

Course: Web Application Development

Lab Instructor: Dr. Nguyen Van Sinh

Email: nvsinh@hcmiu.edu.vn

Lab 4 - Web Session using Cookies (March 22nd, 2018)

Content:

- **Definitions.**
- **Sending cookies to browser**
- **Reading cookies from browser**
- **Simple cookie-handling Servlets**
- **Practices and Exercises**

Duration: 3 hours

Part 1: Definitions

What is Cookies?

A message given to a web browser by a web server. The browser stores the message in a text file. The message is then sent back to the server each time the browser requests a page from the server.

The main purpose of cookies is to identify users and possibly prepare customized web pages for them. When you enter a website using cookies, you may be asked to fill out a form providing such information as your name and interests. This information is packaged into a cookie and sent to your web browser which stores it for later use. The next time you go to the same website, your browser will send the cookie to the webserver. The server can use this information to present you with custom webpages. So, for example, instead of seeing just a generic welcome page you might see a welcome page with your name on it.

What is a Session?

A Session refers to all the request that a single client makes to a server. A session is specific to the user and for each user a new session is created to track all the request from that user. Every user has a separate session and separate session variable is associated with that session. In case of web application, the default time-out value for session variable is 20 minutes, which can be changed as per the requirement.

What is Session cookies?

Also called a transient cookie, a cookie that is erased when the user closes the web browser. The session cookie is stored in temporary memory and is not retained after the browser is closed. Session cookies do not collect information from the user's computer. They typically will store information in the form of a session identification that does not personally identify the user.

What is Persistent cookies?

Also called a permanent cookie, or a stored cookie, a cookie that is stored on a user's hard drive until it expires (persistent cookies are set with expiration dates) or until the user deletes the cookies. Persistent cookies are used to collect identifying information about the user, such as web surfing behavior or user preferences for a specific website.

Part 2: Sending Cookies to client (browser)

Following 3 steps below:

1. Creating a `Cookie` object. You call the `Cookie` constructor with a cookie name and a cookie value, both of which are strings.

Example: `Cookie c = new Cookie("userID", "a1234");`

2. Setting the maximum age. If you want the browser to store the cookie on disk instead of just keeping it in memory, you use `setMaxAge` to specify how long (in seconds) the cookie should be valid.

Example: `c.setMaxAge(60*60*24*7); // One week`

3. Placing the `Cookie` into the HTTP response headers. You use `response.addCookie` to accomplish this. If you forget this step, no cookie is sent to the browser!

Example: `response.addCookie(c);`

Part 3: Reading Cookies from the client (browser)

Following 2 steps:

1. Call `request.getCookies`. This yields an array of `Cookie` objects.
2. Loop down the array, calling `getName` on each one until you find the cookie of interest. You then typically call `getValue` and use the value in some application-specific way.

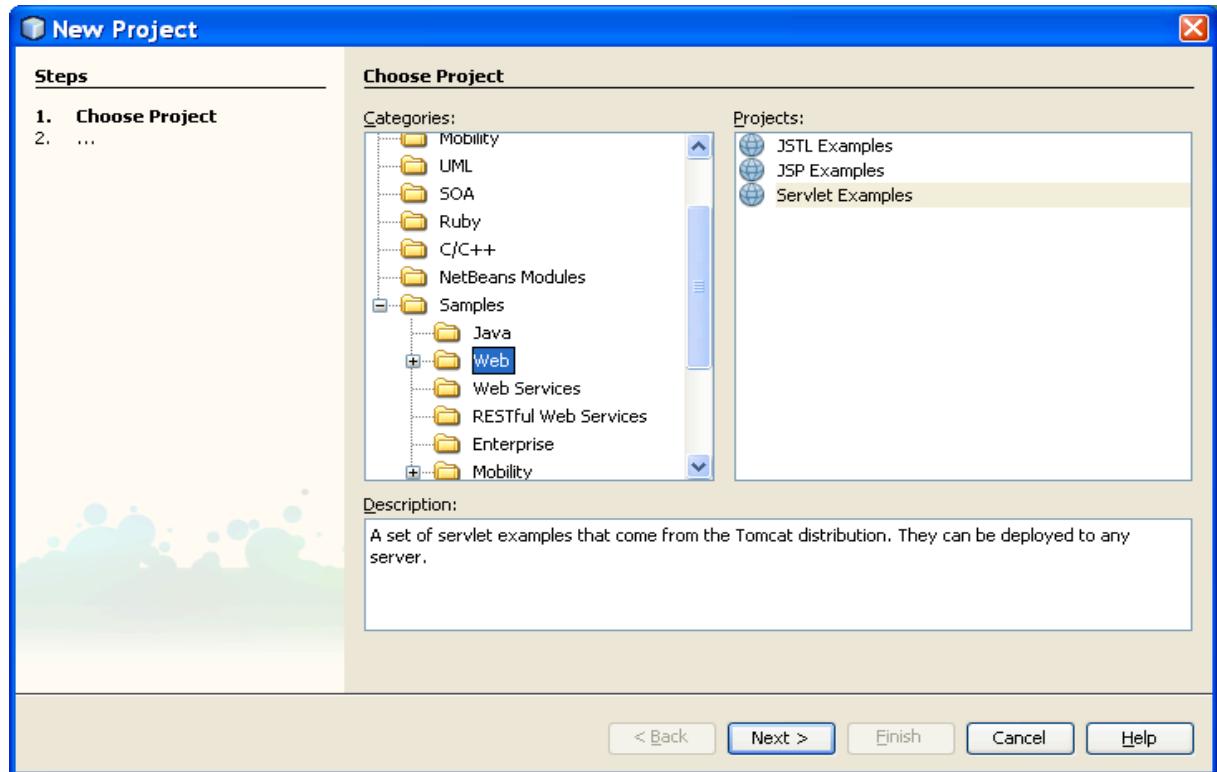
Loop Down the Cookie Array:

