Passing different kinds of data/values

from

Contrller comps to View Comps using Data Rendering Technique

a) Passing simple values

b) Passing collections and arrays

c) Passing Model class obj

d) Passing collection of Model class objects

c) Passing Model class obj

Code in Controller

@GetMapping("/report")

public String sendData(Map<String, Object> map) { //create Model class object with data

}

Employee emp=new Employee();

emp.setEno(1001); emp.setEname("raja"); emp.setSalary(90000.0); emp.setVaccinated(true); //add Model class object to Model attribute map.put("empInfo",emp);

//return LVN

return "show_report";

To work with JSTL core taglibray tags in spring 3.x and tomcat 10.x setup

===============

:==========:

step1) add the following dependency in pom.xml

======================

<!-- https://mvnrepository.com/artifact/org.eclipse.jetty/apache-jstl --> <dependency>

<groupId>org.eclipse.jetty</groupId>

<artifactId>apache-jstl</artifactId>

<version>11.0.0</version>

</dependency>

step2) Import jstl Taglibary in jsp page by specifying its taglib uri and also specify the prefix

JSTL gives collection of jsp taglibrary where each taglibrary is having

set of readymade jsp tags and every Jsp taglibrary is identified with its taglib uri..

JSTL taglibrary

taglib uri

recomanded prefix

Core SQL

http://java.sun.com/jsp/jstl/core

C

we need to import the jsp tag library by using <%@taglib> tag

Formatting Xml

Functions

http://java.sun.com/jsp/jstl/sql http://java.sun.com/jsp/jstl/formatting http://java.sun.com/jsp/jstl/xml http://java.sun.com/jsp/jstl/functions

sql fmt

x

fn

note: To use tags of Jsp taglibary in any jsp page

note: JSTL tags allow us to develop the jsp pages as the Java Code Less Jsp pages

show_report.jsp

```
<%@ page isELIgnored="false" import="java.util.*"%>
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%> | importing JSTL core taglibrary
<h1 style="color:red;text-align:center">Result Page</h1>
<c:if test="${!empty seasonName}">
<h2 style="color:red;text-align:center"> Season name :: ${seasonName}</h2> </c:if>
```

Code in show_data.jsp (view comp)

```
<%@ page isELIgnored="false" import="java.util.*" %>
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %
<c:if test="${!empty empInfo}">
<b> Emp Information :: ${empInfo.eno }, ${empInfo.ename }, ${empinfo.salary },${empInfo.vaccinated} </b>
</c:if>
```

d) Passing collection of Model class objects

Code in controller class

```
@GetMapping("/report")
public String sendData(Map<String, Object> map) { //create Colelction of Model class objects
List<Employee> empsList=List.of(new Employee(1001,"raja", 90000.0,true),
new Employee(1002,"rajesh",80000.0,true), new Employee(1003,"mahesh", 90000.0,true));
<br><br><br>
<c:if test="${!empty name and !empty age}">
<b>SimpleModel attributes are :: ${name},${age}</b>
</c:if>
<br><br><br>
<c:if test="${!empty nickNames and !empty friends and !empty phoneNumbers and !empty idDetails}"> <b>
collection model attributes are</b> <%=Arrays.toString(((String[])request.getAttribute("nickNames"))) %>
<br>,${friends},<br>${phoneNumbers},<br>${idDetails}
</c:if>
<br><br><br><br><br>
<c:if test="${!empty empData}">
```

<b> Model class obj as Model attribute value :: ${empData} </b>

</c:if>

<br><br>

<a href="./">home</a>

}

//add Collecton of Model class object to Model attribute map.put("empsListInfo",empsList);

//return LVN

return "show_report";

show_data.jsp (data rendering)

<%@ page isELIgnored="false" import="java.util.*" %>

<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>

<c:choose>

<c:when test="${!empty empsListInfo }">

<table border="1" bgcolor="cyan" align="center">

<tr><th>eno </th> <th>ename </th> <th>salary </th> <th>IsVaccinated </th> </tr>

<c:forEach var="emp" items="${empsListInfo}" >

<tr>

<td>${emp.eno } </td>

<td>${emp.ename}</td>

<td>${emp.salary } </td>

<td>${emp.vaccinated } </td>

</tr>

</c:forEach>

</table>

<c:choose>, <c:when>, <c:otherwise> can be use as if --else if --else

or as switch.. case conditions

</c:when>

<c:otherwise>

<h1 style="color:red;text-align:center"> Employee not found </h1> </c:otherwise>

</c:choose>

note:: JSTL (Jsp Standard Tag library) is gives set of jsp tag libaries having bunch of jsp tags which can be used to minimize or avoid java code from jsp page i.e JSTL allows to make jsp pages as the

(java

scriptles

pages.

