Web-Programmierung (WPR)

Vorlesung VI.

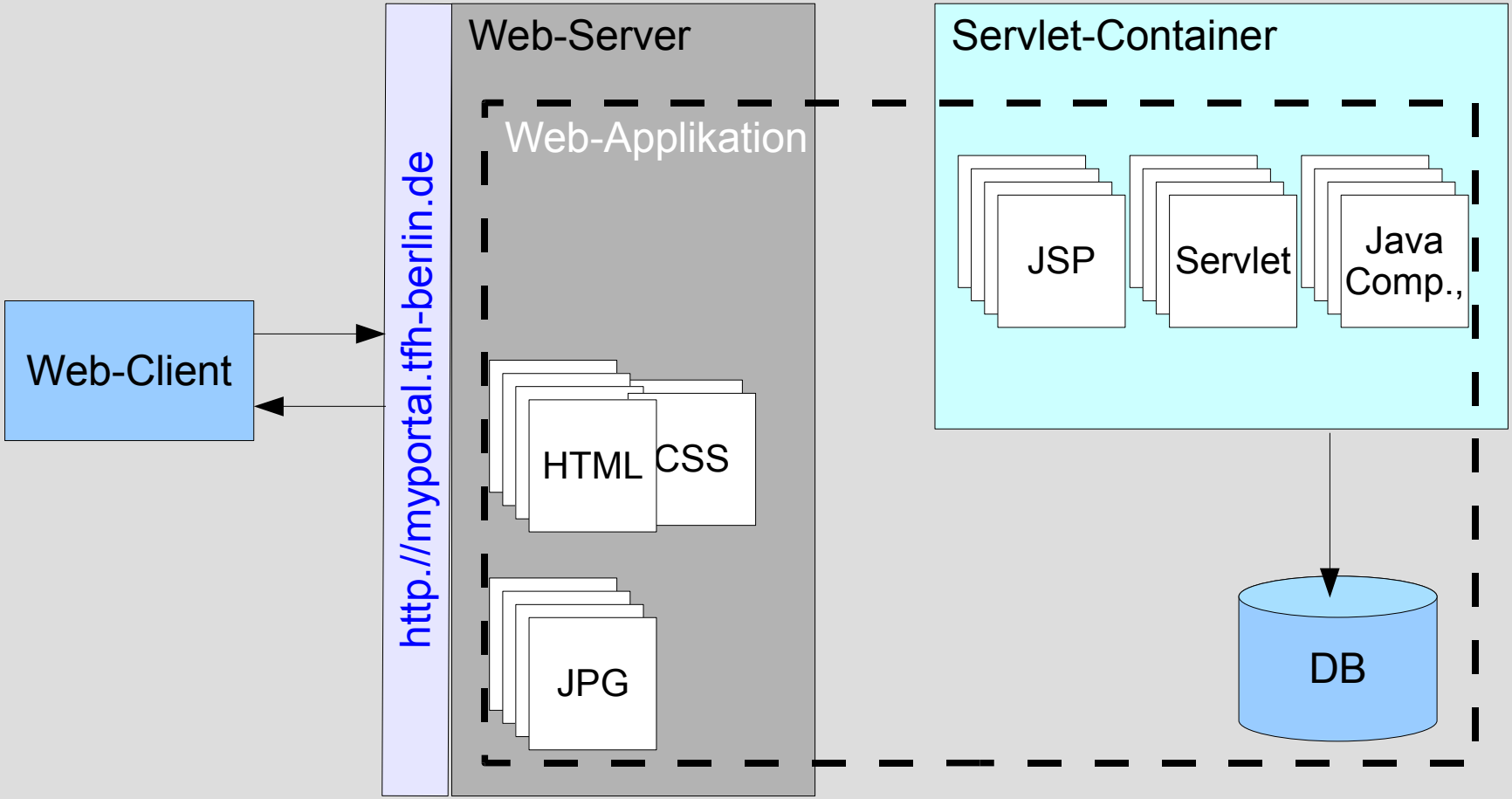
JavaServets u. Java Server Pages (JSP)

mailto:wpr@gruner.org

14.1 Java Servlets

- Java Servlets = Java-Klassen
- Dynamisches Laden beim Aufruf
- Laufzeitumgebung = Servlet-Container
  - Standalone Servlet Engine (Apache Tomcat)
  - Application Server (IBM-Websphere Appl. Server)
  - Einbindung in Standard-WebServer (Apache HTTP-Server)

