Course : Diploma in Multimedia & Infocomm Technology (EGDF15)

Module : Java Enterprise Development (EG3752)

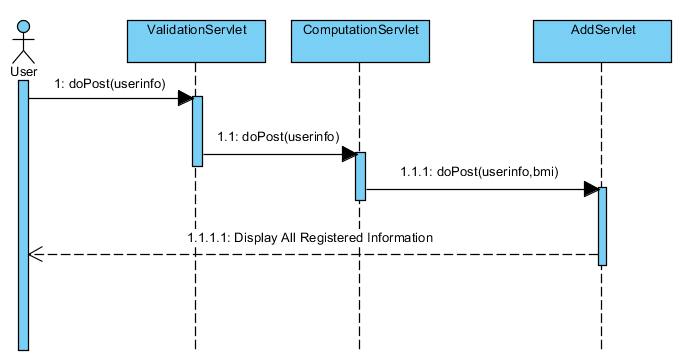
|  |  |
| --- | --- |
| Laboratory : | Lab 3 – Session Management |
| Objectives : | This lab will guide you through the building of web applications using servlets  At the end of this session, you should know how to:   * Develop a Java Object that conforms to Java Bean convention * Retrieve a HttpSession object within the servlet and JSP * Storing values into and retrieving values from a HttpSession object through the use of attributes * Invalidate a HttpSession programmatically |
| Software Used : | Java Standard Development Toolkit (JDK™) 8.0  NetBeans IDE 8.0 with GlassFish Server 4.0 bundle |

**Simple Registration Extended**

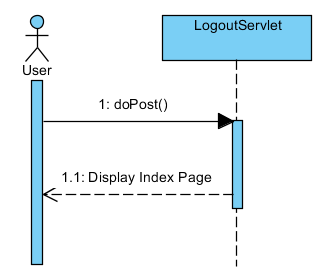
In this lab, we will extend on our work from Lab 2 to allow multiple registration details to be remembered by the web application. This can be achieved by saving all registration records into a HttpSession object. Additional requirement specifications are listed in the table below:

|  |  |
| --- | --- |
| **Requirement Reference** | **Requirement Specifications** |
| 1.3a Display  **(Note: This is a change in requirement. 1.3a will supersede Requirement Reference 1.3 in Lab 1)** | The system will display all information that is submitted by the user.  The system shall allow the user to submit multiple records. |
| 3.1 Registration Consolidation | The system shall be able to store multiple sets of registration details submitted. Each record will comprise a name, admin number, email address, gender, choice of 3rd year specialisation, height, weight and BMI. |
| 3.2 Logout | The system shall allow the user to logout. When logging out, the registration details will be erased. |

There are two main functions to implement, the add function and the log out function. They are represented by **Sequence Diagram 1** and **Sequence Diagram 2** respectively.



Sequence Diagram 1: Add Function



Sequence Diagram 2: Log Out Function

Experimental Task: Visible Difference between Client-Side and Server-Side Redirects

Answer the following questions to clarify the definitions for the following terminologies (*Hint: Answer can be found in Lab 2*) :

A. forward() method is a form of \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

i. Server-side Redirect  
 ii. Client-side Redirect

B. sendRedirect() method is a form of \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

i. Server-side Redirect  
 ii. Client-side Redirect

E.1 Run your web application and input valid information for the registration page.

E.2 Observe the information in the address bar and the information that is output by the page.

E.3 Edit the **ComputationServlet** codes such that it does a client-side redirect to **results.jsp** instead.

E.4 Repeat Step E.1 and E.2 again. What differences did you observe between using server-side redirect and client-side redirect?

Creating and Using a Java Bean compliant object to store Registration Details

To better manage multiple registration details, we will create an object to store the registration details.

* 1. Create a new Java class named **RegistrationRecord** in the project.
  2. Add in all fields shown in the form, together with the bmi, as member variables to the class. You may do that by replacing all the codes in **RegistrationRecord.java** with the ones in **Code Snippet Lab 3 P 1.2**.
  3. Make the class Java Bean compliant by doing the following:

1. Implement **Serializable**.
2. Add a new member variable to the class named **serialVersionUID** and initialised it to **-1L**:

private final long serialVersionUID **=** **-**1L**;**

1. Add a default constructor to the class.
2. Implement getters and setters for all member variables except for the serialVersionUID.

