

# Spring MVC form handling annotation example

By mkyong (<http://www.mkyong.com/author/mkyong/>) | August 20, 2010 | Updated : August 29, 2012 | Viewed : 328,964 times +572 pv/w

In this tutorial, we show you how to do form handling by using annotation in Spring MVC web application.

## Note

This annotation-based example is converted from the last Spring MVC form handling XML-based example (<http://www.mkyong.com/spring-mvc/spring-mvc-form-handling-example/>). So, please compare and spots the different.

## 1. SimpleFormController vs @Controller

In XML-based Spring MVC web application, you create a form controller by extending the `SimpleFormController` class.

In annotation-based, you can use `@Controller` instead.

*SimpleFormController*

```
public class CustomerController extends SimpleFormController{
    //...
}
```

*Annotation*

```
@Controller
@RequestMapping("/customer.htm")
public class CustomerController{
    //...
}
```

## 2. formBackingObject() vs RequestMethod.GET

In `SimpleFormController`, you can initialize the command object for binding in the `formBackingObject()` method. In annotation-based, you can do the same by annotated the method name with `@RequestMapping(method = RequestMethod.GET)`.

*SimpleFormController*

```
@Override
protected Object formBackingObject(HttpServletRequest request)
    throws Exception {

    Customer cust = new Customer();
    //Make "Spring MVC" as default checked value
    cust.setFavFramework(new String []{"Spring MVC"});

    return cust;
}
```

*Annotation*

```

@RequestMapping(method = RequestMethod.GET)
public String initForm(ModelMap model){

    Customer cust = new Customer();
    //Make "Spring MVC" as default checked value
    cust.setFavFramework(new String []{"Spring MVC"});

    //command object
    model.addAttribute("customer", cust);

    //return form view
    return "CustomerForm";
}

```

### 3. onSubmit() vs RequestMethod.POST

In SimpleFormController, the form submission is handle by the **onSubmit()** method. In annotation-based, you can do the same by annotated the method name with **@RequestMapping(method = RequestMethod.POST)**.

*SimpleFormController*

```

@Override
protected ModelAndView onSubmit(HttpServletRequest request,
    HttpServletResponse response, Object command, BindException errors)
    throws Exception {

    Customer customer = (Customer)command;
    return new ModelAndView("CustomerSuccess");
}

```

*Annotation*

```

@RequestMapping(method = RequestMethod.POST)
public String processSubmit(
    @ModelAttribute("customer") Customer customer,
    BindingResult result, SessionStatus status) {

    //clear the command object from the session
    status.setComplete();

    //return form success view
    return "CustomerSuccess";
}

```

### 4. referenceData() vs @ModelAttribute

In SimpleFormController, usually you put the reference data in model via **referenceData()** method, so that the form view can access it. In annotation-based, you can do the same by annotated the method name with **@ModelAttribute**.

*SimpleFormController*

```

@Override
protected Map referenceData(HttpServletRequest request) throws Exception {

    Map referenceData = new HashMap();

    //Data referencing for web framework checkboxes
    List<String> webFrameworkList = new ArrayList<String>();
    webFrameworkList.add("Spring MVC");
    webFrameworkList.add("Struts 1");
    webFrameworkList.add("Struts 2");
    webFrameworkList.add("JSF");
    webFrameworkList.add("Apache Wicket");
    referenceData.put("webFrameworkList", webFrameworkList);

    return referenceData;
}

```

*Spring's form*

```
<form:checkboxes items="${webFrameworkList}" path="favFramework" />
```

### Annotation

```

@ModelAttribute("webFrameworkList")
public List<String> populateWebFrameworkList() {

    //Data referencing for web framework checkboxes
    List<String> webFrameworkList = new ArrayList<String>();
    webFrameworkList.add("Spring MVC");
    webFrameworkList.add("Struts 1");
    webFrameworkList.add("Struts 2");
    webFrameworkList.add("JSF");
    webFrameworkList.add("Apache Wicket");

    return webFrameworkList;
}

```

*Spring's form*

```
<form:checkboxes items="${webFrameworkList}" path="favFramework" />
```

## 5. initBinder() vs @InitBinder

In `SimpleFormController`, you define the binding or register the custom property editor via `initBinder()` method. In annotation-based, you can do the same by annotated the method name with `@InitBinder`.

