Slide 1

Web ontwikkeling 2

Les 6: No Scriptlets

Slide 2

AGENDA

- Expression language
- JSP actions

Slide 3

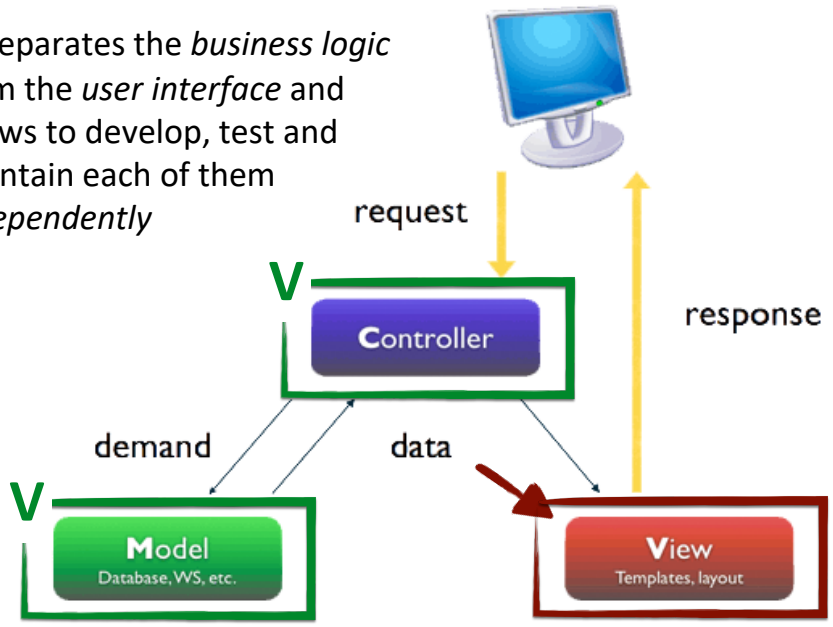
EXAM QUESTIONS...

- What is the advantage of using expression language instead of JSP scriptlets and JSP expressions
- What is the difference between the . operator and the [ ]
- What does JSTL stand for?
- Name the 6 language constructs of JSP
- Is JSTL used in the View, Model or Controller?
- When is a Java class a JavaBean?
- …

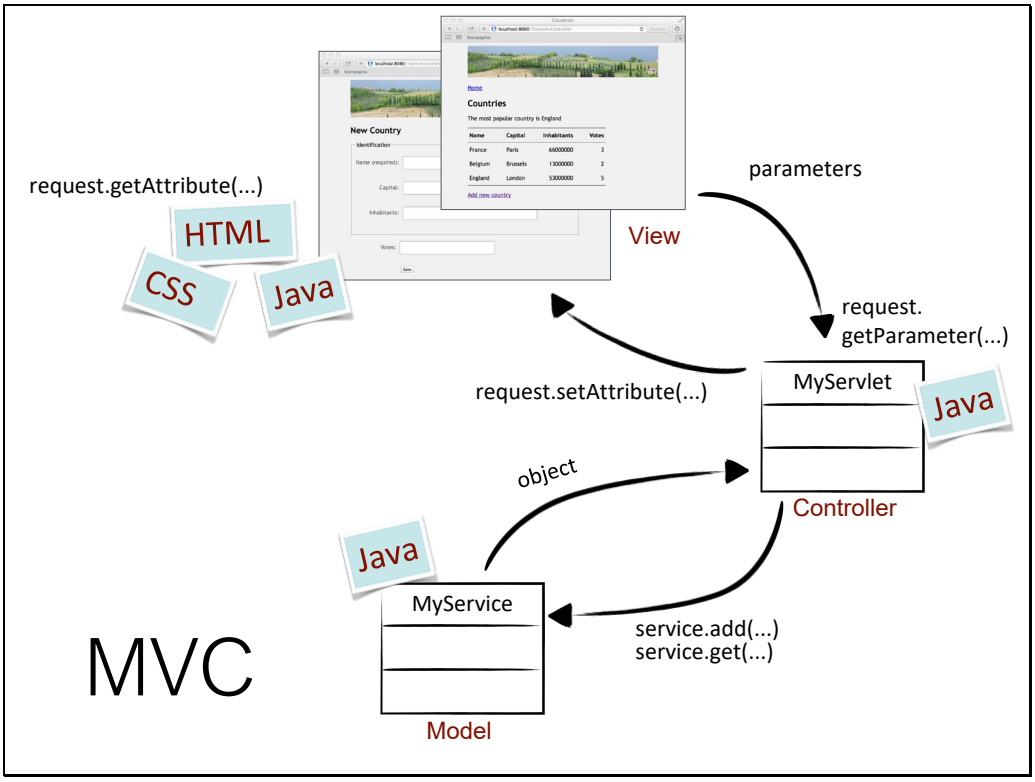
Slide 4

MVC

... separates the *business logic* from the *user interface* and allows to develop, test and maintain each of them *independently*



