



Java Code Geeks
JAVA 2 JAVA DEVELOPERS RESOURCE CENTER

Java Persistence Tools

OpenJPA, Toplink, Hibernate Suppt No Lock-in, Eclipse-Based



You are here: [Home](#) | [Enterprise Java](#) | [Ajax with Spring MVC 3 using Annotations and JQuery](#)



About **Rahul Mondal**



Ajax with Spring MVC 3 using Annotations and JQuery

by Rahul Mondal on February 27th, 2012 | Filed in: [Enterprise Java](#) Tags: [jQuery](#), [Spring](#), [Spring MVC](#)

InterServer.net
www.interserver.net
15 YEARS HOSTING EXCELLENCE
UNLIMITED WEB HOSTING
BUY NOW \$0.01 for first month

Its always been fun for me to work with Ajax! Is not it ? I will make it easy for you to use Ajax with **Spring MVC 3 and JQuery**. This post will illustrate you how to use Ajax in real life practices of industrial coding. As usual, we will take an practical example of Ajax in Spring MVC 3 framework and will implement it and I will make the implementation easy by make you understand the topic.

Let us see what is our example's requirement and how Spring MVC 3 Ajax facility will fulfill it :

In our example, we will make a list of students with name and highest education level, to send the list to the placement office so that the students can get chance. We will make the "Add Student Form" available to the student online so that they can submit their name online and get registered. As a lot of students will use the system, so the

performance of the system may very much low. To increase to performance of the web application we will use Ajax with Spring MVC 3 Framework and JQuery.

Following steps we have to go through to implement our example :

1. First of all, we will create a domain class (User.java) that will hold the value of student information.
2. After that we will create our controller class (UserController.java) to handle HTTP request. Our controller will handle three types of requests. First, to show the "Add Student Form", second to handle Ajax request came from "Add Student Form" and add the students to a list, third to show the student information as a list.
3. Then, we will create jsp page (AddUser.jsp) to show "Add Student Form" that will use JQuery to send Ajax request to the Spring MVC Controller. The jsp will also confirm to the user that Student has been added to the list.
4. Then, we will create a jsp (ShowUsers.jsp) that will list all users in the list.

User.java

User.java has two properties name and education to store the student information. Following is the code of User.java :

```
1 package com.raistudies.domain;
2
3 public class User {
4
5     private String name = null;
6     private String education = null;
7     // Getter and Setter are omitted for making the code short
8 }
```

UserController.java

Controllers has three method to handle three request urls. "showForm" method handle the request for showing the form to the user. Bellow code shows the UserController.java :

```
01 package com.raistudies.controllers;
02
03 import java.util.ArrayList;
04 import java.util.List;
05
06 import org.springframework.stereotype.Controller;
07 import org.springframework.ui.ModelMap;
08 import org.springframework.validation.BindingResult;
09 import org.springframework.web.bind.annotation.ModelAttribute;
10 import org.springframework.web.bind.annotation.RequestMapping;
11 import org.springframework.web.bind.annotation.RequestMethod;
12 import org.springframework.web.bind.annotation.ResponseBody;
13
14 import com.raistudies.domain.User;
15
16 @Controller
17 public class UserController {
```

BuySellAds
A better way to sell ads
Dashboard
Earnings
\$110.00
\$7,268.70

Take a look at how
code runs - in pro
Free Monitoring
APPDYNAMICS

Newsletter

73722 insiders are already en
updates and complimentary whi
Join them now to gain **ex**
to the latest news in the Java w
insights about Android, Scala, G
related technologies.

