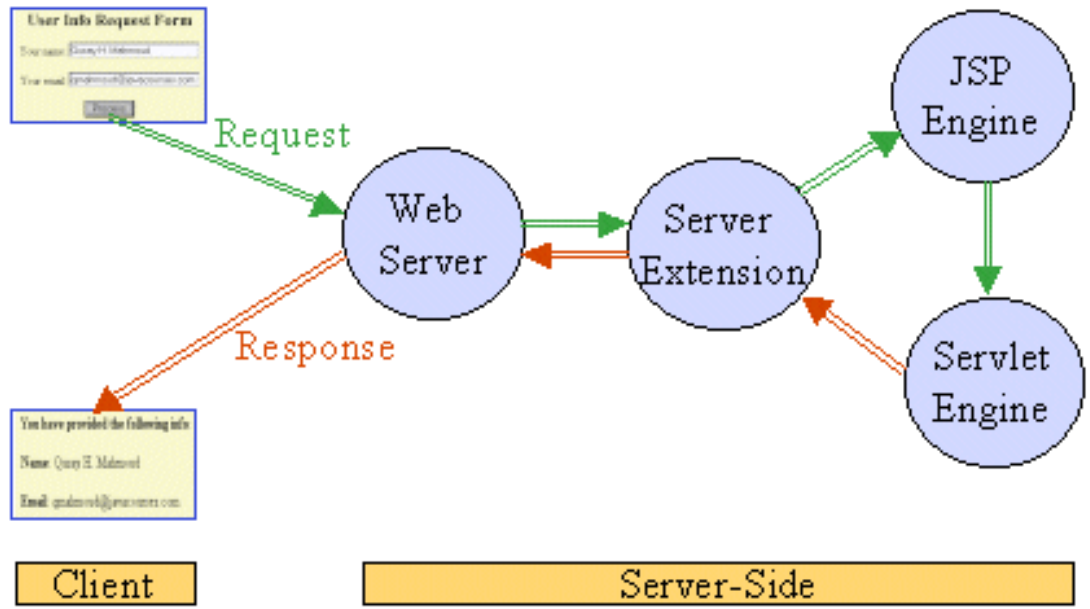
Admin

- Final: take-home (like prelim)
  - Available: 13 May
  - Due: 20 May *firm*
- Project presentations:
  - 10:00, 1:00, 3:00 weekdays
    - between 3:00 6 May and 3:00 19 May
    - Signup sheet Kathy, 4115 Upson

