

公告

昵称: Sunnier
园龄: 4年1个月
粉丝: 394
关注: 0
+加关注

< 2019年1月 >						
日	一	二	三	四	五	六
30	31	1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31	1	2
3	4	5	6	7	8	9

搜索

找找看

谷歌搜索

常用链接

- 我的随笔
- 我的评论
- 我的参与
- 最新评论
- 我的标签

随笔分类

- .net(24)
- CodeSmith模板(1)
- Java(10)
- Oracle(2)
- Python(1)
- SqlServer(1)
- unity3d(1)
- WCF(1)
- WorkFlow(1)
- 编译原理(1)
- 操作系统(1)
- 高并发、分布式(4)
- 软件工程经济学(3)
- 设计模式(2)
- 数据结构及算法(5)
- 数据与计算机通信(1)
- 微信公众平台开发(1)

随笔档案

- 2016年11月 (2)
- 2015年11月 (1)
- 2015年7月 (1)
- 2015年6月 (7)
- 2015年5月 (5)
- 2015年4月 (13)
- 2015年3月 (4)









随笔-50 文章-0 评论-250

史上最全最强SpringMVC详细示例实战教程

SpringMVC学习笔记----

一、SpringMVC基础入门，创建一个HelloWorld程序

1.首先，导入SpringMVC需要的jar包。

-  commons-logging-1.2.jar
-  spring-aop-4.1.6.RELEASE.jar
-  spring-beans-4.1.6.RELEASE.jar
-  spring-context-4.1.6.RELEASE.jar
-  spring-core-4.1.6.RELEASE.jar
-  spring-expression-4.1.6.RELEASE.jar
-  spring-web-4.1.6.RELEASE.jar
-  spring-webmvc-4.1.6.RELEASE.jar

2.添加Web.xml配置文件中关于SpringMVC的配置

```
<!--configure the setting of springmvcDispatcherServlet and configure the mapping-->
<servlet>
    <servlet-name>springmvc</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
    <init-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>classpath:springmvc-servlet.xml</param-value>
    </init-param>
    <!-- <load-on-startup>1</load-on-startup> -->
</servlet>

<servlet-mapping>
    <servlet-name>springmvc</servlet-name>
    <url-pattern></url-pattern>
</servlet-mapping>
```

3.在src下添加springmvc-servlet.xml配置文件

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:mvc="http://www.springframework.org/schema/mvc"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/beans/spring-beans.xsd
        http://www.springframework.org/schema/context
        http://www.springframework.org/schema/context/spring-context-4.1.xsd
        http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc-4.1.xsd">

    <!-- scan the package and the sub package -->
    <context:component-scan base-package="test.SpringMVC"/>

    <!-- don't handle the static resource -->
```

- 2015年1月 (1)
- 2014年12月 (9)
- 2014年11月 (7)
- 最新评论
1. Re:史上最全最强SpringMVC详细示例实战教程
- 给楼主一个赞，真的很适合我这样对springmvc了解很少的人用
- 逍遥人生MIAO
2. Re:史上最全最强SpringMVC详细示例实战教程
- 免费视频教程：
- 分享的世界
3. Re:全面解析Java类加载器
- 这个在实际开发中有用吗啊？
- 一毛不拔的朋友
4. Re:史上最全最强SpringMVC详细示例实战教程
- @god_Summer今天听老师说post和get的原因...
- 天堂里的另一天
5. Re:Spring详细教程
- 真的好，写的真心不错
- 为java付出的男人

- 阅读排行榜
1. 史上最全最强SpringMVC详细示例实战教程(335127)
2. Hibernate详细教程(50117)
3. Spring详细教程(38247)
4. 深入理解java垃圾回收机制(25758)
5. SqlServer存储过程详解(24503)

- 评论排行榜
1. 史上最全最强SpringMVC详细示例实战教程(74)
2. ASP.NET MVC项目框架快速搭建实战(63)
3. 2016阿里巴巴校招offer面经(15)
4. 几个面试经典算法题Java解答(15)
5. Hibernate详细教程(10)

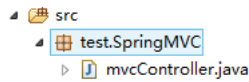
- 推荐排行榜
1. 史上最全最强SpringMVC详细示例实战教程(124)
2. ASP.NET MVC项目框架快速搭建实战(16)
3. 深入理解java垃圾回收机制(12)
4. Spring详细教程(8)
5. Hibernate详细教程(8)

