Cookies

Cookies: what are they

A Cookie is a small amount of information sent by a servlet to a Web browser, saved by the browser, and later sent back to the server.

A cookie's value can uniquely identify a client, so cookies are commonly used for session management.

A cookie has a name, a single value, and optional attributes such as a comment, path and domain qualifiers, a maximum age, and a version number.

Some Web browsers have bugs in how they handle the optional attributes, so use them sparingly to improve the interoperability of your servlets.

Cookies

Cookies affect the caching of the Web pages that use them. HTTP 1.0 does not cache pages that use cookies created with this class.

The Java class "Cookie" does not support the cache control defined with HTTP 1.1. This class supports both the Version 0 (by Netscape) and Version 1 (by RFC 2109) cookie specifications. By default, cookies are created using Version 0 to ensure the best interoperability

Cookies: why?

To maintain status across a "user session"

To maintan infos across sessions

- Customer identification
- Targeted advertisement
- Elimination of username e password

Cookies: i metodi

public void setMaxAge(int c)

public int getMaxAge()

Sets (gets) the maximum age of the cookie in seconds. A positive value indicates that the cookie will expire after that many seconds have passed. Note that the value is the maximum age when the cookie will expire, not the cookie's current age. A negative value means that the cookie is not stored persistently and will be deleted when the Web browser exits. A zero value causes the cookie to be deleted

Cookies: i metodi

public void setDomain(String c)

public String getDomain()

Specifies the domain within which this cookie should be presented. The form of the domain name is specified by RFC 2109. A domain name begins with a dot (.foo.com) and means that the cookie is visible to servers in a specified Domain Name System (DNS) zone (for example, www.foo.com, but not a.b.foo.com). By default, cookies are only returned to the server that sent them.

public void setPath(int c)

public int getPath()

Returns the path on the server to which the browser returns this cookie. The cookie is visible to all subpaths on the server.

Cookies: i metodi

public void setComment(String c)

public String getComment()

Acts on comment describing the purpose of this cookie, or null if the cookie has no comment.

public void setVersion(int c)

public int getVersion ()

Version 0: Netscape standard

Version 1: RFC 2109

Cookies: i metodi

public void setSecure(boolean v)

public boolean setSecure()

Indicates to the browser whether the cookie should only be sent using a secure protocol, such as HTTPS or SSL. The default value is false.

Cookies: esempio

```
Cookie userCookie = new Cookie("user","uid1234");
userCookie.setMaxAge(60*60*24*365);
response.addCookie(userCookie);
```

SetCookies

```
import java.io.*;   import javax.servlet.*; import javax.servlet.http.*;
/** Sets six cookies: three that apply only to the current session
 * (regardless of how long that session lasts) and three that persist for an hour
 * (regardless of whether the browser is restarted).
*/
public class SetCookies extends HttpServlet {
  public void doGet(HttpServletRequest request, HttpServletResponse
    response)
                    throws ServletException, IOException {
    for(int i=0; i<3; i++) {
      // Default maxAge is -1, indicating cookie
      // applies only to current browsing session.
      Cookie cookie = new Cookie("Session-Cookie-" + i,
                                 "Cookie-Value-S" + i);
      response.addCookie(cookie);
```

SetCookies

```
cookie = new Cookie("Persistent-Cookie-" + i,"Cookie-Value-P" + i);
     // Cookie is valid for an hour, regardless of whether
     // user quits browser, reboots computer, or whatever.
     cookie.setMaxAge(3600);
     response.addCookie(cookie);
   }
   response.setContentType("text/html");
   PrintWriter out = response.getWriter();
   String title = "Setting Cookies";
   out.println (("<HTML><HEAD><TITLE>" +title+ "</TITLE></HEAD>" +
     "<BODY BGCOLOR=\"#FDF5E6\">\n" +"<H1 ALIGN=\"CENTER\">"
    + title + "</H1>\n" +"There are six cookies associated with this page.\n" +
"</BODY></HTML>");
  }
}
```

ShowCookies

```
import java.io.*;  import javax.servlet.*;   import javax.servlet.http.*;
/** Creates a table of the cookies associated with the current page. */
public class ShowCookies extends HttpServlet {
  public void doGet(HttpServletRequest request, HttpServletResponse
    response)
                    throws ServletException, IOException {
   response.setContentType("text/html");
   PrintWriter out = response.getWriter();
   String title = "Active Cookies";
   out.println(("<HTML><HEAD><TITLE>" +title+ "</TITLE></HEAD>" +
           "<BODY BGCOLOR=\"#FDF5E6\">\n" +
           "<H1 ALIGN=\"CENTER\">" + title + "</H1>\n" +
           "<TABLE BORDER=1 ALIGN=\"CENTER\">\n" +
           "<TR BGCOLOR=\"#FFAD00\">\n" +
           "  <TH>Cookie Name\n" + "  <TH>Cookie Value");
```

