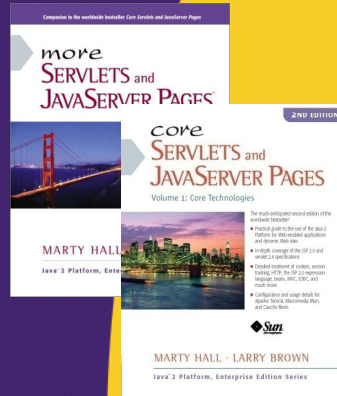




Session Tracking

Originals of Slides and Source Code for Examples:
<http://courses.coreservlets.com/Course-Materials/csajsp2.html>

Customized Java EE Training: <http://courses.coreservlets.com/>
Servlets, JSP, JSF 2.0, Java 6, Ajax, jQuery, GWT, Spring, Hibernate, RESTful Web Services, Android.
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.



**For live Java EE training, please see training courses
at <http://courses.coreservlets.com/>.**

**Servlets, JSP, JSF 2, Ajax (with jQuery, Dojo, Prototype, Ext-JS, etc.),
GWT, Java 6, SOAP-based and RESTful Web Services, Spring,
Hibernate/JPA, Android, and customized combinations of topics.**



Taught by the author of *Core Servlets and JSP*, *More Servlets and JSP*, and this tutorial. Available at public venues, or customized versions can be held on-site at your organization. Contact hall@coreservlets.com for details.

Agenda

- Implementing session tracking from scratch
- Using basic session tracking
- Understanding the session-tracking API
- Differentiating between server and browser sessions
- Encoding URLs
- Tracking user access counts
- Accumulating user purchases
- Implementing a shopping cart
- Building an online store

5

© 2011 Marty Hall



Overview

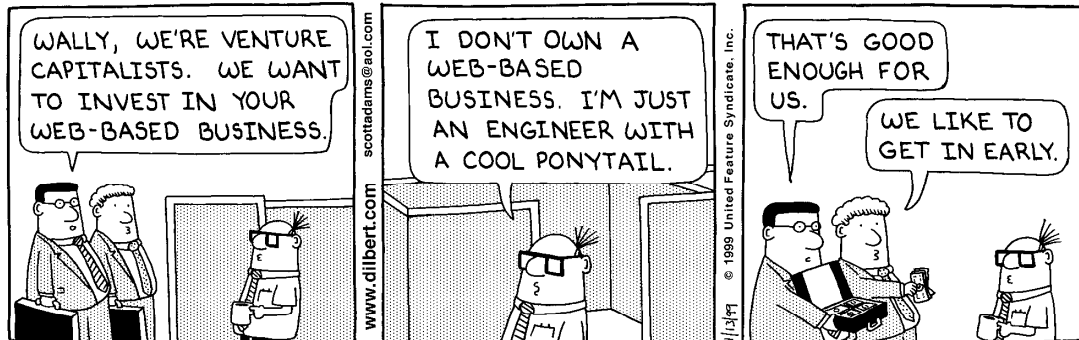
Customized Java EE Training: <http://courses.coreservlets.com/>
Servlets, JSP, JSF 2.0, Java 6, Ajax, jQuery, GWT, Spring, Hibernate, RESTful Web Services, Android.
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

6

Session Tracking and E-Commerce

- **Why session tracking?**

- When clients at on-line store add item to their shopping cart, how does server know what's already in cart?
- When clients decide to proceed to checkout, how can server determine which previously created cart is theirs?



Dilbert used with permission of United Syndicates Inc.

Rolling Your Own Session Tracking: Cookies

- **Idea: associate cookie with data on server**

```
String sessionId = makeUniqueString();
HashMap sessionInfo = new HashMap();
HashMap globalTable = findTableStoringSessions();
globalTable.put(sessionID, sessionInfo);
Cookie sessionCookie =
    new Cookie("JSESSIONID", sessionId);
sessionCookie.setPath("/");
response.addCookie(sessionCookie);
```

- **Still to be done:**

- Extracting cookie that stores session identifier
- Setting appropriate expiration time for cookie
- Associating the hash tables with each request
- Generating the unique session identifiers