JSTL Taglibararies are :: core, sql, formatting, fuctions, xml

Y

localhost:2525/MVCBootProj3-AllDataRenderingOptions/report

eno

ename salary Is Vaccinated

1001 raja 90000.0 true

1002 rajesh 80000.0 true

1003 mahesh 90000.0 true

Data Binding

===▬▬▬▬▬

Data Rendering :: Passing data from controller comp to view comp through DispatcherServlet using (C ---- V) the SharedMemory Support (BindingAwareModelMap object)

Data Binding :: Passing data from View comp to Controller comp through DispatcherServlet (V ----C) (eg:: passing form data, passing req param values and etc..)

Two ways of Data Binding

===============

========

(Java bean class obj)

(a) Passing Form data atController class handler method as Model class obj using @ModelAttribute(-) annotation. (Form binding/Form Data binding)

values

(b) Passing hyperlink supplied addtional request param to

values

Binds all request param generated by form page to Model class /Java bean class obj (Form data binding)

handler method params through DispatcherServlet using @RequestParam annotation (param binding/param data binding)

(a) Passing Form data Controller class handler method as Model class obj

to

using @ModelAttribute(-) annotation.

(Form binding /Form Data binding)

DS

form page

In @Controller class

000

JAva Bean class obj (Student class obj)

Form Data binding

public String register(@ModelAttribute("stud") Student st){

}

(Here All request param values of form page are bound to Model class obj.. Bulk Data binding)

Dispatcher Servlet

In controller class

url?sno=101&sname=raja

Binds each req param value to one param of, handler method query String having req params (Request Param Binding) (Here each request param value is bound each parameter of handler method)

In Servlet grogramming, to write form data to Model class obj

we need to lots of manual work like reading form data from request obj (req.getParameter(-)) as param values and writing that data to Model class obj using setter methods. (lots of manual work)

101

public String findData(@RequestParam("sno") int sno, raja @RequestParam("sname") String name){

\=> In spring MVC/spring boot MVC we need not to do this work manually.. if we specify Model class name along with @ModelAttribute(-) in the controler class handler method

(new) (old) To bind form data (form page form comp values) to Model class /Command class obj we need to do the DS itself binds form data to Model class object and gives that Model class obj (old) following operations

(new)

data

as handler method arg value.

(a) count form page form comps and take same number of properties of required type in Model class/

(b) Match Form page comp names with Model class property names

(Java bean class)

(eg:: p String <method> (@ModelAttribute(-) classname

model

)

(a)

#1

(c) add getter setter methods those properties in in Model class/

(d) In Controller class Handler method take Model class type parameter having @ModelAttribute(-) annotation

Form page using html tags

=====

student_register.jsp (form page)

#3

<form action="register" method="POST">

student id :: <input type="text" name="sno"/><br> 1001

student name :: <input type="text" name="sname"/> <br> raja fill form page student address:: <input type="text" name="sadd"/><br> hyd student avg:: <input type="text" name="avg"/><br> 90.66 <input type="submit"

value="register">

DispatcherSevlet

RequestMappingHandlerMapping

#

(FrontController) #8

#5

#7

#4

SharedMemory for BindingAwareModelMap obj (request scope)

#9

#10

Student class obj(stud) snd: 1001 #11

(request scope)

Data binding Operation

</form>

#2

@Controller

sname:raja (binding) sadd: hyd

avg: 90.66

Request param Binding

note:: The form submitted data goes along with request in the form of request params either appended to url of query String (GET mode) or as request body content (POST mode)

data

note:: hyperlink generated request can carry limited amount of along with request as the query String appended to the href url (Always GET mode)

=> The request generated using url typed in the browser address bar is GET mode request

=> The request generated using hyperlink is GET mode request

=> The Form page generated request is GET mode request by default

=> The Form page can generate event POST mode request

(b)

//Model class

@Data (c)

public class Student{

private Integer sno;

private String sname;

private Float avg;

private Stirng sadd;

public class StudentRegistrationController{

@PostMapping("/register")

public String registerStudent(Map<String, Object>> map, #12 @ModelAttribute("stud") Student st){ (d)

...

//pass model class obj to service,DAO classes for registration and get results

// keep results a model attributes #13

}

Internal activities end to end

===

//return LVN

return "result"; #14 }

#1 & #2) Enduser fills up the form page and submits the form page

#3 &4) based on the action path kept in "action" attribute of <form>

tag the DS traps and takes the request

#5) DS hand overs the request to Handler Mapping comp

#6 &#7)Handler mapping comp searches form Handler method

in controller classes whose request path is "/regiser", PostMapping method and

gets registerStudent(-,-) method signature and relevant controller bean id

#8) HandlerMapping passes controller class bean id and handler method signature back to DS

#9) Based on Map<String,Object> of HandlerMethod signature the SharedMemory will be created-

#10) Based on @ModelAttribute("stud") Student st of handler method signature

it creates Student class obj as the model class obj (Best pratice) Student stud =new Student(); (Model attribute name as the obj name)

(DS)

out

note:: if @ModelAttribute is taking with attribute name like "stud" then

the Model class obj will be created having the class name as the attribute name (student) Student student=new Student();

the

(model class name as object name)

