Tecnologie di Sviluppo per il Web

Applicazioni Web J2EE: Introduzione

versione 3.1

G. Mecca – mecca@unibas.it – Università della Basilicata

Sommario

- Server Applicativo
- Servlet e JSP
- Architettura di Riferimento
- Progetti di Riferimento
  - ⇨Indovina il Numero
  - ⇨Il Sistema Informativo dell'ACI

Server Applicativo

- In questa parte del corso
  - ⇨sviluppo di applicazioni Web con la piattaforma J2EE
- In particolare
  - ⇨Java Servlet
  - ⇨Java Server Pages
  - ⇨architetture di riferimento per lo sviluppo di applicazioni Web con Java

Server Applicativo

- Un componente fondamentale
  - ⇨il server applicativo
  - ⇨o “contenitore dei servlet”
- In questo corso
  - ⇨utilizziamo Apache Tomcat
  - ⇨l’implementazione di riferimento della specifica dei servlet e delle pagine JSP

Server Applicativo

- Tomcat
  - progetto Open source della Apache Software Foundation
  - parte del sottoprogetto Jakarta, finalizzato allo sviluppo di soluzioni open-source per J2EE
  - distribuito su jakarta.apache.org
  - prodotto molto maturo
  - richiede J2SE installato sulla macchina

Server Applicativo

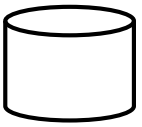
- Nota
  - Tomcat è essenzialmente un server applicativo
  - dovrebbe essere utilizzato assieme ad un server HTTP (es: Apache) per ottenere le migliori prestazioni
  - ma include un proprio server HTTP interno che può essere utilizzato per scopi di sviluppo

Server Applicativo

Apache (server HTTP porta 80)

Connettore JK2

Tomcat (server applicativo)

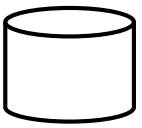


Configurazione di produzione

Server Applicativo

Tomcat (server HTTP porta 8080)

Tomcat (server applicativo)



Configurazione di sviluppo

Server Applicativo

- Installazione di Tomcat
  - ⇨eseguire l'installabile prelevato dal sito
  - ⇨richiedere l'installazione come servizio (installa un servizio per l'esecuzione del server HTTP sulla porta 8080)
  - ⇨richiedere l'installazione delle applicazioni a corredo ("webapps")
  - ⇨verifica dell'installazione: http://localhost:8080
  - ⇨da lì si accede alla documentazione

Server Applicativo

- Post-configurazione
  - ⇨configurazione degli utenti
- I ruoli di Tomcat
  - ⇨il ruolo "admin" – amministratore del contenitore (può modificare la configurazione)
  - ⇨il ruolo "manager" – amministratore delle applicazioni (può aggiungere e rimuovere applicazioni Web)

Server Applicativo

>> tomcat-users.xml

- Configurazione degli utenti
  - ⇨cambiare il valore del file %TOMCAT_HOME%\conf\tomcat-users.xml
  - ⇨aggiungendo un utente che abbia ruolo manager
  - ⇨cambiando le password predefinite per admin e manager

Servlet e JSP

- Applicazione Web J2EE
  - ⇨collezione di servlet e Java Server Pages
  - ⇨componenti Java (classi, file jar)
  - ⇨pagine HTML, CSS, immagini, altri file
- Un concetto importante
  - ⇨il rapporto tra pagine JSP e servlet

Servlet e JSP

- Idea
  - ⇨le richieste HTTP vengono gestite da opportune classi Java dette servlet
  - ⇨il programmatore NON scrive direttamente i servlet, ma scrive pagine JSP
  - ⇨le pagine JSP vengono trasformate dal contenitore in servlet opportuni, che vengono compilati ed istanziati dal contenitore

Servlet e JSP

- Servlet
  - ⇨classe Java orientata alla comunicazione client-server
  - ⇨riceve messaggi di richiesta
  - ⇨produce messaggi di risposta
  - ⇨viene eseguita dal contenitore secondo un opportuno “ciclo di vita”
  - ⇨basata sull’API javax.servlet.*

Servlet e JSP

- Due metodi principali
  - ⇨doGet(): operazioni per rispondere a richieste di tipo GET
  - ⇨doPost(): operazioni per rispondere a richieste di tipo POST
- Parametri
  - ⇨richiesta: HttpServletRequest request
  - ⇨risposta: HttpServletResponse response

Servlet e JSP

- Java Server Page (JSP)
  - ⇨strumento rapido per la scrittura di servlet
  - ⇨pagina server in cui sono mischiati codice HTML e codice Java
  - ⇨viene automaticamente tradotta dal contenitore in un servlet equivalente, che viene compilato e istanziato dal contenitore
  - ⇨ogni volta che una richiesta viene indirizzata ad una pagina JSP, il contenitore la gestisce utilizzando il servlet corrispondente

