Cookies

Some of this material adapted from Core Servlets and JavaServer Pages, © 2008, Marty Hall. Reused with permission

The Potential of Cookies

Design Idea

- Servlet sends a simple name and value to client.
- Client returns same name and value when it connects to same site (or same domain, depending on cookie settings).

Cookies are an Internet propsed standard IETF 6265

- https://tools.ietf.org/html/rfc6265
- In HTTP, a response header *Set-Cookie* is used to request that the browser/client store a cookie.
  - The browser/client dos NOT have to accept the cookie, but if it does, it sends it back to the server on each request

Typical Uses of Cookies

- Identifying a user during an e-commerce session
  - Servlets have a higher-level API for this task
- Customizing a site (personalization)
- Focusing advertising (recommenders)

Using Cookies

Standard approach for adding a Cookie to a Response:

```
Cookie c = new Cookie("name", "value");
c.setMaxAge(...); // Means cookie persists on disk
// Set other attributes.
response.addCookie(c);
```

Standard approach for reading a Cookie from the Request:

```
Cookie[] cookies = request.getCookies();
if (cookies != null) {
  for(int i=0; i<cookies.length; i++) {
    Cookie c = cookies[i];
    if (c.getName().equals("someName")) {
      doSomethingWith(c);
      break;
    }
  }
}
```

Methods in the Cookie API

getDomain/setDomain

- Lets you specify domain to which cookie applies. Current host must be part of domain specified.

getMaxAge/setMaxAge

- Gets/sets the cookie expiration time (in seconds). If you fail to set this, cookie applies to current browsing session only.

getName

- Gets the cookie name. Names are only set in the constructor.

getPath/setPath

- Gets/sets the path to which cookie applies. If unspecified, cookie applies to URLs within or below directory containing current page.

getSecure/setSecure

- Gets/sets flag indicating whether cookie should apply only to SSL connections or to all connections.

getValue/setValue

- Gets/sets value associated with cookie.

