JSP
Intro and Overview
Agenda

• Understanding the need for JSP
• Evaluating the benefits of JSP
• Comparing JSP to other technologies
• Avoiding JSP misconceptions
• Understanding the JSP lifecycle
• Installing JSP pages
• Looking at JSP in the real world
The Need for JSP

• With servlets, it is easy to
  – Read form data
  – Read HTTP request headers
  – Set HTTP status codes and response headers
  – Use cookies and session tracking
  – Share data among servlets
  – Remember data between requests
  – Get fun, high-paying jobs

• But, it sure is a pain to
  – Use those println statements to generate HTML
  – Maintain that HTML
The JSP Framework

• **Idea:**
  – Use regular HTML for most of page
  – Mark servlet code with special tags
  – Entire JSP page gets translated into a servlet (once), and servlet is what actually gets invoked (for each request)

• **Example:**
  ```html
  <!DOCTYPE ...>
  <HTML>
  <HEAD>
  <TITLE>Order Confirmation</TITLE>
  <LINK REL=STYLESHEET
       HREF="JSP-Styles.css"
       TYPE="text/css">
  </HEAD>
  <BODY>
  <H2>Order Confirmation</H2>
  Thanks for ordering
  <I><%= request.getParameter("title") %></I>!
  </BODY></HTML>
  ```
Benefits of JSP

• Although JSP technically can't do anything servlets can't do, JSP makes it easier to:
  – Write HTML
  – Read and maintain the HTML

• JSP makes it possible to:
  – Use standard HTML tools such as Macromedia DreamWeaver or Adobe GoLive.
  – Have different members of your team do the HTML layout than do the Java programming

• JSP encourages you to
  – Separate the (Java) code that creates the content from the (HTML) code that presents it
Advantages of JSP Over Competing Technologies

• **Versus ASP or ColdFusion**
  - Better language for dynamic part
  - Portable to multiple servers and operating systems

• **Versus PHP**
  - Better language for dynamic part
  - Better tool support

• **Versus pure servlets**
  - More convenient to create HTML
  - Can use standard tools (e.g., DreamWeaver)
  - Divide and conquer
  - JSP programmers still need to know servlet programming
Advantages of JSP (Continued)

- **Versus Velocity or WebMacro**
  - Standard

- **Versus client-side JavaScript (in browser)**
  - Capabilities mostly do not overlap with JSP, but
    - You control server, not client
    - Richer language

- **Versus server-side JavaScript (e.g., LiveWire, BroadVision)**
  - Richer language

- **Versus static HTML**
  - Dynamic features
  - Adding dynamic features no longer “all or nothing" decision
Setting Up Your Environment

- Set your CLASSPATH. Not.
- Compile your code. Not.
- Use packages to avoid name conflicts. Not.
- Put JSP page in special directory. Not.
  - `install_dir\webapps\ROOT\` (HTML and JSP -- Tomcat)
  - `install_dir\servers\default\default-app` (JRun)
- Use special URLs to invoke JSP page. Not.
  - Use same URLs as for HTML pages (except for file extensions)
- Caveats
  - Previous rules about CLASSPATH, install dirs, etc., still apply to regular Java classes used by a JSP page
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
<title>JSP Expressions</title>
<meta name="keywords" content="JSP,expressions,JavaServer Pages">
<meta name="description" content="A quick example of JSP expressions.">
<link rel="stylesheet" type="text/css" href="JSP-Styles.css">
</head>
</html>
Example (Continued)

<BODY>
<H2>JSP Expressions</H2>
<UL>
    <LI>Current time: <%= new java.util.Date() %>
    <LI>Server: <%= application.getServerInfo() %>
    <LI>Session ID: <%= session.getId() %>
    <LI>The <CODE>testParam</CODE> form parameter: <%= request.getParameter("testParam") %>
</UL>
</BODY>
Example: Result

• If location was
  C:\jakarta-tomcat-xx\webapps\ROOT\jsp-scripting\Expressions.jsp
• URL would be
  – http://localhost/jsp-scripting/Expressions.jsp

JSP Expressions
• Current time: Mon Oct 31 15:49:45 EST 2005
• Server: Apache Tomcat/5.5.9
• Session ID: 0388502C8442AE5A21D2512DEA09C4A4
• The testParam form parameter: null
Translation/Request Time Confusion

• What happens at page translation time?
  – JSP constructs get translated into servlet code.

• What happens at request time?
  – Servlet code gets executed. No interpretation of JSP occurs at request time. The original JSP page is totally ignored at request time; only the servlet that resulted from it is used.

• When does page translation occur?
  – Typically, the first time JSP page is accessed after it is modified. This should never happen to real user (developers should test all JSP pages they install).
  – Page translation does not occur for each request.
## The JSP Lifecycle

<table>
<thead>
<tr>
<th>JSP page translated into servlet</th>
<th>Page first written</th>
<th>Request #1</th>
<th>Request #2</th>
<th>Request #3</th>
<th>Request #4</th>
<th>Request #5</th>
<th>Request #6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

- **Servlet compiled**
- **Servlet instantiated and loaded into server's memory**
- **init (or equivalent) called**
- **doGet (or equivalent) called**
JSP/Servers in the Real World: Airlines

- Delta Airlines
- United Airlines
- AirTran
- American Airlines
- British Airways
- KLM
- Air China
- Saudi Arabian Airlines
- Iceland Air
JSP/Servlets in the Real World: Travel Sites

- Travelocity.com
- Orbitz.com
- HotWire.com
- Hotels.com
JSP/Servlets in the Real World: Financial Services

- American Century
- Vanguard
- Fidelity
- NY Stock Exchange
- First USA Bank
- Royal Bank of Scotland
- Banco Popular de Puerto Rico
- Bank of America
- China Construction Bank
JSP/Servlets in the Real World: Retail

- Sears.com
- Walmart.com
- SamsClub.com
- Macys.com
- llbean.com
- Kohls.com
- Ikea.com
- REI.com
- Longaberger.com
JSP/Servlets in the Real World: Entertainment

- WarnerBrothers.com
- Billboard.com
- E! (eonline.com)
- PBS.org
JSP/Servlets in the Real World: Military and Federal Government

- DHS
- TSA
- FAA
- CIA
- NSA
- GSA
- IRS
- Army
- Navy
JSP/Servlets in the Real World: Sports

- Baltimore Orioles
- Major League Baseball (mlb.com)
- NHL.com
- Nascar.com
- The Sports Authority
- Dicks Sporting Goods
JSP/Servlets in the Real World: Search/Portals

- Google
- Half.ebay.com
- netscape.com
- excite.com
- dice.com
Summary

• JSP makes it easier to create and maintain HTML, while still providing full access to servlet code
  
• JSP pages get translated into servlets
  – It is the servlets that run at request time
  – Client does not see anything JSP-related

• You still need to understand servlets
  – Understanding how JSP really works
  – Servlet code called from JSP
  – Knowing when servlets are better than JSP
  – Mixing servlets and JSP

• Other technologies use similar approach, but aren't as portable and don't let you use Java for the "real code"