CHAPTER 13

The Stateless Nature of the Web

A web application does not maintain state. It has no memory. Each request to the web server is an independent event. Each request does not know about previous requests. When you send your username and password, the web server views this as an independent event. It does not keep track of what you're logging in to. The information is simply sent to the server.

When you submit a form, all the information about what to do with the form data must be sent along with the form. Why? Because each request is an independent transaction.

In real life this is what it would be like if you went to the bank and got a new teller after each question. And the tellers don't talk to each other—only to you. And each teller would want to see your ID and check your balance and do everything the other teller had already done. To make such a situation easier, you could keep a running log of each transaction that each teller could use to verify what has been done.

So, how does a web application maintain state? The answer is by using either session variables or passing information known as parameters from the previous transaction. Parameters are sent between the client (web browser) and the server via either the URL or as other information sent to the server as part of the request. This is called the request packet; we have touched on this already in our discussion of servlets.

D. Wolf and A.J. Henley, *Java EE Web Application Primer*,
https://doi.org/10.1007/978-1-4842-3195-1_13

Session variables exist in the memory of the web server. Each request includes a session ID. The session ID links the request to the session data for that user. The session ID is automatically passed between requests. You don't have to do anything. It's always there.

Since there is one session per user, you can store variables in each user's session. This is a space in memory that holds data while the user is using the site. Since Java always knows the session ID, it has access to any data in the session.

So, it's the request packet and the session that tie the room together. And you thought it was the rug! (Not funny? Watch *The Big Lebowski* again). A session makes it easy for the server to connect one request to another.

The Process of Passing Data

The following list is a summary of the steps that are followed for data to be sent from a web form to a JSP using a servlet:

1. The form passes the request.
2. Servlet receives the request.
3. The servlet processes the request with `request.getParameter()`.
4. The servlet generates a response based on the data in the request.
5. The servlet constructs a response in an object that will be sent to the JSP.
6. The JSP contains an attribute `${user}`.

7. The servlet sets the attribute `request.setAttribute("user",myUser);`.
8. The servlet sends the JSP back to the originating browser by calling `getServletContext().getRequestDispatcher(url).forward(request, response);`.