

Introduzione alle JSP

Corso di Sistemi Web e Basi Dati
Prof. Antonio d'Acerno

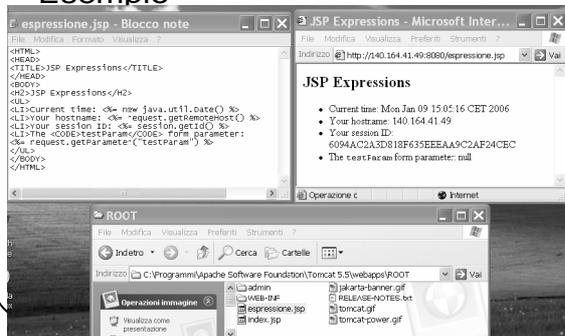
1

Java Server Page

- JSP
- Riferimenti
 - <http://java.sun.com/products/jsp>
 - <http://www.coreservlets.com>
 - <http://www.html.it/jsp>
- Le JSPs consentono di mischiare HTML statico con parti dinamiche attraverso una sintassi molto semplice.

2

Esempio



JSP: Ciclo di vita

- Una JSP è automaticamente tradotta in una normale servlet con l'HTML statico stampato sulla stream di output associata al metodo service.
- La traduzione è normalmente eseguita alla prima richiesta della pagina
- Attenzione:
 - Se faccio un errore nel parte di codice, me ne accorgo solo quando vado sulla pagina!

4

Direttive

- È l'insieme dei marcatori che consentono di definire come il container deve elaborare il documento.
- Le direttive non influenzano la gestione di una singola richiesta HTTP ma influenzano le proprietà generali della JSP e di come questa deve essere tradotta in una servlet.
- Una direttiva è introdotta con il tag `<%@ tipo ...%>` o il suo equivalente XML `<jsp:directive.tipo .../>`.
- Esempi
 - `<%@ page import="java.util.*" %>`
 - `<%@ page contentType="text/plain" %>`

9

Elementi di Scripting

- È il gruppo dei marcatori che permettono di inserire del codice all'interno di un documento.
- Le istruzioni devono essere scritte nel linguaggio di programmazione designato per la pagina (potrebbe anche NON essere Java, anche se attualmente è l'unico supportato) e vengono eseguite ad ogni richiesta della pagina JSP.
- Un elemento di scripting è introdotto con uno dei seguenti tag:
 - `<%!...%>` Dichiarazione
 - `<%=...%>` Espressione
 - `<%...%>` Scriptlet

10

Dichiarazione e Espressione

- Dichiarazione:
 - Si inseriscono dichiarazioni di variabili di classe (comuni a più istanze) o metodi statici
 - `<%! private int x = 4; %>`
- Espressione:
 - `<%= espressione %>`
 - Viene valutata l'espressione e stampato il risultato nella pagina html generata.

11

Scriptlet

- Frammento di codice Java che può modificare anche il flusso del codice html generato.
- Solitamente gli operatori condizionali ed i cicli possono essere utilizzati per produrre dinamicamente porzioni diverse di codice html in funzione dell'input.

12

Esempio

```
<HTML>
<HEAD>
<TITLE>JSP Expressions</TITLE>
</HEAD>
<BODY>
<% For (int i=0; i<10; i++) { %>
<%= i %>
<br>
<% if (i%2==0) { %>
pari
<% } else { %>
dispari
<% } %>
<br>
<% } %>
</BODY>
</HTML>
```

13

Esempio

```
<HTML>
<HEAD>
<TITLE>JSP Expressions</TITLE>
</HEAD>
<BODY>
<% page Import = "java.sql.*" %>
<%
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
Connection conn = DriverManager.getConnection("jdbc:odbc:sakdb");
Statement statement = conn.createStatement();
ResultSet rs = statement.executeQuery("select * from dipartimento");
%>
<br>
<table border="1">
<caption> Risultati della query</caption>
<thead>
<tr>
<th>nome</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amministrazione</td>
</tr>
<tr>
<td>Finanze</td>
</tr>
<tr>
<td>Produzione</td>
</tr>
<tr>
<td>Ingegneria</td>
</tr>
<tr>
<td>Rappresenta</td>
</tr>
</tbody>
</table>
```

