Java Server Pages

Corso di ***Applicazioni Telematiche***

A.A. 2006-07 – Lezione n.18

Prof. Roberto Canonico

Università degli Studi di Napoli Federico II

Facoltà di Ingegneria

Java Server Pages (JSP)

- Java Server Pages è una tecnologia J2EE per costruire applicazioni web a contenuto dinamico, come HTML, DHTML, XHTML ed XML
- JSP permette di creare pagine web a contenuto dinamico con semplicità e flessibilità
- Una pagina JSP è un documento di testo che descrive come elaborare una richiesta HTTP per generare una risposta HTTP
- Idea: usare HTML “normale” per la maggior parte del contenuto della pagina e delimitare il contenuto dinamico mediante tag speciali

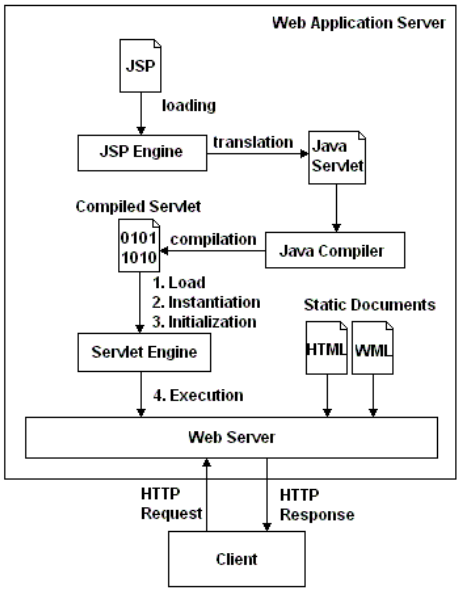
JSP: benefici

- “Write once run anywhere”
- JSP supporta e incoraggia l’uso di componenti riutilizzabili e cross platform come JavaBeans, tag libraries, ed EJB.
- JSP è parte integrante di J2EE ed è uno standard ampiamente supportato da tutti gli application server J2EE.
- E’ supportato dai tool di design web

JSP: benefici (2)

- Favorisce la “separazione dei ruoli”: gli sviluppatori scrivono i componenti che interagiscono col server, i web authors uniscono dati statici e dinamici per realizzare l’applicazione web.
- Incoraggia la separazione tra contenuti statici e contenuti dinamici
- Fornisce lo strato web per architetture J2EE ad “n” livelli.

JSP: funzionamento



JSP: Key Features

- Directives
- Scripting elements
- Standard actions
- Tag Extension mechanism
- Template content

JSP Directives

- Le direttive JSP sono “messaggi” che il programmatore indica al container per influenzare la compilazione della JSP
- Non producono output al client

Esempi:

```
<%@ page import="java.net.*" buffer="10k" %>
<%@ include file="myFile.html" %>
```

JSP: direttive di pagina

<%@ page **language**=”*scriptingLanguage*”
extends=”*className*”
import=”*importList*”
session=”**true**|false”
buffer=”none|*size*kb”
autoFlush=”**true**|false”
isThreadSafe=”**true**|false”
info=”*info_text*”
errorPage=”*error_url*”
isErrorPage=”true|**false**”
contentType=”*ctinfo*”
pageEncoding=”*peinfo*”
isELIgnored=”true|false” >

JSP Scripting Elements

- Gli Scripting Elements sono usati per includere codice (anche non Java) nelle JSP

- ***Declarations:***
  - `<%! int i = 4 %>`
- ***Scriptlets:***
  -
```
<% for (int i=0; i10; i++) {
        out.println("The counter is:" + i);
   }
 %>
```
- ***Expressions:***
  - `<%= myBean.getNumber() %>`

JSP: esempio

```
<html>
<body bgcolor="white">
<% out.println("Hello World"); %>
</body>
</html>
```

JSP: traduzione in Servlet

```
public class hello_jsp extends HttpJspBase {
  public void _jspService(HttpServletRequest request, HttpServletResponse response)
       throws java.io.IOException, ServletException {
    …
    try {
      _jspxFactory = JspFactory.getDefaultFactory();
      response.setContentType("text/html;charset=ISO-8859-1");
      pageContext = _jspxFactory.getPageContext(this, request, response, null, true, 8192, true);
      application = pageContext.getServletContext();
      config = pageContext.getServletConfig();
      session = pageContext.getSession();
      out = pageContext.getOut();
      out.write("<html>\r\n ");
      out.write("<body bgcolor=\"white\">\r\n  ");
      out.println("Hello World");
      out.write("\r\n ");
      out.write("</body>\r\n");
      out.write("</html>");
    } catch (Throwable t) { pageContext.handlePageException(t);
    } finally { if (_jspxFactory != null) _jspxFactory.releasePageContext(pageContext);
    } } }
```

JSP Standard Actions

- JSP Standard Actions sono tag XML specifici che influenzano l'esecuzione della pagina e l'output spedito al client

Esempi:

```
<jsp:useBean id="myBean" scope="application"
   class="mypackage.MyBean" />
<jsp:setProperty name="myBean" property="age" />
<jsp:getProperty name="myBean" property="age" />
<jsp:include page="filename" />
<jsp:forward page="url" />
<jsp:param name="paramName" value="paramValue" />
<jsp:plugin type="applet" code="mypackage.MyApplet"
   codebase="/classes" height="100" width="100" />
```

JSP Tag Extension Mechanism

- In JSP è possibile definire dei TAG che implementano comportamenti “custom” scrivendo delle classi (JavaBeans) che implementino delle interfacce (javax.servlet.jsp.tagext.Tag)
- Queste classi vanno poi dichiarate nel file di configurazione della web application (web.xml) e possono quindi essere utilizzati nelle pagine JSP

JSP Template content

- JSP è nato fondamentalmente per implementare un meccanismo di template flessibile ed estensibile
  - un sistema per incorporare piccole quantità di codice Java in pagine HTML
- Il template content è il contenuto “statico”, generalmente HTML, che rappresenta il layout in cui i dati dinamici trovano posto

JSP Scopes

- Gli oggetti in una JSP possono essere associati ad uno “scope” che definisce da quale contesto sarà accessibile e la sua durata
- Esistono quattro scope JSP:
  1. Application
  2. Session
  3. Request
  4. Page

JSP: ciclo di vita

	Page first written	Request #1	Request #2	Server restarted	Request #3	Request #4	Page modified	Request #5	Request #6
JSP page translated into servlet		Yes	No		No	No		Yes	No
Servlet compiled		Yes	No		No	No		Yes	No
Servlet instantiated and loaded into server's memory		Yes	No		Yes	No		Yes	No
init (or equivalent) called		Yes	No		Yes	No		Yes	No
doGet (or equivalent) called		Yes	Yes		Yes	Yes		Yes	Yes

JSP: ciclo di vita (2)

1. Il metodo jspInit() è invocato dopo il caricamento;
2. Il metodo _jspService() è invocato ad ogni richiesta dando luogo ad un nuovo thread
3. Il metodo jspDestroy() è invocato quando è necessario rimuovere la pagina JSP dal server