Slide 5



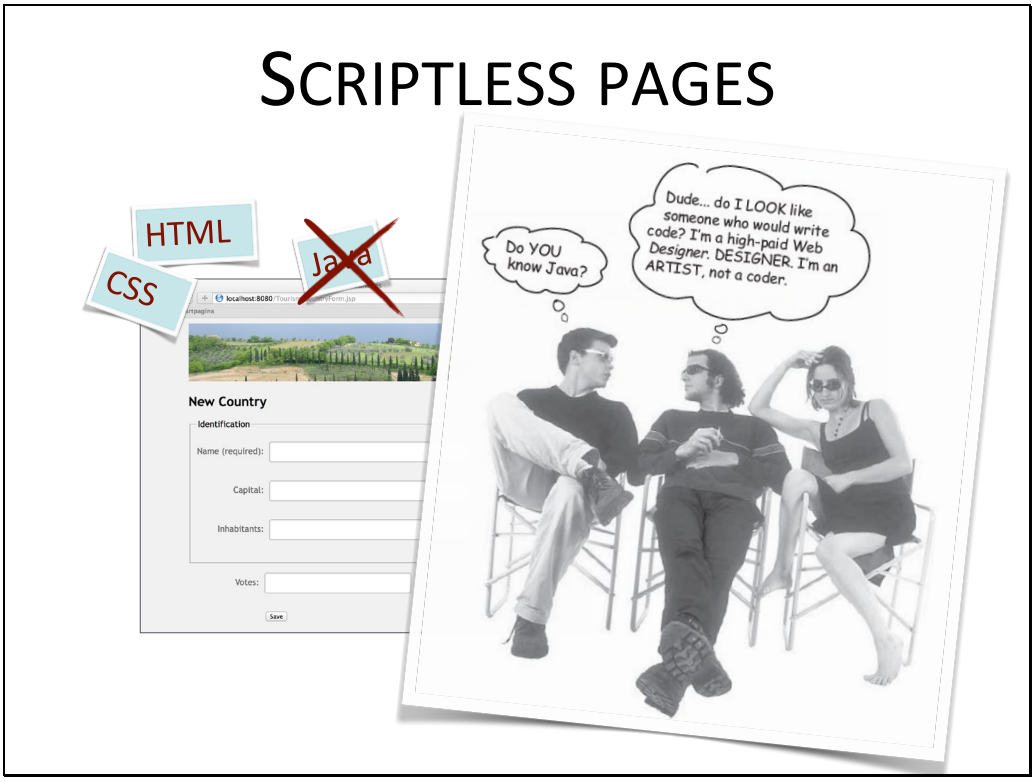
Recap workflow:

- parameters can be passed with HTTP request
- servlet reads them with request.getParameter()
- model is called to add or get an object
- model can return object(s)
- objects can be passed to view with request.setAttribute()
- objects can be read with request.getAttribute()

Technologies used:

- controller: Java
- model: Java
- view: Java, css, html

Slide 6



Problem:

- designers should be able to create view
- good designers or not necessarily good programmers (and vice versa)

Solution: avoid Java in view

Slide 7

JSP 2.0: EXPRESSION LANGUAGE

```
<td>
  <%= ((Dier) request.getAttribute("dier")).getNaam()%>
</td>
```

no cast!

```
<td>
   ${dier.naam}
</td>
```

property

JSP knows:
this must be an attribute

PUBLIC ???

JSP scriplets and expressions are replaced by expression language
Easier:

- JSP assumes you are asking for an attribute —> no request.getAttribute()
- JSP can find out what type you're talking about —> no cast
- JSP expects a property after the dot —> no get-method

Does this mean we should make our properties public?

