C.O.D.E Session

Basic Web Development
Presented by Aaron Korver

Agenda

- Request/Response Cycle
- HTML/DOM
- Servlets/JSP/JSTL

The actors

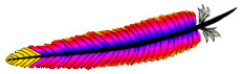
- The Browser
  - GETs pages via a URL
  - POSTs data to a URL
  - Parses HTML to display to the user
  - Not all browsers are equal!

The actors

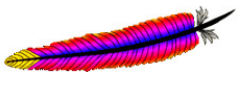
- The Internet
  - Transfers data via the HTTP protocol
  - Also uses many other protocols, but we only care about HTTP at the moment.
  - NOT a series of tubes…

The actors

- The Server
  - Many responsibilities
  - Serves files
  - Executes scripts
  - Responds to requests
  - etc etc etc



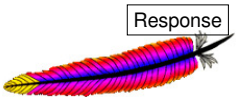
The Request



The Request

- GET /dumprequest.html HTTP/1.1
- Host: fbfs.com
- User-Agent: Mozilla/5.0 (Macintosh; U; Intel Mac OS X; en-US; rv:1.8.1.12) Gecko/20080201 Firefox/2.0.0.12
- Accept: text/xml,
- application/xml,
- application/xhtml+xml,
- text/html;q=0.9,
- text/plain;q=0.8,
- image/png,
- */*;q=0.5
- Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
- Accept-Encoding: gzip,deflate
- Accept-Language: en-us,en;q=0.5
- Connection: keep-alive
- Keep-Alive: 300

The Response



The Response

- Contains a stream of data.
- For Web 1.0 was mainly text (html)
- For Web 2.0 is becoming more binary (Videos, Images, Music, etc)

Everything is STATELESS

- Requests and Responses are a single transaction.
- Data is never shared between requests.

Redirect vs Forward

- Redirect
  - Browser asks server for page A
  - Server tells browser ask for page B instead.
  - Browser asks for page B
  - Server returns page B
- Forward
  - Browser asks client for page A
  - Server gets page A which says to forward to page B
  - Server gets page B
  - Server returns page B

Redirect vs Forward

- Why would I use one or the other?
  - A forward happens on the server, so only one request/response. Thus faster.
  - A redirect will change the URL in the browser, so hitting “refresh” will go to the new URL.
  - All depends on the circumstances.

GET vs POST Form Submission

- GETs places all form data in the URL.
  - http://www.google.com/search?q=GET+vs+POST&ie=utf-8&oe=utf-8&aq=t&rls=org.mozilla:en-US:official&client=firefox-a
- POST places all form data into a message in the request.
- Officially….
  - If the processing of a form is idempotent (i.e. it has no lasting observable effect on the state of the world), then the form method should be GET.
    - From the HTTP 4.0 Specification

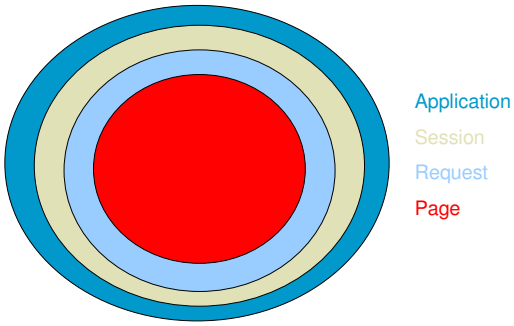
HTML

- Hyper Text Markup Language
  - Browsers use this for layout purposes.
  - An application of SGML (Standard Generalized Markup Language)
  - Contains elements and attributes
  - Mainly delivered over HTTP

DOM

- Document Object Model
  - Treats elements and attributes as objects
  - Has an event model that allows code to hook into
  - Three levels (1,2,3) 2 is the most widely supported
  - DOM is the reason for most of the incompatibility between browsers

Web App Variable Scope



Session

- Variables that are placed into the Session scope are stored on the server.
- The session spans multiple requests.
- A token is stored on the client's machine to track the sessionID.
- Normally, the session ends when the user closes the browser (or a session timeout occurs)

Static vs. Dynamic Pages

- Static pages are html content that is the same for every user
- Dynamic pages contain content that is generated on the server and varies for each user
- The Servlet is Java's foremost method for generating this content (but not the only way)

Servlets

- The 1.0 spec was released in 1997
- Currently we are at spec 2.5 which was released in Sept. 2005
- Servlets are created by the web container (i.e. the server)
- Each request is handled by a separate thread.
  - Which means servlets MUST BE THREADSAFE!

Servlet Lifecycle

- Container creates the Servlet object.
- Container calls init().
- As requests come in, the container gives them to the proper servlet to be handled.
- At the end, the destroy() method is called.

Special Servlet Objects

- ServletContext
  - Shared by all servlets
  - Used to obtain application level information or container details
- ServletConfig
  - Each servlet gets one
  - Used for init parameters

Listeners

- Sometimes you want to act upon events happening inside the container.

Object	Event
Web Context	Initialization and destruction
	Attribute added, removed, or replaced
Session	Creation, invalidation, and timeout
	Attribute added, removed, or replaced

Servlet Example

```
import java.io.*;

import javax.servlet.http.*;
import javax.servlet.*;

public class HelloServlet extends HttpServlet {
  public void doGet (HttpServletRequest request,
                     HttpServletResponse response)
    throws ServletException, IOException
  {response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    out.println("<html>");
    out.println("<body>");
    out.println("<font color=/"green/">");
    out.println("Hello, " + request.getParameter("userName"));
    out.println("</font>");
    out.println("</body>");
    out.println("</html>");
    out.close();
  }
```

Alas, Servlets got old…fast…

- Well, they didn't get old, but doing out.print() over and over did.
- So a better way was born.
- Enter the Java Server Page (JSP)

JSP

- The most important thing to remember about JSPs is this….
  - JSPs == Servlets
- Why? Or better…How?
  - On the server lives a JSP compiler. It takes your JSP page and creates a Java class from it that implements the Servlet API. So every JSP becomes a Servlet on the server.
- Thus, all JSPs are executed on the server. This is Mucho Important to understand!!

JSP example

```
<html>
    <body>
        <font color="green">Hello,
            <%= request.getParameter("userName") %>
        </font>
    </body>
</html>
```

JSTL and EL

- JavaServerPages Standard Tag Library
- Expression Language
- After a few years of JSP and the <% %>, developers saw common functionality being implemented.
  - So a standard tag library was created.
  - Does things like formatting, flow control, and xml parsing

Tag Libraries

- Remember that a JSP == Servlet right?
- And a Servlet is just a Java class.
- And Java classes can call other Java classes.
- So a tag lib, is simply a Java class that is called on the server to do custom stuff.
- There is a Tag Lib API that the class must implement. And some special configuration to use the tag.

JSTL and EL example

```
<html>
    <body>
        <font color="green">
            Hello, <c:out value="${param.userName}">
        </font>
    </body>
</html>
```

More info!

- http://courses.coreservlets.com/Course-Materials/csajsp2.html
- http://java.sun.com/products/jsp/tutorial/TagLibrariesTOC.html
- http://java.sun.com/products/jsp/jstl/1.1/docs/tlddocs/index.html