Programmazione WEB

Lezione del 6 Aprile 2020

Docente: Novella Bartolini

Pagine JSP

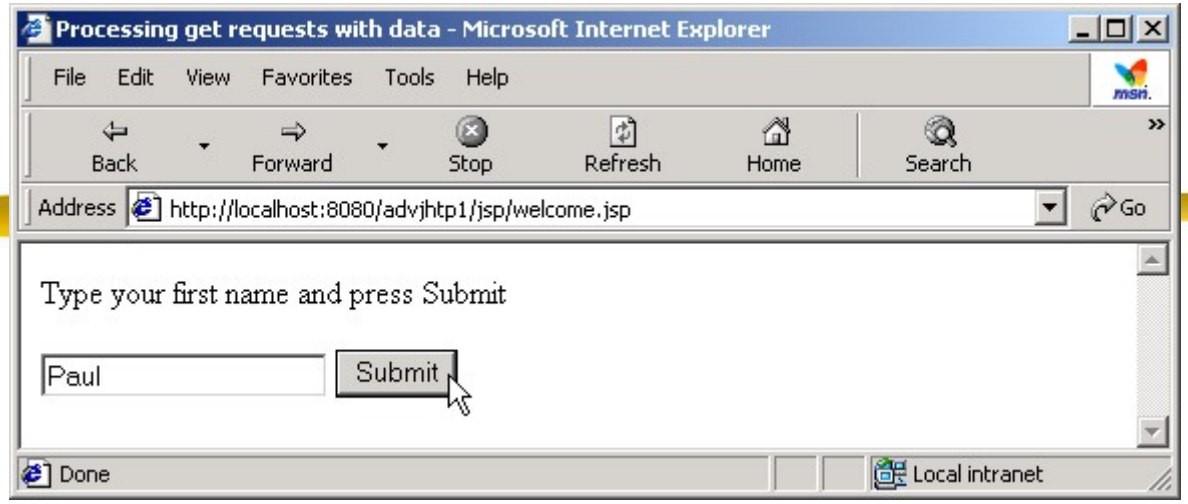
Un esempio, prima di cominciare la trattazione teorica

```
1   <!-- welcome.jsp -->
2   <!-- JSP that processes a "get" request containing data. -->
3
4   <html>
5
6    <!-- head section of document -->
7    <head>
8       <title>Processing "get" requests with data</title>
9    </head>
10
11    <!-- body section of document -->
12    <body>
13       <% // begin scriptlet
14
15          String name = request.getParameter( "firstName" );
16
17          if ( name != null ) {
18
19       %> <%-- end scriptlet to insert fixed template data --%>
20
21             <h1>
22                Hello <%= name %>, <br />
23                Welcome to JavaServer Pages!
24             </h1>
25
26       <% // continue scriptlet
27          }  // end if
```

Uso di scriptlet per inserire codice java

Uso dell'oggetto implicito **`request`** per ottenere il valore di un parametro

```
28          else {
29
30       %> <%-- end scriptlet to insert fixed template data --%>
31
32             <form action = "welcome.jsp" method = "get">
33                <p>Type your first name and press Submit</p>
34
35                <p><input type = "text" name = "firstName" />
36                   <input type = "submit" value = "Submit" />
37                </p>
38             </form>
39
40       <% // continue scriptlet
41
42          }  // end else
43
44       %> <%-- end scriptlet --%>
45    </body>
46
47 </html>  <!-- end XHTML document -->
```


Processing get requests with data - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search
Address http://localhost:8080/advjhtp1/jsp/welcome.jsp
Go
Type your first name and press Submit
Paul
Submit
Done
Local intranet


Processing get requests with data - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search
Address http://localhost:8080/advjhtp1/jsp/welcome.jsp?firstName=Paul
Go
Hello Paul,
Welcome to JavaServer Pages!
Done
Local intranet

Java Server Pages (JSP)

- Java Server Pages
  - Costituiscono un'estensione della tecnologia delle servlet
- Classi e interfacce specifiche per la definizione di pagine JSP:
  - Package **`javax.servlet.jsp`**
  - Package **`javax.servlet.jsp.tagext`**

Java Server Pages (JSP)

- Il JSP container (al momento della prima invocazione della pagina):
  - Legge la pagina JSP
  - Scrive una servlet (corrispondente alla pagina letta)
  - La compila (default usa javac, ma configurabile)
  - La invoca secondo il ciclo di vita della servlet stessa (inizializzazione, servizio)
- Alle invocazioni successive il container fa riferimento alla servlet già caricata in memoria e inizializzata, pertanto esegue solo il metodo di servizio.

