springmvc依赖注入（springmvc常用5种注解）

作者：有故事的人 来源：范文网 www.wtabcd.cn/fanwen/

本文原地址：https://www.wtabcd.cn/fanwen/zuowen/efb1afa7f1c44f06aa19fa8cd822e606.html

初识springmvc

实现步骤：

新建一个web项目导入相关jar包编写web.xml，注册dispatcherservlet编写springmvc配置文件接下来就是去创建对应的控制类，controller最后完善前端视图和controller之间的对应测试运行调试

使用springmvc必须配置的三大件：

处理器映射器、处理器适配器、视图解析器

通常，我们只需要**手动配置视图解析器**，而**处理器映射器**和**处理器适配器**只需要开启**注解驱动**即可，而省去了大段的xml配置

注解实现springmvc

常见注解

@component组件@service服务@controller控制@respositorydao层

控制器

package com.kuang.controller;import org.springframework.stereotype.controller;import org.springframework.ui.model;import org.springframework.web.bind.annotation.requestmapping;//@controller注解的类会自动添加到spring上下文中@controller@requestmapping("/test2")public class controllertest2{//映射访问路径@requestmapping("/t2")public string index(model model){//spring mvc会自动实电荷量的单位例化一个model对象用于向视图中传值model.addattribute("msg", "controllertest2");//返回视图位置return "test";}}

@controller是为了让spring ioc容器初始化时自动扫描到；@requestmapping是为了映射请求路径，这里因为类与方法上都有映射所以访问时应该是/test2/t2；

标准maven依赖

<?xml version="1.0" encoding="utf-8"?><project xmlns="http://maven.apache.org/pom/4.0.0" xmlns:xsi="/d/file/titlepic/xmlschema-instance" xsi:schemalocation ="http://maven.apache.org/pom/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"> <model version>4.0.0</modelversion> <groupid>com.reliable</groupid> <artifactid>springm vc2</artifactid> <packaging>pom</packaging> <version>1.0-snapshot</version> <modules> <module>springmvc-04-controller</m五年级数学题odule> </modules> <dependencies> <dependency> <groupid>junit</groupid> <artifactid>junit</artifactid> <version>4.12</version> </dependency> <depe ndency> <groupid>org.springframework</groupid> <artifactid>spring-webmvc</artifactid> <version>5.1.9.release</vers ion> </dependency> <dependency> <groupid>javax.servlet</groupid> <artifactid>servlet-api</artifactid> <version>2.5</version> </dependency> <dependency> <groupid>javax.servlet.jsp</groupid> <artifactid>jsp-api</artifactid> <version>2.2</version> </dependency> <dependency> <groupid>jav ax.servlet</groupid> <artifactid>jstl</artifactid> <version>1.2</version> </dependency> </dependencies> <buil d> <resources> <resource> <directory>src/main/java</directory> <includes> <include>**/*.properties</include> <include>**/*.xml</include> </inc ludes> <filtering>false</filtering> </ 秋天的日记50字resource> <resource> <directory>src/main/resources</directory> <includes> <include>**/*.properties</include> <include>**/*.xml</include> </inclu des> <filtering>false</filtering> </re source> </resources> </build></project>

一、配置pom.xml

<?xml version="1.0" encoding="utf-8"?><project xmlns="http://maven.apache.org/pom/4.0.0" xmlns:xsi="/d/file/titlepic/xmlschema-instance" xsi:schemalocation ="http://maven.apache.org/pom/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"> <parent > <artifactid>springmvc2</artifactid> <groupid>com.reliable</gr oupid> <version>1.0-snapshot</version> </parent> <modelversio n>4.0.0</modelversion> <artifactid>springmvc-04-controller</artifactid> <dependenc

ies> <dependency> <groupid>javax.servlet</groupid>
<artifactid>servlet-api</artifactid> <version>
2.5</version> </dependency> <dependency>
<groupid>javax.servlet.jsp</groupid> <artifactid>jsp-api</artifact
id> <version>2.2</version> </dependency> <
/dependencies> <build> <resources> <resourc
e> <directory>src/main/java</directory>
<includes> <include>**/*.properties<
/include> <include>**/*.xml</include>
</includes> <filtering>false</filtering>
火腿玉米浓汤 </resource> <resource>
<directory>src/main/resources</directory>
<includes> <include>**/*.properties</in
clude> <include>**/*.xml</include>
</includes> <filtering>false</filtering>
</resource> </resources> </build></project>