Rolling Your Own Session Tracking: URL-Rewriting

- **Idea**
 - Client appends some extra data on the end of each URL that identifies the session
 - Server associates that identifier with data it has stored about that session
 - E.g., `http://host/path/file.html;jsessionid=1234`
- **Advantage**
 - Works even if cookies are disabled or unsupported
- **Disadvantages**
 - Must encode all URLs that refer to your own site
 - All pages must be dynamically generated
 - Fails for bookmarks and links from other sites

9

Rolling Your Own Session Tracking: Hidden Form Fields

- **Idea:**
 - `<INPUT TYPE="HIDDEN" NAME="session" VALUE="...">`
- **Advantage**
 - Works even if cookies are disabled or unsupported
- **Disadvantages**
 - Lots of tedious processing
 - **All pages must be the result of form submissions**

10



The Java Session-Tracking API

Customized Java EE Training: <http://courses.coreservlets.com/>
Servlets, JSP, JSF 2.0, Java 6, Ajax, jQuery, GWT, Spring, Hibernate, RESTful Web Services, Android.
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

11

Session Tracking Basics

- **Access the session object**
 - Call `request.getSession` to get HttpSession object
 - This is a hashtable associated with the user
- **Look up information associated with a session.**
 - Call `getAttribute` on the HttpSession object, cast the return value to the appropriate type, and check whether the result is null.
- **Store information in a session.**
 - Use `setAttribute` with a key and a value.
- **Discard session data.**
 - Call `removeAttribute` discards a specific value.
 - Call `invalidate` to discard an entire session.

12

Session Tracking Basics: Sample Code

```
HttpSession session = request.getSession();
synchronized(session) {
    SomeClass value =
        (SomeClass) session.getAttribute("someID");
    if (value == null) {
        value = new SomeClass(...);
    }
    doSomethingWith(value);
    session.setAttribute("someID", value);
}
```

If SomeClass is a mutable data structure (i.e., you didn't call "new", but just modified the existing object, and you are using a normal (non distributed) application, then the call to setAttribute could be inside the if statement. But if it is an immutable data structure (i.e., you really created a new object, not modified the old one) or you are on a distributed/clustered app, you need to call setAttribute after modifying the value. Since it can't hurt to do this anyhow, it is a good practice to put the call to setAttribute after the part that modifies the session data.

13

To Synchronize or Not to Synchronize?

- **The J2EE blueprints say not to bother**
 - There are no race conditions when multiple *different* users access the page simultaneously
 - On the face of it, it seems practically impossible for the same user to access the session concurrently
- **The rise of Ajax makes synchronization important**
 - With Ajax calls, it is actually quite likely that two requests from the same user could arrive concurrently
- **Performance tip**
 - Don't do "synchronized(this)"!
 - Use the session or perhaps the value from the session as the label of the synchronized block

14

What Changes if Server Uses URL Rewriting?

- **Session tracking code:**
 - No change
- **Code that generates hypertext links back to same site:**
 - Pass URL through `response.encodeURL`.
 - If server is using cookies, this returns URL unchanged
 - If server is using URL rewriting, this appends the session info to the URL
 - E.g.:

```
String url = "order-page.html";  
url = response.encodeURL(url);
```
- **Code that does `sendRedirect` to own site:**
 - Pass URL through `response.encodeRedirectURL`

15

HttpSession Methods

- **getAttribute**
 - Extracts a previously stored value from a session object. Returns null if no value is associated with given name.
- **setAttribute**
 - Associates a value with a name. Monitor changes: values implement `HttpSessionBindingListener`.
- **removeAttribute**
 - Removes values associated with name.
- **getAttributeNames**
 - Returns names of all attributes in the session.
- **getId**
 - Returns the unique identifier.

16

HttpSession Methods (Continued)

- **isNew**
 - Determines if session is new to *client* (not to *page*)
- **getCreationTime**
 - Returns time at which session was first created
- **getLastAccessedTime**
 - Returns time at which session was last sent from client
- **getMaxInactiveInterval, setMaxInactiveInterval**
 - Gets or sets the amount of time session should go without access before being invalidated
- **invalidate**
 - Invalidates current session

17

© 2011 Marty Hall



Storing Simple Values

Customized Java EE Training: <http://courses.coreservlets.com/>
Servlets, JSP, JSF 2.0, Java 6, Ajax, jQuery, GWT, Spring, Hibernate, RESTful Web Services, Android.
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