Java Server Pages (JSP)

- Come per le servlet, le pagine JSP utilizzano un oggetto:
  - request (rappresentativo della richiesta HTTP pervenuta)
  - response (rappresentativo della risposta HTTP da inviare)
  - hanno inoltre accesso a tutti i dati della richiesta, del contesto e dell'applicazione web.

Java Server Pages - componenti

- Elementi che costituiscono una pagina JSP:
  - Direttive, cioè istruzioni dirette al servlet/JSP container che specificano come gestire la JSP
  - Azioni
    - Scriptlet (e varianti, come espressioni, dichiarazioni ecc.)
    - Azioni standard
    - Tag personalizzati

Java Server Pages - direttive

- Direttive
  - Istruzioni dirette al JSP container
    - i.e. programma che gestisce le pagine JSP fino alla loro esecuzione
  - Consentono al programmatore di specificare
    - Impostazioni e opzioni della pagina
    - Contenuti esterni da includere o package da importare
    - Librerie di tag personalizzati utilizzabili nella pagina

Java Server Pages - direttive

- Sintassi:
  - <%@ direttiva {attr="valore"}%>
    - Es: <%@ page language="java" import= "…."%>
      specifica il linguaggio di scripting
    - Es: <%@ include file="relativeURLspec"%>
      specifica il percorso relativo (URL) di un file che deve essere incluso

Java Server Pages - direttiva page

Direttiva	Descrizione
Direttive **page**	
`import="importlist"`	Fornisce un elenco di package da importare. Utile per non dover scrivere tutto il percorso dei package (nomi di classi pienamente qualificati). Per impostazione predefinita vengono importati: *java.lang.*, javax.servlet.*, javax.servlet.jsp.** e *javax.servlet.http.**
`session="true|false"`	Se *true* la pagina ha accesso alla variabile implicita *session*, che fa riferimento alla sessione attuale. Per impostazione predefinita è *true*.
`isThreadSafe="true|false"`	Se *true*, il container può inviare alla pagina nuove richieste prima che vengano completate le richieste in corso. Se questo attributo è su *false* le richieste verranno inviate progressivamente, con possibili conseguenze a livello di prestazioni. L'impostazione predefinita è *true*.
`errorPage="error_url"`	Se su questa pagina si verifica un'eccezione non catturata, il container passerà alla pagina qui indicata.

Java Server Pages - azioni

- Azioni
  - Sono codificate in un linguaggio di programmazione
  - Specificate sotto forma di
    - scriptlet
      - » codice puro <% sorgente scriptlet %>
      - » Varianti: espressioni <%= espressione %>,
        dichiarazioni <%! Dichiarazione %>,
        commento <%-- commento --%>
    - azione standard

      <jsp:actionName attributo="valore">body</jsp:faiqualcosa>
    - tag personalizzati

      <tagPrefix:tagName attributo="valore">body</tag>

JavaServer Pages - scriptlet

- Scriptlet
  - Sono blocchi di codice eseguiti nel contesto della pagina
  - Consentono l'inserimento di codice Java all'interno della pagina JSP
  - Realizzano l'elaborazione della richiesta
    - Interagiscono con gli elementi della pagina e altre componenti per creare pagine dinamiche

JSP: azioni standard e tag personalizzati

- Azioni standard: tag JSP dal comportamento predefinito, comportano l'esecuzione di codice, parametrizzato in base agli attributi del tag.

- Librerie di tag personalizzati
  - Meccanismo di estensione dell'insieme dei tag JSP predefiniti
  - Consente la definizione di nuovi tag da parte del programmatore
    - Nuovi tag possono incapsulare complesse funzionalità

Traduzione della pagina JSP in una servlet

- Il codice contenuto nella pagina JSP
  - Costituirà un blocco di codice all'interno della definizione di un servlet
    - Metodo di inizializzazione _jspInit()
    - Il metodo **`_jspService()`** conterrà il codice degli scriptlet e una sequenza di istruzioni di write("…") per tutti i tag HTML presenti nella pagina .jsp
    - Metodo di distruzione _jspDestroy()

- Ciclo di vita della JSP simile a quello di una servlet.
  - La prima volta che viene invocata la pagina, il container la traduce in una servlet, che viene compilata e mandata in esecuzione (esecuzione del metodo _jspInit).
  - Il container invoca il metodo _jspService ad ogni richiesta successiva della stessa pagina JSP.

