

## 9.5 JavaServer Pages (JSP)

- Grundidee: auf dem Server werden textuelle Beschreibungen abgelegt, wie Anfragen zu Antworten verarbeitet werden
- [active pages](#)
- Vorteil: aktiver Inhalt ohne Eingriff der Server-Administration
- Konkurrenz: ASP, PHP, ...

# Eigenschaften von JSP

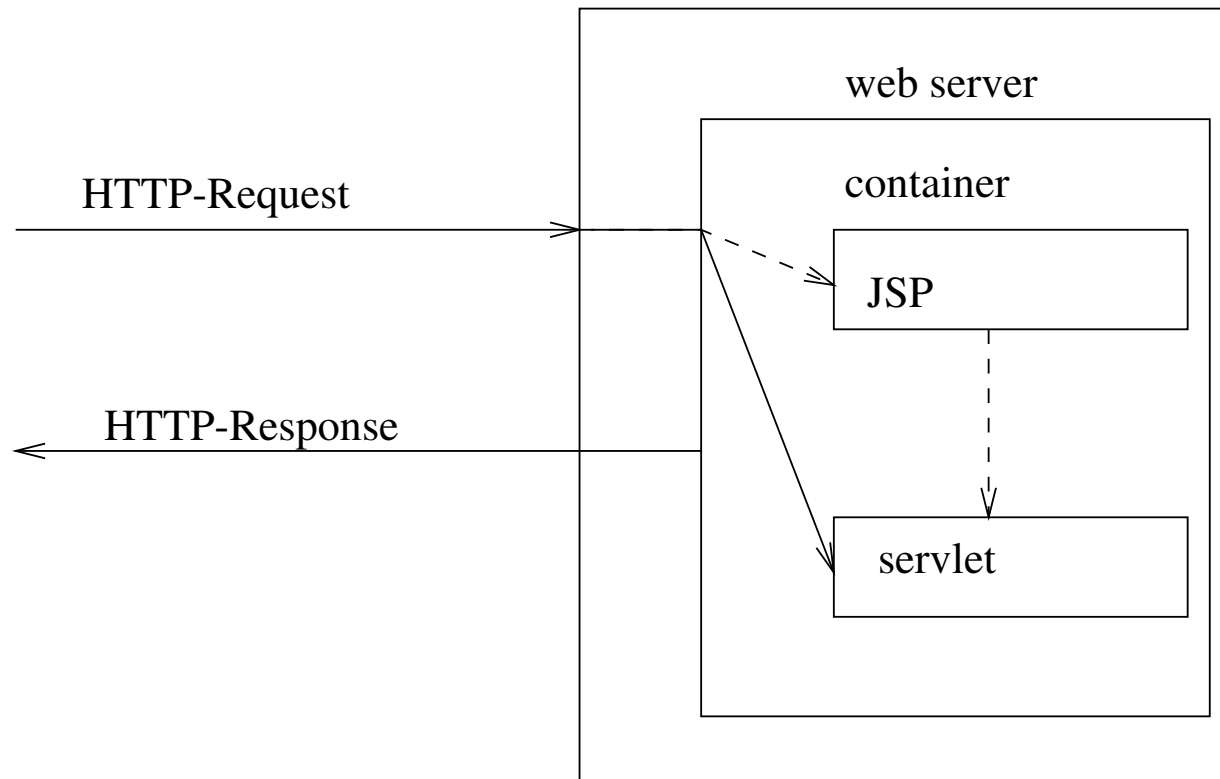
- Flexibles Werkzeug zur Erzeugung von Web-Sites mit dynamischem Inhalt
- Basiert auf Java (*Write Once, Run Anywhere*) und Servlets
- Sämtliche Java-Tools und Packages
- Wiederverwendung: Template-Seiten
- Skripting
- Frontend für Anwendungen mit JavaBeans etc
- Einfache Erzeugung von XML

# Beispiel

```
<html>
  <!-- Zugriff auf einen Kalender -->
  <jsp:useBean id="clock" class="calendar.jspCalendar" />
  <ul>
    <li>Tag: <%=clock.getDayOfMonth () %>
    <!-- Methode für Monat fehlt! --%>
    <li>Jahr: <%=clock.getYear () %>
  </ul>
</html>
```

- Kommentare
- Deklarationen
- vordefinierter Text (Template)
- Methodenaufruf eines Objekts im Server

# JSP



# Inhalt einer JSP

- Standarddirektiven
- Standardaktionen
- Deklarationen, Skripte und Ausdrücke der Skriptsprache  
(Java ist erforderlich, andere können unterstützt werden)
- Mechanismus für Tag-Erweiterung