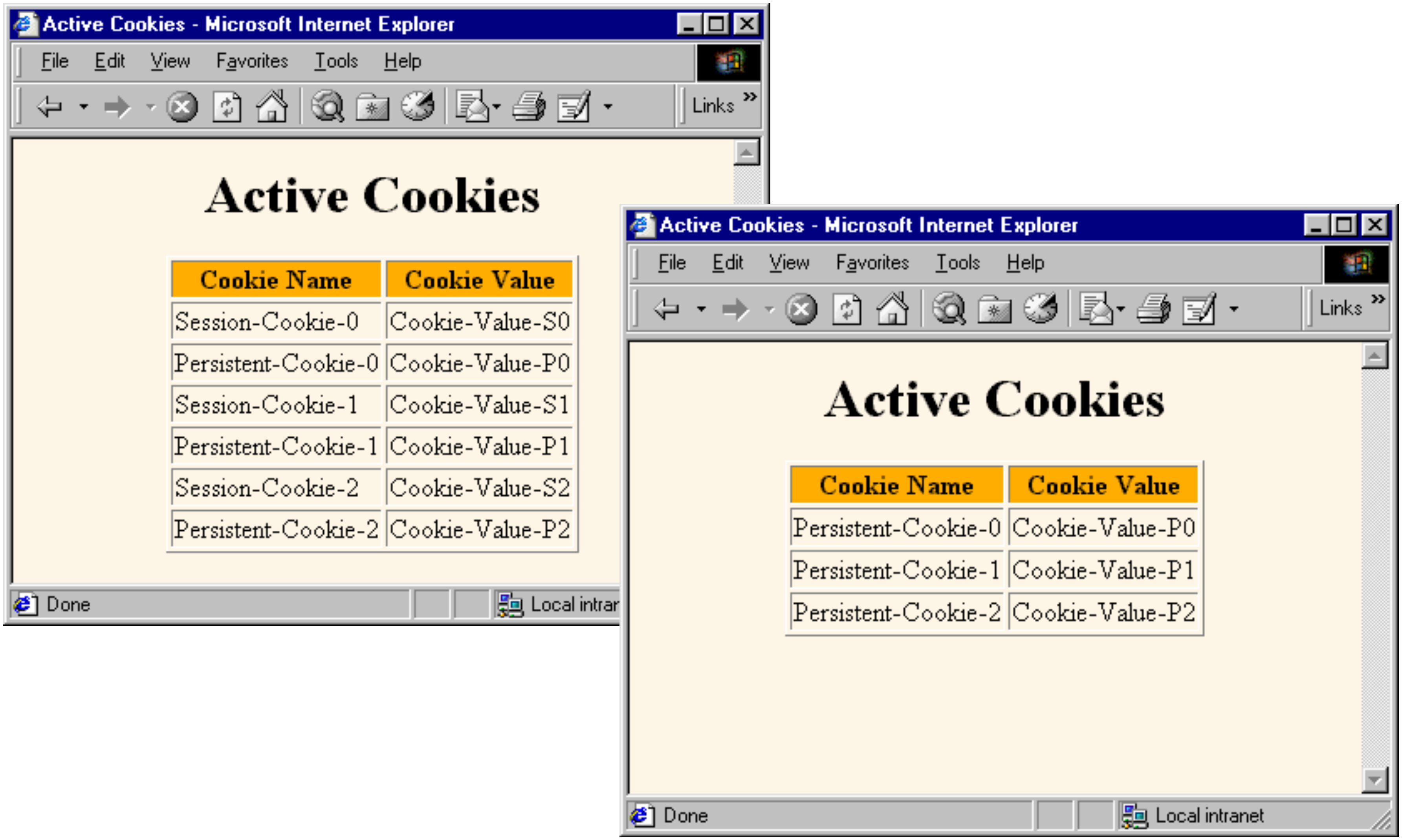
Simple Cookie-Setting Servlet

```
public class SetCookies extends HttpServlet {
  public void doGet(HttpServletRequest request,
                    HttpServletResponse response)
                      throws ServletException, IOException {
    for(int i=0; i<3; i++) {
      Cookie cookie = new Cookie("Session-Cookie-" + i,
                                 "Cookie-Value-S" + i);
      response.addCookie(cookie);
      cookie = new Cookie("Persistent-Cookie-" + i,
                          "Cookie-Value-P" + i);
      cookie.setMaxAge(3600);
      response.addCookie(cookie);
    }
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    out.println(...);
```

Simple Cookie-Viewing Servlet

```
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(ServletUtilities.headWithTitle(title) +
       "<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");
   Cookie[] cookies = request.getCookies();
   if (cookies != null) {
      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>");
  }
 }
```