Once you have the array of cookies, you typically loop down it, calling `getName` on each `Cookie` until you find one matching the name you have in mind. Remember that cookies are specific to your host (or domain), not your servlet (or JSP page). So, although your servlet might send a single cookie, you could get many irrelevant cookies back. Once you find the cookie of interest, you typically call `getValue` on it and finish with some processing specific to the resultant value. For example:

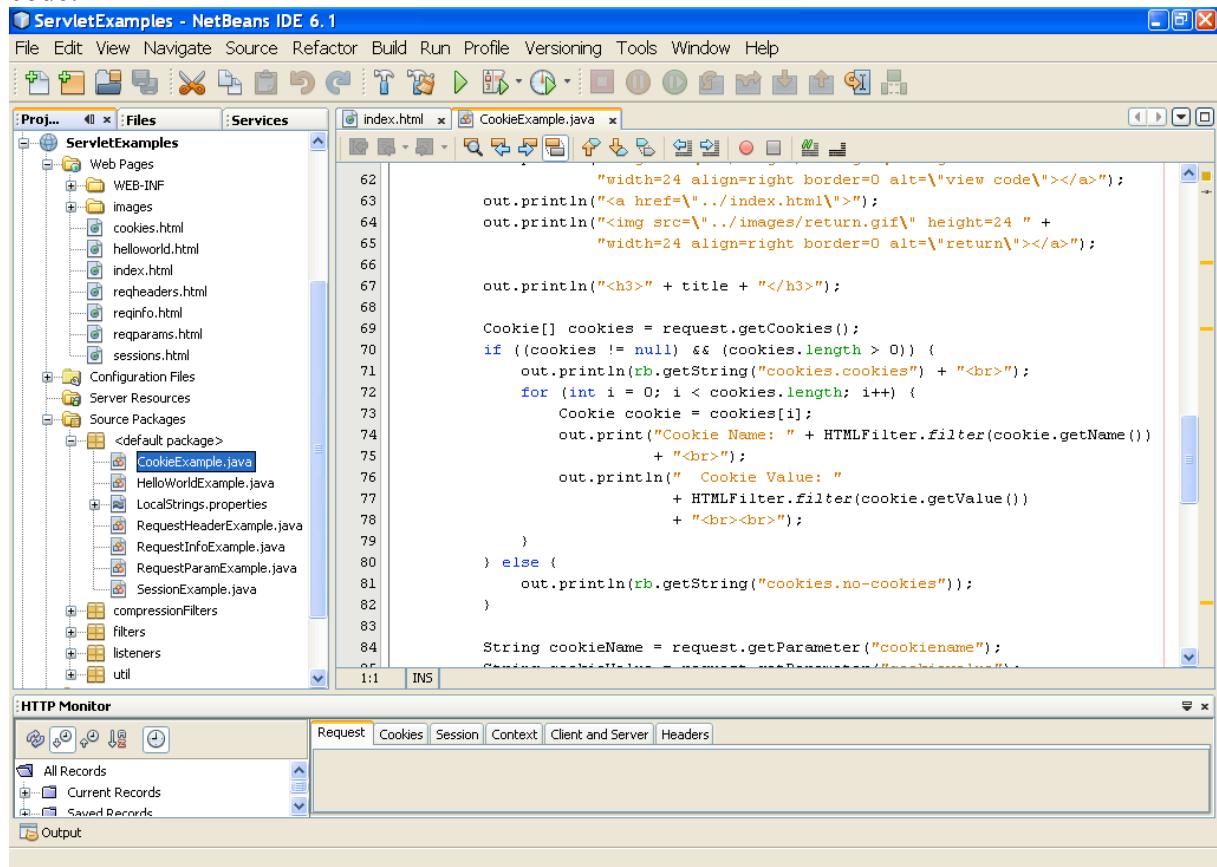
```
String cookieName = "userID";
Cookie[] cookies = request.getCookies();
if (cookies != null) {
    for(int i=0; i<cookies.length; i++) {
        Cookie cookie = cookies[i];
        if (cookieName.equals(cookie.getName())) {
            doSomethingWith(cookie.getValue());
        }
    }
}
```