18

A Servlet that Shows Per-Client Access Counts

```
@WebServlet("/show-session")
public class ShowSession extends HttpServlet {
    public void doGet(HttpServletRequest request,
        HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html");
        HttpSession session = request.getSession();
        synchronized(session) {
            String heading;
            Integer accessCount =
                (Integer)session.getAttribute("accessCount");
            if (accessCount == null) {
                accessCount = new Integer(0);
                heading = "Welcome, Newcomer";
            } else {
                heading = "Welcome Back";
                accessCount =
                    new Integer(accessCount.intValue() + 1);
            }
            session.setAttribute("accessCount", accessCount);
        }
    }
}
```

19

A Servlet that Shows Per-Client Access Counts (Continued)

```
PrintWriter out = response.getWriter();
...
out.println
    (docType +
        "<HTML>\n" +
        "<HEAD><TITLE>" + title + "</TITLE></HEAD>\n" +
        "<BODY BGCOLOR=\"#FDF5E6\">\n" +
        "<CENTER>\n" +
        "<H1>" + heading + "</H1>\n" +
        "<H2>Information on Your Session:</H2>\n" +
        "<TABLE BORDER=1>\n" +
        "<TR BGCOLOR=\"#FFAD00\">\n" +
        "  <TH>Info Type<TH>Value\n" +
        ...
        "  <TD>Number of Previous Accesses\n" +
        "  <TD>" + accessCount + "\n" +
        "</TABLE>\n" +
        "</CENTER></BODY></HTML>");
}
```

20

A Servlet that Shows Per-Client Access Counts: User 1

Session Tracking Example - Mozilla Firefox

http://localhost/session-tracking/show-session

Welcome Back

Information on Your Session:

Info Type	Value
ID	72423AAB3F0E975C5F301409ED35FA24
Creation Time	Thu Sep 30 08:27:00 EDT 2010
Time of Last Access	Thu Sep 30 08:41:11 EDT 2010
Number of Previous Accesses	14

Done

21

A Servlet that Shows Per-Client Access Counts: User 2

Session Tracking Example - Internet Explorer

http://localhost/session-tracking/show-session

Welcome Back

Information on Your Session:

Info Type	Value
ID	ECECB6F319EBBDE30CD187D529AFC101
Creation Time	Thu Sep 30 08:43:12 EDT 2010
Time of Last Access	Thu Sep 30 08:44:10 EDT 2010
Number of Previous Accesses	10

Done

Internet | Protected Mode: On

100%

22



Storing Lists of Values

Customized Java EE Training: <http://courses.coreservlets.com/>
 Servlets, JSP, JSF 2.0, Java 6, Ajax, jQuery, GWT, Spring, Hibernate, RESTful Web Services, Android.
 Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

23

Aside: Compilation Warnings re Unchecked Types

- **HttpSession does not use generics**
 - Since it was written pre-Java5. So, following is illegal:


```
HttpSession<ArrayList<String>> session =
    request.getSession();
```
- **Typecasting to a generic type results in a compilation warning**

```
HttpSession session = request.getSession();
List<String> listOfBooks =
    (List<String>) session.getAttribute("book-list");
...
```

 - The warning is correct, since Java cannot verify that List contains only Strings. Still compiles and runs, but warning is annoying. You don't want to get in habit of ignoring warnings.
- **You can suppress warnings**
 - Put the following before line of code that does typecast:


```
@SuppressWarnings("unchecked")
```

24

Accumulating a List of User Data

```
@WebServlet("/show-items")
public class ShowItems extends HttpServlet {
    public void doPost (HttpServletRequest request,
                       HttpServletResponse response)
        throws ServletException, IOException {
        HttpSession session = request.getSession();
        synchronized(session) {
            @SuppressWarnings("unchecked")
            List<String> previousItems =
                (List<String>) session.getAttribute("previousItems");
            if (previousItems == null) {
                previousItems = new ArrayList<String>();
            }
            String newItem = request.getParameter("newItem");
            if ((newItem != null) &&
                (!newItem.trim().equals("")))) {
                previousItems.add(newItem);
            }
            session.setAttribute("previousItems", previousItems);
        }
    }
}
```

