VEEBIRAKENDUSTE LOOMINE

MTAT.03.230 (6 EAP)

6. Loeng

Helle Hein

Teema: JSP - JavaServer Pages
1. osa

Täna loengus:

- Mis on JSP?
- Miks meil on vaja JSP?
- Kuidas JSP töötab?
- JSP eelised.
- JSP põhielemendid.
- JSP lehe elutsükkel.
- Eeldefineeritud muutujad.

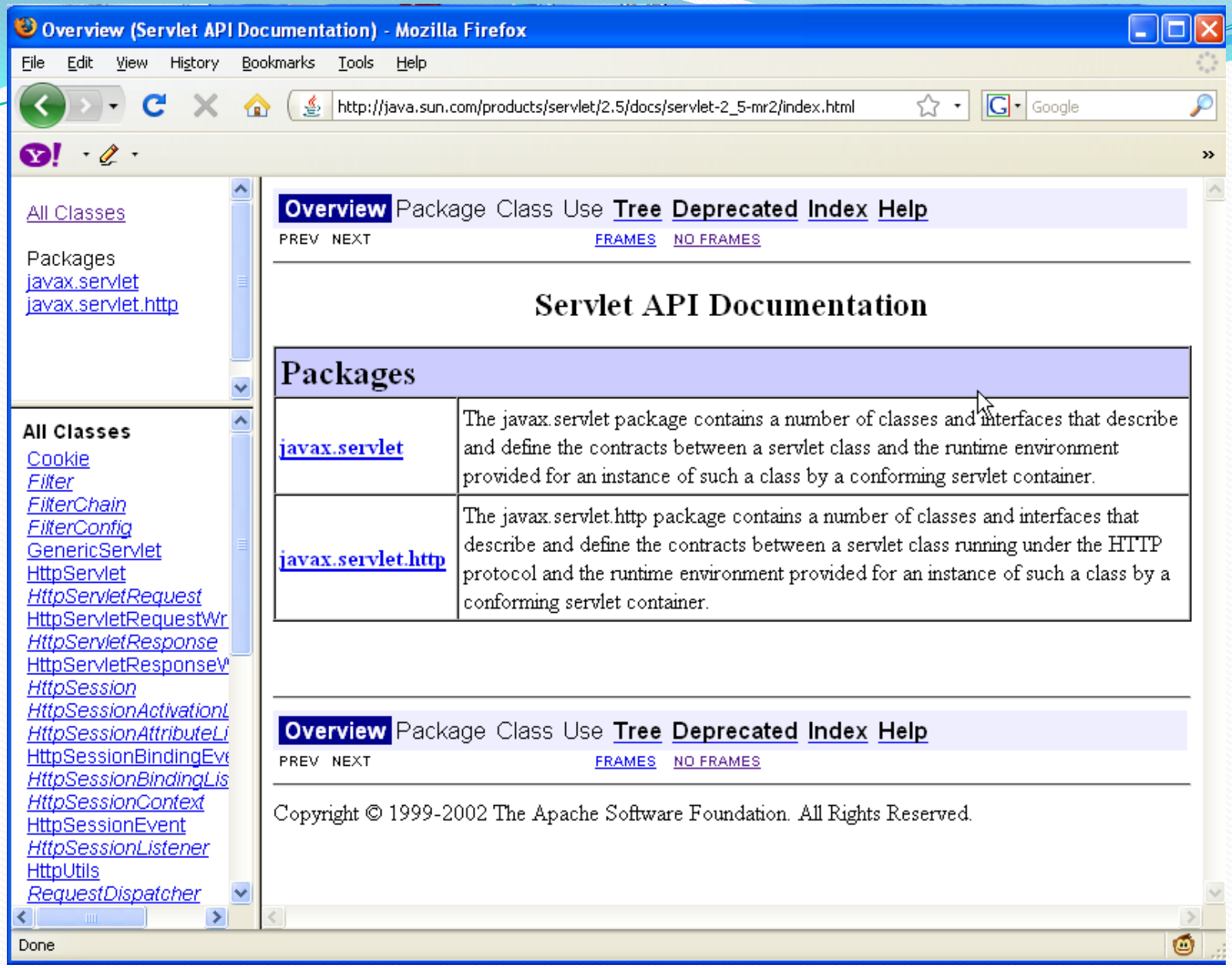
Slaidide koostamisel on kasutatud Marty Halli loal slaide aadressilt

http://www.coreservlets.com/

Versioonid:

- JSP 1.0 - juuni1999
- JSP 1.1 – detsember 1999
- JSP 1.2 - 2001
- JSP 2.0 - november 2003

Lisandusid EL (Expression Language) ja JSTL (JavaServer Pages Standard Tag Library)



http://java.sun.com/products/servlet/2.5/docs/servlet-2_5-mr2/index.html

Miks on vaja JSP?

- Servlettides on mugav:
  - Lugeda vormi (*form)* andmeid.
  - Lugeda HTTP päringu päist.
  - Seada HTTP seisundikoode ja vastuse päist.
  - Vahetada andmeid servlettide vahel.
  - Hallata seanssi.
  - ...
- Kuid - tülikas:
  - Kasutada `println` lauseid HTML genereerimiseks.
  - Lugeda ja säilitada HTML.

Idee:

- Kasutada tavalist HTML enamuse lehtede jaoks.
- Tähistada servletis olev kood eraldi märgistega.
- Kogu JSP leht transleerida servletiks (üks kord) ja tegelikult kasutada servletti (igal päringul).

Näide - kuupäeva servlett

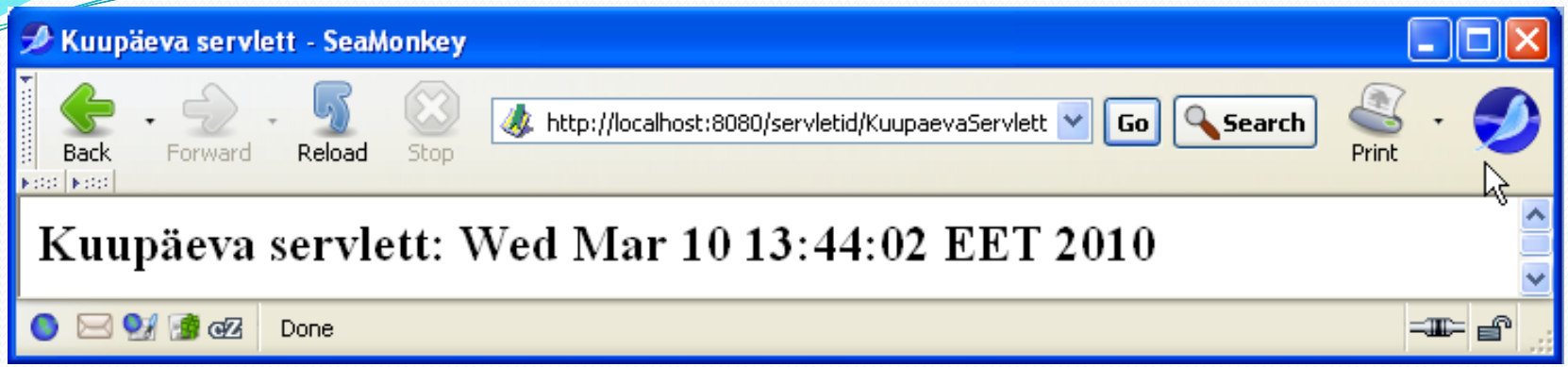
```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.util.*;
public class KuupaevaServlett extends HttpServlet {
 public void doGet( HttpServletRequest request,
  HttpServletResponse response )throws ServletException,
  IOException {
  response.setContentType( "text/html" );
  PrintWriter out = response.getWriter();
  out.println("<html>");
  out.println("<head><title>Kuupäeva servlett " +
   "</title> </head>");
  out.println("<body><h2>Kuupäeva servlett: " +
    (new Date()) + "</h2></body>");
  out.println("</html>");
 }
}
```