14.1 Java Servlets



14.1 Java Servlets

Vorteil	Bedeutung
Portablilität	Vollständige Implementierung in Java.
Leistungsfähigkeit	Multithreading, Serialisierung, sowie Leistungsfähigkeit von Java (Anbindung Legacy Systeme, Datenbank, ...)
Effizienz	Einmal geladen verbleiben Servlets im Speicher des Servlet-Container. Keine Erzeugung von Kind Prozessen wie bei CGI.
Sicherheit	Java stelle robuste Laufzeitumgebung zur Verfügung sowie Sicherheitsmechanismen des Java Security Managers
Einfachheit	Servlet-API = einfach u. Übersichtlich, z.B.: SessionTracking

14.1 Java Servlets

- Java Servlet API - Packages
  - `javax.servlet`
  - `javax.servlet.http`
    - `HttpServlet`
    - `HttpServletRequest`
    - `HttpServletResponse`

14.1 Java Servlets

```
package webmonitorlog;
import javax.servlet.*;
import javax.servlet.http.*;

public class WebMonitorLogServlet extends HttpServlet {

    protected void doGet(HttpServletRequest request,
                         HttpServletResponse response)
    throws ServletException, IOException {
        doPost(request, response);
    }

    protected void doPost(HttpServletRequest request,
                          HttpServletResponse response)
    throws ServletException, IOException {
       ......
    }
}
```

14.1 Java Servlets

- doPost – Servlet-Methode

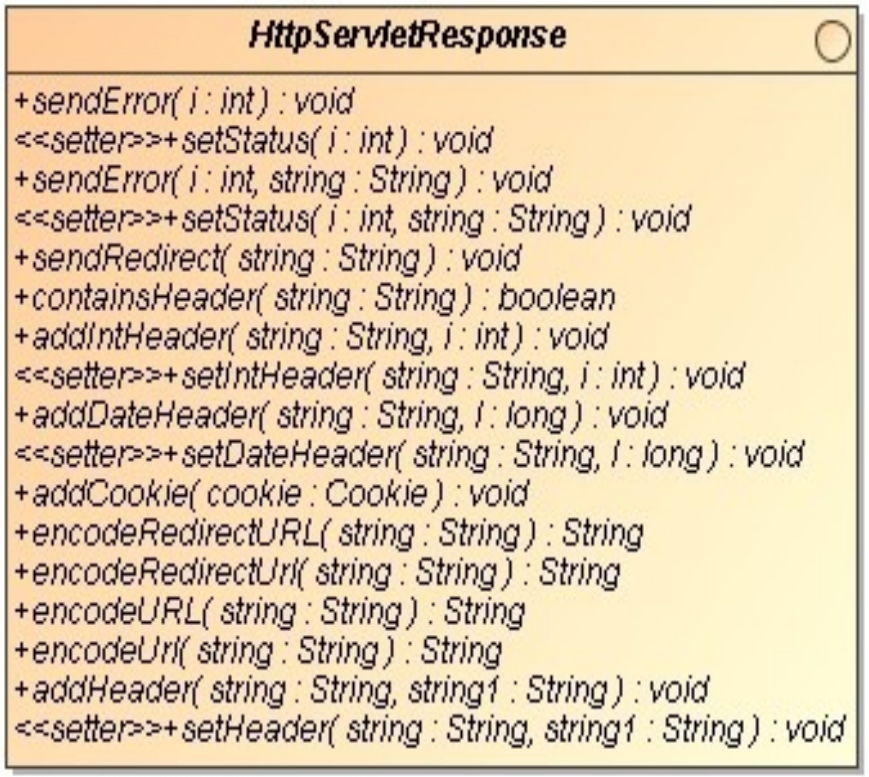
```
 protected void doPost(HttpServletRequest request,
                HttpServletResponse response)
    throws ServletException, IOException {
{

response.setContentType(„text/html“);
PrintWriter printWriter = response.getWriter();
PrintWriter.println(„<html>“);

printWriter.println(„<head><title>WebMonitorLog</title> </head>“);
printWriter.println(„<body“);
printWriter.println(„<h1>Hello MonitorLog</h1>“);
printWriter.println(„</body></html>“);

}
```