|  |
| --- |
| **More Shortcuts when Creating Objects**  If you are unsure what is a default constructor or how to implement getters and setters, try the following shortcut. It will greatly speed up your programming too:  1.3.1 To Create a Default Contructor  1.3.1.1 Place your cursor and select the character just before the close curly braces “}” that ends the class and do either of the following:  a. Press **Alt-Insert**  or  b. Right click and select **Insert Code…**  1.3.1.2 Select **Constructor…**  1.3.1.3 Do **NOT** select any of the fields. Click **Generate**.  1.3.2 To Generate Getters and Setters  1.3.2.1 Place your cursor and select the character just before the close curly braces “}” that ends the class and do either of the following:  a. Press **Alt-Insert**  or  b. Right click and select **Insert Code…**  1.3.2.2 Select **Getter and Setter…**  1.3.2.3 Select **ALL** the fields. Click **Generate**. |

1.4 Create an **AddServlet** for your project. Use “/add” as the urlPattern for the servlet.

1.5 Make your **ComputationServlet** do a **server-side redirect** intothe **AddServlet.**

1.6 In the **AddServlet**, create an instance of the **RegistrationRecord** object and store all the values required that are stored in the **HttpSerlvetRequest** object. (*Hint: Refer to EG1738: Object-Oriented Programming on how this can be done*)

Retrieving a HttpSession in a Servlet and Storing Values

We will be storing the registration record as a list of record in the session. Continuing from where you have left off in **AddServlet**:

2.1 Retrieve the session from the request object:

HttpSession session = request.getSession();

2.2 Attempt to retrieve the **reglist** attribute from the session. The **reglist** uses theList<RegistrationRecord>as its data type.

List<RegistrationRecord> reglist =   
 (List<RegistrationRecord>) session.getAttribute("reglist");

* 1. Check if the registration list actually exists. If it doesn’t, this means that the registration list has not been created before, ie. this is the first time you are going to add a record to the session.

If the list does not exist, instantiate a new list:

if(reglist == null)

{

reglist = new ArrayList<>();

}

2.4 Add the registration record into the list:

reglist.add(registrationRecord);

2.5 Save the list back into the session:

session.setAttribute("reglist",reglist);

* 1. Make a **client-side redirect** into **results.jsp**.

Retrieving Session in a JSP and Displaying Results from a List

Instead of displaying a single result in the JSP, we will be displaying all results stored in the session in the table format.

3.1 Replace the codes in **results.jsp** with **Code Snippet Lab 3 P 3.1**. Test your programme and ensure that the following screenshot is produced after submitting a record.

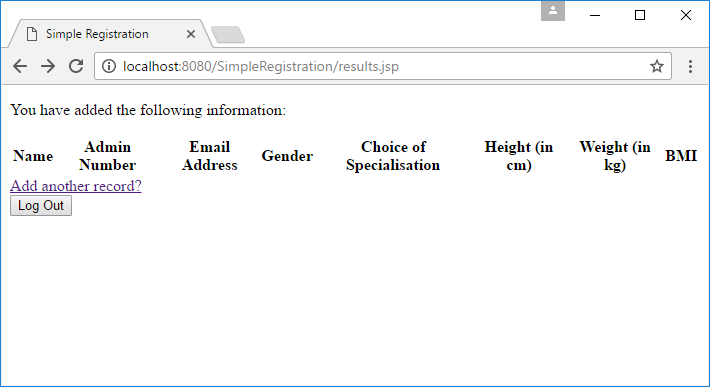


Figure 3.1: Screenshot of results.jsp after replacing the codes

As part of the test, ensure that when you click on “Add another record?” link, it goes back to index.html.

3.2 At line 18, we will be adding **scriptlets** to produce the table of registration records submitted by the user:

a. Retrieve the list of registration records from the session. Remember to add the necessary libraries to the JSP to clear the syntax error before continuing.

<%

List<RegistrationRecord> reglist =

(List<RegistrationRecord>)

session.getAttribute(“reglist”);

%>

b. Continue within the scriptlets tag, create a **for-each** loop to iterate through the list.

for(RegistrationRecord registrationRecord:reglist)

{

%>

<!-- Codes to generate additional rows go here -->

<%

}

c. Within the for loop, we will generate the table elements using HTML while filling it with values from the list using JSP expressions. You may notice that by coding everything within the for-each loop, the HTML portion of the codes will also be repeatedly generated by the loop.

<tr>

<td><%=registrationRecord.getName()%></td>

<!-- Try to complete this set of codes yourself -->

</tr>

* 1. Test your programme. Insert at least two records for the test.

Invalidating the Session Programmatically

* 1. Create a **LogoutServlet** using “/logout” as the URL pattern. Override the **doPost** method. (This time round, we will try to use the **POST** method instead.)
  2. In the doPost method, write the following to invalidate the session:

HttpSession session = request.getSession();

session.invalidate();

//Or Alternatively, request.getSession().invalidate();

* 1. Make a **client-side redirect** to the index page.