Email address:

Join Us



With **1,04**
unique vis
authors w
the top Ja
around. C
the looko
encourag

If you have a blog with unique a
content then you should check c
partners program. You can also
for Java Code Geeks and hone

Recent Jobs

Software Development Engineer
Seattle, WA

Java Developer
Atlanta, GA

Java Developer
San Francisco, CA

Java Developer
San Bruno, CA

Java Developer
Mount Laurel, NJ

[View All](#)

```

18 private List<User> userList = new ArrayList<User>();
19
20 @RequestMapping(value="/AddUser.htm",method=RequestMethod.GET)
21 public String showForm(){
22     return "AddUser";
23 }
24
25 @RequestMapping(value="/AddUser.htm",method=RequestMethod.POST)
26 public @ResponseBody String addUser(@ModelAttribute(value="user") User user, BindingResult result )
27 {
28     String returnText;
29     if(!result.hasErrors()){
30         userList.add(user);
31         returnText = "User has been added to the list. Total number of users are " +
32         userList.size();
33     }else{
34         returnText = "Sorry, an error has occur. User has not been added to list.";
35     }
36     return returnText;
37 }
38
39 @RequestMapping(value="/ShowUsers.htm")
40 public String showUsers(ModelMap model){
41     model.addAttribute("Users", userList);
42     return "ShowUsers";
43 }
44 }

```

"addUsers" is same as the controller method that handle form expect that it also contain annotation **@ResponseBody**, which tells **Spring MVC** that the String returned by the method is the response to the request, it does not have to find view for this string. So the returning String will be send back to the browser as response and hence the Ajax request will work. "showUsers" method is used to show the list of the students to the user.

AddUser.jsp

AddUser.jsp contain a simple form to collect information about the student and uses JQuery JavaScript framework to generate Ajax request to the server. Following is the code in AddUser.jsp :

```

01 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
02 pageEncoding="ISO-8859-1"%>
03 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
04 <html>
05 <head>
06 <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
07 <title>Add Users using ajax</title>
08 <script src="/AjaxWithSpringMVC2Annotations/js/jquery.js"></script>
09 <script type="text/javascript">
10 function doAjaxPost() {
11     // get the form values
12     var name = $('#name').val();
13     var education = $('#education').val();
14
15     $.ajax({
16         type: "POST",
17         url: "/AjaxWithSpringMVC2Annotations/AddUser.htm",
18         data: "name=" + name + "&education=" + education,
19         success: function(response){
20             // we have the response
21             $('#info').html(response);
22             $('#name').val('');
23             $('#education').val('');
24         },
25         error: function(e){
26             alert('Error: ' + e);
27         }
28     });
29 }
30 </script>
31 </head>
32 <body>
33 <h1>Add Users using Ajax .....</h1>
34 <table>
35 <tr><td>Enter your name : </td><td> <input type="text" id="name"><br/></td></tr>
36 <tr><td>Education : </td><td> <input type="text" id="education"><br/></td></tr>
37 <tr><td colspan="2"><input type="button" value="Add Users" onclick="doAjaxPost()"><br/>
38 <tr><td colspan="2"><div id="info" style="color: green;"></div></td></tr>
39 </table>
40 <a href="/AjaxWithSpringMVC2Annotations/ShowUsers.htm">Show All Users</a>
41 </body>
42 </html>

```

You may be little bit confused if you are not aware of JQuery. Here is the explanation of the JQuery code :

1. **var name = \$('#name').val();** :- here the \$ is JQuery selector that uses to select any node in HTML whose identifier is passed as argument. If the identifier is a prefix with # that means it is a id of the HTML node. Here, **\$('#name').val()** contains the value of the HTML node whose is "name". The text box in which user will enter her/his name is with is as name. so java script variable name will contain the name of the user.
2. **\$.ajax()** :- It is the method in \$ variable of JQuery to call Ajax. It has five arguments here. First of all **"type"** which indicated the request type of Ajax. It can be POST or GET. Then, **"url"** which indicates the url to be hit of Ajax submission. **"data"** will contain the

raw data to be sent to the server. "success" will contain the function code that has to be call if the request get success and server sends an response to the browser. "error" will contain the function code that has to be call if the request get any error.

3. `$("#info").html(response);` :- will set the response of the server in to the div. In this way "Hello" + name will be shown in the div whose id is "info".