KuupaevaServlett.java

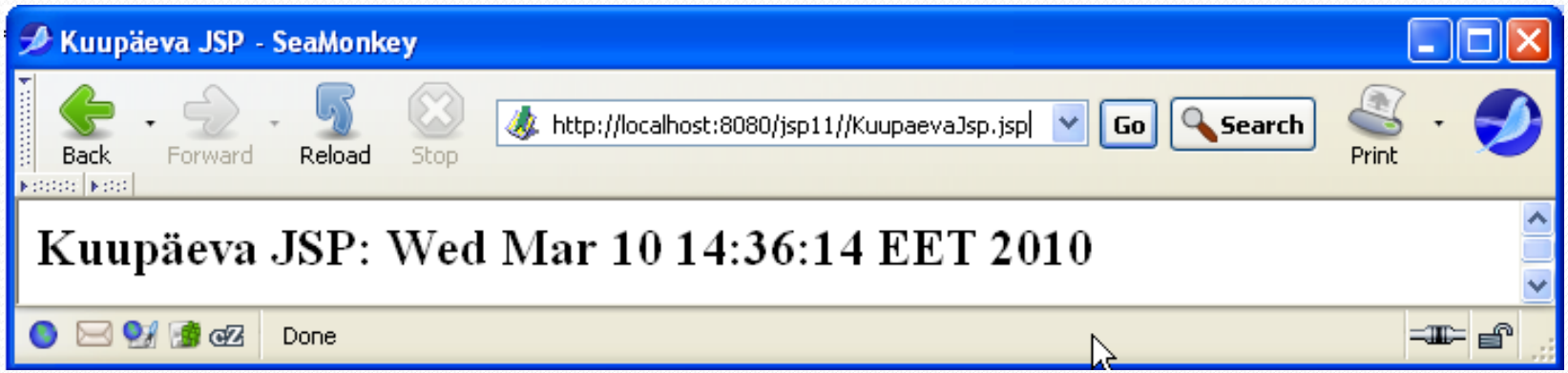
Kuupäeva JSP

```
<html>
 <head>
  <title>Kuup&auml;eva JSP</title>
 </head>
 <body>
  <h2> Kuup&auml;eva JSP: <%= new java.util.Date() %>
  </h2>
 </body>
</html>
```

KuupaevaJsp.jsp



http://localhost:8080/*servletid*/KuupaevaServlett



http://localhost:8080/*jsp11*/KuupaevaJsp.jsp

Mis on JSP?

- JSP on tehnoloogia nimi, mis annab standardse spetsifikatsiooni Java kombineerimiseks HTML märgistega.

- JSP spetsifikatsioon defineerib JSP märgiste süntaksi ja semantika.

- JSP leht on HTML mall, kus on staatilised (passiivsed) HTML märgised ja aktiivsed JSP märgised, mis sisaldavad Java koodi.

- Igal JSP lehel on unikaalne URL.

- JSP lehe päringul täidetakse kood serveris ja see asendatakse koodi poolt genereeritud HTML -ga ja saadetakse kliendile.

Mida on vaja JSP kasutamiseks?

- Ligipääsu JSP konteinerile
  - IBM's WebSphere Application Server
  - Caucho's Resin Server
  - Adobe's JRun Web Server
  - Orion Application Server
  - Borland AppServer
  - Apache Tomcat Server
  - ...
- Java programmeerimisoskust.
- Meil on Apache Tomcat 6.0.xx

NB!

JSP programmeerijad peavad ikkagi oskama servlette programmeerida!

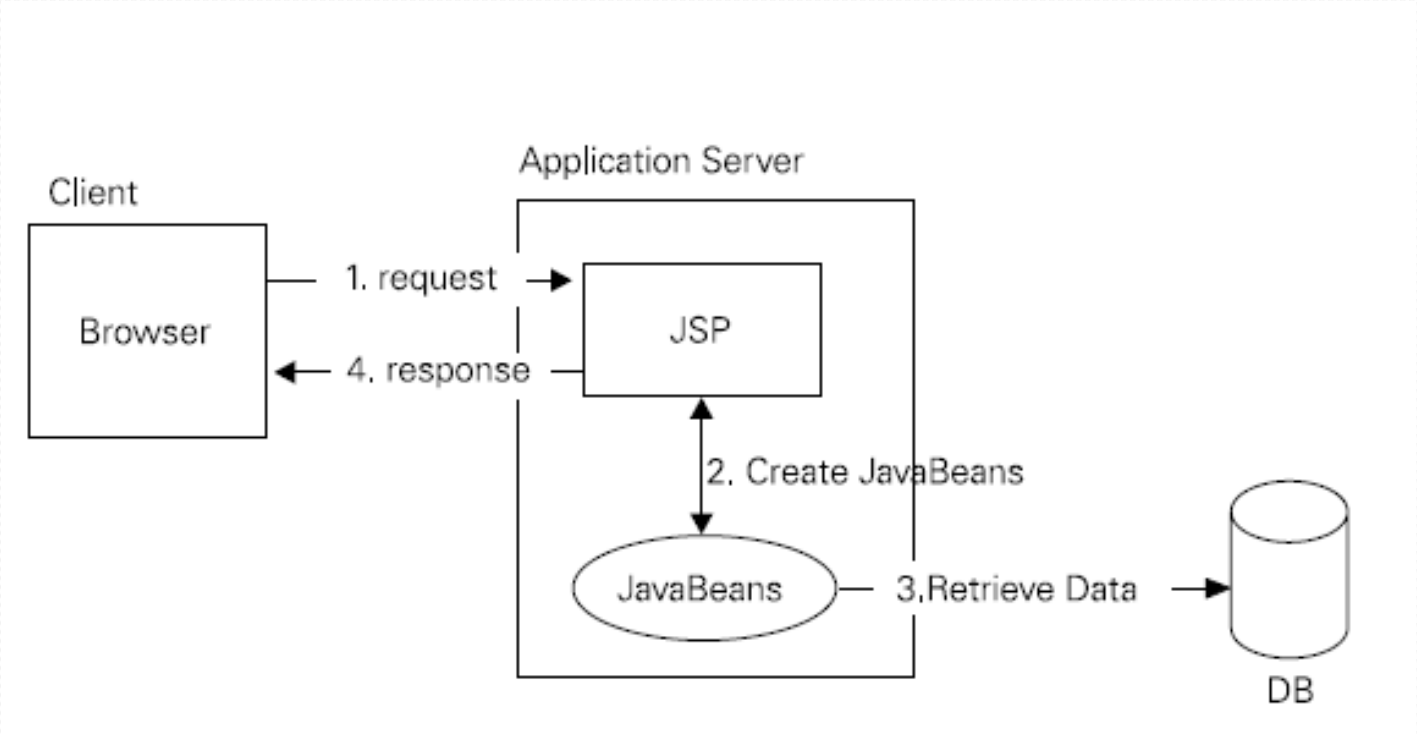
JSP eelised

Kuigi JSP ei tee tehniliselt midagi rohkem kui servletid suudavad teha, JSP lihtsustab:

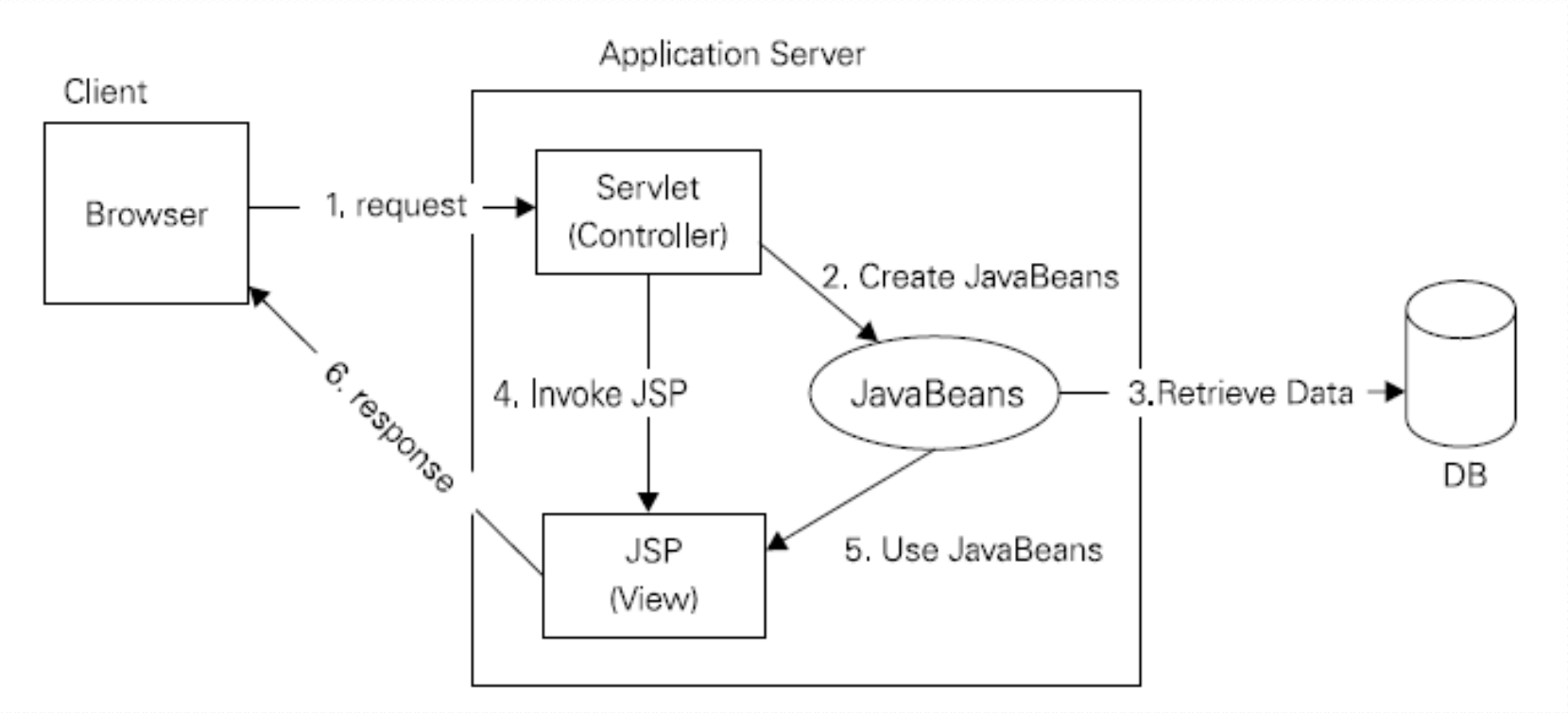
- HTML kirjutamist;
- HTML lugemist ja säilitamist.

JSP võimaldab:

- Meeskonnas eraldada HTML kujundus Java programmeerimisest.
- Eraldada Java kood (sisu ja loogika) koodist, mis seda esitab (HTML).
- Kasutada standard HTML genereerimise vahendeid nagu nt Macromedia DreamWeaver, …



JSP Mudel 1 arhitektuur



JSP Mudel 2 arhitektuur

Keskkonna seadistamine

- Pole vaja seada CLASSPATH.
- Pole vaja koodi kompileerida.
- JSP failid ei pea olema spetsiaalses kaustas.

JSP põhielemendid

- HTML tekst

```
<h1>Tere</h1>
```

  Teisendatakse servleti koodiks

```
out.write("<h1>Tere</h1>");
```

- HTML kommentaarid

```
<!-- kommentaar -->
```

  Saadetakse kliendile

- JSP kommentaarid

```
<%-- kommentaar --%>
```

  Ei saadeta kliendile

JSP põhielemendid

- Avaldised (*expressions*)

```
<%= avaldis %>
```

- Koodielemendid (*scriptlets*)

```
<% kood %>
```

- Kirjeldused (*declarations*)

```
<%! kood %>
```

JSP avaldised

`<%= Java avaldis %>`

- Tulemus
  - Avaldis väärtustatakse ja lisatakse servleti meetodis

    `_jspService()` käsku **`out.print()`**

- Näiteks
  - Hetkeaeg: `<%= new java.util.Date() %>`
  - Kliendi hosti nimi: `<%= request.getRemoteHost() %>`

JSP/Servlett vastavus

JSP fragment

```
<h1>Juhuslik arv</h1>
<%= Math.random() %>
```

Tulemuseks võimalik servleti kood (ainult meetod `_jspService()`)

```
public void _jspService(HttpServletRequest request,
                        HttpServletResponse response)
    throws ServletException, IOException {
  JspWriter out = response.getWriter();
  ...
  out.write("<h1>Juhuslik arv</h1>");
  out.print(Math.random());
  ...
}
```

JSP koodielemendid (*scriptlets*)

`<% Java kood %>`

- Tulemuseks:
  - Kood lisatakse samal kujul servleti meetodisse **`_jspService()`**
- Näiteks

```
<% String queryData = request.getQueryString();

  out.println("Lisatud päringusõne: " + queryData); %>

<% response.setContentType("text/plain"); %>
```

Avaldis ja koodielement

- JSP

```
<%= tee() %>
<% bar(); %>
```

- Tulemuseks võimalik servleti kood

```
public void _jspService(HttpServletRequest request,
  HttpServletResponse response) throws ServletException,
  IOException {
  JspWriter out = response.getWriter();
  ...
  out.print(tee());
  bar();
    ...
 }
```

Näide

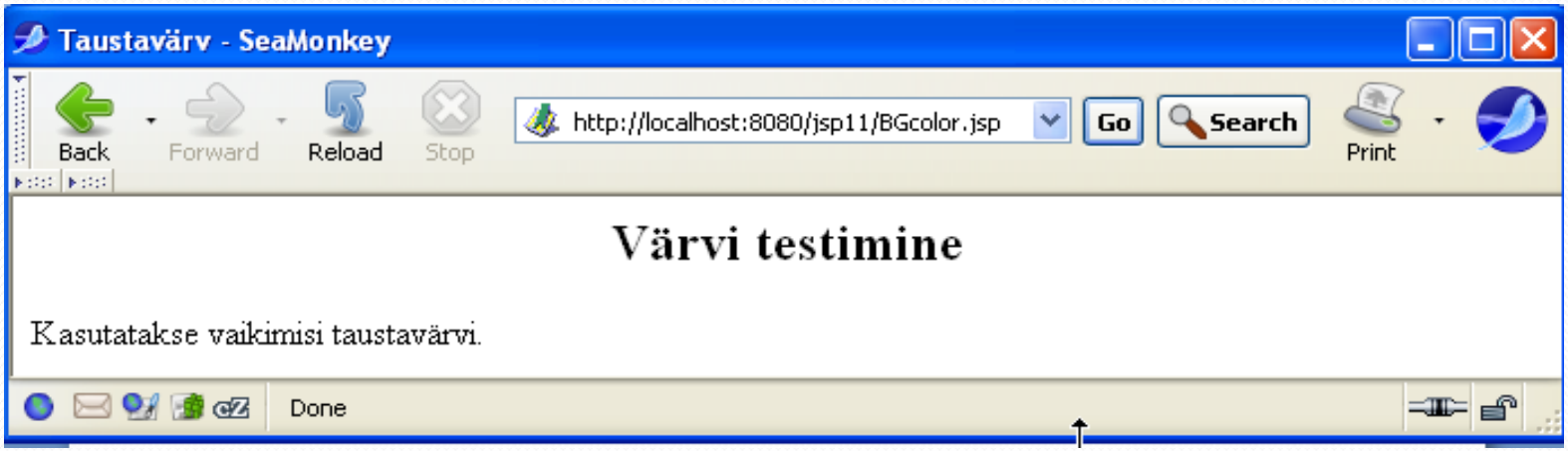
```
…
<html>
<head>  <title>Taustavärv</title> </head>
<% String bgColor = request.getParameter("bgColor");
   boolean hasExplicitColor;
   if (bgColor != null) {
     hasExplicitColor = true;
   } else {
     hasExplicitColor = false;
     bgColor = "WHITE";
   }
%>
```