Un Esempio

istruzioni Java (scriptlet)

oggetti predefiniti

```
<!-- data.jsp -->
<html>
  <body>
  <%
    String nome=(String)request.getParameter("nome");
    session.setAttribute("nome", nome);
  %>
  <p>Benvenuto, <%= nome %>. La data di oggi e':
      <%= new java.util.Date() %>
    </p>
  </body>
</html>
```

espressioni Java

questa pagina JSP viene automaticamente trasformata in un servlet

Il Servlet Corrispondente

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class Data_jsp extends HttpServlet {

  public void doGet (HttpServletRequest request, HttpServletResponse response)
                    throws IOException, ServletException {
     String nome = (String)request.getParameter("nome");
     HttpSession session = request.getSession(true);
     session.setAttribute("nome", nome);
     response.setContentType("text/html");
     PrintWriter out = response.getWriter();
     out.println("<html>\n<body>");
     out.println("<p>Benvenuto, " + nome);
     out.println("La data di oggi e':");
     out.println("<b>" + new java.util.Date()+"</b></p>");
     out.println("</body>\n</html>");
  }
}
```

Servlet e JSP

- Vantaggi delle pagine JSP
  - ⇨la scrittura del codice HTML è decisamente semplificata (non servono println())
  - ⇨non è necessario compilare il codice: la traduzione in servlet e la compilazione viene effettuata dal compilatore
  - ⇨il contenitore, tra l'altro, si accorge anche delle modifiche effettuate nel codice JSP e ripete il processo tutte le volte che è necessario

Servlet e JSP

- Attenzione
  - ⇨i servlet non vengono istanziati dal programmatore
  - ⇨sono classi gestite dal contenitore
  - ⇨il contenitore crea le istanze dei servlet, chiama i metodi di servizio e rimuove le istanze quando non servono ulteriormente
  - ⇨quindi il programmatore non manipola direttamente gli oggetti in questione

Servlet e JSP

- Vantaggi rispetto alle CGI
  - ⇨efficienza (attivazione di thread leggeri rispetto all'attivazione di processi pesanti)
  - ⇨flessibilità (tutti i servlet girano nella stessa macchina virtuale e possono cond. dati)
- Altri vantaggi
  - ⇨potenza del linguaggio Java
  - ⇨portabilità
  - ⇨relativa economicità dei contenitori

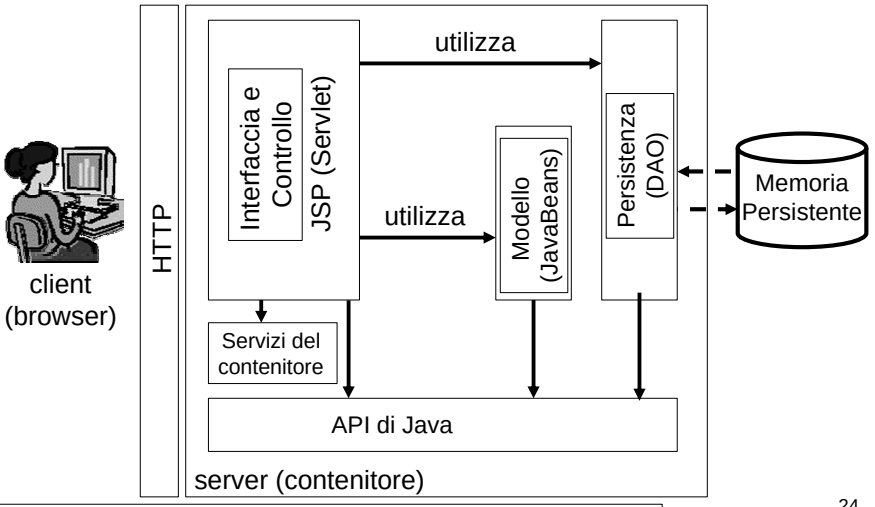
Servlet e JSP

- Riassumendo, in un'applicazione J2EE
  - ⇨i componenti di interfaccia e controllo sono in realtà servlet
  - ⇨le pagine JSP sono un modo più rapido per scrivere questi servlet
  - ⇨nel seguito useremo i servlet come modello per descrivere la semantica
  - ⇨ma svilupperemo prevalentemente pagine JSP

Architettura di Riferimento

- Architettura denominata Modello 1
  - ⇨adotteremo per cominciare un'architettura di riferimento analoga all'architettura di base
  - ⇨interfaccia e controllo insieme, realizzate attraverso pagine JSP >> schermi