*SimpleFormController*

```

protected void initBinder(HttpServletRequest request,
ServletRequestDataBinder binder) throws Exception {

    SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");
    binder.registerCustomEditor(Date.class, new CustomDateEditor(dateFormat, true));
}

```

*Annotation*

```

@InitBinder
public void initBinder(WebDataBinder binder) {
    SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");

    binder.registerCustomEditor(Date.class, new CustomDateEditor(dateFormat, true));
}

```

## From Validation

In SimpleFormController, you have to register and map the validator class to the controller class via XML bean configuration file, and the validation checking and work flows will be executed automatically.

In annotation-based, you have to explicitly execute the validator and define the validation flow in the **@Controller** class manually. See the different :

#### SimpleFormController

```
<bean class="com.mkyong.customer.controller.CustomerController">
    <property name="formView" value="CustomerForm" />
    <property name="successView" value="CustomerSuccess" />

    <!-- Map a validator -->
    <property name="validator">
        <bean class="com.mkyong.customer.validator.CustomerValidator" />
    </property>
</bean>
```

#### Annotation

```
@Controller
@RequestMapping("/customer.htm")
public class CustomerController{

    CustomerValidator customerValidator;

    @Autowired
    public CustomerController(CustomerValidator customerValidator){
        this.customerValidator = customerValidator;
    }

    @RequestMapping(method = RequestMethod.POST)
    public String processSubmit(
        @ModelAttribute("customer") Customer customer,
        BindingResult result, SessionStatus status) {

        customerValidator.validate(customer, result);

        if (result.hasErrors()) {
            //if validator failed
            return "CustomerForm";
        } else {
            status.setComplete();
            //form success
            return "CustomerSuccess";
        }
    }
    //...
```

## Full Example

See a complete @Controller example.

```
package com.mkyong.customer.controller;

import java.sql.Date;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.LinkedHashMap;
import java.util.List;
import java.util.Map;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.propertyeditors.CustomDateEditor;
import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.WebDataBinder;
import org.springframework.web.bind.annotation.InitBinder;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.support.SessionStatus;

import com.mkyong.customer.model.Customer;
import com.mkyong.customer.validator.CustomerValidator;

@Controller
@RequestMapping("/customer.htm")
public class CustomerController{

    CustomerValidator customerValidator;

    @Autowired
    public CustomerController(CustomerValidator customerValidator){
        this.customerValidator = customerValidator;
    }

    @RequestMapping(method = RequestMethod.POST)
    public String processSubmit(
        @ModelAttribute("customer") Customer customer,
        BindingResult result, SessionStatus status) {

        customerValidator.validate(customer, result);

        if (result.hasErrors()) {
            //if validator failed
            return "CustomerForm";
        } else {
            status.setComplete();
            //form success
            return "CustomerSuccess";
        }
    }

    @RequestMapping(method = RequestMethod.GET)
    public String initForm(ModelMap model){

        Customer cust = new Customer();
        //Make "Spring MVC" as default checked value
        cust.setFavFramework(new String []{"Spring MVC"});

        //Make "Make" as default radio button selected value
        cust.setSex("M");

        //make "Hibernate" as the default java skills selection
        cust.setJavaSkills("Hibernate");

        //initilize a hidden value
        cust.setSecretValue("I'm hidden value");

        //command object
        model.addAttribute("customer", cust);

        //return form view
        return "CustomerForm";
    }
}
```

```
@ModelAttribute("webFrameworkList")
public List<String> populateWebFrameworkList() {

    //Data referencing for web framework checkboxes
    List<String> webFrameworkList = new ArrayList<String>();
    webFrameworkList.add("Spring MVC");
    webFrameworkList.add("Struts 1");
    webFrameworkList.add("Struts 2");
    webFrameworkList.add("JSF");
    webFrameworkList.add("Apache Wicket");

    return webFrameworkList;
}

@InitBinder
public void initBinder(WebDataBinder binder) {
    SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd");

    binder.registerCustomEditor(Date.class, new CustomDateEditor(dateFormat, true));
}

@ModelAttribute("numberList")
public List<String> populateNumberList() {

    //Data referencing for number radiobuttons
    List<String> numberList = new ArrayList<String>();
    numberList.add("Number 1");
    numberList.add("Number 2");
    numberList.add("Number 3");
    numberList.add("Number 4");
    numberList.add("Number 5");

    return numberList;
}

@ModelAttribute("javaSkillsList")
public Map<String,String> populateJavaSkillList() {

    //Data referencing for java skills list box
    Map<String,String> javaSkill = new LinkedHashMap<String,String>();
    javaSkill.put("Hibernate", "Hibernate");
    javaSkill.put("Spring", "Spring");
    javaSkill.put("Apache Wicket", "Apache Wicket");
    javaSkill.put("Struts", "Struts");

    return javaSkill;
}

@ModelAttribute("countryList")
public Map<String,String> populateCountryList() {

    //Data referencing for java skills list box
    Map<String,String> country = new LinkedHashMap<String,String>();
    country.put("US", "United Stated");
    country.put("CHINA", "China");
    country.put("SG", "Singapore");
    country.put("MY", "Malaysia");

    return country;
}
}
```