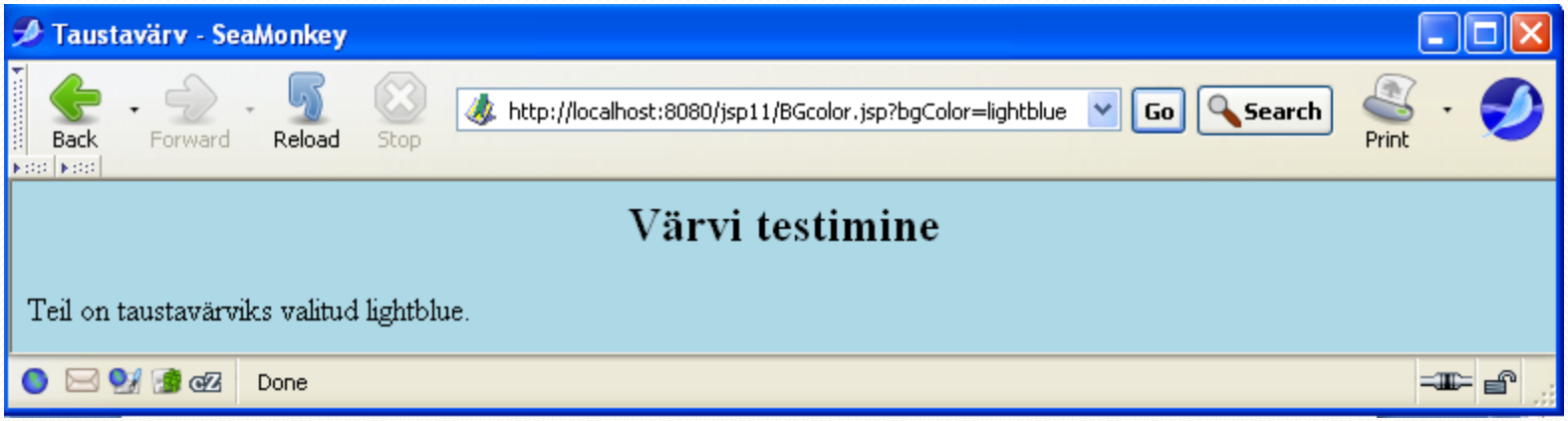
Näide jätkub

```
<body bgcolor="<%= bgColor %>">
 <h2 align="center">Värvi testimine</h2>
  <%
  if (hasExplicitColor) {
    out.println("Teil on taustavärviks valitud " +
    bgColor + ".");
  }
  else {
    out.println("Kasutatakse vaikimisi taustavärvi.");
  }
  %>
</body>
</html>
```

BGcolor.jsp

Tulemus


Taustavärv - SeaMonkey
Back
Forward
Reload
Stop
http://localhost:8080/jsp11/BGcolor.jsp
Go
Search
Print
Värvi testimine
Kasutatakse vaikimisi taustavärvi.
Done


Taustavärv - SeaMonkey
Back
Forward
Reload
Stop
http://localhost:8080/jsp11/BGcolor.jsp?bgColor=lightblue
Go
Search
Print
Värvi testimine
Teil on taustavärviks valitud lightblue.
Done

JSP leht servletina

Tomcat 6.0.xx korral paigutatakse JSP lehele vastav servlett kausta:

Tomcat home**\work\Catalina**\server\rakendus**\org\apache\jsp**

…

```
out.write("<html>\r\n");
out.write("<head>\r\n");
out.write("  <title>Taustavärv</title>\r\n");
out.write("</head>\r\n");
String bgColor = request.getParameter("bgColor");
boolean hasExplicitColor;
if (bgColor != null) {
  hasExplicitColor = true;
}
else {
  hasExplicitColor = false;
  bgColor = "WHITE";
}
```

….

BGcolor_jsp.java

JSP elutsükkel

1

Tom kirjutab **Arvuta.jsp** faili ja paigutab selle veebirakenduse osana

Konteiner loeb **web.xml** ja ei tee selle jsp failiga midagi

Veebikonteiner

web.xml

Fail ootab kliendi päringut

Arvuta.jsp

JSP elutsükkel

2

Klient klikib linki, mis teeb päringu lehele **Arvuta.jsp**

Konteiner püüab transleerida **Arvuta.jsp** servleti Java koodiks

Selles faasis avastatakse JSP süntaksivead.

transleeri

genereeri

Veebikonteiner

Arvuta.jsp

Arvuta_jsp.java

JSP elutsükkel

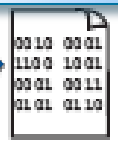
3

Konteiner püüab kompileerida **Arvuta_jsp.java** .class -failiks

Selles faasis avastatakse Java süntaksivead.

kompileeri

genereeri



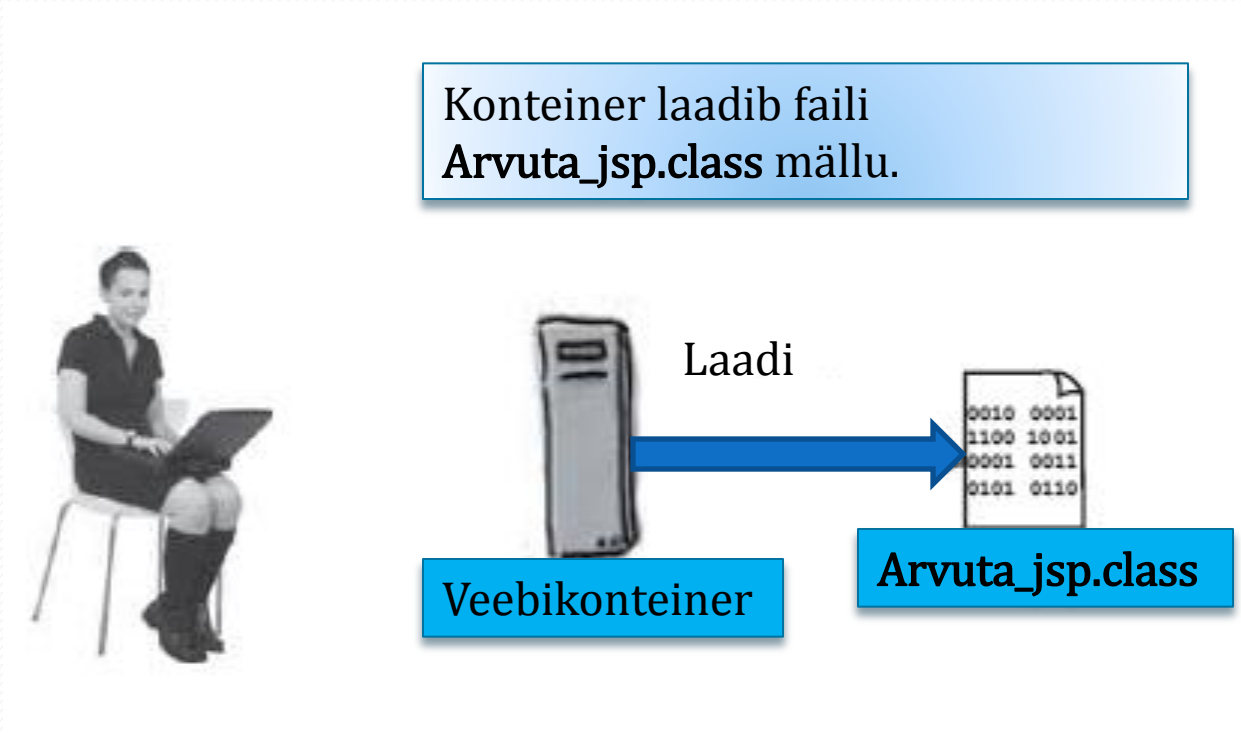
Veebikonteiner

Arvuta_jsp.java

Arvuta_jsp.class

JSP elutsükkel

4

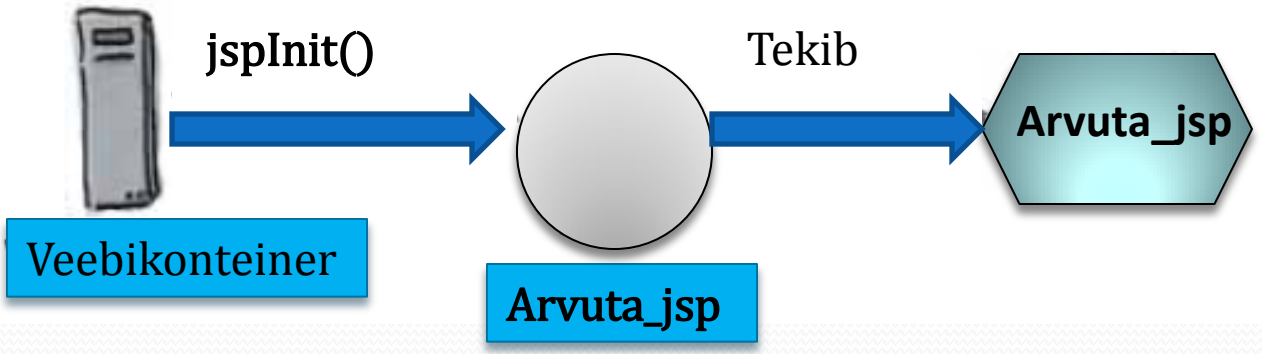


JSP elutsükkel

5

Konteiner loob servletist isendi ja pöördub servleti meetodi **jspInit()** poole

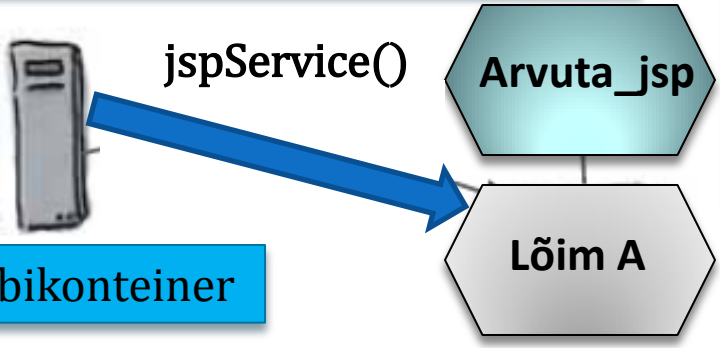
Nüüd on servlett valmis töötlema klientide päringuid



JSP elutsükkel

6

Konteiner loob kliendi päringu töötlemiseks uue lõime ja pöördub servleti meetodi **_jspService()** poole.