Result of Cookie-Viewer



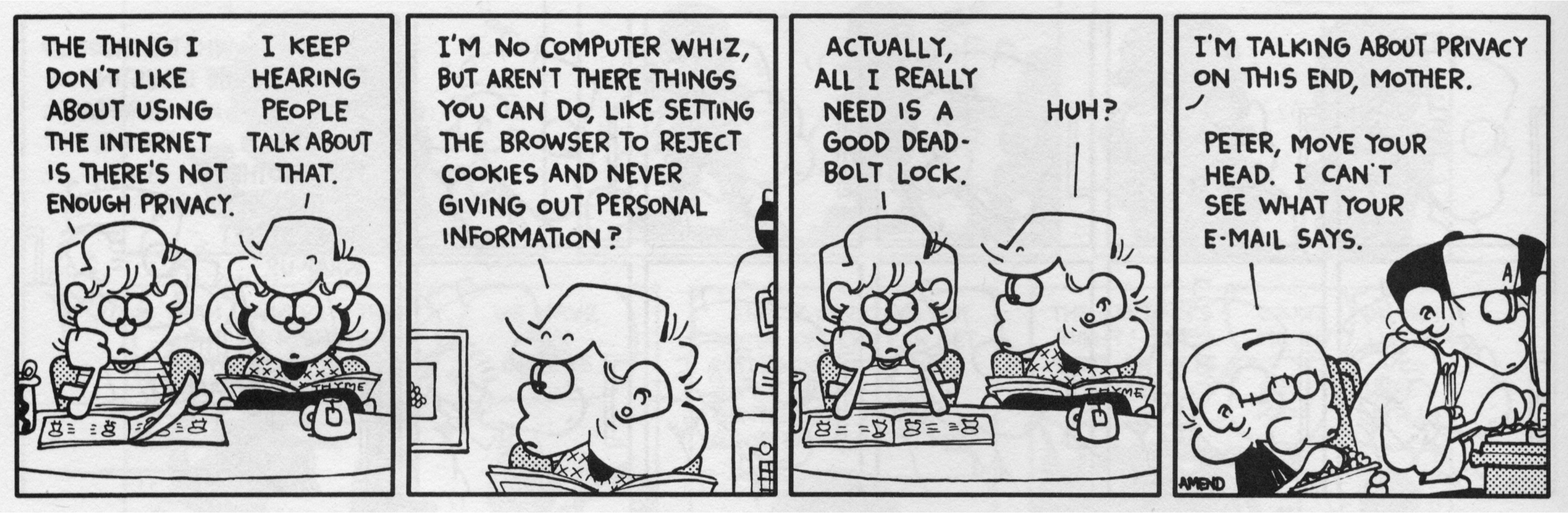
Active Cookies

Cookie Name	Cookie Value
Session-Cookie-0	Cookie-Value-S0
Persistent-Cookie-0	Cookie-Value-P0
Session-Cookie-1	Cookie-Value-S1
Persistent-Cookie-1	Cookie-Value-P1
Session-Cookie-2	Cookie-Value-S2
Persistent-Cookie-2	Cookie-Value-P2

Active Cookies

Cookie Name	Cookie Value
Persistent-Cookie-0	Cookie-Value-P0
Persistent-Cookie-1	Cookie-Value-P1
Persistent-Cookie-2	Cookie-Value-P2

Cookies and Privacy


THE THING I DON'T LIKE ABOUT USING THE INTERNET IS THERE'S NOT ENOUGH PRIVACY.
I KEEP HEARING PEOPLE TALK ABOUT THAT.
I'M NO COMPUTER WHIZ, BUT AREN'T THERE THINGS YOU CAN DO, LIKE SETTING THE BROWSER TO REJECT COOKIES AND NEVER GIVING OUT PERSONAL INFORMATION?
ACTUALLY, ALL I REALLY NEED IS A GOOD DEAD-BOLT LOCK.
HUH?
I'M TALKING ABOUT PRIVACY ON THIS END, MOTHER.
PETER, MOVE YOUR HEAD. I CAN'T SEE WHAT YOUR E-MAIL SAYS.
THYME
AMEND

FoxTrot © 1998 Bill Amend. Reprinted with permission of Universal Press Syndicate. All rights reserved.

The problem is privacy, not security.

- If you give out personal info, servers can link it to previous actions
- Servers can share cookie info through 3rd parties like doubleclick.net
- Poorly designed sites store sensitive info (e.g. credit card #) in cookie

Moral for servlet authors

- If cookies are not critical to your task, avoid servlets that totally fail when cookies are disabled
- Don't put sensitive info in cookies

Summary

Cookies are name/value pairs sent from server to browser and returned when the page/site/domain is visited later.

Cookies allow the servlet developer to

- Track sessions (use higher-level API)
- Permit users to avoid logging in at low-security sites
- Customize sites for different users
- Focus content or advertising

Cookies and State Management

- Cookies are not a great solution for storing conversational state
  - Cookie may be poisoned or stolen
  - Limited in what they can store anyway
  - Especially bad if it is a *persistent cookie*
- Cookies are a more reasonable solution for user preferences
  - Provided that doesn’t mean keeping sensitive info on the client

Your site should be able to function if cookies are disabled!