An Introduction to XML and Web Technologies

Programming Web Applications with JSP

Anders Møller & Michael I. Schwartzbach

Objectives

- How to program Web applications using **JSP**
- How to extend the JSP syntax using **tag libraries**
- The relation to Servlets

The JSP Framework

- Servlets make heavy use of Java and require **sophisticated programming**
- JSP views a Web application as a collection of **active pages**
- The pages are **HTML with *snippets of code***
- JSP pages are **translated into servlets**

A Tiny Example

```
<% response.addDateHeader("Expires", 0); %>
<html>
  <head><title>JSP</title></head>
  <body>
    <h1>Hello World!</h1>
    <%! int hits = 0; %>
    You are visitor number
    <% synchronized(this) { out.println(++hits); } %>
    since the last time the service was restarted.
    <p>
    This page was last updated:
    <%= new java.util.Date().toLocaleString() %>
  </body>
</html>
```

Hello World!

You are visitor number 43 since the last time the service was restarted.

This page was last updated: 28-02-2005 13:56:49

JSP Templates

- A text file with snippets of Java code:
  - *expressions*
  - *statements*
  - *declarations*
- JSP directives
- Implicitly declared variables:
  - `HttpServletRequest request;`
  - `HttpServletResponse response;`
  - `HttpSession session;`
  - `ServletContext application;`
  - `ServletConfig config;`
  - `JspWriter out;`
  - `PageContext pageContext;`

JSP Expressions

```
<html>
  <head><title>Addition</title></head>
  <body>
    The sum of <%= request.getParameter("x") %>
    and <%= request.getParameter("y") %> is
    <%= Integer.parseInt(request.getParameter("x")) +
        Integer.parseInt(request.getParameter("y")) %>
  </body>
</html>
```


JSP Statements

```
<html>
  <head><title>Numbers</title></head>
  <body>
    <ul>
      <% int n =
            Integer.parseInt(request.getParameter("n"));
         for (int i=0; i<n; i++)
           out.println("<li>"+i+"</li>");
      %>
    </ul>
  </body>
</html>
```


JSP Declarations

```
<%! int add(String x, String y) {
      return Integer.parseInt(x)+Integer.parseInt(y);
    }
%>
<html>
  <head><title>Addition</title></head>
  <body>
    The sum of <%= request.getParameter("x") %>
    and <%= request.getParameter("y") %> is
    <%= add(request.getParameter("x"),
            request.getParameter("y")) %>
  </body>
</html>
```

Be Careful About Declarations (1/2)

```
<% response.addDateHeader("Expires", 0); %>
<html>
  <head><title>JSP</title></head>
  <body>
    <h1>Hello World!</h1>
    <%! int hits = 0; %>
    You are visitor number
    <% synchronized(this) { out.println(++hits); } %>
    since the last time the service was restarted.
    <p>
    This page was last updated:
    <%= new java.util.Date().toLocaleString() %>
  </body>
</html>
```


Be Careful About Declarations (2/2)

```
<% response.addDateHeader("Expires", 0); %>
<html>
  <head><title>JSP</title></head>
  <body>
    <h1>Hello World!</h1>
    <% int hits = 0; %>
    You are visitor number
    <% synchronized(this) { out.println(++hits); } %>
    since the last time the service was restarted.
    <p>
    This page was last updated:
    <%= new java.util.Date().toLocaleString() %>
  </body>
</html>
```

This page counter is always 1...

JSP Directives

header.jsp

```
<html>
  <head><title><%= title %></title></head>
  <body>
```

footer.jsp

```
  </body>
</html>
```

```
<%! String title = "Addition"; %>
<%@ include file="header.jsp" %>
  The sum of <%= request.getParameter("x") %>
  and <%= request.getParameter("y") %> is
  <%= Integer.parseInt(request.getParameter("x")) +
      Integer.parseInt(request.getParameter("y")) %>
<%@ include file="footer.jsp" %>
```


Other Directives

- `contentType="`*`type`*`"`
- `info="`*`description`*`"`
- `errorPage="`*`path`*`"`
- `isErrorPage="`*`boolean`*`"`
- `import="`*`package`*`"`