```
<mvc:default-servlet-handler />

<!-- if you use annotation you must configure following setting -->
<mvc:annotation-driven />

<!-- configure the InternalResourceViewResolver -->
<bean class="org.springframework.web.servlet.view.InternalResourceViewResolver"
      id="internalResourceViewResolver">
    <!-- 前缀 -->
    <property name="prefix" value="/WEB-INF/jsp/" />
    <!-- 后缀 -->
    <property name="suffix" value=".jsp" />
</bean>
</beans>
```

- 4.在WEB-INF文件夹下创建名为jsp的文件夹，用来存放jsp视图。创建一个hello.jsp，在body中添加“Hello World”。
- 5.建立包及Controller，如下所示



6.编写Controller代码

```
@Controller
@RequestMapping("/mvc")
public class mvcController {

    @RequestMapping("/hello")
    public String hello(){
        return "hello";
    }
}
```

- 7.启动服务器，键入 http://localhost:8080/项目名/mvc/hello

二、配置解析

1.DispatcherServlet

DispatcherServlet是前置控制器，配置在web.xml文件中的。拦截匹配的请求，Servlet拦截匹配规则要自己定义，把拦截下来的请求，依据相应的规则分发到目标Controller来处理，是配置spring MVC的第一步。

2.InternalResourceViewResolver

视图名称解析器

- 3.以上出现的注解

- @Controller 负责注册一个bean 到spring 上下文中
- @RequestMapping 注解为控制器指定可以处理哪些 URL 请求

三、SpringMVC常用注解

@Controller

负责注册一个bean 到spring 上下文中

@RequestMapping

注解为控制器指定可以处理哪些 URL 请求

@RequestBody

该注解用于读取Request请求的body部分数据,使用系统默认配置的HttpMessageConverter进行解析,然后把相应的数据绑定到要返回的对象上,再把HttpMessageConverter返回的对象数据绑定到 controller中方法的参数上

@ResponseBody

该注解用于将Controller的方法返回的对象,通过适当的HttpMessageConverter转换为指定格式后,写入到Response对象的body数据区

@ModelAttribute

在方法定义上使用 @ModelAttribute 注解: Spring MVC 在调用目标处理方法前,会先逐个调用在方法级上标注了 @ModelAttribute 的方法

在方法的入参前使用 @ModelAttribute 注解: 可以从隐含对象中获取隐含的模型数据中获取对象,再将请求参数 - 绑定到对象中,再传入入参将方法入参对象添加到模型中

@RequestParam

在处理方法入参处使用 @RequestParam 可以把请求参数传递给请求方法

@PathVariable

绑定 URL 占位符到入参

@ExceptionHandler

注解到方法上,出现异常时会执行该方法

@ControllerAdvice

使一个Controller成为全局的异常处理类,类中用@ExceptionHandler方法注解的方法可以处理所有Controller发生的异常

四、自动匹配参数

```
//match automatically
@RequestMapping("/person")
public String toPerson(String name,double age){
    System.out.println(name+" "+age);
    return "hello";
}
```

五、自动装箱

1.编写一个Person实体类

```
package test.SpringMVC.model;

public class Person {
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public int getAge() {
        return age;
    }
    public void setAge(int age) {
        this.age = age;
    }
}
```

```
private String name;
private int age;

}
```

2.在Controller里编写方法

```
//boxing automatically
@RequestMapping("/person1")
public String toPerson(Person p){
    System.out.println(p.getName()+" "+p.getAge());
    return "hello";
}
```

六、使用InitBinder来处理Date类型的参数

```
//the parameter was converted in initBinder
@RequestMapping("/date")
public String date(Date date){
    System.out.println(date);
    return "hello";
}

//At the time of initialization,convert the type "String" to type "date"
@InitBinder
public void initBinder(ServletRequestDataBinder binder){
    binder.registerCustomEditor(Date.class, new CustomDateEditor(new SimpleDateFormat("yyyy-MM-dd"),
        true));
}
```

七、向前台传递参数