14.1 Java Servlets



14.2 Servlet-Lebenszyklus

- Servlet wird ERST instanziiert, wenn es gebraucht wird.
- Alle (auch parallele) Anfragen an dieses Servlet laufen über nur diese eine Instanz.
- Beispiel: Zähler

```
...
counter++;
response.setContentType(„text/html“);
PrintWriter pw = response.getWriter();
pw.println(„<html>“);

pw.println(„<head><title>WebMonitorLog</title> </head>“);
pw.println(„<body“);
pw.println(„<h1>Hello MonitorLog call“+ counter +“</h1>“);
pw.println(„</body></html>“);
...
```

14.2 Servlet-Lebenszyklus

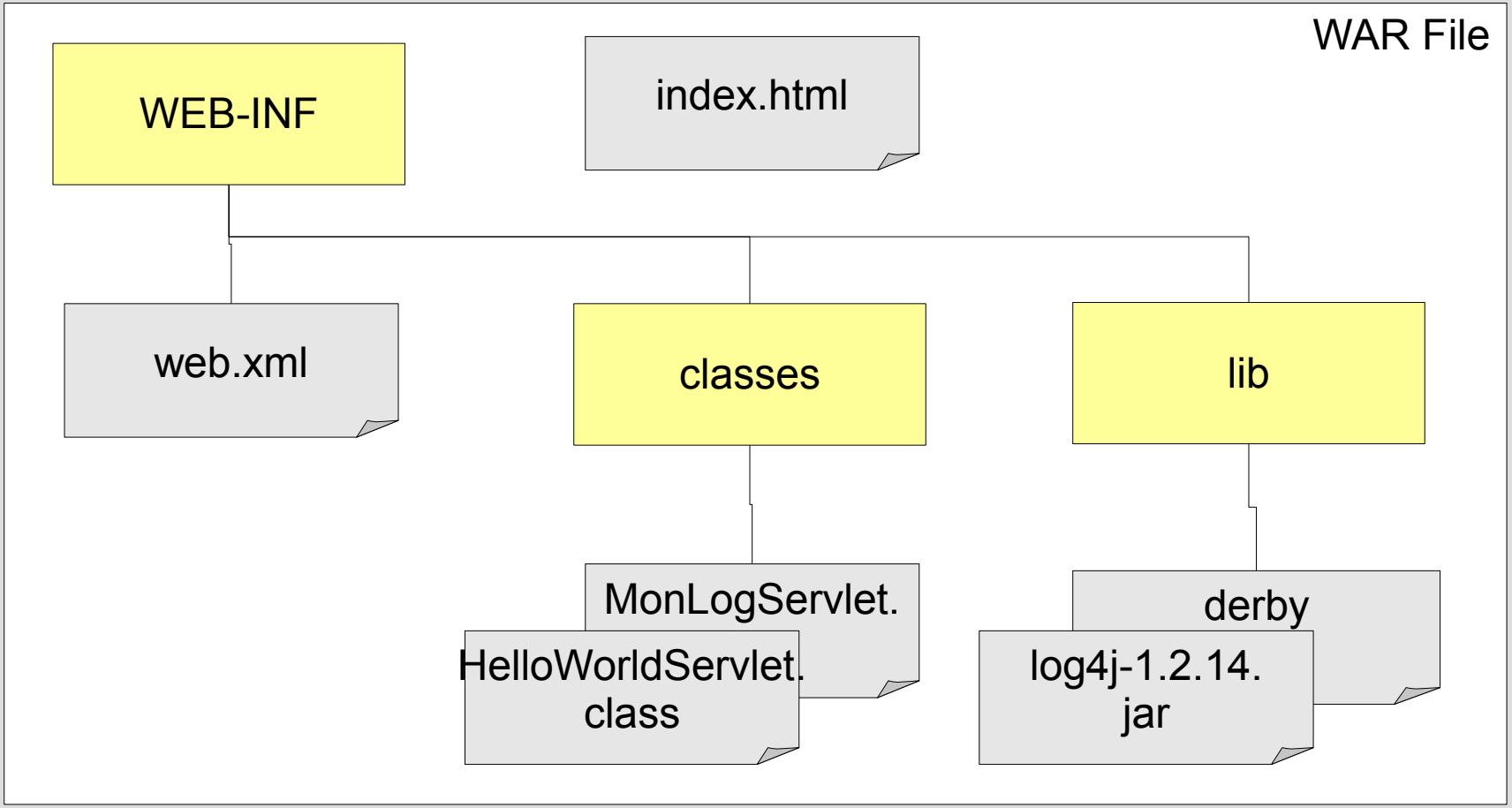
- Initialisierung
  - `init(ServletConfig serletConfig);`
- Zur „Laufzeit“ des Servlet genutzte Methoden
  - `doPost()`
  - `doGet()`
- Aufräumarbeiten
  - `destroy();`