Using Error Pages

```
<%@ page errorPage="error.jsp" %>
<html>
  <head><title>Division</title></head>
  <body>
    <% int n = Integer.parseInt(request.getParameter("n")); %>
    <% int m = Integer.parseInt(request.getParameter("m")); %>
    <%= n %>/<%= m %> equals <%= n/m %>
  </body>
</html>
```

```
<%@ page isErrorPage="true" %>
<html>
  <head><title>Error</title></head>
  <body>
     Something bad happened:
     <%= exception.getMessage() %>
  </body>
</html>
```


Translation into Servlets

```
<% response.addDateHeader("Expires", 0); %>
<html>
  <head><title>JSP</title></head>
  <body>
    <h1>Hello World!</h1>
    <%! int hits = 0; %>
    You are visitor number
    <% synchronized(this) { out.println(++hits); } %>
    since the last time the service was restarted.
    <p>
    This page was last updated:
    <%= new java.util.Date().toLocaleString() %>
  </body>
</html>
```


The Generated Servlet (1/5)

```
package org.apache.jsp;

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

public final class hello_jsp
  extends org.apache.jasper.runtime.HttpJspBase
  implements org.apache.jasper.runtime.JspSourceDependent {

  int hits = 0;
  private static java.util.Vector _jspx_dependants;

  public java.util.List getDependants() {
    return _jspx_dependants;
  }
```


The Generated Servlet (2/5)

```
public void _jspService(HttpServletRequest request,
                        HttpServletResponse response)
    throws java.io.IOException, ServletException {

  JspFactory _jspxFactory = null;
  PageContext pageContext = null;
  HttpSession session = null;
  ServletContext application = null;
  ServletConfig config = null;
  JspWriter out = null;
  Object page = this;
  JspWriter _jspx_out = null;
  PageContext _jspx_page_context = null;
```

The Generated Servlet (3/5)

```
    try {
      _jspxFactory = JspFactory.getDefaultFactory();
      response.setContentType("text/html");
      pageContext = _jspxFactory.getPageContext(
                      this, request, response,
                      null, true, 8192, true);
      _jspx_page_context = pageContext;
      application = pageContext.getServletContext();
      config = pageContext.getServletConfig();
      session = pageContext.getSession();
      out = pageContext.getOut();
      _jspx_out = out;
```


The Generated Servlet (4/5)

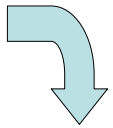
```
      response.addDateHeader("Expires", 0);
      out.write("\n");
      out.write("<html><head><title>JSP</title></head>\n");
      out.write("<body><h1>Hello World!</h1>\n");
      out.write("\n");
      out.write("You are visitor number ");
      synchronized(this) { out.println(++hits); }
      out.write("\n");
      out.write("since the last time the service was restarted.\n");
      out.write("<p>\n");
      out.write("This page was last updated: ");
      out.print(new java.util.Date().toLocaleString());
      out.write("\n");
      out.write("</body></html>\n");
```


The Generated Servlet (5/5)

```
    } catch (Throwable t) {
      if (!(t instanceof SkipPageException)){
        out = _jspx_out;
        if (out != null && out.getBufferSize() != 0)
          out.clearBuffer();
        if (_jspx_page_context != null)
          _jspx_page_context.handlePageException(t);
      }
    } finally {
      if (_jspxFactory != null)
        _jspxFactory.releasePageContext(_jspx_page_context);
    }
  }
}
```


Literal Translation

```
<% if (Math.random() < 0.5) { %>
  Have a <b>nice day!
<% } else { %>
  Have a <b>lousy day!
<% } %>
</b>
```



```
if (Math.random() < 0.5) {
     out.write("\n");
     out.write("  Have a <b>nice day!\n");
} else {
     out.write("\n");
     out.write("  Have a <b>lousy day!\n");
}
out.write("</b>\n");
```

Problematic Whitespace

```
<%@ page contentType="text/plain" %>
<%@ page info="A Tiny Test" %>
This is a tiny test
```

```
This is a tiny test
```


Advanced Features

- XML version of JSP
- The expression language
- Tag files
- JSTL
- The Model-View-Controller pattern

JSP Pages Are Not XML

```
<html>
  <head><title>JSP Color</title></head>
  <body bgcolor=<%= request.getParameter("color") %>>
    <h1>Hello World!</h1>
    <%! int hits = 0; %>
    You are visitor number
    <% synchronized(this) { out.println(++hits); } %>
    since the last time the service was restarted.
    <p>
    This page was last updated:
    <%= new java.util.Date().toLocaleString() %>
  </body>
</html>
```

