Cookies and Counters

Cookies are used to store information about a client on the client's own computer. Session cookies are usually temporary, but they can be given a longer life-time as well. The following example uses a session to create a cookie. It then retrieves the cookie and some data about it. When first run, it will not find a cookie on the client's computer. But after that it will. However, if the client's browser is set to reject all cookies, no cookies will ever be found. To run this servlet, store the class file under the classes folder and use http://localhost/servlet/http_session.Cookies to run it.

```
package http_session;

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

/*    Cookies gets a session and places a cookie on the client's computer. */
public class Cookies extends HttpServlet
{
      public void doGet (HttpServletRequest request, HttpServletResponse response)
      {
            try
            {
                  HttpSession session = request.getSession (true);
                  response.setContentType ("text/html");
                  PrintWriter out = response.getWriter ();

                  // Cookies are stored in an array, so get the array and locate the cookies in it.
                  Cookie cookies [] = request.getCookies ();

                  // Create an output page to display the cookies found.
                  Page.createHeader (out, "Cookies");
                  if ((cookies == null) || (cookies.length == 0))
                        out.println ("<h3>No Cookies Found</h3>");
                  else
                  {
                        out.println ("<h3>Cookies Found</h3>");
                        for (int count = 0; count < cookies.length; count ++)
                        {
                              out.println ("<br>Name: " + cookies [count].getName ());
                              out.println ("<br>Value: " + cookies [count].getValue ());
                              out.println ("<br>Comment: " + cookies [count].getComment ());
                              out.println ("<br>MaxAge: " + cookies [count].getMaxAge ());
                        }
                  }
                  Page.createFooter (out);
            } catch (IOException ex) {System.out.println ("<h3>IO Exception.</h3>");}
      }
} // Cookies
```

If the client’s browser allows cookies, the following program can be used to count the number of times that it accesses the server. The session ID will be preserved from one time to the next. To try it out, just hit refresh each time after loading the page. Use http://localhost/servlet/http_session.Counter to run it.

```
package http_session;

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

/*    Counter gets a session object and uses it to count the number of times the page is accessed. */
public class Counter extends HttpServlet
{
    static final String CounterKey = "Counter.count";

    public void doGet (HttpServletRequest request, HttpServletResponse response)
    {
        try
        {
            HttpSession session = request.getSession (true);
            response.setContentType ("text/html");
            PrintWriter out = response.getWriter ();
            int count = 1;
            Integer i = (Integer) session.getAttribute (CounterKey);
            if (i != null) count = i.intValue () + 1;
            session.setAttribute (CounterKey, new Integer (count));

            // Create the output page including a form with the client’s url.
            Page.createHeader (out, "Session Counter");
            out.println ("Session ID: " + session.getId ());
            out.println ("<br />Count: " + count + "<br />");
            String url = request.getRequestURI ();
            out.println ("<form method = 'get' action = '" + url + "' />");
            out.println ("<input type = 'submit' value = 'Type Again' />");
            out.println ("</form>");
            Page.createFooter (out);
        } catch (IOException ex) {System.out.println ("<h3>IO Exception.</h3>");}
    }
} // Counter
```

If the browser is set to reject all cookies, a servlet will have to use URL rewriting in order to maintain a connection to it. The following servlet can be used for this. It puts the session id into the URL string before sending the page back to the client. Access it using http://localhost/servlet/http_session.CounterRewrite.

```
package http_session;

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
```

```
/*    CounterRewrite is used to count the number of hits on a page from client's that do not accept
cookies. */
public class CounterRewrite extends HttpServlet
{
    static final String CounterKey = "Counter.count";

    public void doGet (HttpServletRequest request, HttpServletResponse response)
    {
        try
        {
            HttpSession session = request.getSession (true);
            response.setContentType ("text/html");
            PrintWriter out = response.getWriter ();
            int count = 1;
            Integer i = (Integer) session.getAttribute (CounterKey);
            if (i != null) count = i.intValue () + 1;
            session.setAttribute (CounterKey, new Integer (count));

            // Create the output page.
            Page.createHeader (out, "Session Counter");
            out.println ("Session ID: " + session.getId ());
            out.println ("<br />Count: " + count + "<br />");

            // Get the client’s url and add the encoded url to the action attribute.
            String url = request.getRequestURI ();
            out.println ("<form method = 'get' action = '" + response.encodeURL (url) + "' />");
            out.println ("<input type = 'submit' value = 'Type Again' />");
            out.println ("</form>");
            Page.createFooter (out);
        } catch (IOException ex) {System.out.println ("<h3>IO Exception.</h3>");}
    }
} // CounterRewrite
```

Reference

Karl Moss, *Java Servlets Developer’s Guide*, chapter 3, McGraw-Hill/Osborne, 2002.