ShowUsers.jsp

Following are the code in ShowUsers.jsp to print all student information from a ArrayList to jsp page :

```
01 <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
02 pageEncoding="ISO-8859-1"%>
03 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
04 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
05 <html>
06 <head>
07 <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
08 <title>Users Added using Ajax</title>
09 </head>
10 <body style="color: green;">
11 The following are the users added in the list :<br>
12 <ul>
13 <c:forEach items="${Users}" var="user">
14 <li>Name : <c:out value="${user.name}" />; Education : <c:out
value="${user.education}" />
15 </c:forEach>
16 </ul>
17 </body>
18 </html>
```

Here, we have used JSTL core taglib to iterate through the ArrayList and show every value in browser.

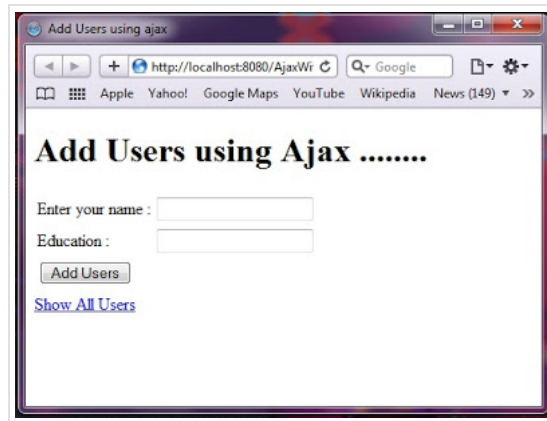
- `<c:forEach items="${Users}" var="user">` : tag is used to iterate through the ArrayList. Property "items" is used to define the bean on which the List object has been stored, so `items="${Users}"` says that the users list is present in "Users" bean. "var" attribute says the name of the variable in which each user will be stored.
- `<c:out value="${user.name}" />` : As, a single user will be stored in variable name "user" so to print the name property in User object we use `${user.name}`.

app-config.xml

Our Spring MVC configuration file should be able to handle annotation driven controllers. The configuration are as follows :

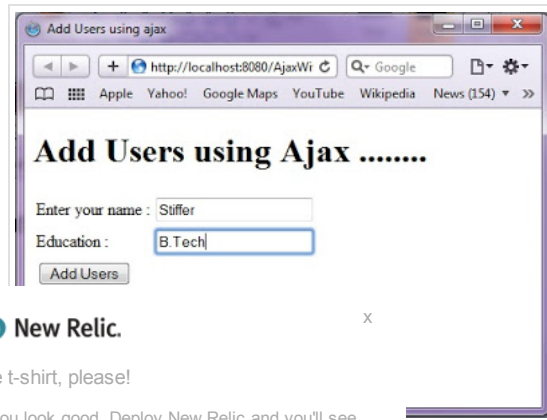
```
01 <?xml version="1.0" encoding="UTF-8"?>
02 <beans xmlns="http://www.springframework.org/schema/beans"
03 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
04 xmlns:context="http://www.springframework.org/schema/context"
05 xmlns:mvc="http://www.springframework.org/schema/mvc"
06 xsi:schemaLocation="
07 http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-
beans-3.0.xsd
08 http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-
context-3.0.xsd
09 http://www.springframework.org/schema/mvc http://www.springframework.org/schema/mvc/spring-mvc-
3.0.xsd">
10
11 <!-- Scans the classpath of this application for @Components to deploy as beans -->
12 <context:component-scan base-package="com.raistudies" />
13
14 <!-- Configures the @Controller programming model -->
15 <mvc:annotation-driven />
16
17 <!-- Resolves view names to protected .jsp resources within the /WEB-INF/views directory -->
18 <bean id="viewResolver" class="org.springframework.web.servlet.view.InternalResourceViewResolver">
19 <property name="prefix" value="/WEB-INF/jsp/" />
20 <property name="suffix" value=".jsp" />
21 </bean>
22
23 </beans>
```

Deploy the war file to tomcat 6 and hit the url in browser, you will get following page displayed :



Ajax Form Using Spring MVC 3

Fill student information :



When your software looks good, you look good. Deploy New Relic and you'll see what we mean.