- This page generates HTML, not XHTML
- <%...%> is not well-formed XML

XML Version of JSP

```
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" version="2.0"
          xmlns="http://http://www.w3.org/1999/xhtml">
  <jsp:directive.page contentType="text/html"/>
  <jsp:scriptlet>
    response.addDateHeader("Expires", 0);
  </jsp:scriptlet>
  <html>
    <head><title>JSP</title></head>
    <jsp:element name="body">
      <jsp:attribute name="bgcolor">
        <jsp:expression>
          request.getParameter("color")
        </jsp:expression>
      </jsp:attribute>
      <h1>Hello World!</h1>
      <jsp:declaration>
        int hits = 0;
      </jsp:declaration>
      You are visitor number
      <jsp:scriptlet>
        synchronized(this) { out.println(++hits); }
      </jsp:scriptlet>
      since the last time the service was restarted.
      <p/>
      This page was last updated:
      <jsp:expression>
        new java.util.Date().toLocaleString()
      </jsp:expression>
    </jsp:element>
  </html>
</jsp:root>
```

- Uses `<jsp:...>`
- No schema seems to be available
- No validation of the output
- No validation of Java code
- but it's there...

The Expression Language

- We want to **avoid explicit Java code** in JSP templates
- The syntax **${exp}** may be used in
  - template text
  - attribute values in markup
- The expression may access
  - variables in the various scopes
  - implicit objects, such as `param`
- The usual operators are available

An Expression Example

```
<html>
  <head><title>Addition</title></head>
  <body bgcolor="${param.color}">
    The sum of ${param.x} and ${param.y} is ${param.x+param.y}
  </body>
</html>
```


Tag Files

Define **abstractions** as **new tags**

wrap.tag:

```
<%@ tag %>
<%@ attribute name="title" required="true" %>
<html>
  <head><title>${title}</title></head>
  <body>
    <jsp:doBody/>
  </body>
</html>
```

```
<%@ taglib prefix="foo" tagdir="/WEB-INF/tags" %>
<foo:wrap title="Addition">
  The sum of ${param.x} and ${param.y} is
  ${param.x+param.y}
</foo:wrap>
```


Content as a Value: A New Image Tag

image.tag:

```
<%@ tag %>
<jsp:doBody var="src"/>
<img src="http://www.brics.dk/ixwt/images/${src}"/>
```

```
<%@ taglib prefix="foo" tagdir="/WEB-INF/tags" %>
<foo:image>widget.jpg</foo:image>
```

Declaring Variables: A Date Context Tag

date.tag:

```
<%@ tag import="java.util.*" %>
<%@ variable name-given="date" %>
<%@ variable name-given="month" %>
<%@ variable name-given="year" %>
<% Calendar cal = new GregorianCalendar();
   int date = cal.get(Calendar.DATE);
   int month = cal.get(Calendar.MONTH)+1;
   int year = cal.get(Calendar.YEAR);
   jspContext.setAttribute("date", String.valueOf(date));
   jspContext.setAttribute("month", String.valueOf(month));
   jspContext.setAttribute("year", String.valueOf(year));
%>
<jsp:doBody/>
```


Using the Date Context

```
<%@ taglib prefix="foo" tagdir="/WEB-INF/tags" %>
<foo:date>
  In the US today is
  ${month}/${date}/${year},
  but in Europe it is
  ${date}/${month}/${year}.
</foo:date>
```


Quick Poll Tags (1/2)

```
<%@ taglib prefix="poll" tagdir="/WEB-INF/tags/poll" %>
<poll:quickpoll title="Quickies" duration="3600">
  <poll:question>
    The question has been set to "${question}".
  </poll:question>
  <poll:ask>
     ${question}?
     <select name="vote">
       <option>yes
       <option>no
     </select>
     <input type="submit" value="vote">
  </poll:ask>
```


Quick Poll Tags (2/2)

```
  <poll:vote>
     You have voted ${vote}.
  </poll:vote>
  <poll:results>
     In favor: ${yes}<br>
     Against: ${no}<br>
     Total: ${total}
  </poll:results>
  <poll:timeout>
     Sorry, the polls have closed.
  </poll:timeout>
</poll:quickpoll>
```

See the tag files in the book...

Tag Libraries

- Libraries of tags capturing common patterns:
  - pagination of large texts
  - date and times
  - database queries
  - regular expressions
  - HTML scraping
  - bar charts
  - cookies
  - e-mail
  - WML
  - ...