To make annotation work, you have to enable the component auto scanning feature in Spring.

```
<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"
  xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
    http://www.springframework.org/schema/context
    http://www.springframework.org/schema/context/spring-context-2.5.xsd">

  <context:component-scan base-package="com.mkyong.customer.controller" />

  <bean class="com.mkyong.customer.validator.CustomerValidator" />

  <!-- Register the Customer.properties -->
  <bean id="messageSource"
    class="org.springframework.context.support.ResourceBundleMessageSource">
    <property name="basename" value="com/mkyong/customer/properties/Customer" />
  </bean>

  <bean id="viewResolver"
    class="org.springframework.web.servlet.view.InternalResourceViewResolver" >
    <property name="prefix">
      <value>/WEB-INF/pages/</value>
    </property>
    <property name="suffix">
      <value>.jsp</value>
    </property>
  </bean>
</beans>
```

## Download Source Code

Download it – SpringMVC-Form-Handling-Annotation-Example.zip (<http://www.mkyong.com/wp-content/uploads/2010/08/SpringMVC-Form-Handling-Annotation-Example.zip>) (12KB)

## Reference

1. Annotated web mvc controllers in spring 2.5 (<http://blog.springsource.com/2007/11/14/annotated-web-mvc-controllers-in-spring-25/>)
2. Spring MVC form handling example – XML version (<http://www.mkyong.com/spring-mvc/spring-mvc-form-handling-example/>)
3. Spring MVC hello world annotation example (<http://www.mkyong.com/spring-mvc/spring-mvc-hello-world-annotation-example/>)
4. Spring MVC MultiActionController annotation example (<http://www.mkyong.com/spring-mvc/spring-mvc-multiactioncontroller-annotation-example/>)

Tags : [annotation](http://www.mkyong.com/tag/annotation/) (<http://www.mkyong.com/tag/annotation/>) [form](http://www.mkyong.com/tag/form/) (<http://www.mkyong.com/tag/form/>) [spring mvc](http://www.mkyong.com/tag/spring-mvc/) (<http://www.mkyong.com/tag/spring-mvc/>)

## Share this article on

Twitter ([https://twitter.com/intent/tweet?text=Spring MVC form handling annotation example&url=http://www.mkyong.com/spring-mvc/spring-mvc-form-handling-annotation-example/&via=mkyong](https://twitter.com/intent/tweet?text=Spring+MVC+form+handling+annotation+example&url=http://www.mkyong.com/spring-mvc/spring-mvc-form-handling-annotation-example/&via=mkyong)) Facebook (<https://www.facebook.com/sharer/sharer.php?u=http://www.mkyong.com/spring-mvc/spring-mvc-form-handling-annotation-example/>) Google+ (<https://plus.google.com/share?url=http://www.mkyong.com/spring-mvc/spring-mvc-form-handling-annotation-example/>)

HOW MUCH DO YOU KNOW ABOUT ROBOTS?


How was the first working robot employed in 1961?

15

▼

- A Searching for gold
- B Spying on Russian-American families
- C Watering at the Olive Garden
- D Making cars for Ford

WANT TO GO TO COLLEGE? WE CAN BE YOUR GUIDE.  
WE'RE ON YOUR PHONE, ONLINE & TOTALLY FREE.