# Lebenszyklus einer JSP

- Client: Anfrage an eine JSP
- Server: Weiterleiten an JSP Container
- Container: Übersetzung JSP → Servlet (falls noch nicht geschehen)
- Container: Laden des Servlets (falls noch nicht geschehen)
- Container: Initialisierung der Servlets; `jspInit ()` (falls noch nicht geschehen)
- Container: Weiterleitung der Anfrage
- Servlet: Ausführung der Anfrage
- Servlet: Generierung der Antwort
- Container, Server: Weiterleitung der Antwort
- Container: Entfernen des Servlets; `jspDestroy ()`

# Skript-Elemente

- Deklaration

```
<%! <Deklarationen der Skript-Sprache> %>
```

```
<jsp:declaration>
```

```
  <![CDATA[ <Deklaration der Skript-Sprache> ]]>
```

```
</jsp:declaration>
```

- Skripte (*scriptlet*)

```
<% <Fragmente von Anweisungen der Skript-Sprache> %>
```

```
<jsp:scriptlet>
```

```
  <Fragmente von Anweisungen der Skript-Sprache>
```

```
</jsp:scriptlet>
```

- Ausdrücke

```
<%= <Ausdruck der Skript-Sprache> %>
```

```
<jsp:expression>
```

```
  <Ausdruck der Skript-Sprache>
```

```
</jsp:expression>
```

# Beispiel: Echo als JSP

```
<!-- echo.jsp --%>  
  
<%@ page info="Echo Servlet als JSP" %>  
  
<%@ page import="java.io.*" %>  
<%@ page import="java.util.*" %>  
<!-- import javax.servlet.*; automatisch importiert --%>  
<!-- import javax.servlet.http.*; automatisch import --%>  
  
<%@ page contentType="text/html" %>
```



# Beispiel: Echo als JSP

```
<html>
<head><title>Echo Results</title></head>
<body>
<h3>Echo Results</h3>
<ul>

<%
    Enumeration e = request.getParameterNames ();
    while (e.hasMoreElements ()) {
        String name = (String)e.nextElement ();
        String value = request.getParameter (name);
%>
<li> <%= name %>: <%= value %>
<%
    }
%>

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

# Beispiel: NumberGuess als JSP

```
<!-- numberGuess.jsp -->

<%@ page info="NumberGuess Servlet als JSP" %>

<%@ page import="java.io.*" %>

<%@ page session="true" %>
<%@ page contentType="text/html" %>

<html>

<head>
  <title>Number Guess</title>
</head>

<body>

<h3>Number Guess</h3>
```

## Beispiel: NumberGuess als JSP

```
<% String guessString = request.getParameter ("guess");
    if (guessString == null) {
        long n = Math.round (Math.random () * 100);
        session.setAttribute ("SN", new Long (n));
%> I am thinking of a number from 1-100 <%
    } else {
        long guess = Long.parseLong (guessString);
        long n = ((Long)session.getAttribute ("SN")).longValue ();
        if (guess == n) {
%> You got it! <%
            } else if (guess > n) {
%> Try lower <%
                } else {
%> Try higher <%
                    }
            }
        String uri = request.getRequestURI ();
%>
<form action ="<%= uri %>" method="get">
    <input type="text" name="guess">
    <input type="submit" value="Make A Guess">
</form>
</body>
</html>
```

# Zugriffszähler

```
<% response.setDateHeader("Expires", 0); %>
<html>
  <head><title>JSP</title></head>
  <body><h1>Hello World!</h1>
  <%! int hits = 0; %>
  You are visitor number
  <% synchronized(this) { out.println(++hits); } %>
  since the last time the service was restarted.
  <p> This page was last updated:
  <%= new java.util.Date().toLocaleString() %>
  </p></body>
</html>
```

## 9.5.1 Kontext eines Servlets

- Automatisch deklarierte Variablen (vgl. Servlets)