Slide 8

JAVABEAN

```
public class Dier {
    private String naam;

    public Dier() {
    }

    public String getNaam() {
        return naam;
    }

    public void setNaam(String naam) {
        this.naam = naam;
    }
}
```

public no-arg constructor

getter and setter
define a property

No, as long as we use the JavaBean conventions:

- getters an setters define a property (public getter to be able to read the property, public setter to modify it)
- a default constructor should be present (no arguments)

Slide 9

. OPERATOR

- Syntax:

```
${attribute.property}
${attribute.property.propertyOfProperty}
```

- Only for **Javabean** or **Map**

Slide 10

EXAMPLES . OPERATOR

```
private Country country = new Country("Belgium", "Brussels", 13000000, 5); ...
```

${country.votes}

```
private Address address = new Address("Herestraat", 49, "Leuven";
private Person customer = new Person("Bert", address; ...
```

${customer.address.street}

```
private Map<String, Country> countries = new HashMap<String, Country>();...
countries.put(country.getName(), country);
```

${countries.Belgium.capital}

```
private List<Country> countries = new ArrayList<Country>(); countries.add(country.getName(), country);
```

→ not possible

[ ] OPERATOR

- Syntax:

```
${attribute["property"]}
${attribute[0]}
```

- For **Javabean, Map, Array, List, …**

Slide 12

EXAMPLES [ ] OPERATOR

```
private Country country = new Country("Belgium", "Brussels", 13000000, 5); ...
```

```
${country["votes"]}
```

```
private Address address = new Address("Herestraat", 49, "Leuven";
private Person customer = new Person("Bert", address; ...
```

```
${customer["address"]["street"]}
```

```
private Map<String, Country> countries = new HashMap<String, Country>();...
countries.put(country.getName(), country);
```

```
${countries["Belgium"]["capital"]}
```

```
private List<Country> countries = new ArrayList<Country>();
countries.add(country.getName(), country);
```

```
${countries[0]}
```

Slide 13

WHAT IF …

I want to show a parameter and not an attribute ?

Slide 14

READ PARAMETER

http://localhost:8080/Servlet?naam=Albert

```
<td>
   ${param.naam}
</td>
```

parameter name

JSP knows:
this is probably not an attribute
I should look for a parameter

Slide 15

WHAT IF …

I want to be sure my project does not contain scriptlets ?

Slide 16

DISABLE SCRIPTLETS

```
<?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">

    <jsp-config>
        <jsp-property-group>
            <url-pattern>*.jsp</url-pattern>
            <scripting-invalid>true</scripting-invalid>
        </jsp-property-group>
    </jsp-config>

    <welcome-file-list>
        <welcome-file>index.html</welcome-file>
    </welcome-file-list>
</web-app>
```

If you want to be sure that all script lets are replaced, you can disable them in your web.xml

Slide 17

HTTP Status 500 - /countryForm.jsp (line: 20, column: 5) Scripting elements (<%!, <jsp:declaration, <%=, <jsp:expression, <%, <jsp:scriptlet) are disallowed here.

---

type Exception report

message /countryForm.jsp (line: 20, column: 5) Scripting elements (<%!, <jsp:declaration, <%=, <jsp:expression, <%, <jsp:scriptlet) are disallowed here.

description The server encountered an internal error that prevented it from fulfilling this request.

exception

```
org.apache.jasper.JasperException: /countryForm.jsp (line: 20, column: 5) Scripting elements (
        org.apache.jasper.compiler.DefaultErrorHandler.jspError(DefaultErrorHandler.java:42)
        org.apache.jasper.compiler.ErrorDispatcher.dispatch(ErrorDispatcher.java:443)
        org.apache.jasper.compiler.ErrorDispatcher.jspError(ErrorDispatcher.java:89)
        org.apache.jasper.compiler.Validator$ValidateVisitor.visit(Validator.java:721)
        org.apache.jasper.compiler.Node$Scriptlet.accept(Node.java:932)
        org.apache.jasper.compiler.Node$Nodes.visit(Node.java:2375)
        org.apache.jasper.compiler.Node$Visitor.visitBody(Node.java:2427)
        org.apache.jasper.compiler.Node$Visitor.visit(Node.java:2433)
        org.apache.jasper.compiler.Node$Root.accept(Node.java:474)
        org.apache.jasper.compiler.Node$Nodes.visit(Node.java:2375)
        org.apache.jasper.compiler.Validator.validateExDirectives(Validator.java:1817)
        org.apache.jasper.compiler.Compiler.generateJava(Compiler.java:217)
        org.apache.jasper.compiler.Compiler.compile(Compiler.java:373)
        org.apache.jasper.compiler.Compiler.compile(Compiler.java:353)
        org.apache.jasper.compiler.Compiler.compile(Compiler.java:340)
        org.apache.jasper.JspCompilationContext.compile(JspCompilationContext.java:657)
        org.apache.jasper.servlet.JspServletWrapper.service(JspServletWrapper.java:357)
        org.apache.jasper.servlet.JspServlet.serviceJspFile(JspServlet.java:390)
        org.apache.jasper.servlet.JspServlet.service(JspServlet.java:334)
```

Slide 18

There is much more you can do with expression language.
This does not mean you have to use all these possibilities: they might lead to bad design.
Remember: your view should contain any other than presentation logic!

