Modelli di sviluppo JSP

dettagli implementativi

Slides parzialmente tratte da materiale di Giansalvatore Mecca (*Tecnologie di Sviluppo per il Web*) e Marty Hall (*http://www.coreservlets.com*)

Architettura delle applicazioni

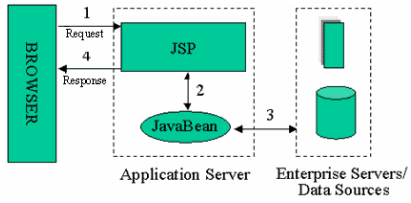
- Tipologie di componenti di un'applicazione
  - Interfaccia (*View*)
    - comunicazione con l'utente
    - possibilità: console, grafica, HTML
  - Logica di controllo (*Control*)
    - gestisce il flusso di esecuzione fra le varie componenti
  - Modello (*Model*)
    - rappresenta lo stato dell'applicazione: dati (temporanei) legati all'esecuzione dell'applicazione
  - Persistenza
    - alcune informazioni devono essere memorizzati in modo persistente al di là delle singole esecuzioni
    - possibilità : base di dati, file di testo, file XML
- Varie architetture fisiche possibili
  - le componenti concettuali possono essere implementate come componenti software separate o meno

Architettura di applicazioni Web

- Due modelli di sviluppo basati sull'uso della tecnologia Servlet/JSP
  - si differenziano i tipi di componenti usati e per l'architettura complessiva dell'applicazione
- Modello JSP/1
  - implementazione dell'*"architettura di base"*
  - combina pagine JSP e Java Bean
    - separazione tra presentazione e logica applicativa
  - adatto per applicazioni di dimensioni medie
- Modello JSP/2
  - implementazione del pattern MVC
  - uso combinato di pagine JSP, Java Bean e Servlet
    - separazione tra presentazione, modello e logica di controllo
  - adatto per applicazioni più complesse con modalità di elaborazione e presentazione molto eterogenei

Modello di sviluppo JSP/1

1. Il client richiede via HTTP un file *.JSP*
2. Il file *.JSP* viene interpretato e accede a componenti lato-server che generano contenuti dinamici
   - tipicamente Java Beans, Servlet
3. il risultato è spedito al client come pagina HTML



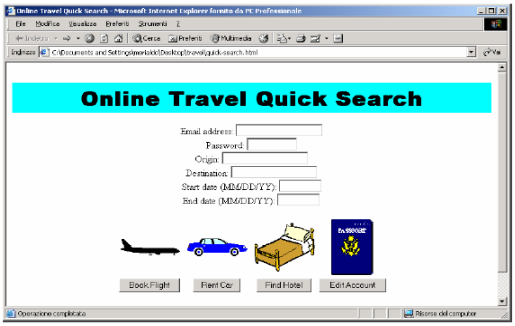
Modello di sviluppo JSP/1: accesso ai bean da JSP

- Installazione della classe bean
  il bytecode del bean (file *.class*) deve essere nella directory WEB-INF\classes dell'applicazione
- Importazione della classe nella pagina
  <%@ page import="Studente"%>
- Dichiarazione/creazione di un oggetto bean
  <jsp:useBean id="st1" class="Studente" />
- Modifica del contenuto del bean
  <jsp:setProperty name="st1" property="nome" value="X" />
  - equivalente a: *<% st1.setName("X"); %>*
- Accesso al contenuto di una proprietà del bean
  <jsp:getProperty name="st1" property="nome" />
  - equivalente a: *<%= st1.getName() %>*

Modello di sviluppo JSP/1: critica

- Vantaggi:
  - JSP semplifica lo sviluppo e la manutenzione della applicazione
  - con i *beans* si separano presentazione e logica
    - i bean "nascondono" il codice a chi sviluppa la pagina JSP
  - ottimo per applicazioni di piccole dimensioni
- Svantaggi:
  - ancora molto codice Java in alcune pagine JSP
  - problemi per lo sviluppo e la manutenzione
  - non appropriata per applicazioni complesse
    - assume che la *singola* pagina JSP assolva ad un singolo compito (gestisce la presentazione di tutti i possibili risultati)

Esempio: un'agenzia viaggi on-line

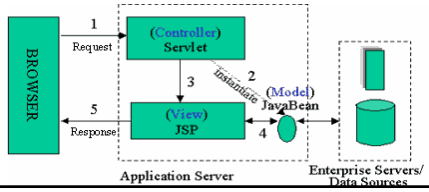


Agenzia viaggi on-line: discussione

- Tutte le richieste includono
  - e-mail, password, partenza, destinazione, data partenza, data ritorno
- Per tutte richieste c'è una elaborazione comune
  - cercare nome, indirizzo, carta di credito, etc., usando email e password come chiave
- La struttura della risposta varia nettamente a seconda della scelta (non basta una sola pagina!):
  - pagina che mostra voli disponibili (con dettagli)
  - pagina che mostra alberghi disponibili (con dettagli)
  - pagina che mostra informazioni su auto in affitto