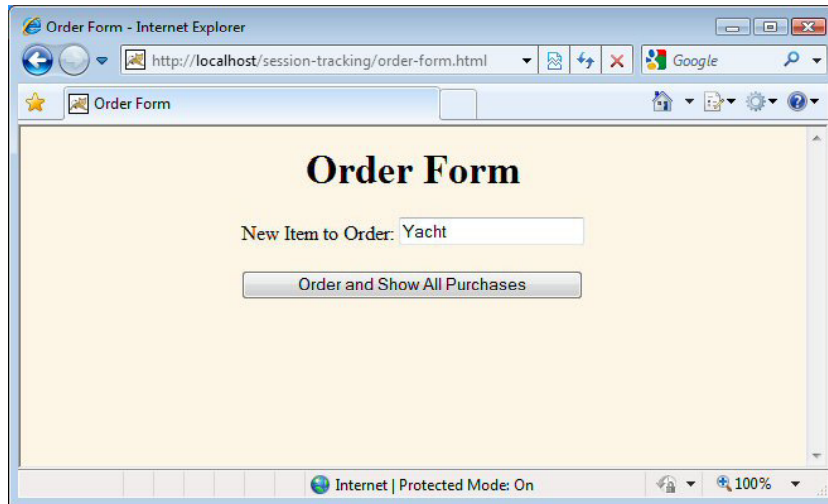
25

Accumulating a List of User Data (Continued)

```
response.setContentType("text/html");
PrintWriter out = response.getWriter();
String title = "Items Purchased";
String docType =
    "<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 4.0 \" +
    \"Transitional//EN\">\n";
out.println(docType +
    "<HTML>\n" +
    "<HEAD><TITLE>" + title + "</TITLE></HEAD>\n" +
    "<BODY BGCOLOR=\"#FDF5E6\">\n" +
    "<H1>" + title + "</H1>");
if (previousItems.size() == 0) {
    out.println("<I>No items</I>");
} else {
    out.println("<UL>");
    for(String item: previousItems) {
        out.println(" <LI>" + item);
    }
    out.println("</UL>");
}
out.println("</BODY></HTML>");
}
```

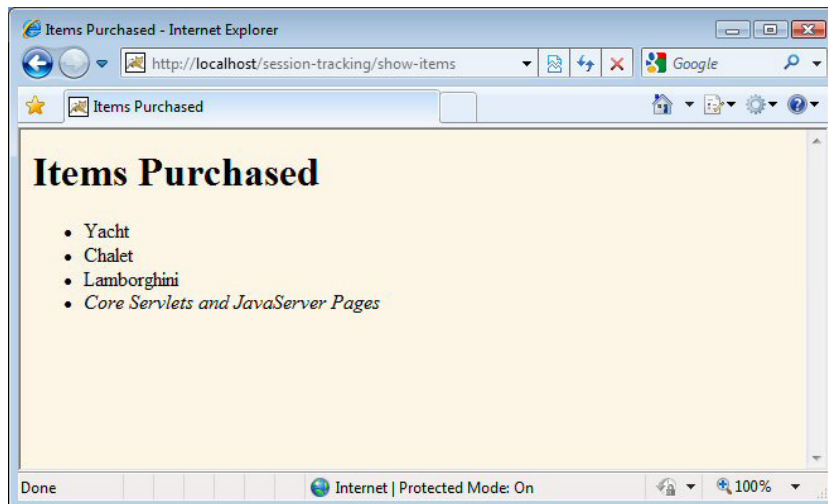
26

Accumulating a List of User Data: Front End



27

Accumulating a List of User Data: Result



28



Advanced Features

Customized Java EE Training: <http://courses.coreservlets.com/>
 Servlets, JSP, JSF 2.0, Java 6, Ajax, jQuery, GWT, Spring, Hibernate, RESTful Web Services, Android.
 Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

29

Distributed and Persistent Sessions

- **Some servers support distributed Web apps**
 - Load balancing used to send different requests to different machines. Sessions should still work even if different hosts are hit.
 - On many servers, you must call `setAttribute` to trigger replication
 - This is a tradeoff: session duplication can be expensive, but gives you better load balancing
- **Some servers support persistent sessions**
 - Session data written to disk and reloaded when server is restarted (as long as browser stays open). **Very important for web4!**
 - Tomcat 5 through 7 support this
- **To support both, make session data Serializable**
 - Classes should implement the `java.io.Serializable` interface
 - There are no methods in this interface; it is just a flag:


```
public class MySessionData implements Serializable
    ...
}
```
 - Builtin classes like `String` and `ArrayList` are already `Serializable`