14.3 Bereitstellung

- Einspielen einer Web-Applikation
  => „Deployment“
- Besteht aus: Servlets,
  HTML-, JSP-Dokumente,
  Hilfsklassen, Grafiken,
  CSS, JavaScript, ... Dateien
- Web-Applikation => 1 Datei => WAR

14.3 Bereitstellung

- WAR-Archiv Aufbau



14.3 Bereitstellung

- Aufbau des Deployment Descriptors „web.xml“

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app version="2.5"
      xmlns="http://java.sun.com/xml/ns/javaee"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
      http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">
    <servlet>..</servlet>
    <servlet-mapping> ... </servlet-mapping>
    <session-config>
        <session-timeout>
            30
        </session-timeout>
    </session-config>
    <welcome-file-list>
        <welcome-file>index.html</welcome-file>
    </welcome-file-list>
</web-app>
```

14.4 Informationsbeschaffung

- Methoden des Servlet API HttpServletRequest

Methode	Beschreibung
GetServerName()	Name des Server-Rechner
GetServerPort()	Port des Web-Servers
GetRemoteAddr()	IP-Adresse des Client
GetRemoteHost()	Hostname des Clients
GetQueryString()	URL-kodierte Query-String
GetMethod()	HTTP-Methode
GetServletPath	Pfadanteil der Servlet-URLs

14.4 Informationsbeschaffung

- HttpServletRequest - Formularzugriffe

Methode	Beschreibung
getParameter(String name)	liefert Wert für Formularfeld „name“
getParameterValues(String name)	liefert Werte für Formularfelder „name“

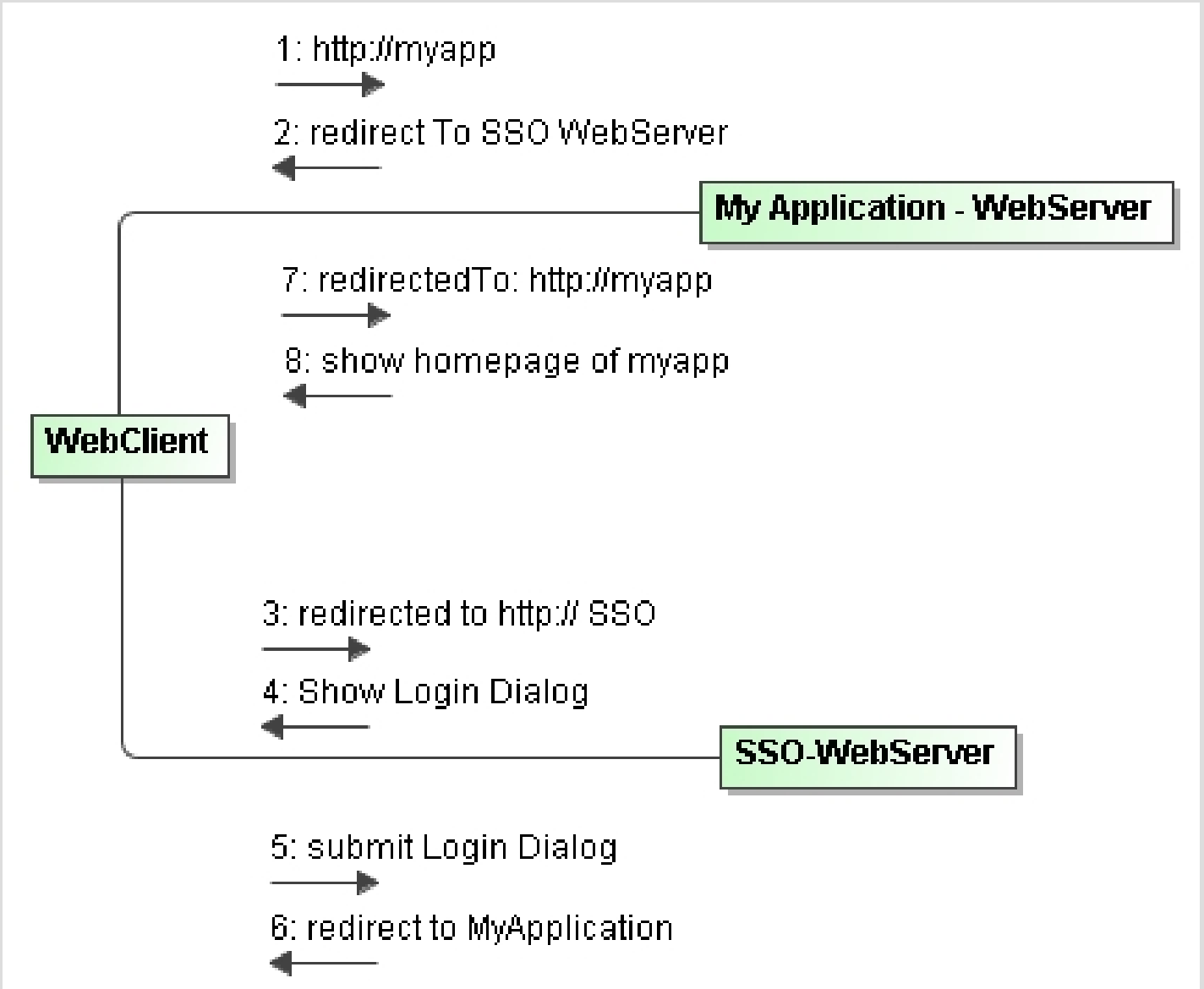
- HTTP-Header

Methode	Beschreibung
getHeaderNames()	Liste aller Header-Namen