二、配置web.xml

<?xml version="1.0" encoding="utf-8"?><web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xmlns:xsi="/d/file/titlepic/xmlschema-instance" xsi:schemalocation
="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd"
version="4.0"> <!--1.注册servlet--> <servlet> <servlet-n
ame>springmvc</servlet-name> <servlet-class>org.springframework.web.servlet.dispat
cherservlet</servlet-class> <!--通过初始化参数指定springmvc配置文件的位置，进
行关联--> <init-param> <param-name>contextconfiglo
cation</param-name> <param-value>classpath:springmvc-servlet.xml</par
am-value> </init-param> <!-- 启动顺序，数字越小，启动越
早 --> <load-on-startup>1</load-on-startup> </servlet> <!--所
有请求都会被springmvc拦截 --> <servlet-mapping> <servlet-name>spri
ngmvc</servlet-name> <url-pattern>/</url-pattern> </servlet-
mapping></web-app>

三、配置springmvc-servlet.xml

<?xml version="1.0" encoding="utf-8"?><beans xmlns="/d/file/titlepic/" xmlns:xsi=
"/d/file/titlepic/xmlschema-instance" xmlns:context="/d/file/titlepic/" xm
lns:mvc="/d/file/titlepic/" xsi:schemalocation="http://www.springframework.org/schema/
beans /d/file/titlepic/spring-beans.xsd http://www.springframework.
org/schema/context /d/file/titlepic/spring-context.xsd http://www.s
pringframework.org/schema/mvc https://www.springframework.org/schema/mvc/sprin
g-mvc.xsd"> <!-- 自动扫描包，让指定包下的注解生效,由ioc容器统一管理 -->
<context:component-scan base-package="com.kuang.controller"/> <!-- 让spring mvc不
处理静态资源：html 等--> <mvc:default-servlet-handler/> <!-- 支持mvc

注解驱动　在spring中一般采用@requestmapping注解来完成映射关系
要想使@requestmapping注解生效　必须向上下文中注册defaultannotationhandlermapping　和一个annotationmethodhandleradapter实例　这两个实例分别在类级别和方法级别处理。　而annotation-driven配置帮助我们自动完成上述两个实例的注入。<mvc:annotation-driven /> 完成了映射和适配（支持用注解完成）　--> <mvc:annotation-driven /> <!-- 视图解析器 --> <bean class="org.springframework.web.servlet.view.internalresourceviewresolver" id="internalresourceviewresolver"> <!-- 前缀 --> <property 康复治疗学专业name="prefix" value="/web-inf/jsp/" /> <!-- 后缀 --> <property name="suffix" value=".jsp" /> </bean></beans>

restful 风格

概念
restful就是一个资源定位及资源操作的风格。不是标准也不是协议，只是一种风格。基于这个风格设计的软件可以更简洁，更有层次，更易于实现缓存等机制。

功能
资源：互联网所有的事物都可以被抽象为资源
资源操作：使用post、delete、put、get，使用不同方法对资源进行操作。
分别对应 添加、 删除、修改、查询。

restfulcontroller(@pathvariable)

package com.kuang.controller;import org.springframework.stereotype.controller;import org.springframework.ui.model;import org.springframework.web.bind.annotation.pathvariable;import org.springframework.web.bind.annotation.requestmapping;import org.springframework.web.bind.annotation.requestmethod;@controllerpublic class restfulcontroller {//映射访问路径@requestmapping("/commit/{p1}/{p2}")public string index(@pathvariable int p1, @pathvariable int p2, model model){int result = p1+p2;//spring mvc会自动实例化一个model对象用于向视图中传值model.addattribute("msg", "结果："+result);//返回视图位置return "test";}//映射访问路径,必须是get请求@requestmapping(value = "/hello",method = {requestmethod.get})public string index2(model model){model.addattribute("msg", "hello!");return "test";}}

使用method属性指定请求类型

用于约束请求的类型，可以收窄请求范围。指定请求谓词的类型如get, post, head, options, put, patch, delete, trace等。

//映射访问路径,必须是get请求@requestmapping(value = "/hello",method = {requestmethod.get})public string index2(model model){model.addattribute("msg", "hello!");return "test";}

除了添加method，还可以使用注解

@getmapping
@postmapping
@putmapping
@deletemapping
@patchmapping

//映射访问路径,必须是get请求@getmapping(value = "/hello")public string index2(model mod
el){model.addattribute("msg", "hello!");return "test";}