30

Letting Sessions Live Across Browser Restarts

- **Issue**
 - By default, Java sessions are based on cookies that live in the browser's memory, but go away when the browser is closed. This is often, but not always, what you want.
- **Solution**
 - Explicitly send out the JSESSIONID cookie.
 - Do this at the beginning of the user's actions
 - Call `setMaxAge` first
- **Problem**
 - Using a cookie with a large `maxAge` makes no sense unless the session timeout (`inactiveInterval`) is also large
 - An overly large session timeout can waste server memory

31

An On-Line Bookstore

- **Session tracking code stays the same as in simple examples**
- **Shopping cart class is relatively complex**
 - Identifies items by a unique catalog ID
 - Does not repeat items in the cart
 - Instead, each entry has a count associated with it
 - If count reaches zero, item is deleted from cart
- **Pages built automatically from objects that have descriptions of books**

32

An On-Line Bookstore

The screenshot shows two browser windows. The left window displays the 'All-Time Best Children's Fantasy Books' page, featuring three book listings: 'The Chronicles of Narnia by C.S. Lewis (\$19.95)', 'The Prydain Series by Lloyd Alexander (\$19.95)', and 'The Harry Potter Series by J.K. Rowling (\$59.95)'. Each listing includes a brief description and an 'Add to Shopping Cart' button. The right window displays the 'All-Time Best Computer Books' page, featuring two listings: 'Core Servlets and JavaServer Pages 2nd Edition (Volume 1) by Marty Hall and Larry Brown (\$39.95)' and 'Core Web Programming, 2nd Edition by Marty Hall and Larry Brown (\$49.99)'. These listings also include descriptions and 'Add to Shopping Cart' buttons.

33

An On-Line Bookstore

The screenshot shows a browser window titled 'Status of Your Order'. The page contains a table with the following data:

Item ID	Description	Unit Cost	Number	Total Cost
alexander001	The Prydain Series by Lloyd Alexander	\$19.95	4 <input type="button" value="Update Order"/>	\$79.80
rowling001	The Harry Potter Series by J.K. Rowling	\$59.95	1 <input type="button" value="Update Order"/>	\$59.95
lewis001	The Chronicles of Narnia by C.S. Lewis	\$19.95	1 <input type="button" value="Update Order"/>	\$19.95
hall001	Core Servlets and JavaServer Pages 2nd Edition (Volume 1) by Marty Hall and Larry Brown	\$39.95	52 <input type="button" value="Update Order"/>	\$2,077.40
hall002	Core Web Programming, 2nd Edition by Marty Hall and Larry Brown	\$49.99	23 <input type="button" value="Update Order"/>	\$1,149.77

Below the table is a 'Proceed to Checkout' button.

34



Wrap-up

Customized Java EE Training: <http://courses.coreservlets.com/>
Servlets, JSP, JSF 2.0, Java 6, Ajax, jQuery, GWT, Spring, Hibernate, RESTful Web Services, Android.
Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

35

Summary

- **Sessions do not travel across network**
 - Only unique identifier does
- **Get the session**
 - `request.getSession`
- **Extract data from session**
 - `session.getAttribute`
 - Do typecast and check for null
 - If you cast to a generic type, use `@SuppressWarnings`
- **Put data in session**
 - `session.setAttribute`
- **Custom classes in sessions**
 - Should implement `Serializable`

36

Summary: Code Template

```
HttpSession session = request.getSession();
synchronized(session) {
    SomeClass value =
        (SomeClass)session.getAttribute("someID");
    if (value == null) {
        value = new SomeClass(...);
    }
    doSomethingWith(value);
    session.setAttribute("someID", value);
}
```

37

© 2011 Marty Hall



Questions?

Customized Java EE Training: <http://courses.coreservlets.com/>

Servlets, JSP, JSF 2.0, Java 6, Ajax, jQuery, GWT, Spring, Hibernate, RESTful Web Services, Android.

Developed and taught by well-known author and developer. At public venues or onsite at *your* location.

38