  - ⇨modello realizzato attraverso JavaBeans
  - ⇨persistenza realizzata utilizzando DAO

Architettura di Riferimento



Architettura di Riferimento

- Codice dell'interfaccia
  - ⇨le pagine JSP (pagine server) generano dinamicamente codice XHTML (pagine client)
  - ⇨la presentazione è stabilita con un foglio di stile CSS
  - ⇨il browser riceve le risposte (pagine client) e le visualizza per l'utente
  - ⇨il codice di controllo è mischiato con quello degli schermi

Architettura di Riferimento

- Utilizzo di modello e persistenza
  - ⇨gli schermi (pagine JSP) istanziano i bean
  - ⇨e ne chiamano i metodi
  - ⇨i bean non chiamano metodi degli schermi (gli schermi non sono oggetti Java ordinari)
  - ⇨i bean "mantengono" lo stato della sessione e dell'applicazione
  - ⇨sono condivisi da schermi diversi

Progetti di Riferimento

- Progetti di riferimento per questo modulo
  - ⇨due progetti di cui vedremo diverse versioni
- Indovina il Numero su Web
  - ⇨un'applicazione a due livelli (senza persistenza)
- Il Sistema Informativo dell'ACI su Web
  - ⇨un'applicazione a tre livelli

Indovina Il Numero su Web

>> indovinam1

- Indovina il Numero su Web
  - ⇨versione J2EE di Indovina il numero
- Tecnologia
  - ⇨realizzata secondo l'architettura di tipo modello 1
  - ⇨con pagine JSP e classi Java
  - ⇨eseguita attraverso un server applicativo per servlet e pagine JSP (Tomcat)

Indovina il Numero su Web

- Struttura dell'applicazione
  - 7 file .jsp
  - index.jsp (potrebbe essere index.html), tentativo.jsp, indovinato.jsp, interruzione.jsp, fine.jsp, errore.jsp
  - 2 classi java (JavaBeans): Partita.java, Record.java
  - NOTA: le classi del modello sono le stesse delle altre versioni dell'applicazione

Indovina il Numero su Web

- Operazioni tipiche nelle pagine JSP
  - gestire le richieste dei client (estrarre dati forniti dall'utente)
  - produrre le risposte
  - gestire le sessioni di lavoro con l'utente
  - gestire il contesto dell'applicazione
  - utilizzare i componenti dell'applicazione
  - inoltrarsi richieste a vicenda

Indovina il Numero su Web

- Idea
  - ⇨il contenitore fornisce l'accesso ad una serie di oggetti che consentono di svolgere in forma semplificata le operazioni essenziali
  - ⇨es: oggetto "request", che consente di prelevare i parametri della query string
  - ⇨es: oggetto "session", in cui è possibile salvare tutti i dati che devono sopravvivere tra una richiesta e l'altra

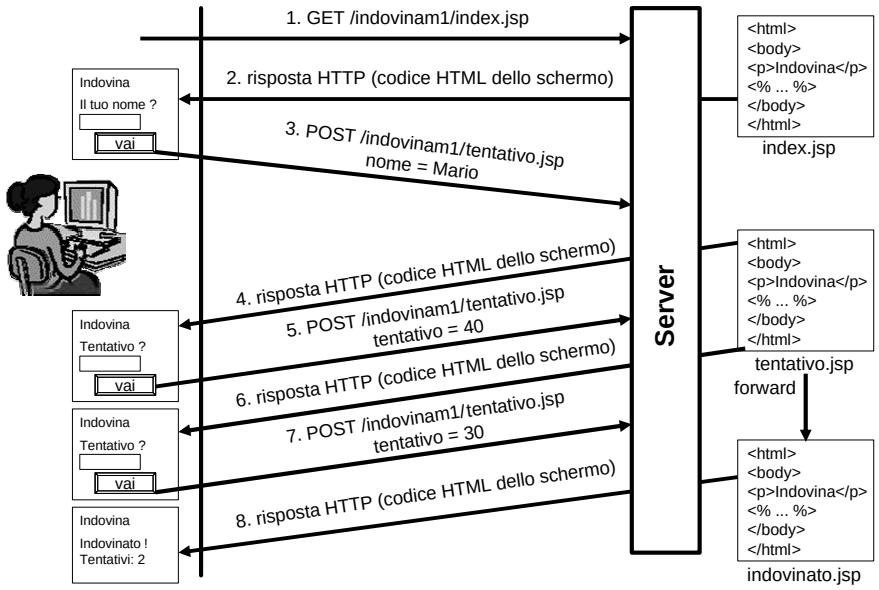
Indovina il Numero su Web

- Un esempio di pagina
  - ⇨interruzione.jsp
  - ⇨codice HTML e codice Java mischiato assieme
  - ⇨nella pagina vengono utilizzati gli oggetti "predefiniti" (request, response, session ...)
  - ⇨in particolare, i bean vengono "salvati" e recuperati dall'oggetto session che rappresenta la sessione di lavoro

