Uttara InfoSolutions

www.uttarainfo.com

Servlets Practicals 2

(Eclipse->Import->Web->WAR->Import the war files given so that you can look into it for clarification and for any doubts, ask us!)

1) Create a HTML with 2 user input elements (name and age) and 1 submit. Take a name and age as input. Submit to a servlet (url-pattern in action attribute with method='get'). In servlet, access the user inputs (request.getParameter("name of input element"), respond with a welcome message depending on age of user. Do user input validations. Change method action to 'post' and see if doPost() is getting called.

   Methods to use in doGet(request,response) are:

   a) String val = request.getParameter("<name of input element>");

   b) PrintWriter pw = response.getWriter();

   c) pw.write("<html>…");

2) Build a Counter using Servlet (CounterApp).In HomePage.html create a hyperlink to send request to "count" url-pattern and register it as welcome-file in web.xml. Right click on project and verify if sending a request to the app will respond with the Home Page. Count requests that are sent for url-pattern /count. The output should be "The Count is <number>". Per refresh the count should get incremented. What type of variable will you create for holding the count? Create it as an instance variable. Override init() and destroy() in CountServlet and ensure that in destroy, the count is getting written into a file named count.txt with a single line (count = value). Read this value in init() and initialise the value to the count instance variable. Test by restarting the server, whether the count is getting incremented per request or not.

3) Build a counter app which counts requests per client, meaning if you open one browser window and send request to http://localhost:8080/SessionCounter/count then

the count should be returned. New requests should increment the count. However if you open another browser and send request to the URL, the count should restart from 1.

4) Create a HomePage.html with 2 hyper links. Clicking on first should send request to “one” url-pattern. Map this to ServletOne servlet. Clicking on second hyper link should send request to “two” url-pattern which should be mapped to ServletTwo. Test linking first by putting SOPs in doXXX() and restarting the server. Once the linking is correct, then in ServletOne, store data in request, session, context under attribute name “data” and some dummy string values. Get the RequestDispatcher and forward to url-pattern “two”. Test it. When you click first hyperlink, the second servlet should get executed. In ServletTwo, get the attributes data from the 3 scopes and then write it into response. Try to click second hyperlink directly. Do you get all the 3 scopes data? If not why not? Is there any exception happening? How to fix it? What is the check you need to do to ensure you get the attribute data without exception?

Add for ServletOne in web.xml, an init-param element with <param-name> and <param-value>. Test in init(), doXXX() and destroy() whether you can access this data using getServletConfig(). Then add outside of all servlets, a <context-param> and then add a param name and value. Check if you can access this by using your ServletContext. (You can ignore init-param and context-param if you don't know about them yet)

Do you understand the 3 scopes and their uses now? Please ask doubts if you have any.

Methods to use:

request.setAttribute(“data”,”requestData”);

HttpSession session = request.getSession(true);

session.setAttribute(“data”,”sessionData”);

ServletContext ctx = getServletContext();

ctx.setAttribute(“data”,”ctxData”);

Similarly invoke ctx.getAttribute(“data”) and downcast to correct data type to get the data.

ServletConfig config = getServletConfig();

String value = config.getInitParameter(“<param name>”);

ServletContext ctx = getServletContext();

String val = ctx.getInitParamter("<param name>");

5) Build a Register / Login / Edit Account / Logout functionality. In HomePage, give the option to user to register. On registering, store the email, pass, name into a file named "users". On Login, verify if the user`s email and pass is correct by verifying in the database. If login succeeds, create a session for the user and then display a Menu with his name at the top and a hyperlink to click logout and a hyperlink to click change password. On clicking of logout, his session should be terminated. On clicking on change password, you should display an input view to accept new password that you should update in the same file.

6) Build an input view to accept name of the user and submit button. Submit to a servlet. Have a text file filled with quotes (1 quote per line). In init() method read all the quotes from the file and store it into an instance List variable of the servlet. In doXXX(), pick a random quote and write it into response so that each time the user sends a request with his name, you will display him a new quote. Do you understand why you need to use init()? Is there any necessity to use destroy() here?