Slide 19

REFACTOR


```
<p>Het dier met naam
<%= request.getParameter(“naam")%>
moet
<%= ((Dier) request.getAttribute("gevondenDier")).getVoedsel()%>
keer per dag eten krijgen.</p>
```

Assignment: refactor this, using expression language.

Slide 20

RESULT

Het dier met naam

${param.naam}

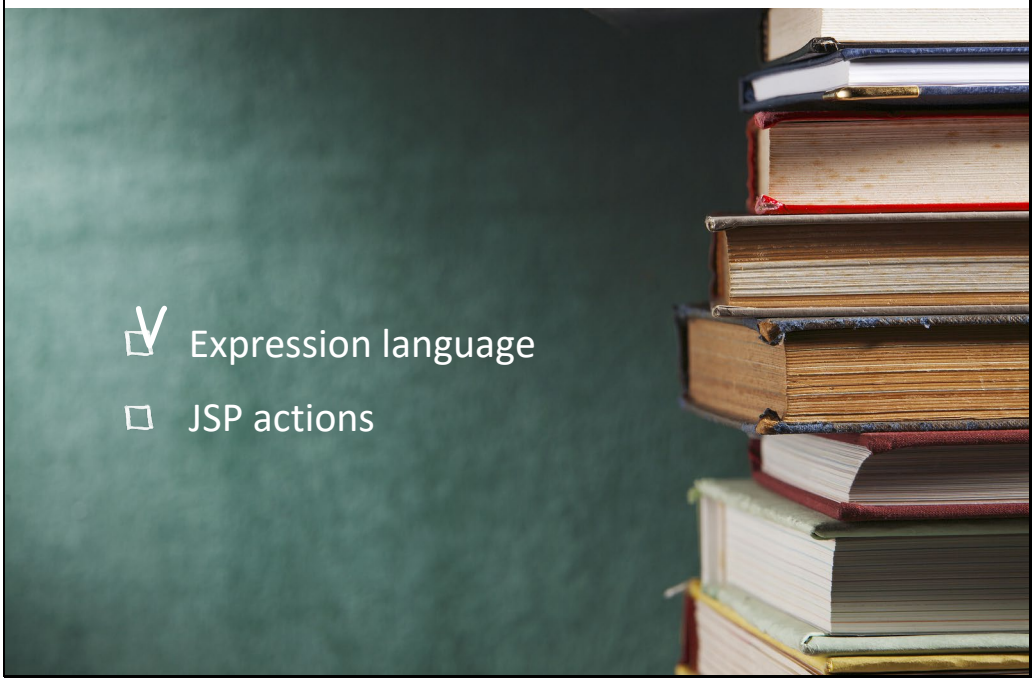
moet

${gevondenDier.voedsel}

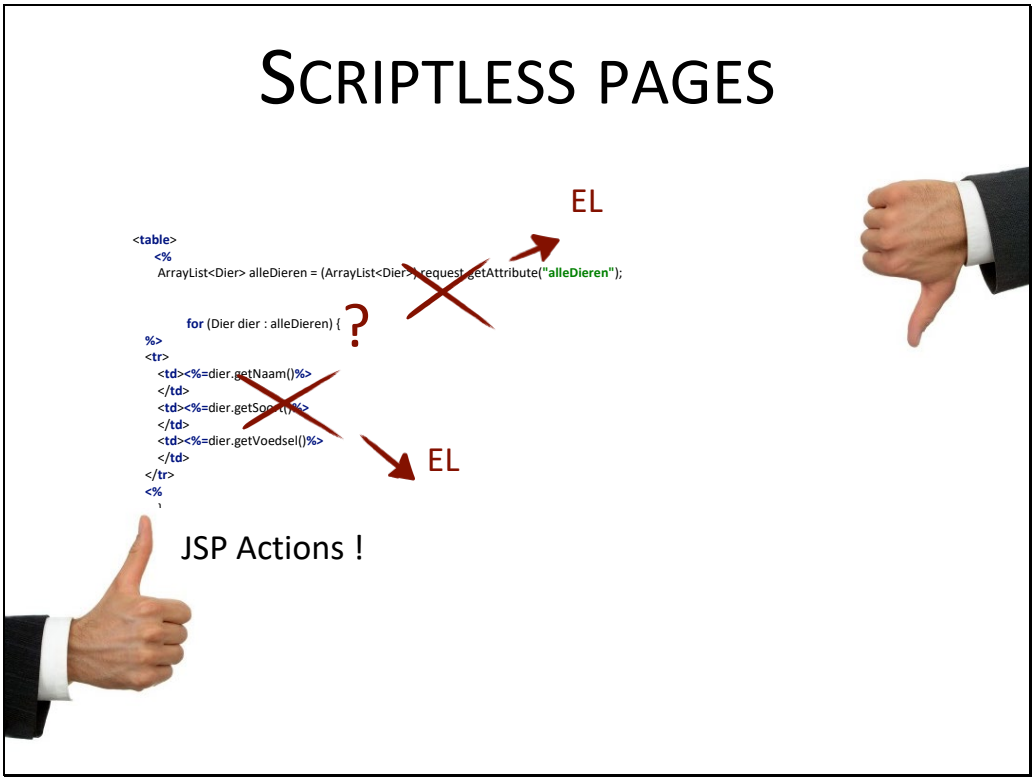
keer per dag eten krijgen.

Slide 21

AGENDA



Slide 22



Example country overview: some parts we can refactor with expression language, but other parts we can't. Expression language is not enough. That's why there is something else called JSP Actions.

Slide 23

JSP ACTIONS

- = predefined XML *tags*

components of a 'tag library'

- Categories:
  - standard `<jsp:include>`
  - custom `<c:forEach>`

    `<mine:homeMade>`

JSP actions are tags you can use next to the normal html tags. They offer you extra functionality, without using to much Java in your pages.

Slide 24

STANDARD ACTIONS

STANDARD ACTIONS

- defined in JSP itself
- begin with default `jsp:` prefix

<jsp:include>	Includes the response from a servlet or JSP page during the request processing phase
<jsp:param>	Adds a parameter value to a request handed off to another servlet
