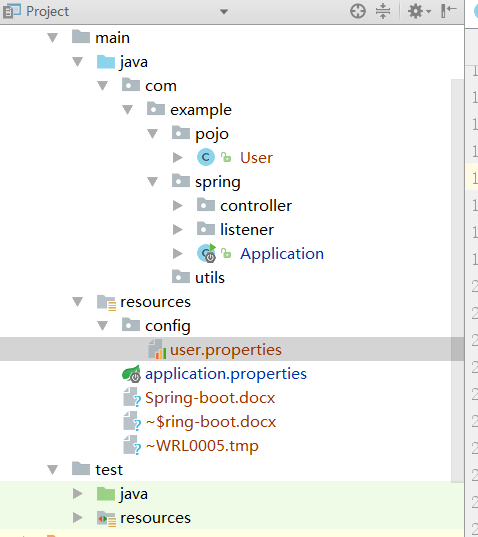
控制台输出：User{userName='hujintao', age=25}

页面显示：



注意：一开始user不能自动注入，经过一番查找，才发现我把User.java声明在Application.java启动类的包外了，spring无法找到User类，[解决办法是在Application.java前添加@ComponentScan](mailto:解决办法是在Application.java前添加@ComponentScan)注解声明自动扫描的包范围

@ComponentScan("com.example")



.yml配置文件同理（注意yml文件的格式）

在类路径下声明application.yml，内容如下：

|  |
| --- |
| **user:  userName:** hujintao  **age:** 22 |

修改User类@PropertySource注解的value值为: classpath:application.yml

启动类启动，控制台输出: User{userName='hujintao', age=22}