```
//pass the parameters to front-end
@RequestMapping("/show")
public String showPerson(Map<String,Object> map){
    Person p =new Person();
    map.put("p", p);
    p.setAge(20);
    p.setName("jayjay");
    return "show";
}
```

前台可在Request域中取到"p"

八、使用Ajax调用

```
//pass the parameters to front-end using ajax
@RequestMapping("/getPerson")
public void getPerson(String name,PrintWriter pw){
    pw.write("hello,"+name);
}

@RequestMapping("/name")
```

```
public String sayHello(){  
    return "name";  
}
```



前台用下面的jQuery代码调用



```
$(function() {  
    $("#btn").click(function() {  
        $.post("mvc/getPerson", {name:$("#name").val()}, function(data) {  
            alert(data);  
        });  
    });  
});
```



九、在Controller中使用redirect方式处理请求

```
//redirect  
@RequestMapping("/redirect")  
public String redirect(){  
    return "redirect:hello";  
}
```

十、文件上传

1.需要导入两个jar包



commons-fileupload-1.3.1.jar



commons-io-2.4.jar



----- 144111

2.在SpringMVC配置文件中加入

```
<!-- upload settings -->  
<bean id="multipartResolver"  
class="org.springframework.web.multipart.commons.CommonsMultipartResolver">  
    <property name="maxUploadSize" value="10240000"></property>  
</bean>
```

3.方法代码



```
@RequestMapping(value="/upload",method=RequestMethod.POST)  
public String upload(HttpServletRequest req) throws Exception{  
    MultipartHttpServletRequest mreq = (MultipartHttpServletRequest)req;  
    MultipartFile file = mreq.getFile("file");  
    String fileName = file.getOriginalFilename();  
    SimpleDateFormat sdf = new SimpleDateFormat("yyyyMMddHHmmss");  
    FileOutputStream fos = new  
FileOutputStream(req.getSession().getServletContext().getRealPath("/") +  
        "upload/"+sdf.format(new Date())+fileName.substring(fileName.lastIndexOf('.')+1));  
    fos.write(file.getBytes());  
    fos.flush();  
    fos.close();  
  
    return "hello";  
}
```



4.前台form表单

```
<form action="mvc/upload" method="post" enctype="multipart/form-data">
    <input type="file" name="file"><br>
    <input type="submit" value="submit">
</form>
```

十一、使用@RequestParam注解指定参数的name

```
@Controller
@RequestMapping("/test")
public class mvcController1 {
    @RequestMapping(value="/param")
    public String testRequestParam(@RequestParam(value="id") Integer id,
        @RequestParam(value="name") String name) {
        System.out.println(id+" "+name);
        return "/hello";
    }
}
```

十二、RESTFul风格的SpringMVC

1.RestController

```
@Controller
@RequestMapping("/rest")
public class RestController {
    @RequestMapping(value="/user/{id}",method=RequestMethod.GET)
    public String get(@PathVariable("id") Integer id){
        System.out.println("get"+id);
        return "/hello";
    }

    @RequestMapping(value="/user/{id}",method=RequestMethod.POST)
    public String post(@PathVariable("id") Integer id){
        System.out.println("post"+id);
        return "/hello";
    }

    @RequestMapping(value="/user/{id}",method=RequestMethod.PUT)
    public String put(@PathVariable("id") Integer id){
        System.out.println("put"+id);
        return "/hello";
    }

    @RequestMapping(value="/user/{id}",method=RequestMethod.DELETE)
    public String delete(@PathVariable("id") Integer id){
        System.out.println("delete"+id);
        return "/hello";
    }
}
```

2.form表单发送put和delete请求

在web.xml中配置

```
<!-- configure the HiddenHttpMethodFilter,convert the post method to put or delete -->
<filter>
```

```
<filter-name>HiddenHttpMethodFilter</filter-name>
<filter-class>org.springframework.web.filter.HiddenHttpMethodFilter</filter-class>
</filter>
<filter-mapping>
<filter-name>HiddenHttpMethodFilter</filter-name>
<url-pattern>/*</url-pattern>
</filter-mapping>
```



在前台可以用以下代码产生请求

```
<form action="rest/user/1" method="post">
<input type="hidden" name="_method" value="PUT">
<input type="submit" value="put">
</form>