<jsp:plugin>	Generates HTML that contains the appropriate browser-dependent elements (OBJECT or EMBED) needed to execute an applet with the Java Plug-in software

Slide 26

STANDARD ACTIONS

```
<jsp:include page="header.jsp"/>
```

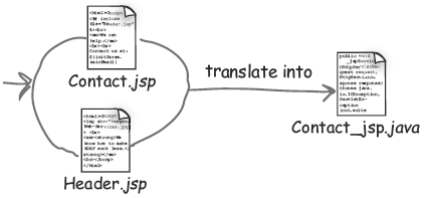
```
<jsp:include page="header.jsp">
   <jsp:param name="title"value="Countries"/>
</jsp:include>
```

```
<jsp:plugin type="applet" code="Molecule.class" codebase="/html">
  <jsp:params>
    <jsp:param name="molecule" value="molecules/benzene.mol" />
  </jsp:params>
  <jsp:fallback>
    <p>Unable to load applet</p>
  </jsp:fallback>
</jsp:plugin>
```

Slide 27

INCLUDE

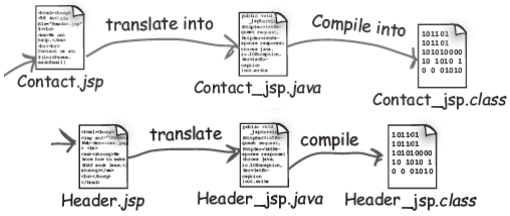
```
<%@include file="header.jsp" %>
```



inserts SOURCE
of “Header.jsp”
at translation time

```
<jsp:include page="header.jsp"/>
```

inserts RESPONSE
of “Header.jsp”
at runtime



Slide 28

DRY

```
<body>
    <%@include file="header.jspf" %>

    <main id="container">
    <p>
        <a href="index.html">Home</a>
    </p>
    <h1>Countries</h1>
    <article> ...
```

```
<body>
    <%@include file="header.jspf" %>

    <main id="container">
    <p>
        <a href="index.html">Home</a>
    </p>
    <h1>New Country</h1>
    <article> ...
```

?