Modello di sviluppo JSP/2

- Uso combinato di Servlet e JSP
  - Servlet per elaborazione e controllo
    - processa la richiesta originale, effettuando il grosso dei calcoli
    - memorizza i risultati in bean
  - JSP per livello di presentazione
    - recupera i bean creati dal servlet
    - usa i dati dei bean per costruire il contenuto della risposta
- uso di più pagine JSP per presentazioni differenti



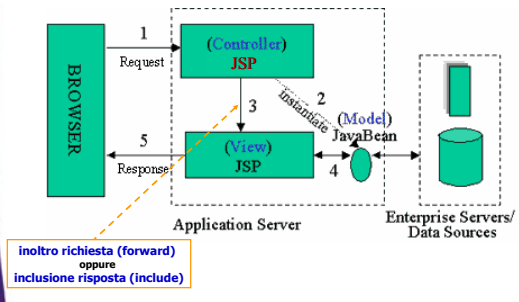
Modello di sviluppo JSP/2: flusso tipico e implementazione

- Pagina controller:
  1. riceve richiesta dal browser
  2. determina l'operazione da eseguire e crea eventuali bean
  3. rende disponibili i bean creati alle pagina JSP
  4. attiva la pagina JSP per la visualizzazione

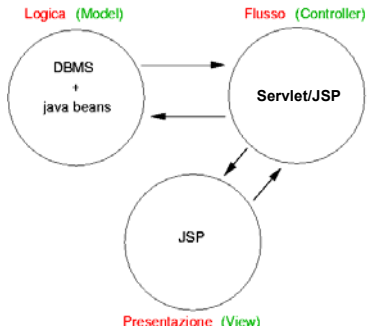
  ➔ di solito è implementata con un Servlet
- Pagina di presentazione:
  1. accede ai bean creati dal controller
  2. utilizza i dati contenuti nei bean per creare la pagina da visualizzare

  ➔ di solito è implementata con una pagina JSP
- *Varianti*:
  - il controller può essere una pagina JSP (poco codice HTML)
  - il controller può essere preceduto da una pagina JSP che produce una parte del risultato

Modello di sviluppo JSP/2: varianti



Modello di sviluppo JSP/2



Modello di sviluppo JSP/2

- Vantaggi:
  - netta separazione tra logica, flusso e presentazione
  - ottimo per applicazioni di medie/grandi dimensioni
- Svantaggi:
  - sistema più complesso da progettare

Strumenti per l'interazione e la condivisione di risorse in JSP

- Interazione con altre pagine (JSP/Servlet)
  - inoltro delle richieste: tag *forward*

```
<jsp:forward page="fine.jsp">
        <jsp:param name="utente" value="Pippo" />
</jsp:forward>
```

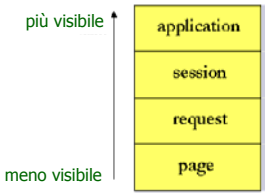
  - inclusione delle risposte: tag *include*

```
<jsp:include page="uri" />
        <jsp:param name="utente" value="Pippo" />
</jsp:include>
```

- Condivisione di dati
  - variabili definite nelle dichiarazioni
  - parametri passati con *forward* e *include*
  - creazione/uso di bean condivisi (scope = page, request, session, application)
    - `<jsp:useBean id="studente" class= "StudenteBean" scope= "…"/>`
    - `<jsp:setProperty name="studente" property ="cognome" … "/>`
    - `<jsp:getProperty name="studente" property = "cognome" />`
  - uso esplicito delle mappe di attributi (ved. avanti)

Condivisione di oggetti bean

- E' possibile costruire oggetti che possono essere condivisi fra vari Servlet/JSP
- Esistono quattro campi di visibilità (scope) in cui può avvenire la condivisione
  - Pagina
  - Richiesta
  - Sessione
  - Applicazione



Campi di visibilità (scope)

- Pagina (Page scope)
  - Oggetti che esistono solo nelle pagine in cui sono definiti
  - Per ogni accesso alla pagina esiste un'istanza di tali oggetti
- Richiesta (Request scope)
  - Esistono per la durata della richiesta del client
  - Muoiono quando la risposta è stata inviata al client
- Sessione (Session scope)
  - Esistono per la durata della sessione di browsing del client
  - Muoiono quando l'utente lascia il sito o avviene un timeout della sessione
- Applicazione (Application scope)
  - Oggetti associati all'intera applicazione web
  - Ogni Servlet/JSP nell'applicazione può accedervi