Errori JSP

- Errori al momento della traduzione
  - Si verificano nel momento in cui viene generata la servlet corrispondente alla JSP
- Errori al momento della richiesta
  - Si verificano durante l'elaborazione della richiesta

JSP o Servlet?

- JSP
  - Hanno l'aspetto e la struttura di pagine XHTML
    - Contengono markup HTML o XHTML
  - Vengono utilizzate quando la maggior parte del contenuto che deve essere visualizzato segue una struttura fissata.
    - In generale una piccola parte del contenuto deve essere generata dinamicamente
- Servlet
  - Utilizzate invece quando solo una piccola porzione del contenuto deve seguire una struttura fissata
    - La maggior parte del contenuto deve essere generata dinamicamente

Componenti di scripting JSP

- Come specificare componenti JSP
  - Scriptlets (delimitate da `<%` and `%>`)
  - Commenti (delimitati da `<%--` and `--%>`)
  - Espressioni (delimitati da `<%=` and `%>`)
  - Dichiarazioni (delimitati da `<%!` and `%>`)

Scriptlet

- Delimitate da <% e %>
- Blocchi di codice Java
- Inserite nel metodo **_jspService** al momento della traduzione

- Scriptlet, espressioni e codice XHTML possono essere intercalati per creare diverse risposte sulla base di informazioni incluse nella richiesta

Componenti di scripting JSP (seq. di escape)

Literal	Escape sequence	Description
`<%`	`<\%`	The character sequence `<%` normally indicates the beginning of a scriptlet. The `<\%` escape sequence places the literal characters `<%` in the response to the client.
`%>`	`%\>`	The character sequence `%>` normally indicates the end of a scriptlet. The `%\>` escape sequence places the literal characters `%>` in the response to the client.

`'`
`"`
`\`
	`\'`
	`\"`
	`\\`
		As with string literals in a Java program, the escape sequences for characters `'`, `"` and `\` allow these characters to appear in attribute values. Remember that the literal text in a JSP becomes string literals in the servlet that represents the translated JSP.

Commenti

- Sono supportati tre tipi di commento:
  - Commento JSP
  - Commento XHTML
  - Commento del linguaggio di scripting

Commento JSP

- Commento JSP
  - <%-- --%>
    - Non si usa all'interno di scriptlet
    - Non è visibile al client

Commento XHTML

- Commento XHTML
  - <!-- -->
    - Non si usa all'interno di scriptlet
    - E' visibile al client

Commento del linguaggio di scripting

- Commento del linguaggio di scripting
  - Single-line //, oppure Multi-line /* */
    - Si usa esclusivamente all'interno di scriptlet
    - E' visibile al client?

Espressioni

- Sono delimitate da <%= e %>
- Contengono espressioni Java che vengono valutate quando il client richiede la pagina che le contiene
- Il container converte il risultato di un'espressione in un oggetto **String** e lo invia in output come parte della risposta

Uso di espressioni in una pagina JSP - esempio

Esempio in cui

- Viene creata la struttura della pagina attraverso markup XHTML
- Viene creato un oggetto java (**java.util.Date**)
- Viene effettuata la conversione automatica di un'espressione JSP in un'oggetto **String**
- Viene usato un **meta**-elemento per ricaricare la pagina a specifici intervalli di tempo

- Si noti alla prima invocazione di **clock.jsp** il ritardo con cui
  - Il JSP container traduce la pagina JSP in una servlet
  - Il JSP container compila la servlet
  - Il JSP container esegue la servlet
- Le successive richieste della pagina JSP non sperimentano questo ritardo.

```
4
5
6   <html xmlns = "http://www.w3.org/1999/xhtml">
7
8
9     <head>
10       <meta http-equiv = "refresh" content = "60" />
11
12       <title>A Simple JSP Example</title>
13
14       <style type = "text/css">
15          .big { font-family: helvetica, arial, sans-serif;
16                 font-weight: bold;
17                 font-size: 2em; }
18       </style>
19     </head>
20
21     <body>
22       <p class = "big">Simple JSP Example</p>
23
24       <table style = "border: 6px outset;">
25          <tr>
26             <td style = "background-color: black;">
27                <p class = "big" style = "color: cyan;">
28
29                   <!-- JSP expression to insert date/time -->
30                   <%= new java.util.Date() %>
31
32                </p>
33             </td>
34          </tr>
35       </table>
36     </body>
37
38  </html>
```