```
HttpServletRequest request;
```

```
HttpServletResponse response;
```

```
HttpSession session;
```

```
ServletContext application;
```

```
ServletConfig config;
```

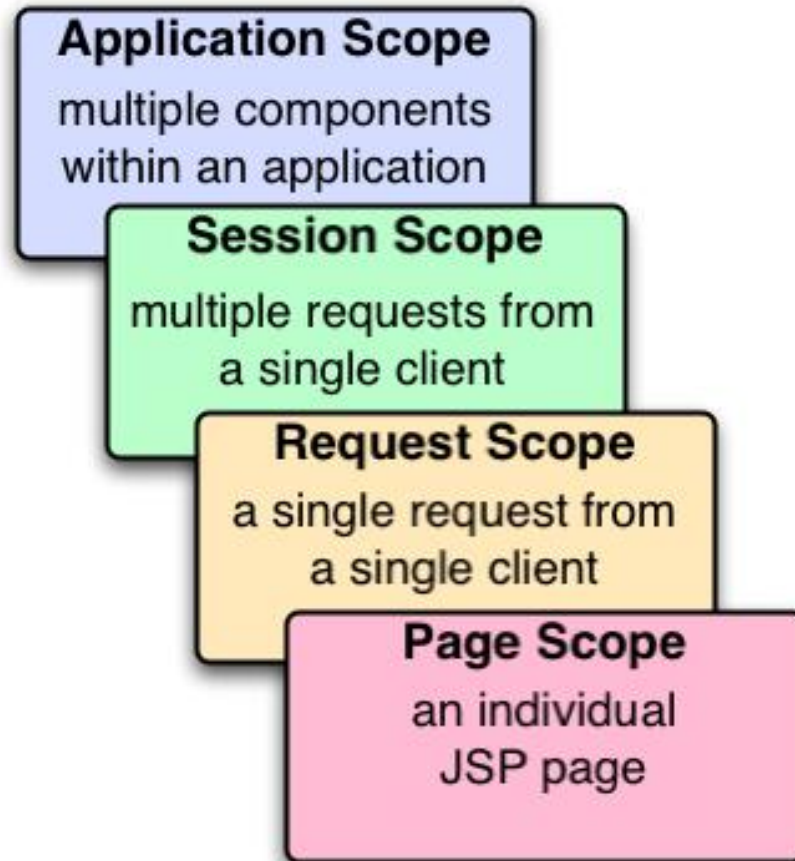
- Ausgabekanal (Exceptions, Pufferung)

```
JspWriter out;
```

# Zustandsverwaltung (Attribute)

- Attribut = benanntes Objekt
- abgelegt in und zugreifbar durch Laufzeitumgebung
- Vier Zustandsebenen (*scoped attributes*)
  - *page scope*  
PageContext pageContext;  
getAttribute, setAttribute (alle Ebenen zugreifbar)  
findAttribute durchsucht alle Ebenen
  - *request scope*
  - *session scope* (auch über Session-Objekt)
  - *application scope* (auch über Kontext-Objekt)

# Funktion der Ebenen



## 9.5.2 Java Ausdrücke

```
<html>
  <head><title>Addition</title></head>
  <body>
    The sum of
    <%= request.getParameter("x") %> and
    <%= request.getParameter("y") %> is
    <%= Integer.parseInt(request.getParameter("x")) +
      Integer.parseInt(request.getParameter("y")) %>
  </body>
</html>
```



## 9.5.3 Java Statements

```
<html>
  <head><title>Numbers</title></head>
  <body>
    <ul>
      <% int n = Integer.parseInt(request.getParameter("n"));
        for (int i=0; i<n; i++)
          out.println("<li>" + i + "</li>");
      %>
    </ul>
  </body>
</html>
```

## 9.5.4 Java Deklarationen

- Syntax `<%! Deklaration %>`
- Variablen Deklarationen (Instanzvariable der Servletklasse)
- Methoden Deklarationen

```
<%! int add(String x, String y) {  
    return Integer.parseInt(x)+Integer.parseInt(y);  
} %>
```

```
<html> <head><title>Addition</title></head>  
<body> The sum of <%= request.getParameter("x") %> and  
    <%= request.getParameter("y") %> is  
    <%= add(request.getParameter("x"),request.getParameter("y")) %>  
</body></html>
```

## 9.5.5 Direktiven

- Parameter für den JSP Prozessor
- Syntax

```
<%@ Directive (<Attribut>="<Wert>")* %>
```

- Abkürzung für XML-Element

```
<jsp:directive.<Directive> (<Attribut>="<Wert>")* />
```

## Direktive include

$\langle \textit{Direktive} \rangle ::= \textit{include}$

- Statisches Einfügen während der Übersetzung in Servlet
- Ersetzt durch Dateiinhalt
- Beispiel

```
<%@ include file="relativeURL" %>
```

## Beispiel für include

- Datei header.jsp

```
<html>
```

```
  <head><title><%= title %></title></head>
```

```
  <body>
```

- Datei footer.jsp

```
  </body>
```

```
</html>
```

## Beispiel für include (Forts.)

```
<%! String title = "Addition"; %>
```

```
<%@ include file="header.jsp" %>
```

The sum of `<%= request.getParameter("x") %>`

and `<%= request.getParameter("y") %>`

is `<%= Integer.parseInt(request.getParameter("x")) +  
Integer.parseInt(request.getParameter("y")) %>`

```
<%@ include file="footer.jsp" %>
```

# Direktive page

- $\langle \text{Direktive} \rangle ::= \text{page}$ 
  - mehrfach erlaubt, aber jedes Attribut nur einmal definierbar
  - Beispiel