<form action="rest/user/1" method="post">
<input type="submit" value="post">
</form>

<form action="rest/user/1" method="get">
<input type="submit" value="get">
</form>

<form action="rest/user/1" method="post">
<input type="hidden" name="_method" value="DELETE">
<input type="submit" value="delete">
</form>
```



十三、返回json格式的字符串

1.导入以下jar包

jackson-annotations-2.2.1.jar
 jackson-core-2.2.1.jar
 jackson-core-asl-1.8.8.jar
 jackson-databind-2.2.1.jar
 jackson-mapper-asl-1.8.8.jar

2.方法代码


```
@Controller
@RequestMapping("/json")
public class jsonController {


    @ResponseBody
    @RequestMapping("/user")
    public User get() {
        User u = new User();
        u.setId(1);
        u.setName("jayjay");
        u.setBirth(new Date());
        return u;
    }
}
```




十四、异常的处理


1.处理局部异常 (Controller内)

```
  
@ExceptionHandler  
public ModelAndView exceptionHandler(Exception ex){  
    ModelAndView mv = new ModelAndView("error");  
    mv.addObject("exception", ex);  
    System.out.println("in testExceptionHandler");  
    return mv;  
}  
  
@RequestMapping("/error")  
public String error(){  
    int i = 5/0;  
    return "hello";  
}
```



2.处理全局异常 (所有Controller)

```
  
@ControllerAdvice  
public class testControllerAdvice {  
    @ExceptionHandler  
    public ModelAndView exceptionHandler(Exception ex){  
        ModelAndView mv = new ModelAndView("error");  
        mv.addObject("exception", ex);  
        System.out.println("in testControllerAdvice");  
        return mv;  
    }  
}
```



3.另一种处理全局异常的方法

在SpringMVC配置文件中配置


```
  
<!-- configure SimpleMappingExceptionHandler -->  
<bean class="org.springframework.web.servlet.handler.SimpleMappingExceptionHandler">  
    <property name="exceptionMappings">  
        <props>  
            <prop key="java.lang.ArithmeticException">error</prop>  
        </props>  
    </property>  
</bean>
```



error是出错页面

十五、设置一个自定义拦截器

1.创建一个MyInterceptor类, 并实现HandlerInterceptor接口

```
  
public class MyInterceptor implements HandlerInterceptor {  
  
    @Override  
    public void afterCompletion(HttpServletRequest arg0,  
        HttpServletResponse arg1, Object arg2, Exception arg3)  
        throws Exception {  
        System.out.println("afterCompletion");  
    }  
}
```



```
}

@Override
public void postHandle(HttpServletRequest arg0, HttpServletResponse arg1,
    Object arg2, ModelAndView arg3) throws Exception {
    System.out.println("postHandle");
}

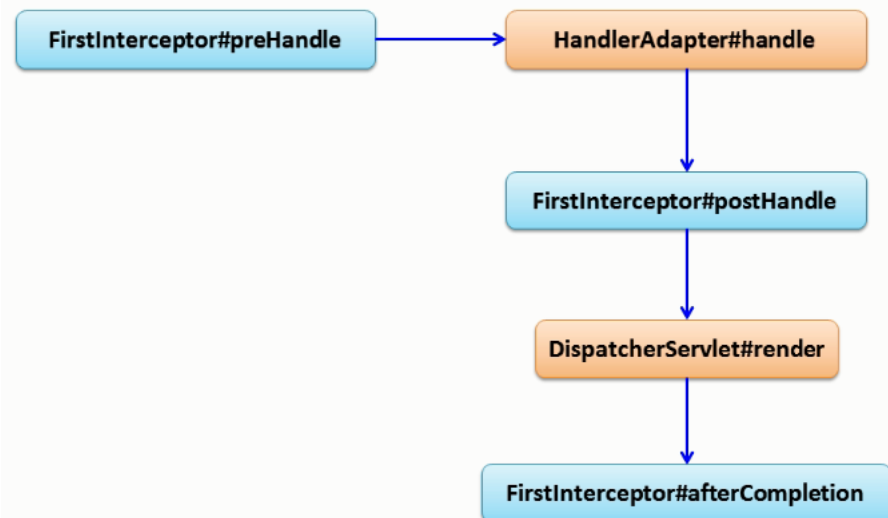
@Override
public boolean preHandle(HttpServletRequest arg0, HttpServletResponse arg1,
    Object arg2) throws Exception {
    System.out.println("preHandle");
    return true;
}
}
```

2.在SpringMVC的配置文件中配置

