Lecture 6
S3 - Summary of Web Development in JEE
Sections 3
Cristian Toma – Business Card

Dorobantiilor Ave., No. 15-17
010572 Bucharest - Romania
http://ism.ase.ro
christian.toma@ie.ase.ro
T +40 21 319 19 00 - 310
F +40 21 319 19 00
Agenda for Lecture 6

1. Java Servlet Technology Intro – S3 Start
2. Exchange Ideas
Java Servlet Tech Recapitulation
1. Java Servlet Technology

What is Java Servlet?

Sun: Java Servlet technology provides Web developers with a simple, consistent mechanism for extending the functionality of a Web server and for accessing existing business systems.

WiKi: Servlets are Java programming language objects that dynamically process requests and construct responses. The Java Servlet API allows a software developer to add dynamic content to a Web server using the Java platform. The generated content is commonly HTML, but may be other data such as XML.

The web server for the Java servlet simple tests is Apache Tomcat 7 in Ubuntu 12
http://tomcat.apache.org
1. Java Servlet Technology

What is Java Servlet?

* Java Servlets Intro & Development Cycle
  - Java Servlet Structure
  - Java Servlet sample that generates “Plain Text”
  - Compiling and testing Java Servlet
  - A Simple Servlet Generating HTML

* Processing the Request: Form Data
  - Introduction (Format, URL-encoding, GET, POST)
  - Example: Reading Specific Parameters
  - Example: Making Table of All Parameters
1. Java Servlet Technology

What is Java Servlet?

* HTTP Request Headers
  - Common Request Headers
  - Sample: Java Servlet for displaying HTML table of the Request Headers

* HTTP Status Codes & HTTP Response Headers
  - Overview: Status Codes & Response Headers
  - Set Status Codes from Java Servlets
  - Set Response Headers from Java Servlets
  - Sample: Refresh at each 3 seconds based on Response Headers

* Handling Cookies
  - Cookies Intro
  - Java Servlet Cookie API
  - Sample: Set/Get Cookie for Internet Explorer & Mozilla

* Session Tracking
  - Session Tracking Overview
  - Java Servlet Session Tracking API + Sample
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class HelloWorld extends HttpServlet {
    public void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        PrintWriter out = response.getWriter();
        out.println("<!DOCTYPE HTML PUBLIC \"<!--[if IE]><!DOCTYPE HTML 4.01 Transitional/>\"">
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "
        "$
## Servlet Example: Showing Request Headers

**Request Method:** GET  
**Request URI:** /servlet/hall.ShowRequestHeaders  
**Request Protocol:** HTTP/1.0

<table>
<thead>
<tr>
<th>Header Name</th>
<th>Header Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection</td>
<td>Keep-Alive</td>
</tr>
<tr>
<td>User-Agent</td>
<td>Mozilla/4.05 [en] (WinNT; I)</td>
</tr>
<tr>
<td>Host</td>
<td>webdev.apl.jhu.edu</td>
</tr>
<tr>
<td>Accept</td>
<td>image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, image/png, <em>/</em></td>
</tr>
<tr>
<td>Accept-Language</td>
<td>en</td>
</tr>
<tr>
<td>Accept-Charset</td>
<td>iso-8859-1,* utf-8</td>
</tr>
<tr>
<td>Cookie</td>
<td>searchString=java servlet cookies; numResults=10; searchEngine=infoseek</td>
</tr>
</tbody>
</table>
## Servlet Example: Showing Request Headers

**Request Method:** GET  
**Request URI:** /servlet/hall.ShowRequestHeaders  
**Request Protocol:** HTTP/1.1

<table>
<thead>
<tr>
<th>Header Name</th>
<th>Header Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accept</td>
<td>image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/vnd.ms-excel, application/msword, application/vnd.ms-powerpoint, <em>/</em></td>
</tr>
<tr>
<td>Accept-Language</td>
<td>en-us</td>
</tr>
<tr>
<td>Accept-Encoding</td>
<td>gzip, deflate</td>
</tr>
<tr>
<td>User-Agent</td>
<td>Mozilla/4.0 (compatible; MSIE 4.01; Windows NT)</td>
</tr>
<tr>
<td>Host</td>
<td>webdev.apl.jhu.edu</td>
</tr>
<tr>
<td>Connection</td>
<td>Keep-Alive</td>
</tr>
</tbody>
</table>
Java Servlet Technology Intro – Response Codes and Errors

```java
String url =
    response.encodeURL(searchSpec.makeURL(searchString,
        numResults));

    response.sendRedirect(url);
    return;
}

response.sendError(response.SC_NOT_FOUND,
    "No recognized search engine specified.");
```
# 1. Java Servlet Technology

