

JSP: Java Server Pages

Introduction

- File extension: .jsp
- Mix of html and "scripting elements" used to generate a servlet, which, in turn, generates the html for the client
- Enclose in `<% ... %>` (or xml syntax: `<jsp:scriptlet>...</jsp:scriptlet>`)
- Access to all of java
 - networking, multithreading, database connectivity, serialization, RMI, ...
- Scalability:
 - once requested, jsp page kept in memory
 - each invocation in a separate thread (synchronization req'd)
- Extensibility:
 - custom tags can be defined (and placed in libraries, then reused)

Predefined Variables

- `HttpServletRequest request`
- `HttpServletResponse response`
- `javax.servlet.jsp.JspWriter out`
- `HttpSession session`
- `ServletContext application`
- `javax.servlet.jsp.PageContext pageContext`
- `javax.servlet.jsp.JspPage page`
- ...

CSE 794R/ECE 694R

3

Basic Scripting Elements

- **Scriptlets** `<%...%>`
 - bits of java code:
`<% out.println("<h1> Hello </h1>"); %>`
- **Expressions** `<%=...%>`
 - value, which is written: `<%= new java.util.Date() %>`
 - i.e., just gets wrapped in `out.println(...)`
 - any type can be written (`toString()`)
- **Declarations** `<%!...%>`
 - declare (& initialize) "global" variables and methods
`<%! public java.util.Date PrintDate()
{ return (new java.util.Date());} %>`
- **Comments** `<%--...--%>`

CSE 794R/ECE 694R

4

More JSP Features

- Can escape in and out of scripting mode
 - control flow is preserved! (easier than echoing)
`<% if () { %> html stuff <% } else { ... %>`
 - (not the case with Javascript)
 - static html gets wrapped in `out.println(...)`
- Exception blocks often done this way
`<% try { %> ...`
`<% } catch (Exception e) { %> ...`
`<% } %>`
 - and it's important to handle exceptions!

CSE 794R/ECE 694R

5

JSP Engine

- On first page request:
 - translate jsp (dyn + static) into java servlet
 - compile java code into class file
 - load classes
 - run servlet and return resulting html
- On subsequent page requests:
 - just run servlet and return resulting html

CSE 794R/ECE 694R

6

Consequences

- Persistence
 - declarations result in shared data (scriptlets don't)
 - useful for keeping info between requests
 - e.g., session info, database connection pooling, etc.
- Performance
 - noticeable performance hit the first time
 - fast afterwards

CSE 794R/ECE 694R

7

JSP Engine cont.

- Threading model
 - default: THREAD PER REQUEST!!!
 - persistent data means danger!
 - use java synchronization to control
- Lifecycle
 - `jspInit()` runs once (on first request)
 - guaranteed to execute before first request
 - `jspDestroy()` runs once (when Servlet unloaded from JVM)
 - not guaranteed to be called

CSE 794R/ECE 694R

8

PageContext

- Keeps track of attributes at four levels:
 - Application
 - Session
 - Request
 - Page
- `javax.servlet.jsp.PageContext pageContext`
 - `findAttribute(name)`
 - `getAttribute(name, scope)`
 - `removeAttribute(name, scope)`
 - `setAttribute(name, value, scope)`
 - `getPage, getRequest, getResponse, getServletContext, getSession, etc.`

CSE 794R/ECE 694R

9

Other Scripting Elements

- Directives
 - `<%@ ... %>`
 - e.g., `<%@ include file="filename" %>`
 - copies static text into JSP source before becoming servlet
 - done once at compile time (similar to C `#include`)
- Actions
 - `<jsp: ... />`
 - e.g., `<jsp:include page="resourcename" flush="true" />`
 - invoke othe URL and merge its output into original JSP
 - performed each time a request is made
 - e.g., `<jsp:forward page="page" />`

CSE 794R/ECE 694R

10

Actions

- Actions `<jsp:xxx />` or `<jsp:xxx id="..." scope="..." />`
 - predefined functions (extendible with "custom tags")
- JavaBean actions
 - create & use objects (beans)
 - `<jsp:useBean id="newBean" scope="page" type="edu.osu.LREngine">`
 - `<jsp:setProperty name="newBean" property="border" value="0" />`
 - `<jsp:useBean />`
- Resource actions
 - make use of external resources
 - `<jsp:include page="..." />` (insert html at this point)
 - `<jsp:forward page="..." />` (forward request)
 - page (static or dynamic) must be on same server
 - return response to client (who sees original url)

Directives

- Directives `<%@ ... %>`
 - control how the JSP engine deals with TRANSLATION
- page
 - `<%@ page import="java.util.Calendar" %>` (java's import)
 - `<%@ page language="java" %>`
 - `<%@ page session="true" %>` (enable sessions)
 - `<%@ page isThreadSafe="true" %>` (all requests in one thread)
 - `<%@ page errorPage="..." %>` (uncaught exception ? forward)
- include
 - `<%@ include file="..." %>` (insert raw data from url)
- taglib
 - declare that this page uses custom tags
 - `<%@ taglib uri="..." prefix="funny" %>`
 - `<funny:xxx id="myTable"> ... </funny:xxx>`
 - `<% myTable.foo(); %>`

Model 1 vs Model 2

- Recall tiers:
 - browser
 - presentation
 - application
 - database

Model 1

- JSP does it all:
 - user requests JSP page
 - JSP performs calculations, db access, etc.
 - JSP page renders its output with HTML
- Code can be written as scriptlets or JavaBeans

Model 2

- Follows Model-View-Controller (MVC) paradigm
 - model—logical “inner” representation
 - view—presentation layer for model with no programming logic
 - controller—handles user input and “directs” model

Model 2 cont.

- All user requests are referred to a single URL, a servlet called the *dispatcher* (controller)
- Based on request's path information, controller dispatches request to proper JSP (action handler). These action handlers constitute the model.
- After doing their thing (access database, perform calculations, etc.), action handlers invoke JSP pages (the view) to present their output

Comparison with ASP

- ASP (Active Server Pages) from MS
- Similarities:
 - both allow us to write pages with dynamic content
 - both separate programming from web page design (sort of)
 - both use code snippets called from parts of the page itself
- Differences:
 - OS reliance: MS Windows vs any
 - Web server reliance:
 - ASP tied to MS IIS (Internet Information Server) [requires ActiveX objects, ports/bridges do exist], JSP can run on any web server
 - closed vs open source
 - ASP uses VBScript or JScript, JSP uses Java (note: in ASP.net these are compiled! not interpreted)
 - ASP pages ???
 - JSP pages access EJB (for business logic) and all standard J2EE services: JNDI, JSBC, JavaMail, JMS

CSE 794R/ECE 694R

17

Comparison with PHP

- PHP: Hypertext Preprocessor (recursive acronym)
- Scripting language (interpreted, weakly typed)
- Filename: .php
- Enclose code in `<?php ... ?>` (or just `<? ... ?>`)
- Opensource (see php.net)

CSE 794R/ECE 694R

18