Part 4: Simple cookie-handling Servlets

Click File -> New Project. Chose Samples\Web on the categories and choose Servlet Example -> Next -> Finish (pls go to google site without Samples directory)



The project is created automatically. Choose the file `CookieExample.java`, look at the source code:



This is the interface when we run project on browser

Servlet Examples - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

http://localhost:8080/servlets-examples/ Go

C# Random String Gen... Customize Links Downloads Home Page... e-books Free Hotmail Windows Marketplace

These examples will only work when viewed via an http URL. They will not work if you are viewing these pages via a "file://" URL. Please refer to the README file provided with this Tomcat release regarding how to configure and start the provided web server.

Wherever you see a form, enter some data and see how the servlet reacts. When playing with the Cookie and Session Examples, jump back to the Headers Example to see exactly what your browser is sending the server.

To navigate your way through the examples, the following icons will help:

- Execute the example
- Look at the source code for the example
- Return to this screen

Tip: To see the cookie interactions with your browser, try turning on the "notify when setting a cookie" option in your browser preferences. This will let you see when a session is created and give some feedback when looking at the cookie demo.

Hello World	Execute	Source
Request Info	Execute	Source
Request Headers	Execute	Source
Request Parameters	Execute	Source
Cookies	Execute	Source
Sessions	Execute	Source

Note: The source code for these examples does not contain all of the source code that is actually in the example, only the important sections of code. Code not important to understand the example has been removed for clarity.

Done

Part 5: Examples

Example 1:

Explore all methods of cookie and session by:

- create a web project
- create a Servlet
- declare an object Cookies cookie -> cookie. ?
- declare an object HttpSession session -> session. ?

Example 2:

Open source code of Cookies and Session from ServletExample and modify Cookies:

JSP Page - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

http://localhost:8084/SessionCookies/TestSimpleCookie.jsp Go

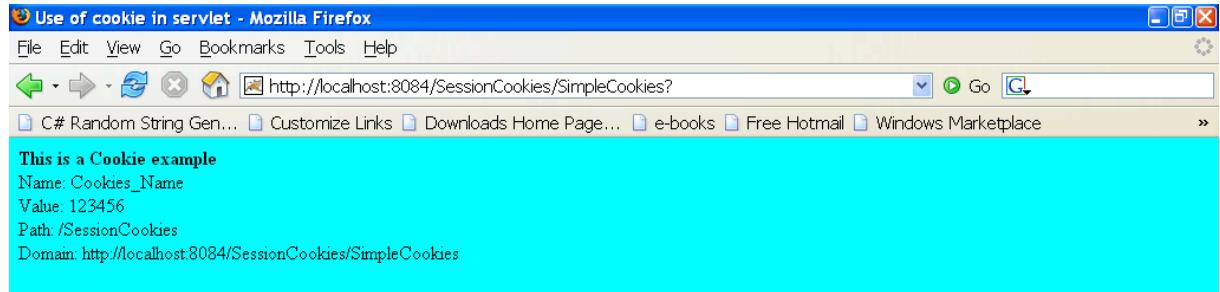
C# Random String Gen... Customize Links Downloads Home Page... e-books Free Hotmail Windows Marketplace

Test Simple Cookies!