```
<!-- interceptor setting -->
<mvc:interceptors>
    <mvc:interceptor>
        <mvc:mapping path="/mvc/**"/>
        <bean class="test.SpringMVC.Interceptor.MyInterceptor"></bean>
    </mvc:interceptor>
</mvc:interceptors>
```

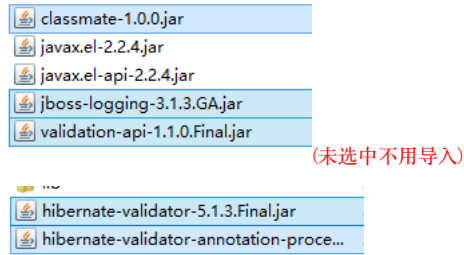
3.拦截器执行顺序

拦截器方法执行顺序



十六、表单的验证（使用Hibernate-validate）及国际化

1.导入Hibernate-validate需要的jar包



2.编写实体类User并加上验证注解

```
public class User {  
    public int getId() {  
        return id;  
    }  
    public void setId(int id) {  
        this.id = id;  
    }  
    public String getName() {  
        return name;  
    }  
    public void setName(String name) {  
        this.name = name;  
    }  
    public Date getBirth() {  
        return birth;  
    }  
    public void setBirth(Date birth) {  
        this.birth = birth;  
    }  
    @Override  
    public String toString() {  
        return "User [id=" + id + ", name=" + name + ", birth=" + birth + "]";  
    }  
    private int id;  
    @NotEmpty  
    private String name;  
  
    @Past  
    @DateTimeFormat(pattern="yyyy-MM-dd")  
    private Date birth;  
}
```

ps:@Past表示时间必须是一个过去值

3.在jsp中使用SpringMVC的form表单

```
<form:form action="form/add" method="post" modelAttribute="user">  
    id:<form:input path="id"/><form:errors path="id"/><br>  
    name:<form:input path="name"/><form:errors path="name"/><br>  
    birth:<form:input path="birth"/><form:errors path="birth"/>  
    <input type="submit" value="submit">  
</form:form>
```

ps:path对应name

4.Controller中代码

```
@Controller
@RequestMapping("/form")
public class formController {

    @RequestMapping(value="/add",method=RequestMethod.POST)
    public String add(@Valid User u,BindingResult br){
        if(br.getErrorCount()>0){
            return "addUser";
        }
        return "showUser";
    }

    @RequestMapping(value="/add",method=RequestMethod.GET)
    public String add(Map<String,Object> map){
        map.put("user",new User());
        return "addUser";
    }
}
```

ps:

- 1.因为jsp中使用了modelAttribute属性，所以必须在request域中有一个"user".
- 2.@Valid 表示按照在实体上标记的注解验证参数
- 3.返回到原页面错误信息回回显，表单也会回显

5.错误信息自定义

在src目录下添加locale.properties

```
NotEmpty.user.name=name can't not be empty
Past.user.birthday=birthday should be a past value
DateTimeFormat.user.birthday=the format of input is wrong
typeMismatch.user.birthday=the format of input is wrong
typeMismatch.user.id=the format of input is wrong
```

在SpringMVC配置文件中配置

```
<!-- configure the locale resource -->
<bean id="messageSource" class="org.springframework.context.support.ResourceBundleMessageSource">
    <property name="basename" value="locale"></property>
</bean>
```

6.国际化显示

在src下添加locale_zh_CN.properties

```
username=账号
password=密码
```

locale.properties中添加

```
username=user name
password=password
```

创建一个locale.jsp

```
<body>
<fmt:message key="username"></fmt:message>
<fmt:message key="password"></fmt:message>
</body>
```

在SpringMVC中配置

```
<!-- make the jsp page can be visited -->
<mvc:view-controller path="/locale" view-name="locale"/>
```

让locale.jsp在WEB-INF下也能直接访问

最后，访问locale.jsp，切换浏览器语言，能看到账号和密码的语言也切换了

十七、压轴大戏-整合SpringIOC和SpringMVC

1.创建一个test.SpringMVC.integrate的包来演示整合，并创建各类