Siit edasi toimub tavaline servleti töötlemine. Servlett saadab vastuse või suunab päringu edasi teisele komponendile

JSP elutsükkel

	Leht valmis kirjutatud esimest korda	Päring #1	Päring #2	Serveri restart	Päring #3	Päring #4	Leht muudetud	Päring #5	Päring #6
JSP lehe transleerimine servletiks		Jah	Ei		Ei	Ei		Jah	Ei
Servleti kompileerimine		Jah	Ei		Ei	Ei		Jah	Ei
Servletist isendi loomine ja mällu lugemine		Jah	Ei		Jah	Ei		Jah	Ei
`init` (või ekv) meetodi väljakutsumine		Jah	Ei		Jah	Ei		Jah	Ei
`doGet` (või ekv) väljakutsumine		Jah	Jah		Jah	Jah		Jah	Jah

Mõned küsimused

Koodielementide kasutamine JSP lehel hargnemise juhtimiseks

Koodifragment ei pea olema tervikuna ühe märgise sees:

Näiteks

```
<% if (Math.random() < 0.5) { %>
  <b> Kaunist </b> hommikut!
<% } else { %>
  <b>Ilusat</b> hommikut!
<% } %>
```

Tulemuseks

```
if (Math.random() < 0.5) {
  out.write("<b>Kaunist</b> hommikut!");
}
else {
  out.write("<b> Ilusat </b> hommikut!");
}
```

Hargnemiste korral tuleb ka ühe käsu korral kasutada loogsulge:

Näiteks

```
<% if (userLoggedIn) %>
Welcome, <%= userName %>
```

Tulemuseks on

```
if (isUserLoggedIn)
 out.print("Welcome, ");
 out.print(userName);
```

JSP kirjeldused (*declarations*)

`<%! Java Code %>`

Java kood lisatakse samal kujul servleti klassi väljapoole olemasolevaid meetodeid

Näiteks:

```
<%! private int muutuja = 5; %>
<%! private void mingiMeetod(...) {...} %>
```

JSP kirjeldused

JSP

```
<h1>Pealkiri</h1>
<%!
  private String randomHeading() {
   return("<h2>" + Math.random() + "</h2>");
  }
%>
<%= randomHeading() %>
```

JSP/Servlett vastavus

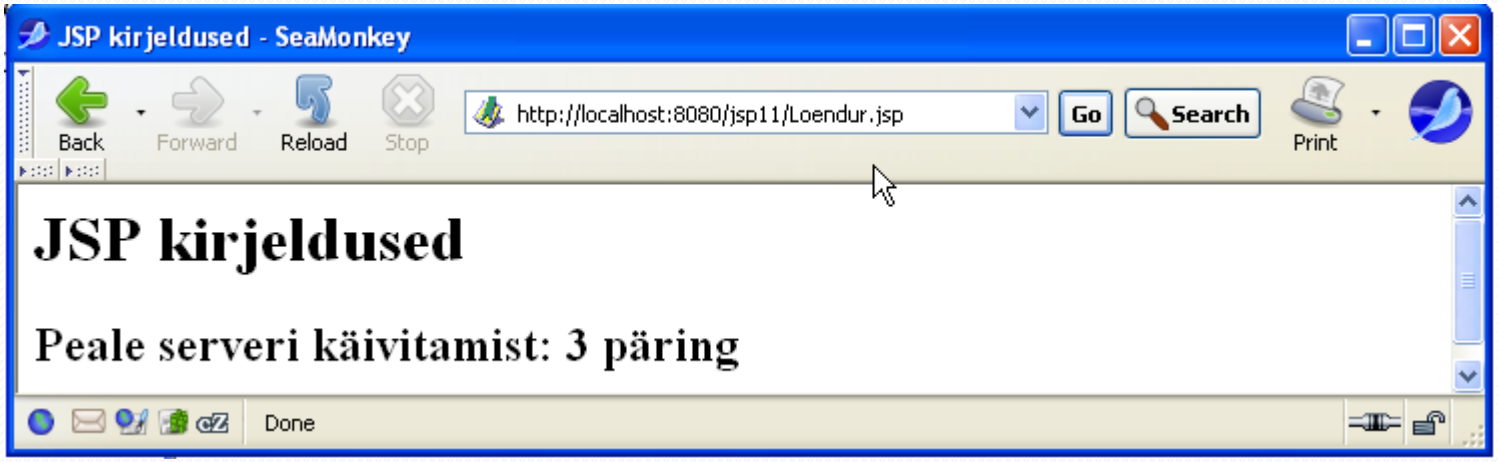
Tulemuseks võimalik servleti kood

```
public final class xxxxx_jsp extends
 org.apache.jasper.runtime.HttpJspBase
    implements org.apache.jasper.runtime.JspSourceDependent {

 private String randomHeading() {
   return("<h2>" + Math.random() + "</h2>");
 }

 public void  _jspService(HttpServletRequest request,
   HttpServletResponse response) throws ServletException,
       IOException {
   response.setContentType("text/html");
   HttpSession session = request.getSession(true);
   JspWriter out = response.getWriter();
   ...
   out.write("<h1>Pealkiri</h1>");
   out.print(randomHeading());
   ...
 } ...
}
```

Alternatiiv: teha staatiline meetod `randomHeading()` eraldi Java klassi

JSP kirjelduste kasutamine Loendur.jsp

```
<html>
  <head><title>JSP deklaratsioonid</title>
  </head>
  <body>
    <h1>JSP deklaratsioonid</h1>
    <%! private int loendur = 0; %>
    <h2>Peale serveri käivitamist:
    <%= ++loendur %> päring</h2>
  </body>
</html>
```



Eeldefineeritud muutujad

- **`request`**

  `HttpServletRequest` (esimene argument meetodile `service/doGet`)

- **`response`**

  `HttpServletResponse` (teine argument meetodile `service/doGet`)

- **`out`**

  `Writer` (puhvriga kirjutaja `JspWriter`) kasutajale väljundi saatmiseks

- **`session`**

  `HttpSession` seotakse kliendi päringuga

- **`application`**

  `ServletContext` (andmete jagamiseks) saadakse servletis meetodi `getServletContext`() abil (meetodid `setAttribute` ja `getAttribute`).

Eeldefineeritud muutujad

- **`config`**

  `ServletConfig` objekt antud lehe jaoks

- **`pageContext`**

  `PageContext` (juurdepääs lehe atribuutidele erineva skoobiga andmetele)

- **`Page`**

  `Object`

Eeldefineeritud muutujad

	In a servlet	In a JSP (using implicit objects)
Application	getServletContext().setAttribute("foo", barObj);	application.setAttribute("foo", barObj);
Request	request.setAttribute("foo", barObj);	request.setAttribute("foo", barObj);
Session	request.getSession().setAttribute("foo", barObj);	session.setAttribute("foo", barObj);
Page	Does not apply!	pageContext.setAttribute("foo", barObj);

Näiteks:

```
<% pageContext.forward("other.jsp"); %>

<%= ((Servlet)page).getServletInfo() %>

<%= this.getServletInfo() %>
```

Meetodid `jspInit` ja `jspDestroy`

- JSP lehti, nagu ka servlette, on vaja initsialiseerida ja hävitada
- Probleem: servletil juba on olemas `init` and `destroy`
  - Nende ülekirjutamine võib põhjustada probleeme.
- Lahendus: kasutada `jspInit` ja `jspDestroy`.