## Java Servlet Technology Intro

<table>
<thead>
<tr>
<th>Status Code</th>
<th>Associated Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>Continue</td>
</tr>
<tr>
<td>101</td>
<td>Switching Protocols</td>
</tr>
<tr>
<td>200</td>
<td>OK</td>
</tr>
<tr>
<td>201</td>
<td>Created</td>
</tr>
<tr>
<td>202</td>
<td>Accepted</td>
</tr>
<tr>
<td>203</td>
<td>Non-Authortitative Information</td>
</tr>
<tr>
<td>204</td>
<td>No Content</td>
</tr>
<tr>
<td>205</td>
<td>Reset Content</td>
</tr>
<tr>
<td>206</td>
<td>Partial Content</td>
</tr>
<tr>
<td>300</td>
<td>Multiple Choices</td>
</tr>
<tr>
<td>301</td>
<td>Moved Permanently</td>
</tr>
<tr>
<td>302</td>
<td>Found</td>
</tr>
<tr>
<td>303</td>
<td>See Other</td>
</tr>
<tr>
<td>304</td>
<td>Not Modified</td>
</tr>
<tr>
<td>305</td>
<td>Use Proxy</td>
</tr>
<tr>
<td>307</td>
<td>Temporary Redirect</td>
</tr>
<tr>
<td>400</td>
<td>Bad Request</td>
</tr>
<tr>
<td>401</td>
<td>Unauthorized</td>
</tr>
<tr>
<td>403</td>
<td>Forbidden</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
</tr>
<tr>
<td>405</td>
<td>Method Not Allowed</td>
</tr>
<tr>
<td>406</td>
<td>Not Acceptable</td>
</tr>
<tr>
<td>407</td>
<td>Proxy Authentication Required</td>
</tr>
<tr>
<td>408</td>
<td>Request Timeout</td>
</tr>
<tr>
<td>409</td>
<td>Conflict</td>
</tr>
<tr>
<td>410</td>
<td>Gone</td>
</tr>
<tr>
<td>411</td>
<td>Length Required</td>
</tr>
<tr>
<td>412</td>
<td>Precondition Failed</td>
</tr>
<tr>
<td>413</td>
<td>Request Entity Too Large</td>
</tr>
<tr>
<td>414</td>
<td>Request URI Too Long</td>
</tr>
<tr>
<td>415</td>
<td>Unsupported Media Type</td>
</tr>
<tr>
<td>416</td>
<td>Requested Range Not Satisfiable</td>
</tr>
<tr>
<td>417</td>
<td>Expectation Failed</td>
</tr>
<tr>
<td>500</td>
<td>Internal Server Error</td>
</tr>
<tr>
<td>501</td>
<td>Not Implemented</td>
</tr>
<tr>
<td>502</td>
<td>Bad Gateway</td>
</tr>
<tr>
<td>503</td>
<td>Service Unavailable</td>
</tr>
<tr>
<td>504</td>
<td>Gateway Timeout</td>
</tr>
<tr>
<td>505</td>
<td>HTTP Version Not Supported</td>
</tr>
</tbody>
</table>
### 1. Java Servlet Technology

#### Java Servlet Technology Intro – Response Headers

<table>
<thead>
<tr>
<th>Header</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow</td>
<td>Refresh</td>
</tr>
<tr>
<td>Content-Encoding</td>
<td>Server</td>
</tr>
<tr>
<td>Content-Length</td>
<td>Set-Cookie</td>
</tr>
<tr>
<td>Content-Type</td>
<td>WWW-Authenticate</td>
</tr>
<tr>
<td>Date</td>
<td></td>
</tr>
<tr>
<td>Expires</td>
<td></td>
</tr>
<tr>
<td>Last-Modified</td>
<td></td>
</tr>
<tr>
<td>Location</td>
<td></td>
</tr>
</tbody>
</table>
There are tech issues with HTTP, because it is a "stateless" protocol.

Usually this may be solved as it follows:

1. **Cookies.** Most used way to transform HTTP from “stateless” to “state-full”. The objects associated to the cookie are NOT going through the network and are stored on the web server side.

2. **URL Rewriting.** For each HTTP request there is attached in the end or the URL an unique char string generated by the web server.

3. **Hidden form fields.** Are used HTML tags such as:

```html
<INPUT TYPE="HIDDEN" NAME="session" VALUE="...">
```
//create cookie 1 - implicit value in seconds of cookie is within the session
Cookie userCookie = new Cookie("CookieGigel", "CucuBau");
response.addCookie(userCookie);

//create cookie 2 - is per year
Cookie userCookie2 = new Cookie("Cookielon", "IONIONION");
userCookie2.setMaxAge(SECONDS_PER_YEAR);
response.addCookie(userCookie2);

Cookie[] cookies = request.getCookies();
if (cookies != null) {
    for(int i=0; i<cookies.length; i++) {
        Cookie cookie = cookies[i];
        if ("CookieGigel".equals(cookie.getName())) {...}
public void processRequest(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

    HttpSession session = request.getSession(true);
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    String title = "Show Session"; String heading;
    Integer accessCount = new Integer(0);

    if (session isNew()) {
        heading = "Welcome, Newcomer";
    } else {
        heading = "Welcome Back";
        Integer oldAccessCount = (Integer)session.getAttribute("accessCount");
        if (oldAccessCount != null) {
            accessCount = new Integer(oldAccessCount.intValue() + 1);
        }
    }

    session.setAttribute("accessCount", ""+accessCount);
Section Conclusion

Fact: DAD needs Web Programming

In few samples it is simple to remember: Java Servlet Programming with HTTP protocol analysis in real time for request headers, responses’ codes and headers, session tracking – generates standards HTML pages as entering gate for distributed computing and systems.
Java Server Page – JSP & Java Servlet Technology Intro

Communicate & Exchange Ideas
Questions & Answers!

But wait...
There’s More!
Thanks!

DAD – Distributed Application Development
End of Lecture 6 – Section 3