`meta` element refresh: ricarica la pagina ogni **`60`** seconds

Crea un oggetto **`Date`** che viene implicitamente convertito in un oggetto **`String`**: si tratta di un'espressione

Dichiarazioni

- Delimitate da <%! e %>
- Consentono la definizione di variabili e metodi attraverso la sintassi di Java
- Le variabili diventano attributi della classe servlet che rappresenta la pagina JSP
- I metodi così dichiarati corrisponderanno ai metodi della classe servlet che rappresenta la pagina JSP
- La stessa variabile senza <%! diventa var locale di _jspService()

Esempio sulla dichiarazione

- Scrivere una pagina.jsp contenente una variabile **contatore** e delle istruzioni di incremento del suo valore.
- Ci sono differenze se la variabile viene utilizzata nei seguenti modi:
  - <% int counter=0; %>
  - <%! int counter=0; %> ?

Cosa scrive nel primo caso?

Nel secondo ?

```
1 <html>
 2 <head>
 3  <title>
 4  Contatore dichiarato come scriptlet
 5  </title>
 6
 7 </head>
 8 <body>
 9
10 <% int counter=0; %>     oppure     <%! int counter=0; %>
11 <% counter++; %>
12 <p>Il contatore vale <%= counter %>.</p>
13 <% counter++; %>
14 <p>Il contatore vale <%= counter %>.</p>
15
16
17 </body>
18 </html>
```

Aree di visibilità (scope) in una pagina JSP

- Le pagine JSP possono accedere ad oggetti definiti in diverse aree di visibilità (scope):
  - Applicazione
    - Oggetti associati al **contesto servlet** della JSP;
    - Per recuperare tali oggetti si ricorre al metodo javax.servlet.ServletContext.**getAttribute()**
  - Pagina
    - Oggetti che sono visibili solo al codice presente sulla stessa pagina
    - Per accedervi si usa javax.servlet.jsp.PageContext.**getAttribute()**
    - Una volta completata la richiesta della pagina il container elimina il riferimento a tali oggetti

Aree di visibilità (continua)

- Richiesta
  - Visibilità uguale alla richiesta
  - Vi si accede con il metodo javax.servlet.ServletRequest.**getAttribute**
  - Viene eliminato il riferimento a tali oggetti quando viene inviata la risposta
- Sessione
  - Oggetti associati ad una sessione utente
  - Vi si accede con il metodo javax.servlet.http.**HTTPSession.getAttribute**
  - I riferimenti a tali oggetti vengono eliminati quando la sessione termina (per volere del client o per timeout)

Oggetti impliciti

Implicit Object	Description
Application Scope	
application	This **javax.servlet.ServletContext** object represents the container in which the JSP executes.
Page Scope	
config	This **javax.servlet.ServletConfig** object represents the JSP configuration options. As with servlets, configuration options can be specified in a Web application descriptor.
exception	This **java.lang.Throwable** object represents the exception that is passed to the JSP error page. This object is available only in a JSP error page.
out	This **javax.servlet.jsp.JspWriter** object writes text as part of the response to a request. This object is used implicitly with JSP expressions and actions that insert string content in a response.
page	This **java.lang.Object** object represents the **this** reference for the current JSP instance.
pageContext	This **javax.servlet.jsp.PageContext** object hides the implementation details of the underlying servlet and JSP container and provides JSP programmers with access to the implicit objects discussed in this table.

Oggetti impliciti (cont.)

Implicit Object	Description
`response`	This object represents the response to the client. The object normally is an instance of a class that implements **`HttpServletResponse`** (package **`javax.servlet.http`**). If a protocol other than HTTP is used, this object is an instance of a class that implements **`javax.servlet.ServletResponse`**.
Request Scope	
`request`	This object represents the client request. The object normally is an instance of a class that implements **`HttpServletRequest`** (package **`javax.servlet.http`**). If a protocol other than HTTP is used, this object is an instance of a subclass of **`javax.servlet.ServletRequest`**.
Session Scope	
`session`	This **`javax.servlet.http.HttpSession`** object represents the client session information if such a session has been created. This object is available only in pages that participate in a session.