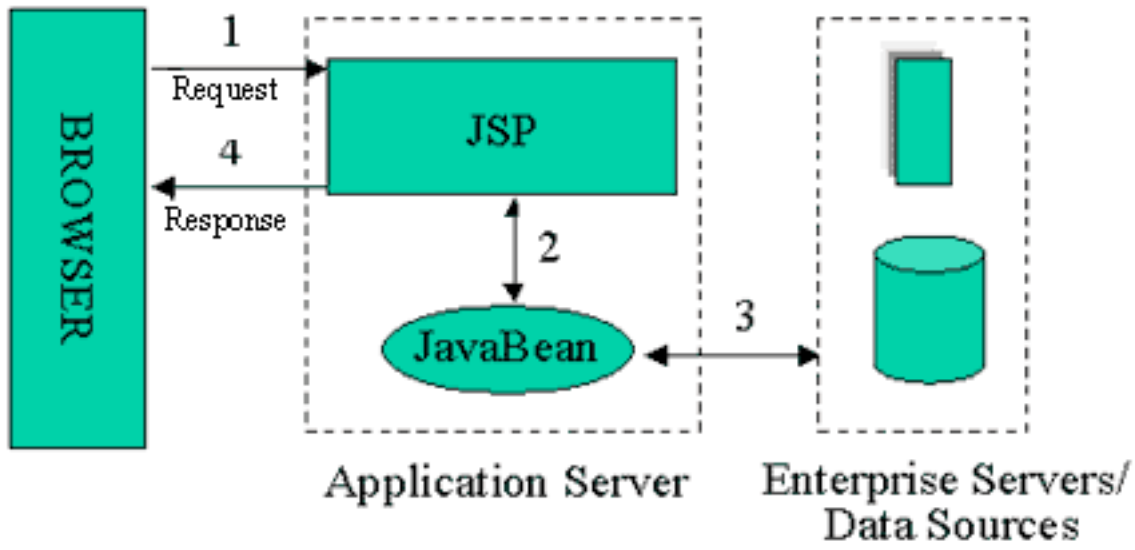
Struts -- MVC Framework

- http://jakarta.apache.org/struts/

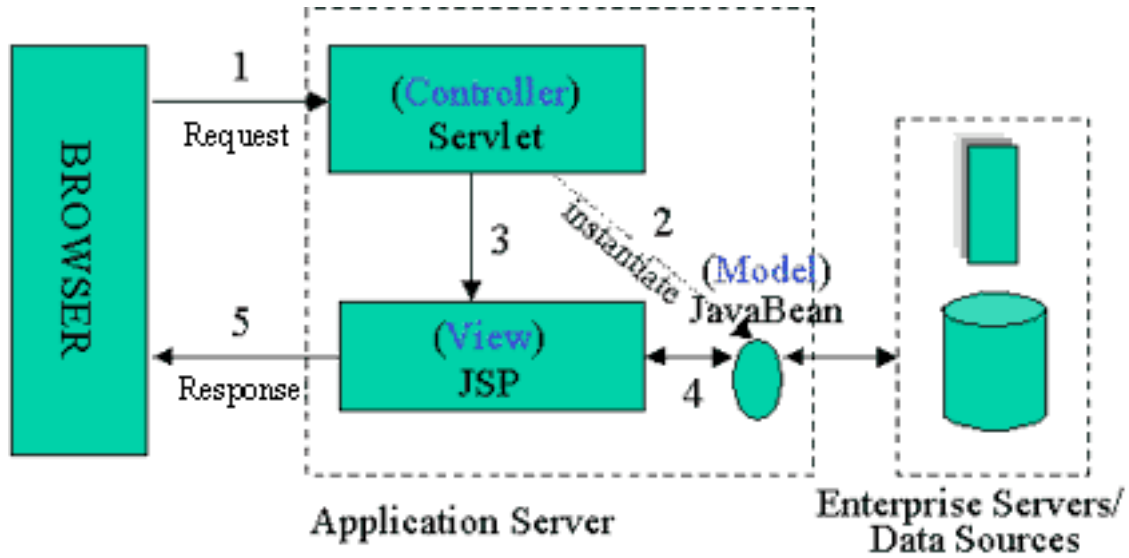
JSP + Servlet Architecture



JSP Model 1


BROWSER
1
Request
4
Response
JSP
2
JavaBean
3
Application Server
Enterprise Servers/
Data Sources

JSP Model 2 / MVC



Shopping Cart View

```
<%
Vector buylist = (Vector) session.getValue("shopping.shoppingcart");
if (buylist != null && (buylist.size() > 0)) {
%>
<table border="0" cellpadding="0" width="100%" bgcolor="#FFFFFF">
 ...
<%
  for (int index=0; index < buylist.size();index++) {
   CD anOrder = (CD) buylist.elementAt(index);
%>
<tr>
  <td><b><%= anOrder.getAlbum() %></b></td>
    ...
  <td>
   <form name="deleteForm"
    action="/examples/servlet/ShoppingServlet"
    method="POST">
   <input type="submit" value="Delete">
   <input type="hidden" name= "delindex" value='<%= index %>'>
  </form>  </td>  </tr>
    <% } %>
  </table>
<% } %>
```

Tags - JSP Standard Tag Library

- Eliminate much Java code from JSP pages
- Custom actions in separate namespaces:
  - Core
  - XML manipulation
  - SQL
  - Internationalization and formatting

