


Elementi fondamentali JSP

Utilizzo degli Oggetti Impliciti (Built-in) in JSP

JSP (JavaServer Pages) semplifica la programmazione web fornendo oggetti impliciti, istanziati automaticamente nell'ambiente JSP. Non è necessario importarli o istanziarli manualmente, rendendo più agevole lo sviluppo delle pagine. Per accedervi, è sufficiente utilizzare la sintassi `nomeOggetto.nomeMetodo`.

Oggetti Impliciti Disponibili in Pagine JSP

Gli oggetti impliciti in JSP sono accessibili direttamente nelle pagine senza la necessità di dichiararli. I principali oggetti sono:

- `out`: Per scrivere codice HTML nella risposta (analogamente a System.out di Java).
- `session`: Contiene dati specifici della sessione utente corrente.
- `request`: Fornisce informazioni sulla richiesta HTTP, inclusi attributi, header, cookie, parametri, ecc.
- `page`: Rappresenta la pagina JSP e le sue proprietà.
- `config`: Contiene dati di configurazione per la pagina.
- `response`: Rappresenta la risposta HTTP e le sue proprietà.
- `application`: Contiene dati condivisi da tutte le pagine della web application.
- `exception`: Utilizzato per gestire eventuali eccezioni lanciate dal server, utile per pagine di errore.
- `pageContext`: Fornisce dati di contesto per l'esecuzione della pagina.

Categorie di Oggetti Impliciti in JSP

Gli oggetti impliciti in JSP possono essere suddivisi in diverse categorie:

- **Oggetti legati alla servlet relativa alla pagina JSP**: Come `out` e `page`.
- **Oggetti legati all'input e all'output della pagina JSP**: Come `request` e `response`.
- **Oggetti che forniscono informazioni sul contesto di esecuzione**: Come `session` e `application`.
- **Oggetti risultanti da eventuali errori**: Come `exception`.

Questo approccio semplifica notevolmente la gestione delle informazioni e l'interazione con l'ambiente di esecuzione all'interno delle pagine JSP.

Ambito delle Variabili in JSP

In JSP (JavaServer Pages), il concetto di scope è applicato attraverso l'uso di oggetti impliciti noti come oggetti di ambito (scope objects). Gli oggetti di ambito consentono di memorizzare e recuperare dati durante il ciclo di vita di una richiesta HTTP. Ci sono quattro tipi principali di oggetti di ambito in JSP:

1. **Page Scope**: La variabile è valida solo per la durata della richiesta e della pagina. Non è condivisa con altre pagine o richieste.
2. **Request Scope**: La variabile è valida solo per la durata della richiesta. Può essere condivisa tra le diverse pagine all'interno della stessa richiesta.
3. **Session Scope**: La variabile è valida per tutta la sessione dell'utente. Può essere condivisa tra le diverse richieste dello stesso utente.
4. **Application Scope (o ServletContext Scope)**: La variabile è valida per l'intera applicazione web. Può essere condivisa tra le diverse sessioni e richieste degli utenti.

Ecco un esempio di utilizzo di variabili con differenti ambiti in un'applicazione JSP:

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@ page import="java.util.ArrayList" %>
<%@ page import="java.util.List" %>

<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Scope Example</title>
</head>
<body>

<%
    // Page Scope: Variabile valida solo per questa pagina
    pageContext.setAttribute("pageVariable", "Page Scope Variable");

    // Request Scope: Variabile valida solo per questa richiesta
    request.setAttribute("requestVariable", "Request Scope Variable");

    // Session Scope: Variabile valida per tutta la sessione dell'utente
    session.setAttribute("sessionVariable", "Session Scope Variable");

    // Application Scope: Variabile valida per l'intera applicazione
    application.setAttribute("applicationVariable", "Application Scope
Variable");
%>

<h2>Page Scope Variable: <%= pageContext.getAttribute("pageVariable") %>
</h2>
<h2>Request Scope Variable: <%= request.getAttribute("requestVariable") %>
</h2>
<h2>Session Scope Variable: <%= session.getAttribute("sessionVariable") %>
</h2>
<h2>Application Scope Variable: <%=
application.getAttribute("applicationVariable") %></h2>
```

```
</body>
</html>
```

In questo esempio:

- `pageVariable` è una variabile con ambito di pagina.
- `requestVariable` è una variabile con ambito di richiesta.
- `sessionVariable` è una variabile con ambito di sessione.
- `applicationVariable` è una variabile con ambito di applicazione.

Le variabili vengono impostate e lette utilizzando gli oggetti di ambito appropriati (`pageContext`, `request`, `session`, `application`).

Ciclo di Vita di una Pagina JSP

Il ciclo di vita di una pagina JSP comprende diverse fasi:

1. **Salvataggio**: La pagina viene salvata nella cartella pubblica del server web alla prima richiesta ricevuta dal Web server.
2. **Traduzione**: La pagina JSP viene automaticamente tradotta in un sorgente Java chiamato **Servlet**.
3. **Compilazione**: La servlet viene compilata come un programma Java.
4. **Esecuzione**: La servlet viene caricata in memoria ed eseguita.
5. **Esecuzione Successiva**: Successivamente, la pagina JSP (la servlet) viene solo eseguita. In fase di debug, il web server verifica se la pagina JSP è più recente del servlet corrispondente.

Rispetto ad altre tecnologie server-side come PHP o ASP, questa differenza è vantaggiosa in termini di velocità di risposta. Dopo la prima esecuzione, il codice è già compilato e disponibile immediatamente. In PHP e ASP, il web server deve interpretare il codice ad ogni richiesta prima di servire la pagina di risposta.

Elementi di una Pagina JSP

Una pagina JSP è costituita da diversi elementi:

- **Codice HTML**: Contenuto HTML normale per la presentazione.
- **Marcatori JSP**: Sono segnali per l'inizio e la fine di porzioni di codice Java all'interno della pagina JSP.
- **Direttive al Server**: `<%@ direttive %>` per fornire informazioni al server, come importazioni e configurazioni.
- **Elementi di Scripting**: `<%! dichiarazioni %>`, `<%= espressioni %>`, `<% scriptlet %>` per inserire codice Java all'interno della pagina.
- **Commenti**: `<%-- commenti --%>` per aggiungere commenti che non saranno visibili nella pagina generata.
- **Azioni Standard**: Utilizzo di tag JSP come `<jsp:include>`, `<jsp:forward>`, `<jsp:param>`, `<jsp:useBean>`, `<jsp:setProperty>`, `<jsp:getProperty>`, `<jsp:plugin>`.
- **Azioni Personalizzate (Custom Tags)**: Utilizzo di tag personalizzati tramite librerie di tag (Tag Libraries), come ad esempio le tag libraries JSTL (JavaServer Pages Standard Tag Library).

Questi elementi consentono di creare pagine dinamiche e interattive, incorporando logica Java all'interno del contenuto HTML.

Inclusioni JSP

m27-222-851