Oggetto Page

- L'oggetto page rappresenta l'istanza corrente della servlet corrispondente alla pagina JSP
- Ha come tipo l'interfaccia HTTPJspPage che discende da JSP page, la quale a sua volta estende Servlet
- Può quindi essere quindi utilizzato per accedere a tutti i metodi definiti nelle servlet

```
<%@ page info="Esempio di uso page." %>
<p>Page info: <%=page.getServletInfo() %> </p>
```

JSP

```
<p>Page info: Esempio di uso di page</p>
```

HTML

Oggetto PageContext

- Oggetto che fornisce informazioni sul contesto di esecuzione della JSP
- Rappresenta l'insieme degli oggetti impliciti di una JSP
- Consente l'accesso a tutti gli oggetti impliciti e ai loro attributi attraverso i corrispondenti metodi *get*:

  getPage()
  getRequest()
  getResponse()
  getSession()
  getServletContext()
  getException() ecc.

- Poco usato per lo scripting, utile per costruire custom tags

Oggetto Application

- Oggetto che fornisce informazioni sul contesto di esecuzione della JSP (è il ServletContext)
- Rappresenta la web application a cui la JSP appartiene
- Consente di interagire con l'ambiente di esecuzione:
  - garantisce l'accesso a risorse server-side
  - permette accesso ai parametri di inizializzazione relativi all'applicazione
  - consente di gestire gli attributi di un'applicazione

Oggetto Exception

- Oggetto connesso alla gestione degli errori
- Rappresenta l'eccezione che non viene gestita da nessun blocco catch
- Non è automaticamente disponibile in tutte le pagine ma solo nelle Error Page (quelle dichiarate con l'attributo errorPage impostato a true)

```
<%@ page isErrorPage="true" %>
<h1>Attenzione!</h1>
E' stato rilevato il seguente errore:<br/>
<b><%= exception %></b><br/>
<% exception.printStackTrace(out); %>
```

Azione standard `<jsp:include>`

Reminder: redirezione attraverso direttiva include
<%@ include file="relativeURLspec"%>
specifica il percorso relativo (URL) di un file che deve essere incluso

- **`<jsp:include ...>`**
  - Consente l'inclusione di contenuto **dinamico** in una pagina JSP
  - Più flessibile della direttiva **`include`**
    - Richiede maggiore overhead quando il contenuto della pagina cambia frequentemente
    - La direttiva include il codice al momento della traduzione (l'inclusione corrisponde ad una serie di print), mentre l'azione standard include il codice solo al momento dell'esecuzione

