Module 7

Cookie API

Course Objectives

- After completing this module, you should be able to:
  - Describe what a cookie is
  - Describe the use of cookies for client (or session) management
  - Explain how cookies are implemented in the architecture
  - Examine the cookie API
  - Explain how long cookies last

Cookie API

- Creating cookies
  - Cookie(String name, String value)
- Sending a cookie back to the browser
  - HttpServletResponse.addCookie(Cookie aCookie)
- Retrieving cookies
  - HttpServletRequest.getCookies()
- Retrieving a cookie’s name
  - aCookie.getName()
- Retrieving a cookie’s value
  - aCookie.getValue()
- Changing a cookie's value
  - aCookie.setValue(String)

Cookie usage example

```
public void doGet(HttpServletRequest req,
                  HttpServletResponse res) {
  String userType = "novice";
  Cookie[] cookies = req.getCookies();
  if (cookies != null) {
    for (int i=0; i<cookies.length; i++) {
      if(cookies[i].getName().equals("userType"))
        userType = cookies[i].getValue();
    }
  }
  if (userType.equals("expert"))
      // do expert HTML
  else     // do novice HTML
}
```

Proper cookie usage

- Because cookies are stored as plain text on the client machine, cookies can be viewed and altered by the client
- Cookies should not be used for information such as:
  - Validation information
  - Secure information (credit card numbers and the like)

Cookie applicability

- Cookies have an expiration date
  - setMaxAge(int expiryInSeconds)
- Default expiration date is -1
  - This means that the cookie is not stored persistently
  - It lasts only as long as the browser is open
- Can restrict the applicable URLs to which a cookie will be sent
  - setDomain(String)
  - setPath(String)

Client data and session tracking with cookies

- Cookies can be made to persist within or across browser interactions
- Cookies are passed to the Web server in the header of the request
- Any updates are passed back in the header of the response
- Session data tracking via HTTP cookies is the most commonly used session tracking mechanism
  - Required to be supported by all Web containers

Checking if cookies are accepted on browser

- Most browsers can be configured to accept or decline cookies
- There is no single method to directly check the browser's cookie configuration
- To check the browser's cookie configuration:
  - Use the **addCookie()** method to send a test cookie to the browser
  - Use **getCookies()** method to read the test cookie
  - Execute appropriate code based on the result of **getCookies()**

Checking cookie config: sample code

```
// Check to see if cookietest parameter is set
if (req.getParameter("cookietest") == null) {
  resp.addCookie(new Cookie("CookieTest", "ok"));
  String url = req.getRequestURI() + "?cookietest=ok";
  resp.sendRedirect(url);
  return;
}
// Callback from sendRedirect() above - check for cookie
ServletContext ctx = getServletContext();
if (req.getCookies() != null) {
  // Cookies were accepted, so handle appropriately
  ctx.getRequestDispatcher("/HandleCookie").
  forward(req, resp);
} else {
  // Cookies were declined, so handle appropriately
  ctx.getRequestDispatcher("/HandleNoCookie").
  forward(req, resp);
}
```

Checkpoint

1. What types of data are best not kept in a cookie?
2. Why use cookies?
3. How do you send a cookie to a browser?

Module Summary

- This unit covered the following topics:
  - Describe what a cookie is
  - Describe the use of cookies for client (or session) management
  - Create a cookie and add it to a response
  - Find a cookie in a request
  - Set cookie properties such as:
    - Name
    - Value
    - Age
    - Domain
    - Path
  - Test for cookie support in a browser