```
test.SpringMVC.integrate
├── User.java
├── UserController.java
└── UserService.java
```

2.User实体类

```
public class User {
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public Date getBirth() {
        return birth;
    }
    public void setBirth(Date birth) {
        this.birth = birth;
    }
    @Override
    public String toString() {
        return "User [id=" + id + ", name=" + name + ", birth=" + birth + "]";
    }
    private int id;
    @NotEmpty
    private String name;

    @Past
    @DateTimeFormat(pattern="yyyy-MM-dd")
    private Date birth;
}
```

3.UserService类

```
@Component
public class UserService {
    public UserService() {
        System.out.println("UserService Constructor...\n\n\n\n\n");
    }

    public void save() {
        System.out.println("save");
    }
}
```



4. UserController



```
@Controller
@RequestMapping("/integrate")
public class UserController {

    @Autowired
    private UserService userService;

    @RequestMapping("/user")
    public String saveUser(@RequestBody @ModelAttribute User u) {
        System.out.println(u);
        userService.save();
        return "hello";
    }
}
```



5. Spring配置文件

在src目录下创建SpringIOC的配置文件applicationContext.xml



```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
           http://www.springframework.org/schema/beans/spring-beans.xsd
           http://www.springframework.org/schema/util
           http://www.springframework.org/schema/util/spring-util-4.0.xsd
           http://www.springframework.org/schema/context
           http://www.springframework.org/schema/context/spring-context.xsd"
       "
       xmlns:util="http://www.springframework.org/schema/util"
       xmlns:p="http://www.springframework.org/schema/p"
       xmlns:context="http://www.springframework.org/schema/context"
       >
    <context:component-scan base-package="test.SpringMVC.integrate">
        <context:exclude-filter type="annotation"
            expression="org.springframework.stereotype.Controller"/>
        <context:exclude-filter type="annotation"
            expression="org.springframework.web.bind.annotation.ControllerAdvice"/>
    </context:component-scan>
</beans>
```



在Web.xml中添加配置



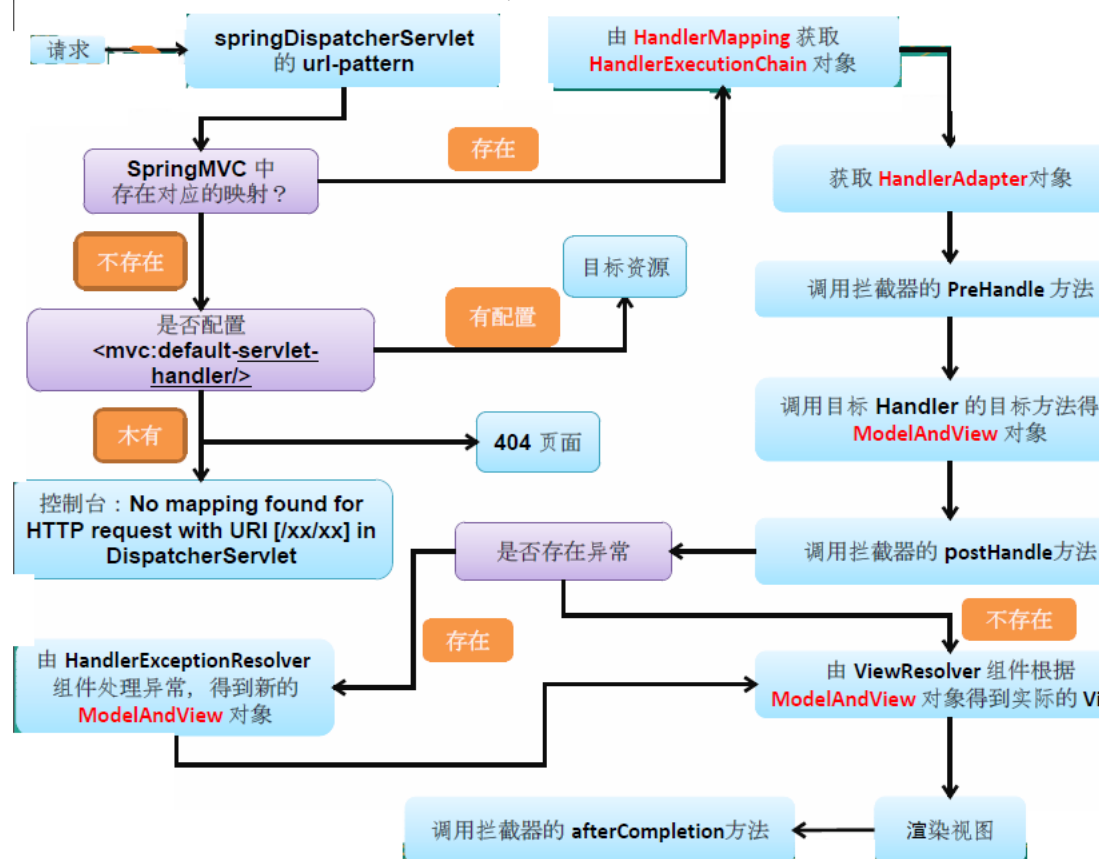
```
<!-- configure the springIOC -->
<listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
</listener>
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>classpath:applicationContext.xml</param-value>
</context-param>
```



6.在SpringMVC中进行一些配置，防止SpringMVC和SpringIOC对同一个对象的管理重合