```
<%@ page info="meine erste JSP" %>
<%@ page buffer="none" isThreadSafe="yes" errorPage="/fehler.jsp" %>
```
  - Attribute
    - \* `buffer="⟨bufferSpec⟩"`  
Größe oder none
    - \* `contentType="⟨contentType⟩"`
    - \* `pageEncoding="⟨encoding⟩"`  
ISO-8859-1
    - \* `info="⟨ServletInfoString⟩"`
    - \* `import="⟨importList⟩"`
    - \* `isThreadSafe="⟨yes-or-no⟩"`
    - \* `errorPage="⟨URL⟩"`  
bei uncaught exception
    - \* `isErrorPage="⟨boolean⟩"`  
Fehler über Variable exception

## Beispiel page Direktive

```
<%@ page errorPage="error.jsp" %>
<html>
  <head><title>Division</title></head>
  <body>
    <% int n = Integer.parseInt(request.getParameter("n")); %>
    <% int m = Integer.parseInt(request.getParameter("m")); %>
    <%= n %>/<%= m %> equals <%= n/m %>
  </body>
</html>
```



## Beispiel page Direktive

```
<%@ page isErrorPage="true" %>
<html>
  <head><title>Error</title></head>
  <body>
    Something bad happened:
    <%= exception.getMessage() %>
  </body>
</html>
```

## Direktive taglib

- $\langle \textit{Direktive} \rangle ::= \textit{taglib}$ 
  - Def. neuer Tags zu Objekten und Methoden aus  $\langle \textit{libraryURI} \rangle$   
 $\langle \%@ \textit{taglib} \textit{uri} = \langle \textit{libraryURI} \rangle \textit{prefix} = \langle \textit{tagPrefix} \rangle \% \rangle$
  - Beispiel  
 $\langle \%@ \textit{taglib} \textit{uri} = \textit{lib.cream.com} \textit{prefix} = \textit{cream} \% \rangle$   
...  
 $\langle \textit{cream:whip} \rangle$   
...  
 $\langle / \textit{cream:whip} \rangle$

## 9.5.6 JSP Expression Language

- Ziel: Anwendungen ohne Java Syntax / Kenntnisse
- Syntax `${ expression }`
- Überall außerhalb von Skriptlets
- Ersetzt durch Wert des *expression*
- Syntax von *expression*: vergleichbar mit JavaScript (vgl. JSP-Spezifikation)
- Objektoperationen  
`${shoppingCart.price}`  
wird übersetzt nach  
`pageContext.findAttribute ("shoppingCart").getPrice()`

# Implizite Objekte in der EL

- param Request Parameter
- pageContext
- cookie
- Beispiel

```
<html>  
  <head><title>Addition</title></head>  
  <body bgcolor="{param.color}">  
    The sum of {param.x} and {param.y} is  
    {param.x+param.y}  
  </body>  
</html>
```

## 9.5.7 Beispiel: Einkaufswagen

```
<%@ page import="java.util.*" %>
<%! void addToCart(Map cart, String item, int amount) {
    if (amount<0)
        return;
    Integer a = (Integer)cart.get(item);
    if (a==null)
        a = new Integer(0);
    cart.put(item, new Integer(a.intValue()+amount));
}
%>
```

```
<html>
  <head>
    <title>Widget Inc.</title>
  </head>
  <body>
    <h1>Widget Inc. Online Shopping</h1>
    <% Map cart;
      if (session.isNew()) {
        cart = new TreeMap();
        session.setAttribute("cart", cart);
      } else
        cart = (Map)session.getAttribute("cart");
```

```
if (request.getMethod().equals("POST")) {
    String item = request.getParameter("item");
    String amount = request.getParameter("amount");
    if (item!=null)
    try {
        addToCart(cart, item, Integer.parseInt(amount));
    } catch (Exception e) {
        response.sendError(response.SC_BAD_REQUEST,
            "malformed request");
    }
}
String url = request.getRequestURI();
%>
```

```
<form method="post" action="<%= response.encodeURL(url) %>">
  Item: <input type="text" name="item" size="20"/>
  Amount: <input type="text" name="amount" size="5"/>
  <input type="submit" name="submit"
    value="Add to shopping cart"/>
</form>
<br/>
<% if (cart.isEmpty()) { %>
  Your shopping cart is empty.
<% } else { %>
  Your shopping cart now contains:<br/>
```



```

<table border="1"><tr><th>Item</th><th>Amount</th></tr>
<% Iterator i = cart.entrySet().iterator();
    while (i.hasNext()) {
        Map.Entry me = (Map.Entry)i.next();
%>
        <tr><td><%= me.getKey() %></td>
        <td align="right"><%= me.getValue() %></td></tr>
<% } /* if (cart.isEmpty) */ %>
</table><br/>
<form method="post" action="<%= response.encodeURL("buy") %>">
    <input type="submit" name="submit"
        value="Proceed to cashier" />
</form>
<% } /* if (session.isNew()) */ %>
</body>
</html>

```