Initsialiseerimisparameetrite konfigureerimine

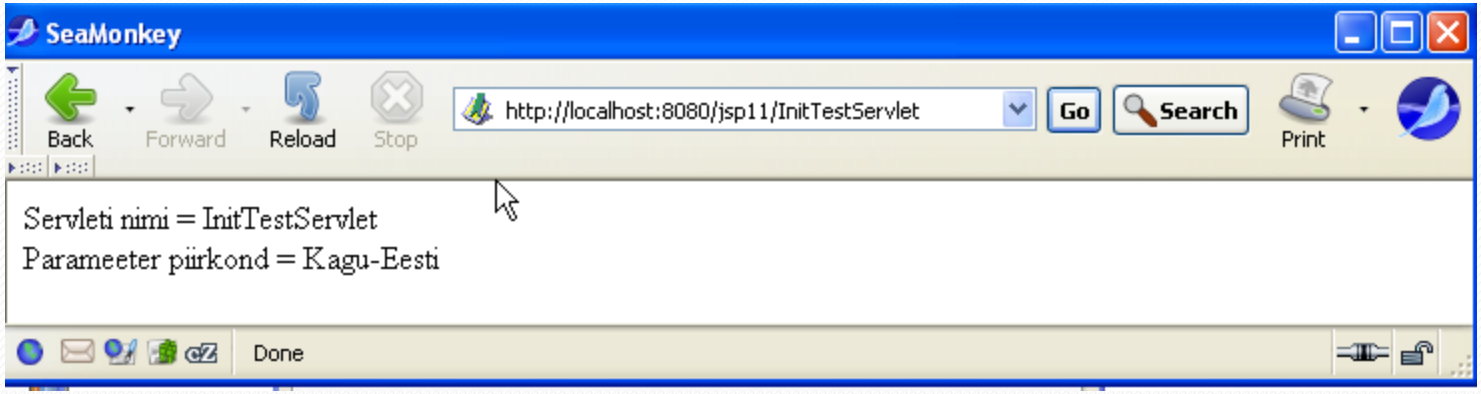
```
<web-app ...>
....
  <servlet>
    <servlet-name>MinuTestServlett</servlet-name>
      <jsp-file>/MinuServlett.jsp</jsp-file>
        <init-param>
          <param-name>email</param-name>
          <param-value>lilled@tulbiaed.com</param-value>
        </init-param>
  </servlet>
....
</web-app>
```

Initsialiseerimisparameetrite konfigureerimine

```
…
<servlet>
  <servlet-name>InitTestServlet</servlet-name>
  <jsp-file>/initTest.jsp</jsp-file>
  <init-param>
    <param-name>piirkond</param-name>
    <param-value>Kagu-Eesti</param-value>
  </init-param>
</servlet>
<servlet-mapping>
  <servlet-name>InitTestServlet</servlet-name>
  <url-pattern>/InitTestServlet</url-pattern>
</servlet-mapping>
…
```

initTest.jsp

```
<html>
 <body>
  Servleti nimi = <%=config.getServletName()%><br>
  Parameeter piirkond = <%=config.getInitParameter("piirkond")%>
 </body>
</html>
```



Meetodi `jspInit` ülekatmine

```
<%!
public void jspInit(){
  //antud servleti jaoks
  ServletConfig sConfig = getServletConfig();
  String email = aConfig.getInitParameter("email");
   //kogu rakenduse jaoks
  ServletContext ctx = getServletContext();
  ctx.setAttribute("mail", email);
 }
%>
```

JSP kirjeldused ja eeldefineeritud muutujad

- Probleem
  - Eeldefineeritud muutujad (`request, response, out, session`, ...) on meetodis `_jspService` **lokaalsed.** Nad ei ole kättesaadavad deklaratsioonides defineeritud meetodites.
- Lahendus: edastada osuti (*reference*) neile argumendina.

```
<%!
private void someMethod(HttpSession s) {
  doSomethingWith(s);
}
%>
<% someMethod(session); %>
```

JSP direktiivid

- Mõjuvad kompileerimise ajal servletile, mis tuleneb antud JSP lehest.
- Direktiividel võivad olla atribuudid.
- Direktiive on kolm: `page`, `include` ja `taglib`

```
<%@ directive attribute1="value1"
      attribute1="value1" …  %>
```

Direktiivi `<%@ page %>` atribuudid

- `import`
- `contentType`
- `session`
- `errorPage`
- `isErrorPage`
- `buffer`
- …

Atribuut `import`

```
<%@ page import="package.class1,...,package.classN" %>
```

Näiteks,

```
<%@ page import="java.util.*" %>
```

Vaikimisi loetakse sisse

```
java.lang.*,
javax.servlet.*,
javax.servlet.jsp.*,
javax.servlet.http.*
```

Atribuut `contentType`

```
<%@ page contentType="MIME-Type" %>

<%@ page contentType="MIME-Type; charset=Character-Set" %>
```

Näiteks,

```
<%@ page contentType="text/plain" %>
```

direktiivil on sama efekt, mis koodielemendil

```
<% response.setContentType("text/plain"); %>
```

Atribuut `session`

```
<%@ page session="true" %>
```

```
<%@ page session="false" %>
```

- Määrab, kas leht osaleb seanssides või mitte
- Võimaldab kokku hoida mälu
- Vaikimisi `session = true`

Atribuudid errorPage ja isErrorPage

errorPage

```
<%@ page errorPage="Relative URL" %>
```

isErrorPage

```
<%@ page isErrorPage="true" %>

<%@ page isErrorPage="false" %> <%!-- Default --%>
```

Atribuut `buffer`

```
<%@ page buffer="sizekb" %>
```

- Vaikimisi `vähemalt 8kB`

Direktiiv `include`

```
<%@ include file= "relative URL" %>
```

- Faili lisamiseks antud kohta (kompileerimise ajal)
  - Sisestatud faili käsitletakse JSP- na (võib olla HTML, XML, tekstifail)
- Kasulik korduvate osade kasutamiseks (menüü, navigeerimine, päis, jalus)

Failide lisamine päringu ajal

```
<jsp:include page="Relative URL" flush="true" />
```

JSP toimingud (*actions*)

- Mõjuvad lehe täitmise ajal.

- Toiminguteks on näiteks:

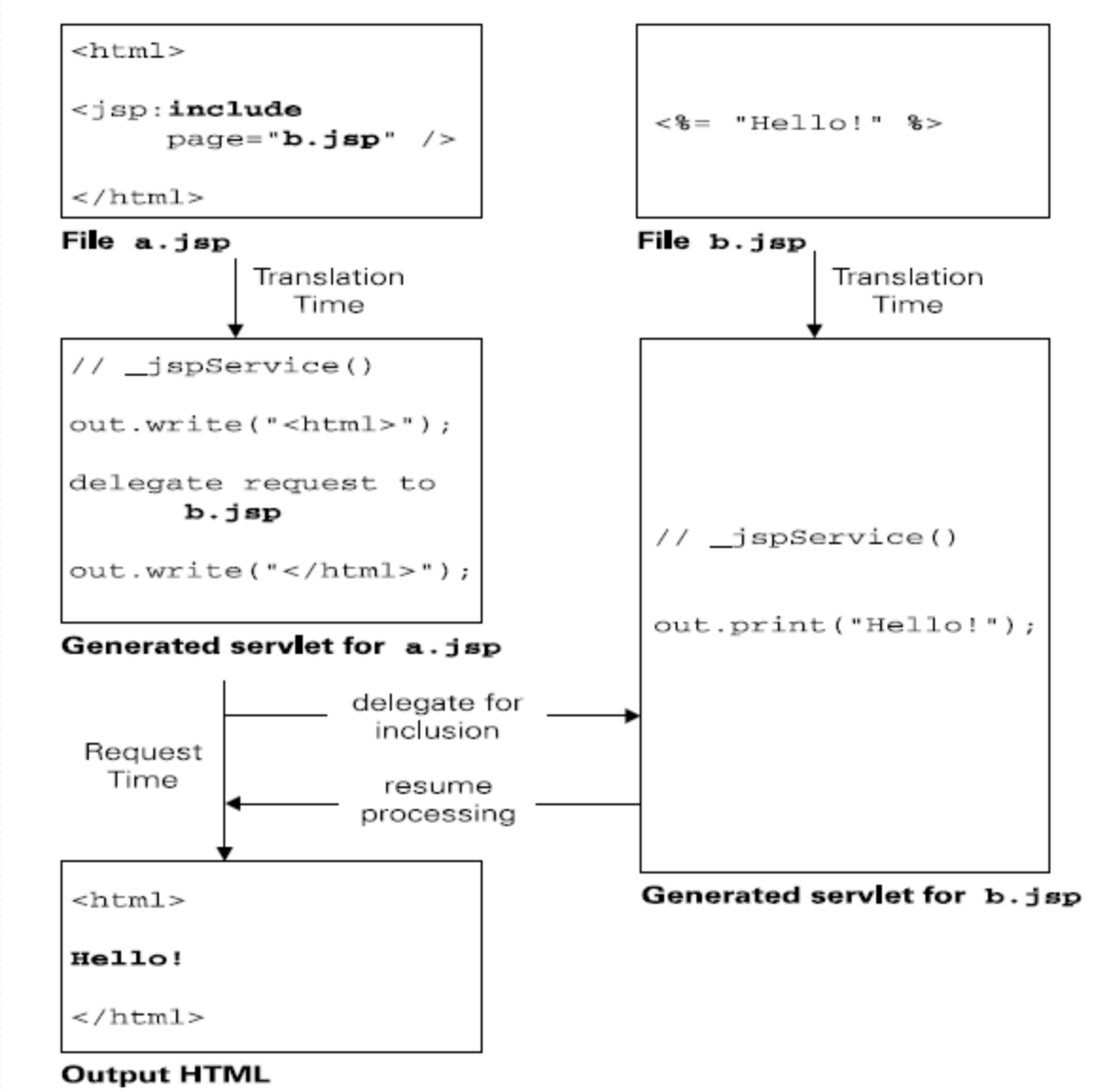
```
jsp:include
jsp:forward
jsp:useBean
jsp:setProperty
jsp:getProperty
```

Üldkuju:

```
<jsp:actionName attribute-list />
```

Näiteks:

JSP toimingud (*actions*)

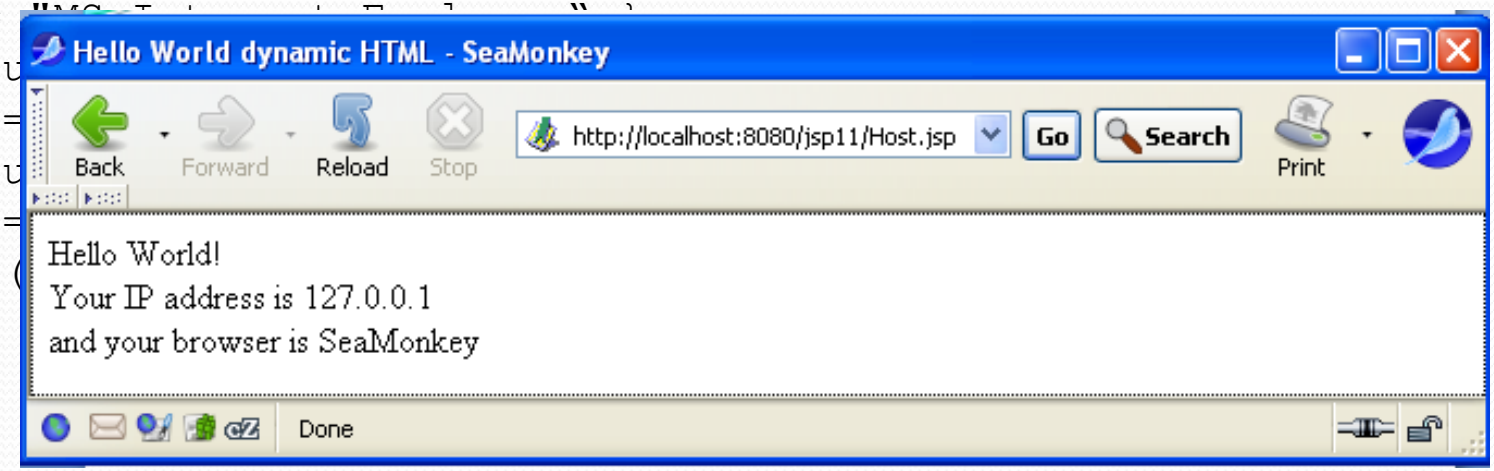


Host.jsp

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<%@page language="java" contentType="text/html"%>
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head><title>Hello World dynamic HTML</title></head>
<body>
Hello World!
<% out.println("<br/>Your IP address is " + request.getRemoteAddr());
 String userAgent = request.getHeader("user-agent");
 String browser = "unknown";
 out.print("<br/>and your browser is ");
 if (userAgent != null) {
  if (userAgent.indexOf("MSIE") > -1) {
   browser = "MS Internet Explorer";}
   else if (userAgent.indexOf("Firefox") > -1) {
    browser = "Mozilla Firefox";}
   else if (userAgent.indexOf("SeaMonkey") > -1) {
    browser = "SeaMonkey";}}
 out.println(browser);
%>
</body>
</html>
```

Host.jsp

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<%@page language="java" contentType="text/html"%>
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head><title>Hello World dynamic HTML</title></head>
<body>
Hello World!
<% out.println("<br/>Your IP address is " + request.getRemoteAddr());
 String userAgent = request.getHeader("user-agent");
 String browser = "unknown";
 out.print("<br/>and your browser is ");
 if (userAgent != null) {
  if (userAgent.indexOf("MSIE") > -1) {
   browser =
   else if (u
    browser =
   else if (u
    browser =
 out.println(
%>
</body>
</html>
```



JSP XML versioon

- JSP faile võib kirjutada nii standardse JSP süntaksi kui ka XML süntaksi abil.
- JSP faile, mis on kirjutatud XML süntaksis, nimetatakse JSP dokumentideks.
- Ühes failis ei tohi neid segamini kasutada.
- Kõikidel JSP dokumentidel on märgis `<jsp:root>`, mille sees on kõik ülejäänud elemendid.

JSP ja XML

- HTML: `<%= avaldis %>`
  XML: `<jsp:expression>avaldis</jsp:expression>`
- HTML: `<% Java kood %>`
  XML: `<jsp:scriptlet> Java kood </jsp:scriptlet>`
- HTML: `<%! kirjeldused %>`
  XML: `<jsp:declaration> kirjeldused </jsp:declaration>`
- HTML: `<%@ include file= "relatiivne URL" %>`
  XML: `<jsp:directive.include file= "relatiivne URL" />`

Direktiivide XML süntaks

Üldkuju:

```
<jsp:directive.directiveType attribute="value" />
```

Näiteks,

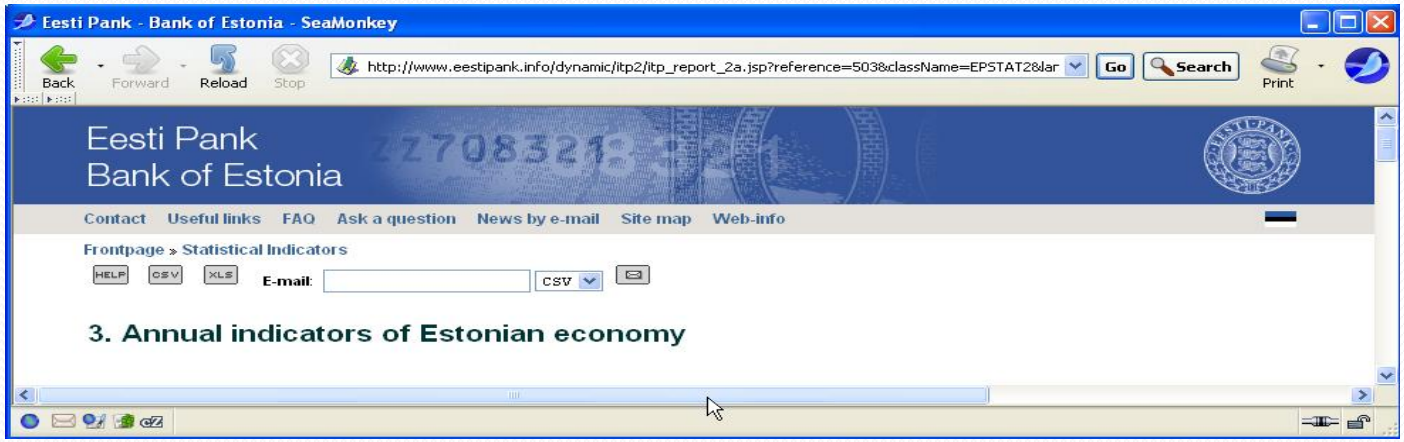
```
<%@ page import="java.util.*" %>
```