```
<!-- scan the package and the sub package -->
<context:component-scan base-package="test.SpringMVC.integrate">
    <context:include-filter type="annotation"
        expression="org.springframework.stereotype.Controller"/>
    <context:include-filter type="annotation"
        expression="org.springframework.web.bind.annotation.ControllerAdvice"/>
</context:component-scan>
```

十八、SpringMVC详细运行流程图



十九、SpringMVC运行原理

1. 客户端请求提交到DispatcherServlet
2. 由DispatcherServlet控制器查询一个或多个HandlerMapping，找到处理请求的Controller
3. DispatcherServlet将请求提交到Controller
4. Controller调用业务逻辑处理后，返回ModelAndView
5. DispatcherServlet查询一个或多个ViewResolver视图解析器，找到ModelAndView指定的视图
6. 视图负责将结果显示到客户端

二十、SpringMVC与struts2的区别

- 1、springmvc基于方法开发的，struts2基于类开发的。springmvc将url和controller里的方法映射。映射成功后springmvc生成一个Handler对象，对象中只包括了一个method。方法执行结束，形参数据销毁。springmvc的controller开发类似web service开发。
- 2、springmvc可以进行单例开发，并且建议使用单例开发，struts2通过类的成员变量接收参数，无法使用单例，只能使用多例。
- 3、经过实际测试，struts2速度慢，在于使用struts标签，如果使用struts建议使用jsdl。

分类: [Java](#)

好文要顶

关注我

收藏该文



Sunnier

关注 - 0

粉丝 - 394

124

3

+加关注

« 上一篇: [Spring详细教程](#)
» 下一篇: [全面解析Java类加载器](#)

posted @ 2015-06-05 23:26 Sunnier 阅读(335127) 评论(74) 编辑 收藏

< Prev

1

2

评论列表

#51楼	2016-09-21 23:01	沧海一滴	<pre>//boxing automatically @RequestMapping("/person1") public String toPerson(Person p){ System.out.println(p.getName()+" "+p.getAge()); return "hello"; } ===== 想把request Header中的数据，自动设置到Person 中的一个字段，譬如id中，需要怎么操作呢？</pre>	支持(0) 反对(0)
#52楼	2016-10-23 23:39	_dafeng	32个👍	支持(1) 反对(3)
#53楼	2016-11-07 21:00	欧阳静秋	要是从数据库传输到前台是要用Spring 接口的方式吗 SpringMVC有没有直接的方法？	支持(0) 反对(0)
#54楼[楼主]	2016-11-07 21:03	Sunnier	@ 欧阳静秋 map就是springmvc隐含的一个参数啊	支持(0) 反对(0)
#55楼	2017-02-20 14:52	Variazioni	没有jsp的代码。。 新手根本就运行不起来。。	支持(8) 反对(2)
#56楼	2017-05-10 15:15	god_Summer	restful风格put提交，返回页面报405方法冲突，这个问题能解决吗？或者说是根本不会有这样的应用？	支持(0) 反对(0)
#57楼	2017-05-11 11:14	liuchaoxuan	jsp代码让你吃了啊。。。。。。。。	支持(7) 反对(1)
#58楼	2017-07-26 16:14	Marsom		

哈哈 57楼说jsp代码让你吃了。哈哈 给你32个赞够不够，不够再加一个