Modello di sviluppo JSP/2

Agenzia di viaggio on-line: la pagina JSP controller

```
…
<jsp:useBean id="customer"
    class="viaggi.TravelCustomer"  scope="session" />
…
// memorizza i dati nell'oggetto "customer" della classe TravelCustomer
<% if (req.getParameter("flights") != null) %>
        <jsp:forward page="fine.jsp" />
<% else if (…) %>
        <jsp:forward page="…" />
```

Modello di sviluppo JSP/2:

Agenzia di viaggio on-line: la pagina JSP dei voli

```
<BODY>
<H1>Best Available Flights</H1>
<CENTER>
<jsp:useBean id="customer"
        class="viaggi.TravelCustomer"
        scope="session" />
Finding flights for
<jsp:getProperty name="customer"
        property="fullName" />
<P>
<jsp:getProperty name="customer"
        property="flights" />
…
```

Modello JSP/2: conclusioni

- Quando usare l'approccio JSP/2:
  - una sottomissione che può portare a molte soluzioni
  - elaborazione condivisa
- Architettura
  - Un servlet/JSP *controller* riceve la risposta originale, elabora i dati e memorizza i risultati in oggetti bean
  - I bean sono memorizzati in contesti condivisi (request, session, o application)
  - Il servlet/JSP *controller* passa il controllo ad una pagina JSP attraverso un'operazione di *forward* o *include*
  - La pagina JSP legge i dati dai bean per mezzo di *jsp:useBean* con lo scope appropriato (*request*, *session*, o *application*)

Condivisione di dati con le mappe di attributi

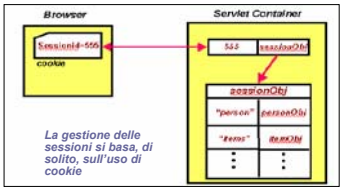
Condivisione di dati

Mappe di attributi

- Alcuni oggetti predefiniti hanno mappe associative
  - ogni mappa contiene gli oggetti con lo *scope* corrispondente (application, session, request, …)
  - insieme di coppie (nome, oggetto)

Esempio:

Mappa dell'oggetto `session`



- Consentono lo scambio di dati fra Servlet/JSP
  alternativa (semplice ed efficace) per le pagine JSP: uso di bean

Condivisione di dati

Mappe: metodi di accesso

- Metodi per l'accesso agli elementi di una mappa:
  - **void setAttribute (String nome, Object valore)**
    - metodo per aggiungere un attributo
    - inserisce una coppia (nome, valore)
  - **Object getAttribute (String nome)**
    - metodo per prelevare il valore di un attributo
    - ritorna l'oggetto associato al nome (riferimento a *Object*)
    - attenzione: necessaria la conversione al tipo effettivo dell'oggetto
- request ha anche una mappa per i parametri della richiesta
  - tali parametri sono stringhe
    - provenienti da una form o trasferiti da un'altra pagina tramite inoltro o inclusione
  - metodi di accesso:
    - request.getParameter("nome")
    - request.getParameterValues("nome")
    - request.getParameterNames()

Condivisione di dati

Mappe di attributi: esempio

```
<%@ page errorPage="errorpage.jsp" %>
<html>  <head> <title>UseSession</title> </head>
<body>
<%
   // Prova a recuperare il contatore corrente dalla mappa di sessione
   Integer count = (Integer) session.getAttribute("COUNT");
   // Se non trova COUNT, lo crea e lo aggiunge alla mappa di sessione
   if ( count == null ) {
         count = new Integer(1);
         session.setAttribute("COUNT", count);
   }
   else {
     // se COUNT esiste, viene incrementato e rimesso nella mappa
         count = new Integer(count.intValue() + 1);
         session.setAttribute("COUNT", count);
   }
   // Stampa il numero di accessi
   out.println("<b>Hai visitato questo sito: " + count + " volte.</b>");
%>
</body></html>
```

Le mappe possono contenere solo oggetti e non variabili dei tipi elementari

Gestione della sessione: metodi utili dell'oggetto *session*

- getAttribute, setAttribute, removeAttribute
  - operano sull'attributo con il nome passato come parametro
- getAttributeNames
  - ritorna i nomi di tutti gli attributi presenti nella sessione
- getId
  - ritorna l'identificatore (unico) della sessione
- isNew
  - Indica se la sessione è nuova per il *client* (non per la *pagina*)
- getCreationTime, getLastAccessedTime
  - ritornano i tempi del primo e dell'ultimo accesso alla sessione
- getMaxInactiveInterval, setMaxInactiveInterval
  - ritorna (imposta) il tempo massimo per cui la sessione può restare inattiva prima di essere invalidata
- invalidate
  - invalida la sessione e distrugge gli oggetti ad essa associati