GET STARTED AT **GET SCHOOLED** .COM Ad



## About the Author



**mkyong**

Founder of Mkyong.com (<http://mkyong.com>), love Java and open source stuff. Follow him on Twitter (<https://twitter.com/mkyong>), or befriend him on Facebook (<http://www.facebook.com/java.tutorial>) or Google Plus (<https://plus.google.com/110948163568945735692?rel=author>). If you like my tutorials, consider make a donation to these charities (<http://www.mkyong.com/blog/donate-to-charity/>).

## Related Posts

142k	<a href="#">Spring MVC InternalResourceViewResolver example (/spring-mvc/spring-mvc-internalresourceviewresolver-example/)</a>
126k	<a href="#">Spring MVC hidden value example (/spring-mvc/spring-mvc-hidden-value-example/)</a>
606k	<a href="#">Spring MVC form handling example (/spring-mvc/spring-mvc-form-handling-example/)</a>
308k	<a href="#">Spring MVC hello world annotation example (/spring-mvc/spring-mvc-hello-world-annotation-example/)</a>
398k	<a href="#">Spring MVC file upload example (/spring-mvc/spring-mvc-file-upload-example/)</a>
36k	<a href="#">Spring 3 MVC and RSS feed example (/spring-mvc/spring-3-mvc-and-rss-feed-example/)</a>
5k	<a href="#">ClassNotFoundException : com.sun.syndication.feed.WireFeed (/spring-mvc/classnotfoundexception-com-sun-syndication-feed-wirefeed/)</a>
279k	<a href="#">Gradle - Spring 4 MVC Hello World Example (/spring-mvc/gradle-spring-mvc-web-project-example/)</a>
467k	<a href="#">Spring AOP + AspectJ annotation example (/spring3/spring-aop-aspectj-annotation-example/)</a>
31k	<a href="#">Google App Engine + JDO + Spring MVC, CRUD example (/google-app-engine/google-app-engine-jdo-spring-mvc-crud-example/)</a>

## Popular Posts

<a href="#">Struts 2 Tutorial (/tutorials/struts-2-tutorials/)</a>	874k
<a href="#">How to get current timestamps in Java (/java/how-to-get-current-timestamps-in-java/)</a>	678k
<a href="#">RESTful Java client with Jersey client (/webservices/jax-rs/restful-java-client-with-jersey-client/)</a>	655k
<a href="#">Spring JdbcTemplate Querying examples (/spring/spring-jdbctemplate-querying-examples/)</a>	710k
<a href="#">JavaMail API - Sending email via Gmail SMTP example (/java/javamail-api-sending-email-via-gmail-smtp-example/)</a>	940k
<a href="#">How to read and parse CSV file in Java (/java/how-to-read-and-parse-csv-file-in-java/)</a>	875k
<a href="#">How to write to file in Java - BufferedWriter (/java/how-to-write-to-file-in-java-bufferedwriter-example/)</a>	1.2m
<a href="#">Spring Tutorial (/tutorials/spring-tutorials/)</a>	2.2m
<a href="#">Spring Security Custom Login Form Example (/spring-security/spring-security-form-login-example/)</a>	671k
<a href="#">JSF 2.0 hello world example (/jsf2/jsf-2-0-hello-world-example/)</a>	743k
<a href="#">How to convert String to Date - Java (/java/how-to-convert-string-to-date-java/)</a>	963k
<a href="#">Maven + Spring + Hibernate + MySQL Example (/spring/maven-spring-hibernate-mysql-example/)</a>	655k
<a href="#">log4j.properties example (/logging/log4j-log4j-properties-examples/)</a>	852k
<a href="#">JSF 2.0 Tutorial (/tutorials/jsf-2-0-tutorials/)</a>	1.1m
<a href="#">How to convert Java object to / from JSON (Gson) (/java/how-do-convert-java-object-to-from-json-format-gson-api/)</a>	1.2m
<a href="#">Java XML Tutorial (/tutorials/java-xml-tutorials/)</a>	767k



2.2m

[Java - How to get current date time \(/java/java-how-to-get-current-date-time-date-and-calender/\)](#)

1.6m

[Hibernate Tutorial \(/tutorials/hibernate-tutorials/\)](#)

1.3m

[JAX-RS Tutorial \(/tutorials/jax-rs-tutorials/\)](#)