```
<%@ page import="it.unibas.indovina.modello.*" %>
<%@ page errorPage="errore.jsp" %>
<%
      Partita partita = (Partita)session.getAttribute("partita");
      if (session.isNew() || partita == null){
         throw new javax.servlet.jsp.JspException("L'accesso alla pagina richiesta non e' consentito");
      }
      session.invalidate();
%>
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="it">
<head>
<title>Partita Interrotta</title>
<link rel="stylesheet" href="indovina.css" type="text/css" />
</head>
<body>
<p>Corso di Tecnologie di Sviluppo per il Web - Universit&agrave; della Basilicata</p>
<div class="pageTitle">
   <img src="icons/titoloWeb.gif" width="562" height="61" />
</div>
<h2>Partita Interrotta</h2>
<p>In numero da indovinare era <strong><%= partita.getNumeroDaIndovinare() %></strong></p>
<p>Hai effettuato in totale <strong><%= partita.getNumeroDiTentativi() %></strong> tentativi</p>
...
</html>
```


Indovina il Numero su Web

- Flusso di controllo tipico
  - ⇨il client effettua una richiesta (evento scatenato dall'utente con un click)
  - ⇨lo schermo corrispondente all'URI richiesto riceve la richiesta e la analizza
  - ⇨se è di sua competenza, produce la risposta per il browser (HTML) utilizzando i bean
  - ⇨altrimenti inoltra la richiesta ad un altro schermo che può gestirla



Indovina il Numero su Web

- Stile di controllo nelle applicazioni Web
  - ⇨ si tratta di uno stile di programmazione asincrona in cui il flusso di controllo è frammentato nelle varie transazioni HTTP
  - ⇨ in cui i componenti di controllo (pagine server) reagiscono ad eventi scatenati dal client
  - ⇨ ogni evento corrisponde ad una richiesta HTTP
  - ⇨ richiesta – elaborazione/inoltro – risposta
  - ⇨ ogni volta un componente viene eseguito sul server, effettua le sue elaborazioni e poi termina

Indovina il Numero su Web

- Esempio
  - la pagina tentativo.jsp viene chiamata in due circostanze diverse
  - per cominciare la partita (l'utente sottomette il nome)
  - per continuare la partita (l'utente sottomette un tentativo)
  - il codice di controllo nella pagina deve per prima cosa capire cosa fare

Indovina il Numero su Web

- Verifica delle richieste
  - l'URI /indovinam1/tentativo.jsp può essere chiamato da un client anche al di fuori di una partita
  - queste chiamate sono da considerarsi scorrette
  - e in questo caso l'applicazione deve sollevare un'eccezione

Indovina il Numero su Web

>> servlet generati

- Attenzione
  - ⇨nell'applicazione, per ogni pagina JSP il contenitore genera un servlet corrispondente
  - ⇨nella cartella %TOMCAT_HOME%\work
  - ⇨modificando una pagina jsp, il servlet corrispondente viene rigenerato, ricompilato, istanziato e utilizzato per gestire le richieste

Indovina il Numero su Web

- In effetti
  - ⇨le due tecnologie sono completamente equivalenti dal punto di vista delle funzionalità
  - ⇨tutto quello che è possibile fare in una pagina JSP è fattibile anche scrivendo direttamente un servlet e viceversa
- Ma...
  - ⇨la scrittura diretta dei servlet è più laboriosa

Indovina il Numero su Web

>> indovinaM1Servlet

- Un esempio
  - ⇨indovinaM1Servlet
  - ⇨un'applicazione funzionalmente identica a quella sviluppata con pagine JSP
  - ⇨ma scritta direttamente utilizzando i servlet

Il Sistema Informativo dell'ACI

- In sistema informativo dell'ACI su Web
  - ⇨versione Web dell'applicazione client/server dell'ACI
- Struttura dell'applicazione
  - ⇨stessa architettura applicativa (modello 1)
  - ⇨maggiore complessità degli schermi (20 file .jsp)
  - ⇨anche in questo caso modello e persistenza sono identici a quelli della versione desktop

Il Sistema Informativo dell'ACI

>> aciM1

- Alcuni elementi aggiuntivi
  - ⇨l'applicazione è realmente basata su un'architettura a tre livelli (client – server – dbms)
  - ⇨la gestione delle form è più complessa e prevede convalide articolate dei dati
  - ⇨l'applicazione prevede un meccanismo di autorizzazione e autenticazione, e l'accesso agli schermi è protetto

Riassumendo

- Server Applicativo
- Servlet e JSP
- Architettura di Riferimento
- Progetti di Riferimento
  - ⇨Indovina il Numero
  - ⇨Il Sistema Informativo dell'ACI

Termini della Licenza