Tags and Expressions

```
<c:if test="${book.orderQuantity > book.inStock}">
The book <c:out value="${book.title}"/> is currently out of
stock.
</c:if>
```

or

```
<c:if test="${book.orderQuantity >book.inStock}">
The book ${book.title} is currently out of stock.
</c:if>
```

- Eliminate many uses of Java code for control structure
- Allow use of field selection rather than getter/setter invocations

Implicit Objects

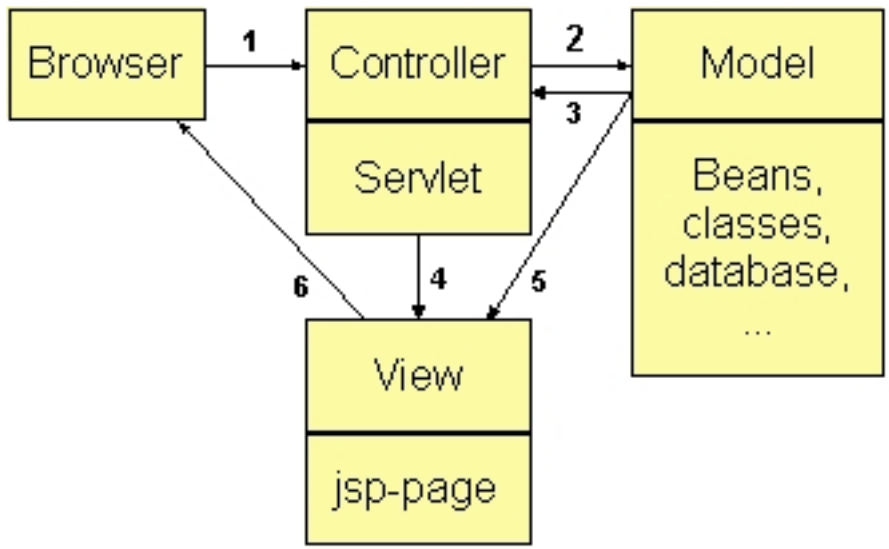
- ${pageContext.request.servletPath}
  - servlet path from HttpRequest
- ${sessionScope.loginId}
- ${param.bookId}
- ${paramValues.bookId}
  - String[] of all param.bookId values

- All common Java operators
- Implicit coercions
- Exception handling

Output

- <c:out value="${sessionScope.bookInfo.title}">
- Use escapeXml parameter to control interpretation of XML/HTML tags in output