Azione <jsp:include>

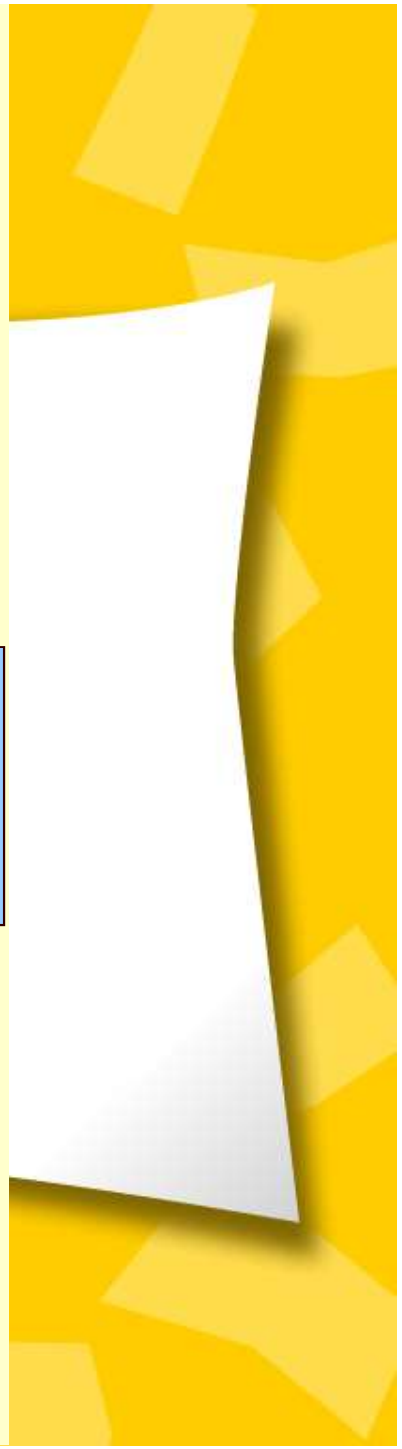
Attribute	Description
page	Specifies the relative URI path of the resource to include. The resource must be part of the same Web application.
flush	Specifies whether the buffer should be flushed before the **include** is performed. In JSP 1.1, this attribute is required to be **true**.

```
<!-- banner.html              -->
<!-- banner to include in another document -->
<div style = "width: 580px">
  <p>
    Corso di
    Laboratorio <br /> Internet and
    World Wide Web Programming Training <br />
    On-Site Seminars Delivered Worldwide
  </p>

  <p>
    <a href = "mailto:novella@di.uniroma1.it">
      novella@di.uniroma1.it</a><br />

    978.579.9911<br />
    490B Boston Post Road, Suite 200,
    Sudbury, MA 01776
  </p>
</div>
```

```
<!-- toc.html                    -->
<!-- contents to include in another document -->

<p><a href = "http://www.uniroma1.it/books/index.html">
  Publications/BookStore
</a></p>

<p><a href = "http://www.uniroma1.it/whatsnew.html">
  What's New
</a></p>

<p><a href = "http://www.uniroma1.it/books/downloads.html">
  Downloads/Resources
</a></p>

<p><a href = "http://www.uniroma1.it/faq/index.html">
  FAQ (Frequently Asked Questions)
</a></p>

<p><a href = "http://www.uniroma1.it/intro.html">
  Who we are
</a></p>

<p><a href = "http://www.uniroma1.it/index.html">
  Home Page
</a></p>

<p>Send questions or comments about this site to
  <a href = "mailto:novella@di.uniroma1.it">
    novella@di.uniroma1.it
  </a><br />


</p>
```

```
1  <!-- Fig. 10.9: clock2.jsp                    -->
2  <!-- date and time to include in another document -->
3
4  <table>
5    <tr>
6      <td style = "background-color: black;">
7        <p class = "big" style = "color: cyan; font-size: 3em;
8          font-weight: bold;">
9
10         <%-- script to determine client local and --%>
11         <%-- format date accordingly             --%>
12         <%
13           // get client locale
14           java.util.Locale locale = request.getLocale();
15
16           // get DateFormat for client's Locale
17           java.text.DateFormat dateFormat =
18             java.text.DateFormat.getDateTimeInstance(
19               java.text.DateFormat.LONG,
20               java.text.DateFormat.LONG, locale );
21
22         %>  <%-- end script --%>
23
24         <%-- output date --%>
25         <%= dateFormat.format( new java.util.Date() ) %>
26       </p>
27     </td>
28   </tr>
29 </table>
```

Use **`Locale`** to format **`Data`** with specified **`DataFormat`**



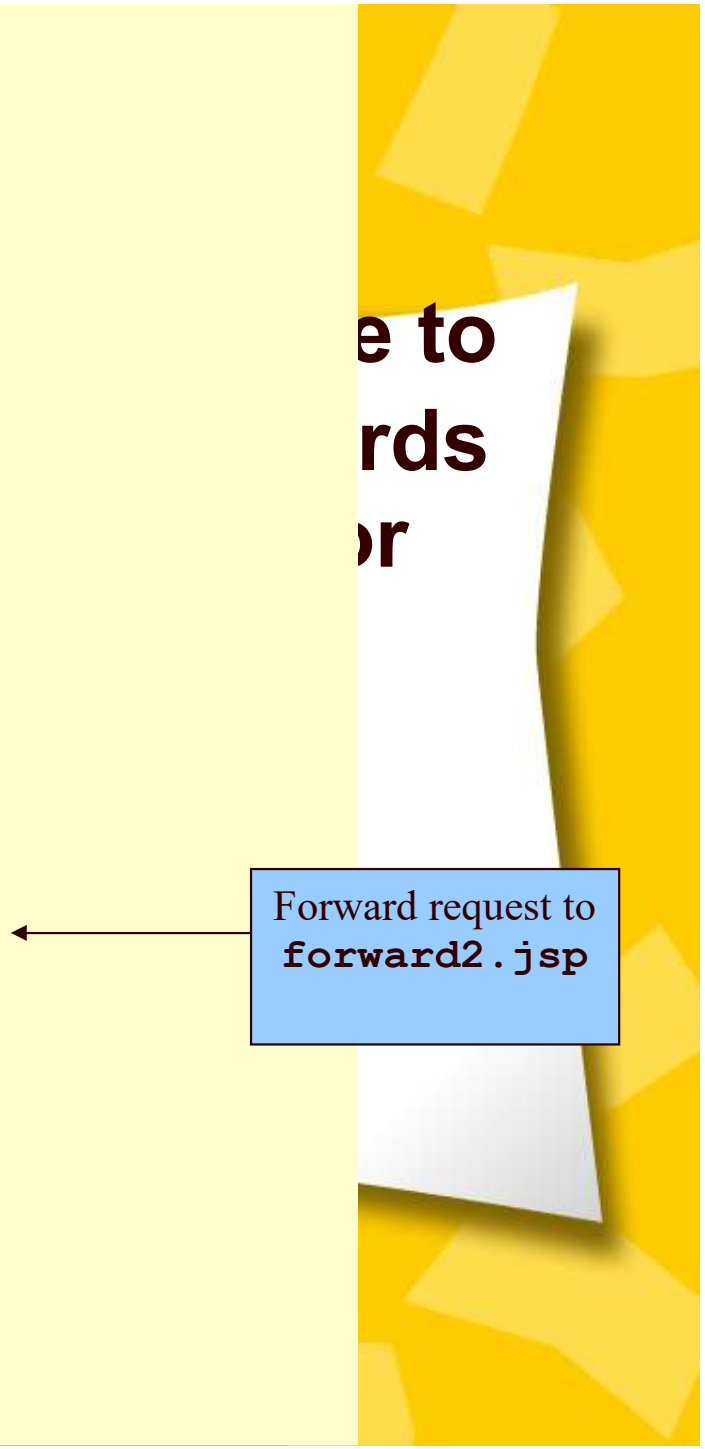
```
<?xml version = "1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- include.jsp -->

<html xmlns = "http://www.w3.org/1999/xhtml">

  <head>
    <title>Using jsp:include</title>

    <style type = "text/css">
      body {
        font-family: tahoma, helvetica, arial, sans-serif;
      }

      table, tr, td {
        font-size: .9em;
        border: 3px groove;
        padding: 5px;
        background-color: #dddddd;
      }
    </style>
  </head>

  <body>
    <table>
      <tr>
        <td style = "width: 160px; text-align: center">
          <img src = "images/logotiny.png"
            width = "140" height = "93"
          />
        </td>