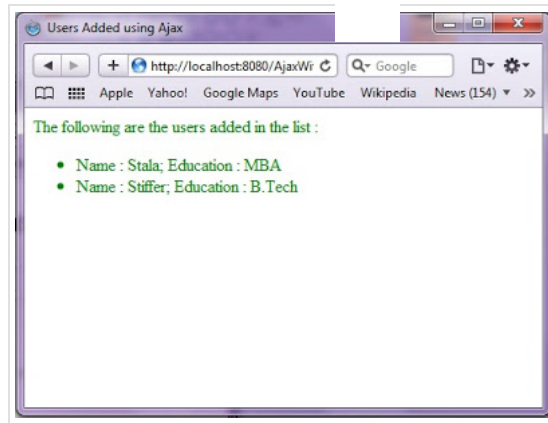
Ajax Filled Form Using Spring MVC 3

After that click on "Add Users" button, you will get message that user has been added to the list :



Ajax Form Submission Confirmation Spring MVC 3

To show all student added to the list click on the button "Show All Users", you will get following page :



Show All Users Spring MVC 3

That is all from Ajax using Spring MVC 3 and JQuery. You can download source from bellow link.

Source : [Download](#)

Reference: [Ajax with Spring MVC 3 using Annotations and JQuery](#) from our JCG partner Rahul Mondal at the [Rai Studies](#) blog.

Do you want to know how to develop your skillset to become a **Java Rockstar**?

Subscribe to our newsletter to start Rocking right now!

To get you started we give you our best selling eBooks for **FREE!**

1. JPA Mini Book
2. JVM Troubleshooting Guide
3. JUnit Tutorial for Unit Testing
4. Java Annotations Tutorial
5. Java Interview Questions
6. Spring Interview Questions
7. Android UI Design

and many more

Email address:

Java Persistence Tools

OpenJPA, Toplink, Hibernate
Suppt No Lock-in, Eclipse-
Based



6 Responses to "Ajax with Spring MVC 3 using Annotations and JQuery"



Marc Schipperheyn

March 2nd, 2012 at 2:02 pm

I would recommend always returning

1 | public atResponseBoy Map

(The form replaces the at symbol automatically through some Twitter parser)so that you have more flexibility in the result and define some standard keys in this response for Forms that you can code against in the frontend,

such as an errors key, which will contain the binding result errors.

and have some kind of general error handler that you put in a BaseController class that you extend. This will prevent stacktraces from showing up in your json messages.

```

1  atExceptionHandler(Exception.class)
2  public atResponseBody Map handleException(Exception e, HttpServletRequest request,
3  HttpServletResponse response) {
4      response.setStatus(HttpServletResponse.SC_INTERNAL_SERVER_ERROR);
5      Map errors = new HashMap();
6      errors.put("errors",new Object[]{new
7  ObjectError("object",messages.getMessage("error.servererror.500",null,
8  RequestUtils.getLocale(request))));
9      return errors;
10 }
```

Also, you will have to define handler mappings in such a way as to prevent a cache header from being generated for those json requests that shouldn't be cached.

JQuery is excellent but personally, I prefer YUI.

BTW this is java "code" geeks, right. What kind of response form is this?!

[Reply](#)



Kalyan Dasika

March 5th, 2012 at 10:50 pm

There should also be an emphasis on a sound domain model. Do you really want a User class to hold Student information? A student can be a user, but not all users are students..

[Reply](#)



hasini

September 18th, 2012 at 2:25 am

it is not working in IE8..