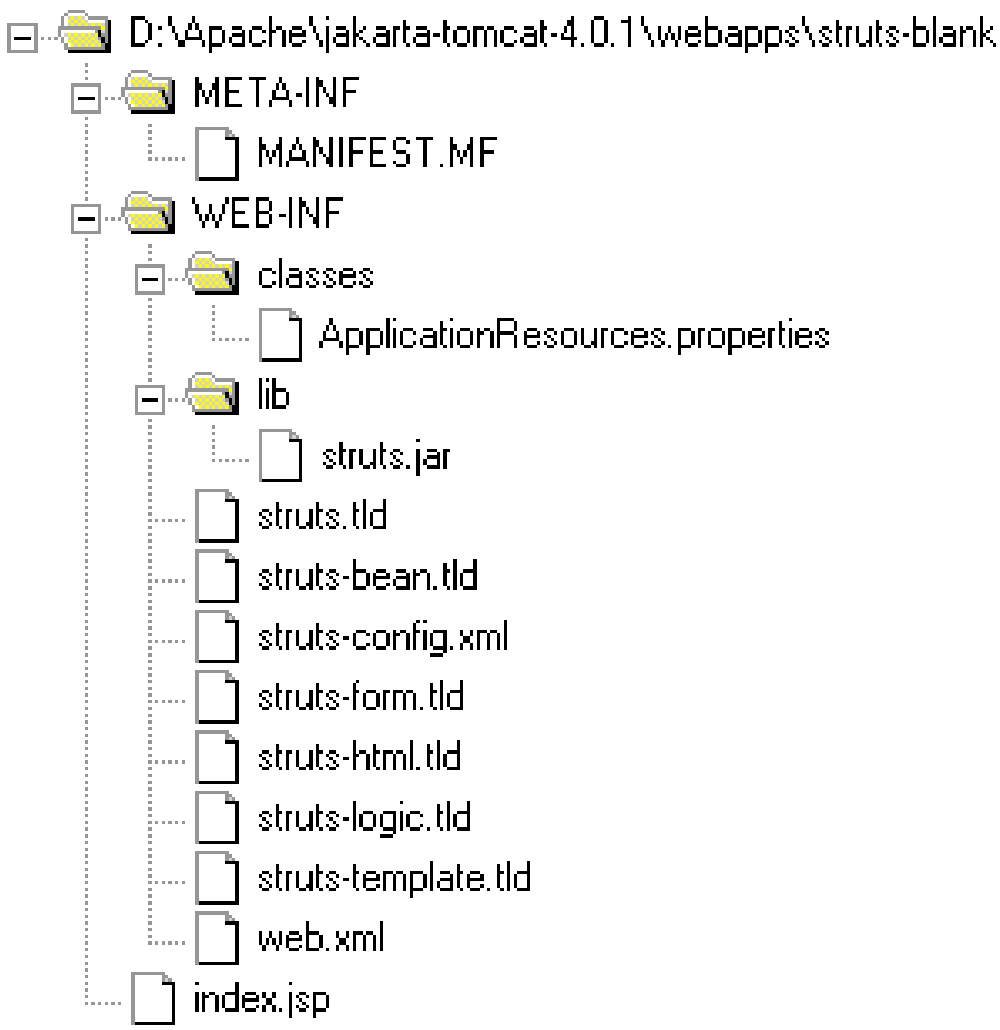
Struts MVC


Browser
1
Controller
2
Model
3
Servlet
Beans,
classes,
database,
...
6
4
5
View
jsp-page

Struts Components

- standard controller servlet for application
- beans and helper classes for use in model
- jsp tag libraries for use in views

Struts Application Directory



File or Directory name	Purpose
META-INF	Contains meta information. Used by utilities etc.
WEB-INF/classes	This is where you place you own Java classes.
WEB-INF/classes/ApplicationResources.properties	Contains the messages (fixed texts) of the application. Error messages are also put here.
WEB-INF/lib/struts.jar	Contains the Struts servlet, helper classes, taglib code etc.
WEB-INF/*.tld	The Struts tag libraries.
WEB-INF/struts-config.xml	A Struts configuration file. More on this later.
WEB-INF/web.xml	The usual configuration file for the servlet container. More on this later.
index.jsp	The jsp-files (and html-files) may be placed in the root of the application directory. "struts-blank" contains this single jsp-file.