#11) DS reads form data as request param values and binds them to the matched Model class obj properties as shown below (it is called form data binding (or)

stud.setSno(Integer.parseInt(req.getParameter("sno")));

stud.setSname(req.getParameter("sname"));

stud.setSadd(req.getParameter("sadd");

request wrapping)

Data binding

from data to Model class obj

for @ModelAttribute Student st

stud.setAvg(Float.parseFloat(req.getParameter("avg")));

to

#12) DS calls handler method having Map object pointing Shared Memory and Model class obj representing from data as the argument values

#13) Handler method uses the service,DAO classes support to process the request and to generate the results and also keeps the result in shared memory as model attributes

#14) Handler method returns LVN back to DispatcherServlet

take

=> While dealing with form page submission related request processing we generally two handler methods

in the controller class

a) GET mode handler method to launch the form page (displaying form page)

of

b) POST mode handler method to process form page sumbisssion request (Processing form page)

note:: Both these methods can have same request path becoz the request modes are different

@Controller

public class StudentRegistrationController{

@GetMapping("/register")

public String showForm(){ //return LVN

Handler method

that launches/displays the form

}

return "student_register"; page

@PostMapping("/register")

public String registerStudent(Map<String,Object> map,

return "show_result";

}

}//class

@ModelAttribute("stud") Student st){

Handler method

to process the form page sumbission

#6? (verifies for

/register request path

handler method with POST mode request)

Example app Story board

===============================

welcome.jsp

=>Model class obj default scope is request scope

page

(form launching) student_register.jsp 1001

(home page)

sno:

register student

GET

sname: raja sadd: hyd avg: 90.66 <register

POST

(form submission related result page) show_result.jsp

This application needs 3 handler methods

in the controller class

//disppaly form data

a) For Launching Home Page

b) For Launching Form page

c) For Submitting the Form page

#1) create spring boot starter project having the following depdencices

a) web b) lombok c) tomcat-embedded-jasper with scope provided

#2) Develop model class

package com.nt.model;

import lombok.Data; @Data

public class Student {

}

private Integer so;

private String sname;

private String sadd;

private Float avg;

#3) Add the following entries in application.properties

application.properties

# For ViewResolver

spring.mvc.view.prefix=/WEB-INF/pages/

spring.mvc.view.suffix=.jsp

# for embedded server

server.port=4041

#4) DEVELOP CONTROLLER CLASS

package com.nt.controller;

import java.util.Map;

import org.springframework.stereotype.Controller; import
org.springframework.web.bind.annotation.GetMapping; import
org.springframework.web.bind.annotation.ModelAttribute; import
org.springframework.web.bind.annotation.PostMapping;

```
import com.nt.model.Student;
@Controller
public class StudentOperationsController {
@GetMapping("/")
//For home page
public String showHomePage() {
//return LVN (home page --welcome.jsp)
return "welcome";
}
@GetMapping("/register") //for displaying form page
public String showStudentFormPage() {
}
//return LVN (form page --> student_register.jsp
return "student_register";
@PostMapping("/register") For data rendering, to place model attributes
public String registerStudent(Map<String,Object> map,
System.out.println(st);
//return lvn
return "show_result";
}
}
```

#5) DEVELOP THE VIEW COMPS (JSP PAGES)

welcome.jsp

```
@ModelAttribute("stud") Student st) { //for processing the form page submission
For data binding, to place form data into Model class obj
<%@ page isELIgnored="false" %>
<h1 style="color:red;text-align:center">Home page</h1>
<br><br>
<h2 style="color:red;text-align:center"><a href="register">register student</a> </h2>
```

student_register.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>
<h1 style="color:red;text-aling:center">student registration page</h1>
<form action="register" method="POST">
<table align="center" bgcolor="cyan">
<tr>
<td>student number </td>
<td><input type="text" name="sno"> </td>
```

```
</tr>
<tr>
<td>student name </td>
<td><input type="text" name="sname"> </td>
</tr>
<tr>
<td>student address </td>
<td><input type="text" name="sadd"> </td>
</tr>
<tr>
<td>student avg </td>
<td><input type="text" name="avg"> </td>
</tr>
<tr>
<td colspan="2"><input type="submit" value="Register"></td>
</tr>
</table>
</form>
```

show result.jsp

```
<%@page isELIgnored="false" %>
M5BootMVCProj04-DataBinding-FormBinding [boot]
> Deployment Descriptor: BootMVCProj04-DataBinding-FormB
> Spring Elements
> JAX-WS Web Services
#src/main/java
✓ com.nt
> BootMvcProj04DataBinding FormBindingApplication.jav
>ServletInitializer.java
com.nt.controller
> StudentOperationsController.java
com.nt.model
> Student.java
src/main/resources
static
templates
<br>
<b> student data is :: ${stud} </b>
```

application.properties

> #src/test/java

> JRE System Library [JavaSE-11]

<br>

> Maven Dependencies

<a href="./">home</a>

› C Deployed Resources

✓ src

main

> java

>

resources

webapp

WEB-INF

•mages

show_result.jsp

student register.jsp

welcome.jsp

> test

> target

WHELP.md

mvnw

mvnw.cmd

Mpom.xml