Slide 29

DRY

```
<body>
    <%@include file="header.jspf" %>

    <main id="container">
    <jsp:include page="title.jsp">
      <jsp:param name="title" value="Countries" />
    </jsp:include>
    <article> ...
```

```
<body>
    <%@include fil

    <main id="container">
    <jsp:include page="title.jsp">
      <jsp:param name="title" value="New Country" />
    </jsp:include>
    <article> ...
```

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
  pageEncoding="UTF-8"%>
<p>
    <a href="index.html">Home</a>
</p>
<h1>${param.title}</h1>
```

title.jsp

Slide 30

CUSTOM ACTIONS

CUSTOM ACTIONS

- user-defined JSP language element
- NOT defined in JSP itself → include !
- Example: **JSP Standard Tag Library**

Custom actions are located in extra libraries. You will have to include them in your build path, and you will have to reference them from your JSP page.

Slide 32

JSTL

- **J**SP **S**tandard **T**ag **L**ibrary
- Bundels recurring functionality:
  - iteration
  - choice
  - ...

Documentation:
https://www.javaworld.com/article/2073217/soa/jsp-standard-tag-library-eases-webpage-development.html

This documentation will be available during exam (pdf).

Slide 33

ITERATION

```
<c:forEach var="dier" items="${alleDieren}">
    <tr>
        <td>${dier.naam}</td>
        ...
    </tr>
</c:forEach>
```

CHOICE: IF

```
<c:if test="${dier.voedsel > 10}">
        ...
</c:if>
```

```
<c:if test="${dier == null ||
                dier.naam == 'Albert'}">
        …
</c:if>
```

CHOICE: IF ELSE

```
<c:choose>
        <c:when test="${dier.voedsel > 20}">
                ...
        </c:when>
        <c:when test="${dier.voedsel > 50}">
                ...
        </c:when>
        <c:otherwise>
                ...
        </c:otherwise>
</c:choose>
```

Slide 36

USE LIBRARIES

Add to dependencies:

jstl-1.2.jar

```
<!-- https://mvnrepository.com/artifact/javax.servlet/jstl -->
<dependency>
  <groupId>javax.servlet</groupId>
  <artifactId>jstl</artifactId>
  <version>1.2</version>
</dependency>
```

JSP page:

```
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
```

Ask IntelliJ for help

LIBRARIES

- Core: http://java.sun.com/jsp/jstl/core
- XML: http://java.sun.com/jsp/jstl/xml
- Internationalization: http://java.sun.com/jsp/jstl/fmt
- SQL: http://java.sun.com/jsp/jstl/sql
- Functions: http://java.sun.com/jsp/jstl/functions

In this course, you will be using the Core library.

Ask IntelliJ for help …

Slide 38

YOUR OWN ACTIONS

Slide 39

SUMMARY JSP

Scriptlet:	`<% List<Country> countries = (List)request.getAttribute("countries"); %>`
Expression:	`<%= country.getName() %>`
Directive:	`<%@ include file="header.jsp" %>`
Declaration:	`<%! int counter = 0; %>`
EL:	`${ country.name }`
Action:	`<jsp:include page="header.jsp"/>`

Slide 40

REFACTOR


```
<table>
<% ArrayList<Dier> alleDieren = (ArrayList<Dier>)
request.getAttribute("alleDieren");

for (Dier dier : alleDieren) { %>
  <tr>
    <td><%=dier.getNaam()%></td>
    <td><%=dier.getSoort()%></td>
    <td><%=dier.getVoedsel()%></td>
  </tr>
  <% } %>
</table>

<p>Het meest hongerige dier is <%= ((Dier)
request.getAttribute("meestHongerige")).getNaam() %>.
</p>
```

Assignment: refactor this, using expression language and JSP actions.

Slide 41

RESULT


```
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ page language="java" contentType="text/html; charset=UTF-8"
      pageEncoding="UTF-8" %>
...
    <table>
      <c:forEach var="dier" items="${alleDieren}">
       <tr>
          <td>${dier.naam}
          </td>
          <td>${dier.soort}
          </td>
          <td>${dier.voedsel}
          </td>
       </tr>
       </c:forEach>
    </table>
     <p>Het meest hongerige dier is ${meestHongerige.naam}./p>
```

Slide 42

AGENDA

- ☑ Expression language
- ☑ JSP actions

Slide 43

DOCUMENTATIE

- https://docs.oracle.com/en/java/javase/12/docs/api/index.html
- https://docs.oracle.com/javaee/7/tutorial/jsf-el005.htm#BNAIK
- JSP Standard Tag Library eases Webpage development JavaWorld page 3.pdf
- JSP Standard Tag Library eases Webpage development JavaWorld page 1.pdf
- JSP Standard Tag Library eases Webpage development JavaWorld page 2.pdf