



Servlet Session Tracking

Qiang Yang

Thanks: Ethan Cerami, New York University

1/29/01 Cookies 1

Sessions: persist over a transaction

- Applications
 - Shopping Carts
 - Personalization Services
 - Maintaining state about the user's preferences.
- Implementation: *transmitted back and forth*
 - Cookies (what if users turn off cookies?)
 - URL Encoding (insecure)
 - `http://abc.com/servlet/Shop?id=99&item1=TV&price1=299.99&...`
 - Hidden Forms
 - `<input type="hidden" name="id" value="123">`
 - Generated dynamically by servlets

1/29/01 Cookies 2

Encoding URLs

- If a browser does not support cookies, you need some other way to maintain the user's session ID.
- The Servlet API takes care of this for you by automatically appending the session ID to URLs if the browser does not support cookies.
- To automatically append the session ID, use the **encodeURL ()** method


```
String url = response.encodeURL(http://www.abc.com/Shop.html);
```

1/29/01 Cookies 3

Sessions: the hard way

- Explicitly use a bunch of cookies or URL encoding
- When the user accepts cookies
 - Use a hashtable of cookies
- When the user does not allow cookies
 - Use URL encoding
- But, this is too complicated for us to do
 - Thus, the Session API does it automatically

1/29/01 Cookies 4

Session API: similar to cookies

- Steps to using the Java Session API
 - Get the Session object from the HTTPRequest object


```
HttpSession session=request.getSession()
```
 - Extract Data from the user's Session Object
 - Extract information about the session object, e.g. when was the session created?
 - Add data to the user's Session Object.

1/29/01 Cookies 5

Using Session Objects

- A session is a list of name/object pairs
- Before using the value, cast to desired type


```
Integer accessCount =
    (Integer) session.getValue("accessCount");
int count = accessCount.intValue()+1;
session.putValue("accessCount", new Integer(count));
```
- If you want to get a list of all "keys" associated with a Session, use the **getValueNames()** method.
 - This method returns an array of strings.

1/29/01 Cookies 6

Example

- Our example tracks the number of visits for each unique visitor.
 - If this is a first time visit, the servlet creates an `accessCount` Integer variable and assigns it to the Session.
 - If the user has visited before, the servlet extracts the `accessCount` variable, increments it, and assigns it to the Session.
- Servlet also displays basic information regarding the session, including: creation time and time of last access.

1/29/01

Cookies

7

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

```
public class ShowSession extends HttpServlet {
    public void doGet(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        String title = "Session Tracking Example";
        HttpSession session = request.getSession(true);
        String heading;
```

1/29/01

Cookies

8

```
Integer accessCount =
(Integer)session.getValue("accessCount");
if (accessCount == null) {
    accessCount = new Integer(0);
    heading = "Welcome, Newcomer";
} else {
    heading = "Welcome Back";
    accessCount = new Integer(accessCount.intValue() + 1);
}
session.putValue("accessCount", accessCount);
out.println("<HTML> <TITLE>" + title + "</TITLE>" +
"<BODY BGCOLOR=#FDF5E6>\n" +
"<H1 ALIGN=CENTER>" + heading + "</H1>\n" +
"<H2>Information on Your Session:</H2>\n" +
"<TABLE BORDER=1 ALIGN=CENTER>\n" +
"<TR BGCOLOR=#FFAD00>\n" +
```

1/29/01

Cookies

9

```
" <TH>Info Type<TH>Value\n" +
"<TR>\n" +
"<TD>ID\n" +
"<TD> + session.getId() + "\n" +
"<TR>\n" +
"<TD>Creation Time\n" +
"<TD> +
new Date(session.getCreationTime()) + "\n" +
"<TR>\n" +
"<TD>Time of Last Access\n" +
"<TD> +
new Date(session.getLastAccessedTime()) + "\n" +
"<TR>\n" +
"<TD>Number of Previous Accesses\n" +
"<TD> + accessCount + "\n" +
"</TR>"+
```

1/29/01

Cookies

10

```
"</TABLE>\n" +
"</BODY></HTML>");
}
```

```
/** Handle GET and POST requests identically. */
public void doPost(HttpServletRequest request,
    HttpServletResponse response)
    throws ServletException, IOException {
    doGet(request, response);
}
}
```

1/29/01

Cookies

11

Shopping Cart

- A shopping cart is a list of name value pairs
 - `ArrayList shoppingCart = new ArrayList();`
- Generate a new session
 - `session.putValue("books", shoppingCart);`
- Retrieving shopping cart items
 - `ArrayList shoppingCart = (ArrayList) session.getValue("books");`
 - Note the cast from object to `ArrayList`!

1/29/01

Cookies

12

Working with more than one servlet

- In an online bookstore, three servlets
 - Kidsbooks.class /* frontend*/
 - Computerbooks.class /* frontend*/
 - Order.class /* Processing Orders Using Sessions*/

1/29/01

Cookies

13

Example Store Front

```
Public class Kidsbooks extends HttpServlet {  
  
    public void doGet (...)  
    {  
  
        PrintWriter out = response.getWriter();  
        /* form stuff ...*/  
        String formURL = "servlet/order";  
        formURL = response.encodeURL(formURL);  
        out.println  
        ("<Form Action=\" + formURL + \" \" + \"\n\" +  
        \" <input type = ...  
    }  
}
```

1/29/01

Cookies

14

Two useful methods to call other servlets

- HttpServletResponse methods
 - String encodeURL (String url)
 - Encodes URL by appending the session ID if cookies are disabled
 - Embed in Form-Action or href
 - String encodeRedirectURL(String url)
 - Encodes URL before redirecting the browser (to the specified URL)
 - response.sendRedirect(encodedURL)

1/29/01

Cookies

15

Terminating a session

- A session can time out or be terminated
 - session.invalidate() kills a session
 - session.isNew() checks if the session is not yet touched by the client
 - session.getLastAccessdTime() allows you to check length of user inactivity
- Web server can set expiration time too.
- Or,
 - session.setMaxInactiveInterval(int seconds)

1/29/01

Cookies

16