JSTL 1.1

- JSP Standard Tag Library covers:
  - assigning to variables
  - writing to the output stream
  - catching exceptions
  - conditionals
  - iterations
  - URL construction
  - string formatting
  - SQL queries
  - XML manipulation

Selecting Some Recipes

Title	Calories	Fat	Carbohydrates	Protein	Alcohol
Beef Parmesan with Garlic Angel Hair Pasta	1167	23%	45%	32%	0%
Linguine Pescadoro	532	12%	59%	29%	0%
Zuppa Inglese	612	49%	45%	4%	2%

Using JSTL for the Menu

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/xml" prefix="x"%>
<c:import url="http://www.brics.dk/ixwt/recipes.xml" var="xml"/>
<x:parse xml="${xml}" var="recipes" scope="session"/>
<html>
  <head><title>Select Some Recipes</title></head>
  <body>
    <form method="post" action="show.jsp">
      <x:forEach select="$recipes//recipe">
        <c:set var="id"><x:out select="@id"/></c:set>
        <input type="checkbox" name="selected" value="${id}"/>
        <x:out select="title/text()"/>
        <br/>
      </x:forEach>
      <input type="submit" value="Select"/>
    </form>
  </body>
</html>
```

Using JSTL for the Table (1/3)

```
<html>
 <head><title>Nutrition Overview</title></head>
 <body>
  <table border="1">
   <tr>
    <td>Title</td>
    <td>Calories</td>
    <td>Fat</td>
    <td>Carbohydrates</td>
    <td>Protein</td>
    <td>Alcohol</td>
   </tr>
```


Using JSTL for the Table (2/3)

```
<x:forEach select="$recipes//recipe">
  <c:forEach var="id" items="${paramValues.selected}">
   <x:if select="@id=$id">
    <tr>
     <td>
      <x:out select=".//title"/>
     </td>
     <td align="right">
      <x:out select=".//nutrition/@calories"/>
     </td>
     <td align="right">
      <x:out select=".//nutrition/@fat"/>
     </td>
     <td align="right">
      <x:out select=".//nutrition/@carbohydrates"/>
     </td>
```


Using JSTL for the Table (3/3)

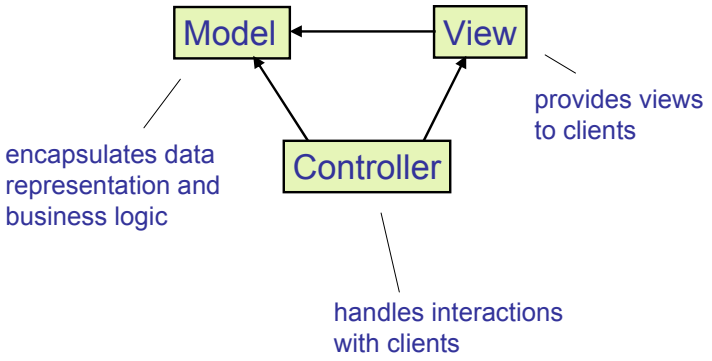
```
      <td align="right">
       <x:out select=".//nutrition/@protein"/>
      </td>
      <td align="right">
       <x:out select=".//nutrition/@alcohol"/>
       <x:if select="not(.//nutrition/@alcohol)">
        0%
       </x:if>
      </td>
     </tr>
    </x:if>
   </c:forEach>
  </x:forEach>
 </table>
</body>
</html>
```


Evaluation of Tags

- Make Web applications available to a wider range of developers
- May be used to structure applications

- A myriad of domain-specific languages
- Brittle implementation, hard to debug

The Model-View-Controller Pattern



The Benefit of MVC

Separation of concerns!

(high cohesion – low coupling)

Using MVC

- Controller: one servlet
- View: JSP pages
- Model: pure Java (e.g. JavaBeans)

[Example in the book: Business Card Server]

Summary

- **JSP templates** are HTML/XHTML pages with embedded code
- The simple **expression language** is often sufficient in place of full-blown Java code
- **Tag files and libraries** allow code to be hidden under a tag-like syntax
- **MVC** provides separation of programming and HTML design tasks

Essential Online Resources

- Sun's home page for JSP:
  http://java.sun.com/products/jsp/
- JSTL:
  http://java.sun.com/products/jsp/jstl/