```

```
35          <td>
36
37            <%-- include banner.html in this JSP --%>
38            <jsp:include page = "banner.html"
39              flush = "true" />
40
41          </td>
42        </tr>
43
44        <tr>
45          <td style = "width: 160px">
46
47            <%-- include toc.html in this JSP --%>
48            <jsp:include page = "toc.html" flush = "true" />
49
50          </td>
51
52          <td style = "vertical-align: top">
53
54            <%-- include clock2.jsp in this JSP --%>
55            <jsp:include page = "clock2.jsp"
56              flush = "true" />
57
58          </td>
59        </tr>
60      </table>
61    </body>
62  </html>
```

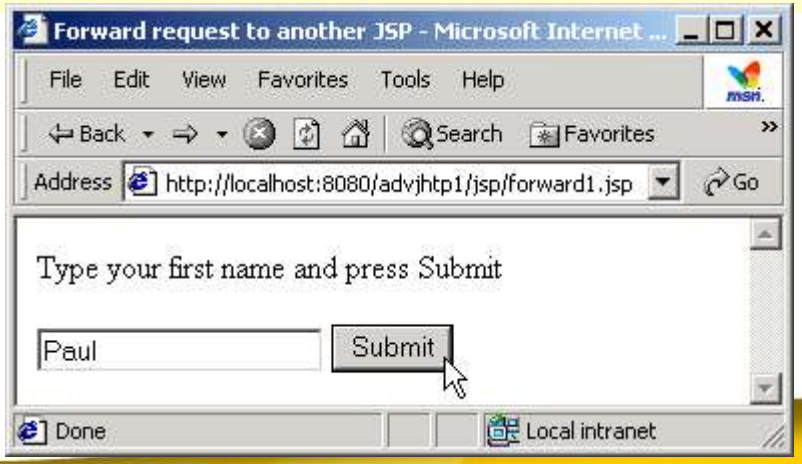
Azione `<jsp:forward>`

- **`<jsp:forward>`**
  - Consente ad una pagina JSP di inoltrare la richiesta ad altre risorse
- L'azione **`<jsp:param>`** (annidata nel forward) specifica coppie nome/valore di dati da allegare ad altre azioni