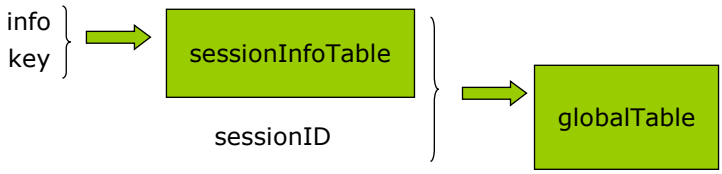
ShowCookies

```
Cookie[] cookies = request.getCookies();
   Cookie cookie;
   for(int i=0; i<cookies.length; i++) {
     cookie = cookies[i];
     out.println("<TR>\n" +
                 "  <TD>" + cookie.getName() + "\n" +
                 "  <TD>" + cookie.getValue());
   }
   out.println("</TABLE></BODY></HTML>");
  }
}
```

Sessions

Session tracking using cookies

```
String sessionID = makeUniqueString();
Hashtable sessionInfoTable = new Hashtable();
Hashtable globalTable = getTableStoringSession();
globalTable.put(sessionID, sessionInfoTable );
Cookie sessionCookie=new Cookie("SessionID",sessionID);
sessionCookie.setPath("/");
response.addCookie(sessionCookie);
```



HttpSession Class

Provides a way to identify a user across more than one page request or visit to a Web site and to store information about that user.

The servlet container uses this interface to create a session between an HTTP client and an HTTP server. The session persists for a specified time period, across more than one connection or page request from the user.

A session usually corresponds to one user, who may visit a site many times. The server can maintain a session in many ways such as using cookies or rewriting URLs.

HttpSession Class

This interface allows servlets to View and manipulate information about a session, such as the session identifier, creation time, and last accessed time Bind objects to sessions, allowing user information to persist across multiple user connections.

When an application stores an object in or removes an object from a session, the session checks whether the object implements HttpSessionBindingListener. If it does, the servlet notifies the object that it has been bound to or unbound from the session.

Session tracking API

```
HttpSession session = request.getSession(true);
ShoppingCart cart = (ShoppingCart)session.getValue("carrello");
   // 2.1
// 2.2   (ShoppingCart)session.getAttribute("carrello");
if (cart==null) {
   cart=new ShoppingCart();
   session.putValue("carrello",cart); //2.1
//2.2 session.putValue("carrello",cart);
}
doSomeThingWith(cart);
```

Session tracking API

```
public void putValue(String name, Object value);
        //2.1
public void setAttribute(String name, Object value);
        //2.2

public void removeValue(String name);   //2.1
public void removeAttribute(String name);       //2.2

public String[] getValueNames()             //2.1
public Enumeration getAttributeNames() //2.2
```

Session tracking API

```
public long getCreationTime();
public long getLastAccessdTime();
   milliseconds since midnight,  1.1.1970

public int getMaxInactiveInterval();
public void setMaxInactiveInterval(int sec);

public void invalidate();
```

ShowSession

```
import java.io.*; import javax.servlet.*; import javax.servlet.http.*;
import java.net.*; import java.util.*;
/** Simple example of session tracking. */
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;
   // Use getAttribute instead of getValue in version 2.2.
   Integer accessCount = (Integer)session.getValue("accessCount");
```

ShowSession

```
if (accessCount == null) {
      accessCount = new Integer(0);
      heading = "Welcome Newcomer";
    } else {
      heading = "Welcome Back";
      accessCount = new Integer(accessCount.intValue() + 1);
    }
    // Use setAttribute instead of putValue in version 2.2.
    session.putValue("accessCount", accessCount);
```

ShowSession

```
out.println(("<HTML><HEAD><TITLE>" +title+ "</TITLE></HEAD>" +
            "<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" +
            "  <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" + "</TABLE>\n" +"</BODY></HTML>");
}
```

ShowSession

```
/** Handle GET and POST requests identically. */

public void doPost(HttpServletRequest request,
                   HttpServletResponse response)
  throws ServletException, IOException {
  doGet(request, response);
 }
}
```

Sessions in JSP

Predefined Objects

out	Writer
request	HttpServletRequest
response	HttpServletResponse
session	HttpSession
page	this nel Servlet
application	servlet.getServletContext
	area condivisa tra i servlet
config	ServletConfig
exception	solo nella errorPage
pageContext	sorgente degli oggetti, raramente usato

session

```
<%@ page errorPage="errorpage.jsp" %>

<html> <head> <title>UseSession</title> </head>
 <body>
  <%
    // Try and get the current count from the session
    Integer count = (Integer)session.getAttribute("COUNT");
    // If COUNT is not found, create it and add it to the session
    if ( count == null ) {
      count = new Integer(1);
      session.setAttribute("COUNT", count);
    }
```

session

```
    else {
      count = new Integer(count.intValue() + 1);
      session.setAttribute("COUNT", count);
    }
    // Get the User's Name from the request
    out.println("<b>Hello you have visited this site: "
      + count + " times.</b>");
  %>
 </body>
</html>
```