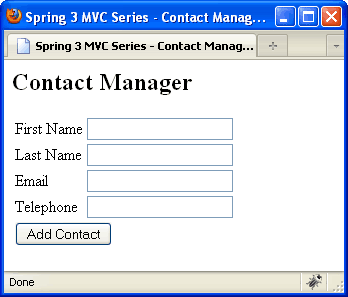
Our Goal

Our goal is to create basic Contact Manager application. This app will have a form to take contact details from user. For now we will just print the details in logs. We will learn how to capture the form data in Spring 3 MVC.



Getting Started

Let us add the contact form to our Spring 3 MVC Hello World application. Open the index.jsp file and change it to following:

*File: WebContent/index.jsp*

|  |
| --- |
| <jsp:forward page="contacts.html"></jsp:forward> |
| The above code will just redirect the user to contacts.html page. The View- contact.jsp |

Create a JSP file that will display Contact form to our users.  
File: /WebContent/WEB-INF/jsp/contact.jsp

|  |
| --- |
| <%@taglib uri="http://www.springframework.org/tags/form" prefix="form"%>  <html>  <head>      <title>Spring 3 MVC Series - Contact Manager</title>  </head>  <body>  <h2>Contact Manager</h2>  <form:form method="post" action="addContact.html">        <table>      <tr>          <td><form:label path="firstname">First Name</form:label></td>          <td><form:input path="firstname" /></td>      </tr>      <tr>          <td><form:label path="lastname">Last Name</form:label></td>          <td><form:input path="lastname" /></td>      </tr>      <tr>          <td><form:label path="lastname">Email</form:label></td>          <td><form:input path="email" /></td>      </tr>      <tr>          <td><form:label path="lastname">Telephone</form:label></td>          <td><form:input path="telephone" /></td>      </tr>      <tr>          <td colspan="2">              <input type="submit" value="Add Contact"/>          </td>      </tr>  </table>    </form:form>  </body>  </html> |

Here in above JSP, we have displayed a form. Note that the form is getting submitted to **addContact.html** page.

## Adding Form and Controller in Spring 3

We will now add the logic in Spring 3 to display the form and fetch the values from it. For that we will create two java files. First the Contact.java which is nothing but the form to display/retrieve data from screen and second the ContactController.java which is the spring controller class.

*File: net.spring3.form.Contact*

|  |
| --- |
| package net.spring3.form;    public class Contact {      private String firstname;      private String lastname;      private String email;      private String telephone;        //.. getter and setter for all above fields.    } |

The above file is the contact form which holds the data from screen. Note that I haven’t showed the getter and setter methods. You can generate these methods by pressiong **Alt + Shift + S, R**.

File: net.spring3.controller.ContactController

package net.spring3.controller;

import net.spring3.form.Contact;

import org.springframework.stereotype.Controller;

import org.springframework.validation.BindingResult;

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.annotation.SessionAttributes;

import org.springframework.web.servlet.ModelAndView;

@Controller

@SessionAttributes

public class ContactController {

    @RequestMapping(value = "/addContact", method = RequestMethod.POST)

    public String addContact(@ModelAttribute("contact")

                            Contact contact, BindingResult result) {

        System.out.println("First Name:" + contact.getFirstname() +

                    "Last Name:" + contact.getLastname());

        return "redirect:contacts.html";

    }

@RequestMapping("/contacts")

    public ModelAndView showContacts() {

        return new ModelAndView("contact", "command", new Contact());

    }

}

In above controller class, note that we have created two methods with Request Mapping **/contacts** and**/addContact**. The method showContacts() will be called when user request for a url contacts.html. This method will render a model with name “contact”. Note that in the ModelAndView object we have passed a blank Contact object with name “command”. The spring framework expects an object with name command if you are using <form:form> in your JSP file.

Also note that in method addContact() we have annotated this method with RequestMapping and passed an attribute method=”RequestMethod.POST”. Thus the method will be called only when user generates a POST method request to the url /addContact.html. We have annotated the argument Contact with annotation @ModelAttribute. This will binds the data from request to the object Contact. In this method we just have printed values of Firstname and Lastname and redirected the view to cotnacts.html.

Configuration File:

**<?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:p="http://www.springframework.org/schema/p"**

**xmlns:context="http://www.springframework.org/schema/context"**

**xsi:schemaLocation="http://www.springframework.org/schema/beans**

**http://www.springframework.org/schema/beans/spring-beans-3.0.xsd**

**http://www.springframework.org/schema/context**

**http://www.springframework.org/schema/context/spring-context-3.0.xsd">**

**<context:component-scan**

**base-package="net.spring3.controller" />**

**<bean id="viewResolver"**

**class="org.springframework.web.servlet.view.UrlBasedViewResolver">**

**<property name="viewClass"**

**value="org.springframework.web.servlet.view.JstlView" />**

**<property name="prefix" value="/WEB-INF/jsp/" />**

**<property name="suffix" value=".jsp" />**

**</bean>**

**</beans>**

**<?xml version="1.0" encoding="UTF-8"?>**

**<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"**

**xmlns="http://java.sun.com/xml/ns/javaee"**

**xmlns:web="http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd"**

**xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd"**

**id="WebApp\_ID" version="2.5">**

**<display-name>Spring3MVC</display-name>**

**<welcome-file-list>**

**<welcome-file>index.jsp</welcome-file>**

**</welcome-file-list>**

**<servlet>**

**<servlet-name>spring</servlet-name>**

**<servlet-class>**

**org.springframework.web.servlet.DispatcherServlet**

**</servlet-class>**

**<load-on-startup>1</load-on-startup>**

**</servlet>**

**<servlet-mapping>**

**<servlet-name>spring</servlet-name>**

**<url-pattern>\*.html</url-pattern>**

**</servlet-mapping>**

**</web-app>**