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. 10.11: forward1.jsp -->
6
7  <html xmlns = "http://www.w3.org/1999/xhtml">
8
9  <head>
10   <title>Forward request to another JSP</title>
11 </head>
12
13 <body>
14   <% // begin scriptlet
15
16     String name = request.getParameter( "firstName" );
17
18     if ( name != null ) {
19
20   %> <%-- end scriptlet to insert fixed template data --%>
21
22       <jsp:forward page = "forward2.jsp">
23         <jsp:param name = "date"
24           value = "<%= new java.util.Date() %>" />
25       </jsp:forward>
26
27   <% // continue scriptlet
28
29     } // end if
30     else {
31
32   %> <%-- end scriptlet to insert fixed template data --%>
33
```



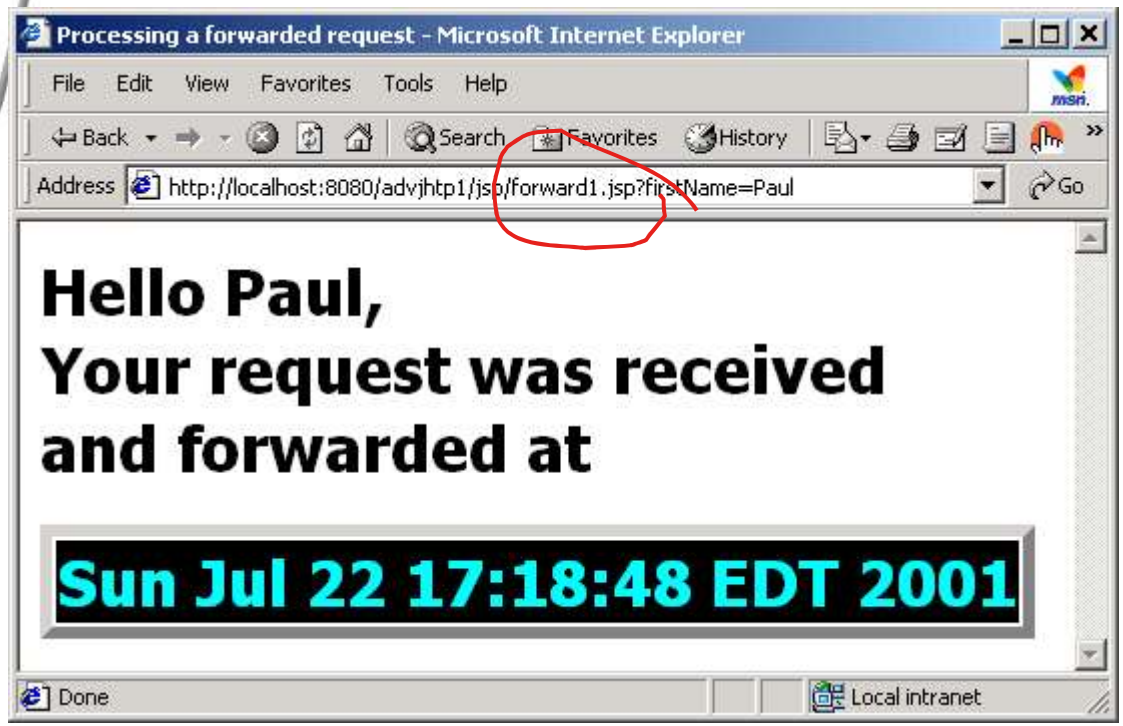
```
34        <form action = "forward1.jsp" method = "get">
35          <p>Type your first name and press Submit</p>
36
37          <p><input type = "text" name = "firstName" />
38            <input type = "submit" value = "Submit" />
39          </p>
40        </form>
41
42    <%  // continue scriptlet
43
44      }  // end else
45
46    %> <%-- end scriptlet --%>
47  </body>
48
49  </html>  <!-- end XHTML document -->
```



```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- forward2.jsp -->
6
7  <html xmlns = "http://www.w3.org/1999/xhtml"v
8
9  <head>
10   <title>Processing a forwarded request</title>
11
12   <style type = "text/css">
13     .big {
14       font-family: tahoma, helvetica, arial, sans-serif;
15       font-weight: bold;
16       font-size: 2em;
17     }
18   </style>
19 </head>
20
21 <body>
22   <p class = "big">
23     Hello <%= request.getParameter( "firstName" ) %>, <br />
24     Your request was received <br /> and forwarded at
25   </p>
26
27   <table style = "border: 6px outset;">
28     <tr>
29       <td style = "background-color: black;">
30         <p class = "big" style = "color: cyan;">
31           <%= request.getParameter( "date" ) %>
32         </p>
33       </td>
34     </tr>
35   </table>
```

Receive request from **`forward1.jsp`**, then get **`firstName`** parameter from request

Get **`data`** parameter from request



Redirezione
INTERNA