Create a servlet with name SimpleCookie.java, modify source code below:

```
PrintWriter out;
response.setContentType("text/html");
out = response.getWriter();
Cookie cookie = new Cookie("Cookies_Name", "123456");
cookie.setMaxAge(100);
response.addCookie(cookie);
```

This is the result on browser:



Example 3:

Design a form like this:

Print out the cookies

Input Cookie Name:

Input Cookie Value:

Copy the following source code and modify:

```
public class CookieExample extends HttpServlet {

    public void doGet(HttpServletRequest request, HttpServletResponse
response)
        throws IOException, ServletException
    {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();

        // print out cookies

        Cookie[] cookies = request.getCookies();
        for (int i = 0; i < cookies.length; i++) {
            Cookie c = cookies[i];
            String name = c.getName();
            String value = c.getValue();
            out.println(name + " = " + value);
        }

        // set a cookie

        String name = request.getParameter("cookieName");
        if (name != null && name.length() > 0) {
            String value = request.getParameter("cookieValue");
            Cookie c = new Cookie(name, value);
            response.addCookie(c);
        }
    }
}
```

This is the result:

Print out the cookies:
Name: JSESSIONID
Content: 93788FAD219662ABB5125F25F07284F6

Set a new cookie:
Cookie name: CookieValue
Cookie value: 12345

Go to browser click Tool -> Option -> Cookies -> View Cookies to check information of name and value which is sent to browser.

Exercise 3: Base on chapter 8 in the text-book (8.5 Using Cookies to Detect First-Time Visitors) copy source code and modify to create a cookie:

This is the first running, counter = 0.

Welcome Aboard

Counter: 0

This is the result when click “Refresh” button, counter = 1.

Welcome Back

Counter: 1

Exercise 4: Refer example “Listing 9.1 ShowSession.java” from text-book, you can modify source code to understand session.

Test Session

Submit

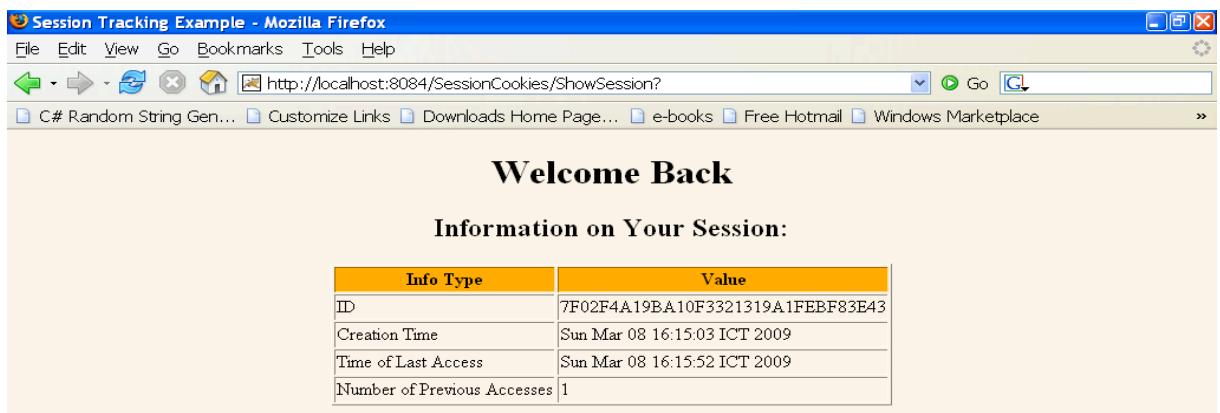
The first running on browser:

Welcome, Newcomer

Information on Your Session:

Info Type	Value
ID	7F02F4A19BA10F3321319A1FEBF83E43
Creation Time	Sun Mar 08 16:15:03 ICT 2009
Time of Last Access	Sun Mar 08 16:15:03 ICT 2009
Number of Previous Accesses	0

The next time (refresh button):



Extra Exercise: **Implementation of “Keep me logged in” feature**

You are required to do the following tasks to implement a feature of an e-commerce website:

1. Create a graphical user interface of login window (user name, password and login button) using HTML and CSS.
2. Implement a Cookie to store user name and password of user. After the first login, your browser will be able to remember your user name and password. Closing your browser and opening it again, your browser should recognize your identification without logging in.
3. Encrypt the cookie using AES-256. The Advanced Encryption Standard (AES) is an encryption algorithm for securing electronic data established by the U.S National Institute of Standards and Technology. To prevent unauthorized access of user name and password stored in Cookies, you need to encrypt the Cookies using a reliable encryption standard AES-256.

Hint: you can refer in the textbook and the source code in the CD