[Reply](#)



Binh Thanh Nguyen

June 11th, 2014 at 7:08 am

Thanks, nice tips

[Reply](#)



Parth

July 9th, 2014 at 12:22 pm

Nice tutorial

[Reply](#)



ray jhon

July 31st, 2014 at 12:14 pm

Plz tell me what's wrong in this code:——

IN JSP PAGE:——trying to passsing req to code...

```

$("#id_complaint").change(function(){
var v_ttType = $("#id_complaint").val();
alert(v_ttType);
$.getJSON("/attributeListcreateTT.*",{TTType:v_ttType,ajax:'true'}, function(resp)
{
alert('in json');
for(var i=0;i<resp.length;i++)
{
alert(resp[i].ATTRIBUTE);
alert(resp[i].OUTCOME);
}
```

```

}
});
});
});
/-----in java
public String attributeList()
{
//templist2=(ArrayList)CnmRefData.getTTAttributes(prod, ttType);
//return templist2;
System.out.println("in attributeList.....RequestCame....");
try{
JSONObject object = null;
JSONArray array = null;
List jsonAttrList = new ArrayList();
String ttType = httpServletRequest.getParameter("TTType");
System.out.println("In *>>> :"+ttType);
templist2=(ArrayList)CnmRefData.getTTAttributes("RON", ttType);
System.out.println(":::"+templist2.size());
Iterator itr = templist2.iterator();
while(itr.hasNext()){
CNMTTAttribute objCNMTTAttribute = (CNMTTAttribute)itr.next();
object = new JSONObject();
object.put("ATTRIBUTE", objCNMTTAttribute.getAttribute());
object.put("OUTCOME",objCNMTTAttribute.getValidateOutcome());
jsonAttrList.add(object);
}
array = JSONArray.fromObject(jsonAttrList);
response.setHeader("Cache-Control","no-cache");
response.getWriter().write(array.toString());
}catch(NullPointerException e){
addActionError(e.getMessage());
e.printStackTrace();
}catch(RuntimeException e){
addActionError(e.getMessage());
e.printStackTrace();
}catch(OutOfMemoryError e){
addActionError(e.getMessage());
e.printStackTrace();
}catch(Exception e){
addActionError(e.getMessage());
e.printStackTrace();
}
return null;
}
}
plz tell me what's wrong in this... stuck here...

```

[Reply](#)

Leave a Reply

Name (Required)

Mail (will not be published) (Required)

Website

× two = 12

☐ Notify me of followup comments via e-mail. You can also [subscribe](#) without commenting.

☒ Sign me up for the newsletter!

Knowledge Base

[Academy](#)
[Examples](#)
[Resources](#)
[Tutorials](#)
[Whitepapers](#)

Partners

[Mkyong](#)

The Code Geeks Network

[Java Code Geeks](#)
[.NET Code Geeks](#)
[Web Code Geeks](#)

Hall Of Fame

[“Android Full Application Tutorial” series](#)
[GWT 2 Spring 3 JPA 2 Hibernate 3.5 Tutorial](#)
[Advantages and Disadvantages of Cloud Computing – Cloud computing pros and cons](#)
[Android Google Maps Tutorial](#)
[Android Location Based Services Application – GPS location](#)
[11 Online Learning websites that you should check out](#)
[Java Best Practices – Vector vs ArrayList vs HashSet](#)
[Android JSON Parsing with Gson Tutorial](#)
[Android Quick Preferences Tutorial](#)
[Difference between Comparator and Comparable in Java](#)

About Java Code Geeks

JCGs (Java Code Geeks) is an independent online community creating the ultimate Java to Java developers resource center. JCGs serve the Java, SOA, Agile and communities with daily news written by domain experts, articles, reviews, announcements, code snippets and open source projects.

Java Code Geeks and all content copyright © 2010-2015, Exelixis Media Ltd | [Terms of Use](#) | [Privacy Policy](#) | [Contact](#)
 All trademarks and registered trademarks appearing on Java Code Geeks are the property of their respective owners.
 Java is a trademark or registered trademark of Oracle Corporation in the United States and other countries.
 Java Code Geeks is not connected to Oracle Corporation and is not sponsored by Oracle Corporation.