708k

[Spring Auto-Wiring Beans with @Autowired annotation \(/spring/spring-auto-wiring-beans-with-autowired-annotation/\)](#)

## Leave a Reply



Join the discussion

Sort by: newest | oldest | **most voted**

Guest

Thomas Sangjoon Kim

Hi mkyong, you suggest `@RequestMapping(method = RequestMethod.GET)` for `formBackingObject`. But I found `formBackingObject` is called by GET and POST (if that command not is in session). So `@RequestMapping` might not enough to replace `formBackingObject`. in that case we can use `@ModelAttribute`. `@ModelAttribute("numberList")` public Customer bindCustomer(HttpServletRequest request ... ) { Customer cust = new Customer(); cust.setFavFramework(new String []{"Spring MVC"}); return cust; } `@RequestMapping(method = RequestMethod.GET)` public String initForm(ModelMap model, `@ModelAttribute(customer)` Cust cust){ //command object model.addAttribute("customer", cust); //return form view return "CustomerForm"; }

0 REPLY

🕒 1 year 8 months ago



Guest

mqsj

What if in the submission form, you collect information about two entities (information about a shop(name, country, city, address and phone number) and information about the shop's manager (id, name, surname, address, phone number and date of birth) ).

Is it possible to cast the command object in two beans? I guess it's possible, but how is it internally done?

0 REPLY

🕒 2 years 4 months ago



Guest

Yogesh Bhosale

java.lang.IllegalStateException: Neither BindingResult nor plain target object for bean name 'userEditCommand' available as request attribute

0 REPLY

🕒 2 years 11 months ago



Guest

Sanjay Patil

Hi Mkyong, thanks for the article. it was really useful.

I had issue replacing `formBackingObject` method in my case so I created a `ModelAttributeMethod` annotating that with my command name and it worked for me.

Do you see any issues here ?

Cheers,  
Sanjay Patil

0 REPLY

🕒 2 years 11 months ago



Guest

neha agrawal

Hi Mkyong,  
Thanks for your tutorials!! It has helped me a lot to learn Spring. Though I have one doubt in this tutorial. Why are you using custom validator in this example and not jsr-303 validator? Is there any limitation with JSR-303 validator?  
Also I don't understand the use of `@InitBinder` in this example as you haven't used date field in your model/jsp. I also read somewhere that you register your custom Validator in the Controller using `@InitBinder`. But I don't see that you are registering it. Can you please clear by doubts. I would really appreciate it. Thanks!!

Regards,  
Neha



🕒 3 years 6 months ago ^



Victor

That was for elaboration purpose , u can skip that part.

Guest



🕒 2 years 2 months ago



safe

Hello,

Guest

I asked almost the same question just one week ago (see below) but it seems to be a site left abandoned.

No reaction. It's not cool !

Sorry.



🕒 3 years 6 months ago



safeghost

Hi,

Guest

Good job, thank you.

However, i didn't understand the role of @InitBinder in this example since you never invoked a random date issue in all your tutorials.

Can you explain what is the roler of your @InitBinder with date issue and its role in a general manner since i google it and it's not yet clear for me.

Regards.



🕒 3 years 7 months ago ^



Victor

That was for elaboration purpose , u can skip that part.

Guest



🕒 2 years 2 months ago



abhishek

hello

Guest

in apring annotation based form handling how do i identify that the value from the form will be mapped to CCustomer pojo



🕒 3 years 8 months ago ^



Victor

See the constructor of the controller.

Guest



🕒 2 years 2 months ago



kumar

thank u

Guest

i'm facing problem with including .js and .css file in jsp please help on this



🕒 3 years 10 months ago ^



VP

Place your resources (static files ) folder inside webapp ( but not inside WEB-INF).

Guest

For example, if you want to include js/\*

Go to your dispatcher servlet xml, add the following code after the line where you've declared the base-package.



🕒 3 years 7 months ago ^



VP

```
<mvc:resources mapping="/js/**" location="/js/" />
```

Guest



🕒 3 years 7 months ago



VP

;

Guest



🕒 3 years 7 months ago



Fateme

You show differences very good, thanks for this good article.  
good luck

Guest



🕒 3 years 10 months ago



rajesh

Dear sir,  
i got it with the knowledge given by you with the previous tutorials i rectified that error. thanks for your posts.