14

Azioni

- Questi marcatori permettono di supportare diversi comportamenti della pagina JSP.
- Vengono processati ad ogni invocazione della pagina JSP.
- Permettono, ad esempio, di trasferire il controllo da una JSP all'altra
- Un'azione è introdotta con un tag del tipo:
 - <jsp:tipoAzione.../>
- Ad esempio, se si vuole includere all'interno di una JSP dinamicamente un'altra JSP, è sufficiente inserire nel punto dove si vuole l'inclusione l'azione:
 - <jsp:include page="localURL" flush="true"/>

15

Commenti

- Questi marcatori permettono di inserire diversi tipi di commenti all'interno delle JSP.
 - <!-- commenti di contenuto -->
 - sono i tipici commenti di HTML.
 - <%-- commenti JSP --%>
 - sono i commenti visibili solo nel sorgente della JSP. (la servlet equivalente non conterrà nulla di questi commenti).
 - <% /* commenti di scripting */ %>
 - sono i commenti all'interno della parte di codice della JSP e sono visibili anche nella servlet equivalente.

16

Oggetti Impliciti

- Ogni pagina JSP rende disponibile un insieme di oggetti che possono essere utilizzati all'interno della pagina.
- Tra i più importanti
 - request
 - response
 - out
 - session

17

Cosa è un BEAN? (1)

- L'API JavaBeans mette a disposizione un format standard per classi Java.
- Rispettando tale formato è possibile, ad esempio, che tools visuali manipolino classi per costruire applicazioni complesse.
- Per approfondimenti un buon punto di partenza è
 - <http://java.sun.com/developer/onlineTraining/Beans/JBShortCourse/index.html>

18

Cosa è un BEAN? (2)

- Per i nostri scopi, una classe è un bean sse:
 - ha un costruttore vuoto
 - non ha variabili pubbliche
 - ogni variabile è accessibile tramite metodi del tipo getXxx (isXxx per dati boolean) e setXxx
 - se manca il get si parla di varabile writeonly
 - se manca il set si parla di variabile readonly

19

Esempi di BEAN - 1

```
public class StringBean {
    private String message = "No message specified";

    public String getMessage() {
        return(message);
    }

    public void setMessage(String message) {
        this.message = message;
    }
}
```

20

Esempi di BEAN - 2

```
package swbd;
import java.sql.*;

public class Utente {
    private String nome;
    private String password;
    private boolean autorizzato;

    public Utente() {
        nome = "sconosciuto";
        password = "sconosciuta";
        autorizzato = false;
    }

    public Utente(String n, String p) {
        nome = n;
        password = p;
        autorizzato = false;
    }

    public String getNome() {
        return nome;
    }

    public boolean isAutorizzato() {
        try {
            ..... //Caricamento del driver e
            ..... //connessione
            ..... //Costruzione query
            ResultSet rst = stm.executeQuery(query);
            if (rst.next())
                if (password.equals(rst.getString("password")))
                    autorizzato = true;
            return true;
        } catch (SQLException e) { ..... }
        return false;
    }
}
```

21

JSP e BEANS - 1

- Tra le varie azioni esiste la `jsp:useBean` che consente di usare un bean all'interno di una pagina
- Tale tecnica è particolarmente utile in quanto consente
 - il riutilizzo di classe java
 - di mantenere la logica di business fuori dalle pagine

22

JSP e BEANS - 2

- `<jsp:useBean id="book1" class="coreservlets.Book" />`
- Si può pensare che tale riga sia equivalente a `<% coreservlets.Book book1 = new coreservlets.Book(); %>` anzi che la seconda sia più potente essendo possibile, ad esempio, usare costruttori diversi da quello vuoto.
- In realtà la `jsp:useBeans` permette di usare l'attributo `scope` che consente la condivisione di Beans
- Quindi `<jsp:useBean id="book1" class="coreservlets.Book" />` istanzia un Bean solo se non esiste uno con lo stesso nome e con lo stesso `scope`. In caso contrario ci si collega a quello esistente.

23

Condivisione di Beans: SCOPE

- **page**
 - Valore di default. Il bean è visibile solo all'interno della pagina.
- **session**
 - Il bean, una volta creato, vive fino a che la sessione viene invalidata (esplicitamente o implicitamente)
- **request**
 - Il bean vive per il tempo della richiesta (che potrebbe coinvolgere più pagine)
- **application**
 - Il bean è condiviso da tutta l'applicazione.

24