支持(0) 反对(0)

#59楼 2017-09-16 13:10 长安怎乱

不错不错，收藏了。

推荐下，分库分表中间件 Sharding-JDBC 源码解析 17 篇：www.yunai.me/categories/Sharding-JDBC/?cnblog&601

苟

支持(0) 反对(0)

#60楼 2017-10-11 11:34 望仔爱好程序

总结的很详细，赞一下贴主

支持(0) 反对(0)

#61楼 2017-11-14 15:47 JamesDragon

适合新手对SpringMVC有个大概的了解，整理的还是不错的，赞楼主一个！

支持(0) 反对(0)

#62楼 2017-11-25 21:09 一丶阳光

厉害的不行不行的，对新手帮助很大。看见其他人说没有jsp页面代码。jsp逗给你了干脆给个demo给你不是更好？刚开始学习的时候觉得还是自己写的demo映像才更深才更有帮助

支持(1) 反对(0)

#63楼 2017-12-09 15:33 微暖丶雪下落日月葵

很不错，适合新手和转java的同学，够精简~ 怒赞

支持(0) 反对(0)

#64楼 2017-12-22 09:25 Gavin_Gao

总结的不错

支持(0) 反对(0)

#65楼 2018-01-12 17:12 jokerZer0

1024个赞

支持(0) 反对(0)

#66楼 2018-05-28 08:23 奋斗^{TEL}

楼主，能给一波对应的jar包么？

支持(0) 反对(0)

#67楼[楼主] 2018-05-29 11:24 Sunnier

@ 奋斗^{TEL}
网上有的，百度一下就行

支持(0) 反对(0)

#68楼 2018-07-23 23:54 槐殇树

为什么我的springmvc-servlet.xml 文件总是报错 Referenced file contains errors (<http://www.springframework.org/schema/beans/spring-beans-4.1.xsd>). For more information, right click on the message in the Problems View and select "Show Details..."

支持(0) 反对(0)

#69楼 2018-08-20 22:56 wizard_Q

jsp代码自己写呗

支持(0) 反对(0)

#70楼 2018-09-16 21:58 stonebye

写的不错，收藏了

支持(0) 反对(0)

#71楼 2018-10-11 23:31 琴_酒

友情提示,如果按博主的AOP的版本会报

1 | java.lang.NoSuchMethodError: org.springframework.aop.framework.AopProxyUtils.getSingletonTarget

- 需要把AOP的版本提高

支持(1) 反对(0)
- #72楼 2018-10-29 22:18 天堂里的另一天

@_god_Summer
今天听老师说post和get的原因

支持(0) 反对(0)
- #73楼 2018-12-05 17:32 分享的世界

免费视频教程: <http://www.chilangedu.com>

支持(0) 反对(0)
- #74楼 2018-12-11 11:41 逍遥人生MIAO

给楼主一个赞，真的很适合我这样对springmvc了解很少的人用

支持(0) 反对(0)

注册用户登录后才能发表评论，请 [登录](#) 或 [注册](#)，[访问网站首页](#)。

【推荐】超50万VC++源码：大型组态工控、电力仿真CAD与GIS源码库！

- 相关博文：
- SpringMVC详细示例实战教程

· 史上最全最强SpringMVC详细示例实战教程 【good】

· 转：史上最全最强SpringMVC详细示例实战教程

· SpringMVC详细示例实战教程（较全开发教程）

· SpringMVC+RestFul详细示例实战教程

- 最新新闻：
- 马云最新演讲全文：企业发展不好 怪宏观不如怪自己

· 继财产冻结后 锤子又被法院保全：涉及金额1577万元

· 苹果周四开盘大跌8.8% 盘中股价创下52周新低

· 定了：嫦娥四号成功落月 月球车命名“玉兔二号”

· 网曝华为用iPhone发推文处罚曝光：责任人月薪下调5千
- » 更多新闻...