getHeader(String name)	Header Wert als String

14.5 Response-Generierung

- **response.setContentType(String)**
- **PrintWriter pw = response.getPrintWriter()**
- HTTP-Status => WebServer (default) oder
- **response.setStatus(int status)**
  `status={SC_OK; SC_NOT_FOUND; SC_INTERNAL_SERVER_ERROR}`
- **response.setHeader(String name, String value)**
- **response.sendRedirect()**

14.5 Single Sign On (SSO)



14.6 Web-Sessions

- Benutzerinformationen über einen Request hinaus speichern (z.B.: Einkaufswagen)
- Cookies sind möglich
- Session Tracking API
  `HttpSession`

Methode	**Beschreibung**
setAttribute(String name, Object value)	Sessiondaten speichern
getAttribute(String name)	Sessiondaten lesen
getAttributeNames()	Namen aller Sessiondaten erfragen
removeAttribute(String name)	Sessiondaten entfernen

14.6 Web-Sessions

- **`HttpSession`**

Methode	Beschreibung
isNew()	true, wenn getSession() neue Session angelegt hat
invalidate()	Session wird manuel beendet
getCreationDate()	Zeitpunkt des Session-Beginns
getLastAccessedTime()	Zeitpunkt der letzten Client-Anfrage
getId()	liefert Session Id

14.7 Java Server Pages

- Trennung von Präsentation und Inhalt
- deklarative API für Web-Designer
- Reduzierung von Java auf die dynamischen Teile eines Dokumentes
- Erstellung statischer Bestandteile mit gewöhnlichen HTML-Tools
- vollständiger Zugriff aufs Java API

14.8 JSP-Konstrukte

- Scriptlets
- JSP-Ausdrücke
- Deklarationsanweisungen
- Interpreteranweisungen

14.8.1 Scriptlets

- Kodierung beliebiger Java Anweisung in JSP-Dokument