ekvivalent on

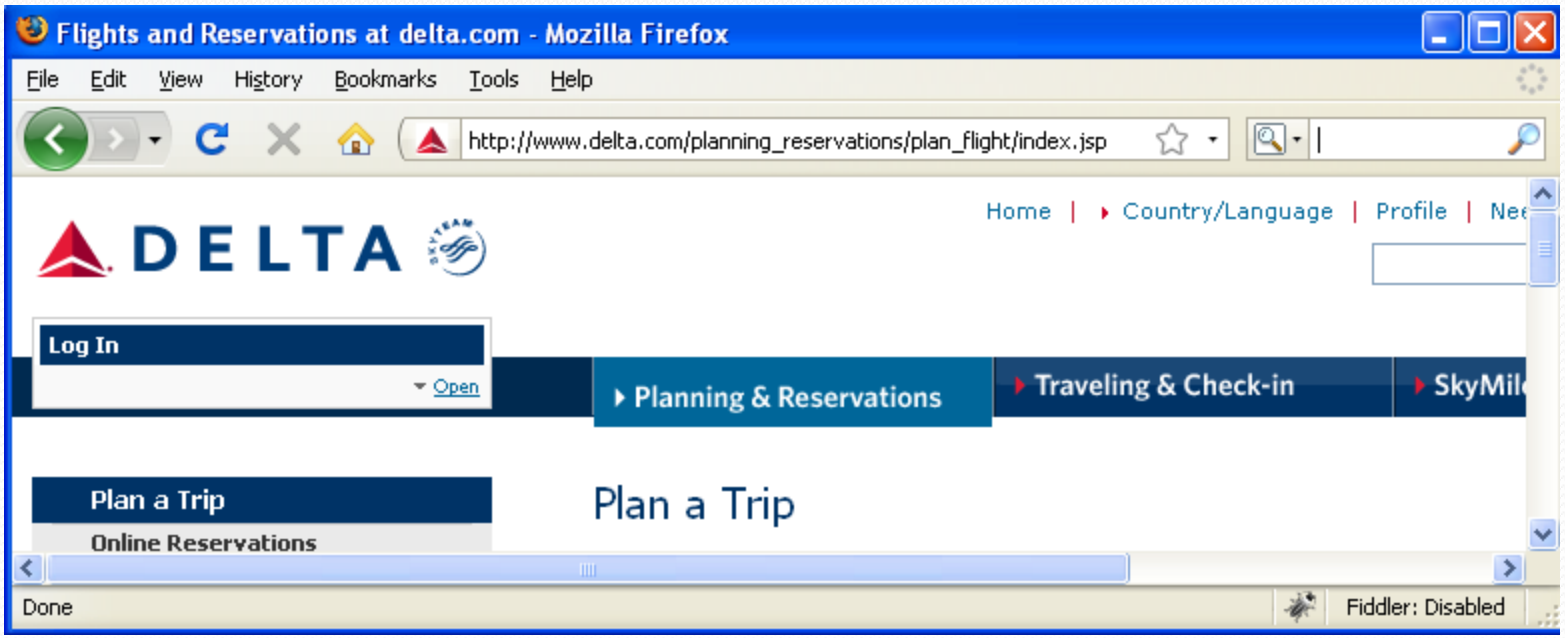
```
<jsp:directive.page import="java.util.*" />
```

```
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page"
version="1.2 | 2.0">
 <html>
    <body>
      <jsp:directive.page language="java" />
      <jsp:declaration> int count = 0; </jsp:declaration>
      <jsp:scriptlet>  count++; </jsp:scriptlet>
      <jsp:text> Welcome! You are visitor number </jsp:text>
      <jsp:expression> count </jsp:expression>
    </body>
  </html>
 </jsp:root>
```

Kes kasutab JSP tehnoloogiat?




GoArmy.com > Soldier Life > Becoming a Soldier - SeaMonkey
Back
Forward
Reload
Stop
http://www.goarmy.com/life/becoming_a_soldier.jsp?bl=Army+Reserve
Go
Search
Print
HOME
ABOUT THE ARMY
CAREERS & JOBS
BENEFITS
SOLDIER LIFE
FOR PARENTS
U.S.ARMY
ARMY STRONG®
Soldier Life
Overview
Becoming a Soldier
Working with a Recruiter
Basic Combat Training
Advanced Individual Training
Being a Soldier
SEARCH GOARMY.COM
GO
more search options
MY GOARMY.COM
Overview
Done


Flights and Reservations at delta.com - Mozilla Firefox
File
Edit
View
History
Bookmarks
Tools
Help
http://www.delta.com/planning_reservations/plan_flight/index.jsp
Home
Country/Language
Profile
DELTA
Log In
Open
Planning & Reservations
Traveling & Check-in
Plan a Trip
Online Reservations
Plan a Trip
Done
Fiddler: Disabled

Kokkuvõte

- JSP teeb lihtsamaks HTML kirjutamise ja hoidmise
- JSP lehed transleeritakse servletiks
  - Servletid käivitatakse päringu ajal
  - Klient ei näe midagi JSP-ga seotut

JSP 2.0

JSTL- JSP Standard Tag Library

```
xmlns:c=http://java.sun.com/jsp/jstl/core
xmlns:c="http://java.sun.com/jsp/jstl/xml
xmlns:c="http://java.sun.com/jsp/jstl/fmt
xmlns:c="http://java.sun.com/jsp/jstl/sql
xmlns:c="http://java.sun.com/jsp/jstl/fn
```

Näide

```
<html
  xmlns="http://www.w3.org/1999/xhtml"
  xmlns:jsp="http://java.sun.com/JSP/Page"
  xmlns:c="http://java.sun.com/jsp/jstl/core">
  <jsp:directive.page contentType="text/html" />
  <jsp:output
    omit-xml-declaration="yes"
    doctype-root-element="html"
    doctype-public="-//W3C//DTD XHTML 1.0 Strict//EN"
    doctype-
system="http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd"
/>
  <head>
    <title>
      HelloCounter.jspx
    </title>
  </head>
  <body>
```

```
<jsp:scriptlet>  /* Initialize and update the "visits" variable. */
 </jsp:scriptlet>
    <c:if test="${empty visits}">
      <c:set var="visits" scope="application" value="0"
/>
    </c:if>
    <c:set var="visits" scope="application"
value="${visits+1}" />
    <p>
       Hello World!
    </p>
    <p>
       This page has been viewed
         ${visits}
       times since the most recent
       application restart.
    </p>
  </body>
```

Avaldiste keel – Expression Language JSP 2.0

```
${avaldis}
```

EL Avaldis	Tulemus
`${1 <= (1/2)}`	`false`
`${5.0 > 3}`	`true`
`${100.0 == 100}`	`true`
`${'a' < 'b'}`	`true`
`${'fluke' gt 'flute'}`	`false`
`${1.5E2 + 1.5}`	`151.5`
`${1 div 2}`	`0.5`
`${12 mod 5}`	`2`

```
${first.second.third}
```

on ekvivalentne

```
<%=first.getSecond().getThird()%>
```

Avaldiste keele reserveeritud sõnad:

```
and  div  empty eq  false ge  gt

 instanceof le  lt  mod  ne not    null

or   true
```

Avaldiste keel – Expression Language

EL avaldis:

```
Please contact: ${applicationScope.mail}
```

Sama Java avaldisena:

```
Please contact: <%= application.getAttribute("mail")%>
```

Mõned JSTL tegevused `core` teegist

Tegevus	Eesmärk
`set`	muutujale väärtuse omistamine, muutuja loomine;
`remove`	muutuja kustutamine;
`out`	andmete kirjutamine ilmutamata objekti out, escaping XML special characters;
`url`	URL loomine koos päringusõnega;
`if`	tingimuslause (if-then);
`choose`	tingimus (valiku)lause (if-then-elseif);
`forEach`	tsükkel üle kollektsiooni elementide.