Guest



🕒 3 years 10 months ago



rajesh

Yhanks mkyong. Your tutorials are helped me a lot. For this tutorial(annotation based form validation) i did as you said above but i am getting error as :-

Guest

```
org.springframework.web.util.NestedServletException: Request processing failed; nested exception is java.lang.IllegalStateException:
Neither BindingResult nor plain target object for bean name 'customerForm' available as request attribute
```

root cause

```
java.lang.IllegalStateException: Neither BindingResult nor plain target object for bean name 'customerForm' available as request
attribute
```

note The full stack traces of the exception and its root causes are available in the GlassFish Server Open Source Edition 3.1.2 logs.



🕒 3 years 10 months ago



Shoaib Chikate

When i am using the Validator  
it is wroking for blank space

Guest

but not for int value....as i want the value int value min 4 and max 12

And i m using @Min and @Max annotation as

```
public class Student{
```

```
@Id
```

```
@GeneratedValue(strategy=GenerationType.Identity)
```

```
private int studentId
```

```
@NotBlank(message="Please enter your name")
```

```
private String studentName;
```

```
@Min(value="18",message="Age cannot be less 18")
@Max(value="99",message="Age cannot be greater than 99")
private int age;

//setters and getters

Validation on studentName is working but not on age
Any specific reason??

Reply your answer
I will be waiting

}
```



3 years 11 months ago ^



Good Learner

This validation will work only for String elements. Not for int.

Guest



2 years 8 months ago



replica hermes new

([http://www.luxurydreamhermes.com/hermes-new-bags-c-10\\_28.html](http://www.luxurydreamhermes.com/hermes-new-bags-c-10_28.html))

Guest

Hello, your articles here Spring MVC form handling annotation example to write well, thanks for sharing!



4 years 26 days ago



Thiru

Hi Mkyong,

Guest

I copied the same code, but i'm unable to access the MVCForm using this following URL.  
Please help me out to resolve this issue.

<http://localhost:8080/SpringMVCForm/customer.htm> (<http://localhost:8080/SpringMVCForm/customer.htm>)

Thanks 'n' Regards,  
Thirunavukkarasu



4 years 1 month ago ^



Docaohuynh

U can see SpringMVC in file pom.xml  
and used this link  
<http://localhost:8080/SpringMVC/customer.htm> (<http://localhost:8080/SpringMVC/customer.htm>)

Guest

If you want to used this link <http://localhost:8080/SpringMVCForm/customer.htm>  
(<http://localhost:8080/SpringMVCForm/customer.htm>)  
You must change value in SpringMVC to SpringMVCForm



3 years 9 months ago ^



Victor

Adding some information: just use your war File name in URL as

Guest

[localhost:8080/customer.htm](http://localhost:8080/customer.htm)



2 years 2 months ago



at work (<http://weekdayworker.blogspot.co.uk/>)

Guest

An intriguing discussion is worth comment. I believe that you ought to publish more about this issue, it may not be a taboo subject but usually people do not discuss these issues. To the next! All the best!!



🕒 4 years 1 month ago



Simple navigation through pages spring |

BlogoSfera

(<http://www.blogosfera.co.uk/2013/04/simple-navigation-through-pages-spring/>) [...]

[...] I am a Spring beginner and was recently looking at this tutorial <http://www.mkyong.com/spring-mvc/spring-mvc-form-handling-annotation-example/> (<http://www.mkyong.com/spring-mvc/spring-mvc-form-handling-annotation-example/>) [...]



🕒 4 years 1 month ago



Simple navigation through pages spring - How-

To Video (<http://howtovideo.eu/simple-navigation-through-pages-spring/>)

<http://www.mkyong.com/spring-mvc/spring-mvc-form-handling-annotation-example/> [...]

[...] I am a Spring beginner and was recently looking at this tutorial <http://www.mkyong.com/spring-mvc/spring-mvc-form-handling-annotation-example/> (<http://www.mkyong.com/spring-mvc/spring-mvc-form-handling-annotation-example/>) [...]



🕒 4 years 1 month ago



Guest

chandan modi

you are real hero in java in internet till now i have read.  
really appreciate u a lot , to write such a fantastic example.  
thankx a lot mkyong.



🕒 4 years 5 months ago



Guest

kunal

If u have any good tutorial on spring JPA pls share it



🕒 4 years 8 months ago



Guest

kunal

good one...