```
<% Scriptlet Code %>
```

```
...
<% if(username.startsWith(„G“)){%>
   <h1>Aktueller Benutzer G</h1>
<%} else { %>
   <h1>Aktueller Benutzer not G</h1>
<% }%>
...
```

14.8.1 Scriptlets

- Vordefinierte Objekte eines JSP-Dokumentes

Objekt	**Beschreibung**
request	HttpServletRequest – Objekt
response	HttpServletResponse – Objekt
out	PrintWriter – Objekt
session	HttpSession – Objekt
application	ServletContext – Objekt
config	ServletConfig – Objekt

14.8.2 JSP-Ausdrücke

- Bewertung eines Java Ausdrucks zur Laufzeit

```
<%= Java Ausdruck %>
```

entspricht

```
<% out.println(Java Ausdruck) %>
```

```
<h1>
<%= request.getParameter(„name“)%>
</h1>
```

14.8.3 Deklarationsanweisungen

- Definition zusätzlicher Attribute und Methoden
  für die resultierende Servletklasse

```
                                    <%! Java Code %>

<%!
    private String strasse;
```

```
    private boolean checkAddress(request)
    {
        ............
    }
%>
```

14.8.4 Interpreter-Anweisungen

- Konfiguration des resultierenden Servlets

```
<%@ page attributename="attributevalue" %>
  <%@ page attribute1="attributeV1"
           attribute2="attributeV2" %>
```

- Import von Java-Packages und Einbinden externer Dateien

```
<%@ includefile="relativer_url" %>
<%@ taglib uri="taglibrary.tld"
        prefix="pre" %>
```

14.8.4 Interpreter-Anweisungen

- Servlet Configuration

Attribut	Bedeutung
import	import von Java Klassen in den aktuellen Namensraum
errorPage	Registrierung einer anderen JSP als ErrorHandler
isErrorPage	Konfiguration einer JSP als ErrorHandler. Zugriff auf Fehlerinformation über „exception“ Objekt
contentType	MIME-Type des Dokumentes
session	Festlegung, ob JSP automatisch an einer Session geteiligt ist. Default = „true“
buffer	Definition der Puffergröße des Printwriters in KB oder „none“ für Ausschalten
autoflush	Festlegung, ob obiger Puffer automatisch gefluscht werden soll, falls er voll ist (default)

14.9 JavaBeans und JSPs

- JavaBeans = Komponentenmodell von Java
- JavaBean ist eine Java-Klasse, wenn
  - getter/setter Methoden für Attribut-Zugriffe zur Verfügung stellt.
- JavaBean => Datencontainer für Darstellung

14.9.1 JSP-Aktionen

- Zugriff auf JavaBeans von JSP
  - durch deklarative Aktionen (JSP-Tags)
  - Objekt erzeuge

```
<jsp:useBean id="bezeichner" value="classname"/>
```

  - Property lesen

```
<jsp:getProperty name="strasse" value="classname"/>
```

  - Property setzen

```
<jsp:setProperty name="bezeichner"
  property="propertyName" value="classname"/>
```