🕒 4 years 8 months ago



Guest

samuel

Hi Mkyong,  
Could you add tutorials on Spring JPA?

Thanks,  
Samuel



🕒 4 years 8 months ago



Guest

Tapas

Thank you very much.It was very useful blog I have came across so far.



🕒 4 years 9 months ago



Guest

Balachandar Chinnaraj

Hi Mkyong,  
Your example /tutorials are very nice and very useful. The only problem i am facing is, Not able identify which one is Spring 3.0 except one /two examples( Top).

If possible kindly mention the spring version in each and every example, So that it will be more helpful for me and others also.  
If you don't mind, Can you kindly send me Professional (Wrox publication) pdf for spring3.0 version.  
Look forward your positive reply.

Thanks & Regards,  
Balachandar.C  
651 315 2180



🕒 4 years 9 months ago ^



Victor

For Spring version see the POM.xml file

Guest



🕒 2 years 2 months ago



Rady

Hi

Guest

<http://localhost:8080/SpringMVC/customer.htm> (<http://localhost:8080/SpringMVC/customer.htm>)

It displays nothing , just see something at console:

15:04:02,937 WARN [org.springframework.web.servlet.PageNotFound] (http-localhost-127.0.0.1-8080-1) No mapping found for HTTP request with URI [/SpringMVC/customer.htm] in DispatcherServlet with name 'mvc-dispatcher'



🕒 4 years 10 months ago ^



Rady

Guest

Because of JBoss AS 7.1 doesn't work with Spring 2.5.6.  
(it should upgrade to Spring 3.0.5.RELEASE).

Spring 2.5.6 works with JBoss AS 4.2.



🕒 4 years 10 months ago ^



mkyong (<http://www.mkyong.com>)

Thanks for your extra info.  
(<http://www.mkyong.com/author/mkyong/>)

Author



🕒 4 years 10 months ago ^



veeru

Guest

hi tell me how to configure springmvc+hibernate(hibernetatemplateclass) for crud operations in spring 3.0



🕒 4 years 4 days ago



Spring Guy

Guest

<http://static.springsource.org/spring/docs/3.0.0.RC3/reference/html/ch05s07.html>  
(<http://static.springsource.org/spring/docs/3.0.0.RC3/reference/html/ch05s07.html>) describes configuring annotation based auto-executing validators. It also emphasizes the fact that Spring 3 is JSR-303 compliant in and of itself without the need for Hibernate validators. This is relatively cutting edge so it's understandable that the information is not covered here, but it's important knowledge. You no longer have to manually execute validators to do annotation based config if you use the proper mechanisms covered in that spring article.



🕒 4 years 11 months ago



JAVA: Registration form with Spring « since 06  
– 28 – 2011

(<http://samjdev.wordpress.com/2012/04/07/java-registration-form-with-spring/>)

[...] Mkyong Share this:TwitterFacebookLike this:LikeBe the first to like this [...]



🕒 5 years 1 month ago



Shrileckha

-Shrileckha Chaithanya

Guest



Thank you. This post is very useful. I understood the user of @ModelAttribute

🕒 5 years 1 month ago



John

Guest

Thanks for the article. It really helped me put some the annotation-driven Spring pieces together. There seems to be one missing piece for me... If I have a controller method similar to your processSubmit() method, where HTTP Parameters will be bound to available setters in the data object (Customer in your example), is there a way to explicitly supply that Consumer object as a Spring bean? For example, I might want to configure a Consumer object in a Spring XML file (so I can inject values into some members via XML where those members won't be populated from HTTP parameters).... Read more »



🕒 5 years 3 months ago



Maruf Hassan

Guest

Thanks a lot...You save my life...



🕒 5 years 5 months ago



Spring MVC 2 vs Spring MVC 3 form handling |

Imran Tariq's Blog

(http://www.imrantariq.com/blog/?p=667)


[...] Reference Posted by Imran Computer, Java, Java, web Subscribe to RSS feed [...]



🕒 5 years 6 months ago

Load More Comments

(http://www.mkyong.com/spring-mvc/spring-mvc-form-handling-annotation-example/#!/parentId=72993)



Mkyong

53,505 likes

Like Page

Sign Up

Be the first of your friends to like this