Struts *web.xml* File

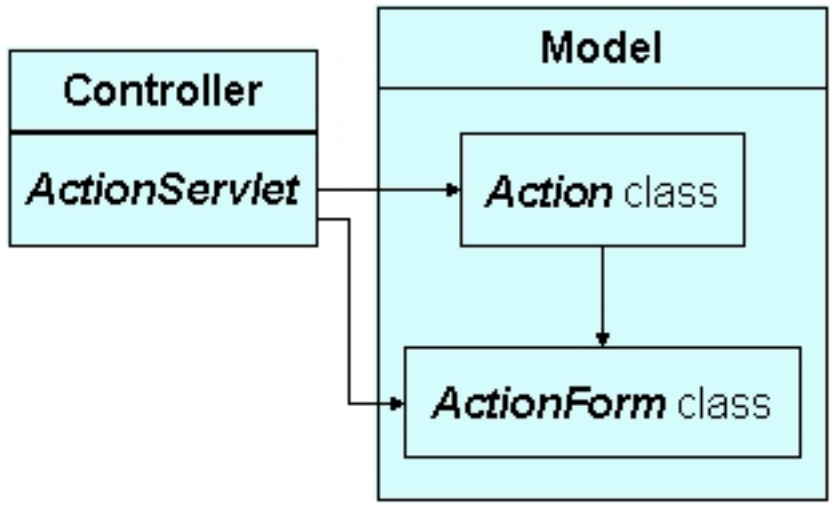
```
<web-app>

  <!-- Standard Action Servlet Configuration (with debugging) -->
  <servlet>
    <servlet-name>action</servlet-name>
    <servlet-class>
          org.apache.struts.action.ActionServlet
        </servlet-class>
      ...
  </servlet>


  <!-- Standard Action Servlet Mapping -->
  <servlet-mapping>
    <servlet-name>action</servlet-name>
    <url-pattern>*.do</url-pattern>
  </servlet-mapping>


  <!-- Struts Tag Library Descriptors -->
  <taglib>
    <taglib-uri>/WEB-INF/struts-html.tld</taglib-uri>
    <taglib-location>/WEB-INF/struts-html.tld</taglib-location>
  </taglib>
    ...

</web-app>
```



- Struts controller ActionServlet
  - transfers form data into ActionForm object
  - invokes Action class method

Struts *struts-config.xml* File

```
<struts-config>

  <!-- ========== Form Bean Definitions ================== -->
  <form-beans>

    <form-bean      name="submitForm"
                    type="hansen.playground.SubmitForm"/>

  </form-beans>

  <!-- ========== Action Mapping Definitions ============= -->
  <action-mappings>

    <action    path="/submit"
               type="hansen.playground.SubmitAction"
               name="submitForm"
               input="/submit.jsp"
               scope="request">
    <forward name="success" path="/submit.jsp"/>
    <forward name="failure" path="/submit.jsp"/>
    </action>

  </action-mappings>

</struts-config>
```

Simple Form JSP

```
<%@ page language="java" %>
<%@ taglib uri="/WEB-INF/struts-html.tld" prefix="html" %>
<html>
<head><title>Submit example</title></head>
<body>
<h3>Example Submit Page</h3>
<html:errors/>
<html:form action="submit.do">
Last Name: <html:text property="lastName"/><br>
Address:   <html:textarea property="address"/><br>
Sex:       <html:radio property="sex" value="M"/>Male
           <html:radio property="sex" value="F"/>Female<br>
Married:   <html:checkbox property="married"/><br>
Age:       <html:select property="age">
             <html:option value="a">0-19</html:option>
             <html:option value="b">20-49</html:option>
             <html:option value="c">50-</html:option>
           </html:select><br>
           <html:submit/>
</html:form>
</body>
</html>
```

Sample ActionForm Class

```
import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.*;

public final class SubmitForm extends ActionForm {

  /* Last Name */
  private String lastName = "Bush"; // default value
  public String getLastName() {
    return (this.lastName);
  }
  public void setLastName(String lastName) {
    this.lastName = lastName;
  }

  ...

  /* Age */
  private String age = null;
  public String getAge() {
    return (this.age);
  }
  public void setAge(String age) {
    this.age = age;
  }
}
```

Sample Action Class

```
import javax.servlet.http.*;
import org.apache.struts.action.*;

public final class SubmitAction extends Action {

  public ActionForward perform(ActionMapping mapping,
      ActionForm form,
      HttpServletRequest request,
      HttpServletResponse response) {

    SubmitForm f = (SubmitForm) form; // get the form bean
    // manipulate the last name value
    String lastName = f.getLastName();
    request.setAttribute("lastName", lastName.toUpperCase());

    // Forward control to the specified success target
    return (mapping.findForward("success"));
  }
}
```

Validation in ActionForm Class

```
public ActionErrors validate(ActionMapping mapping,
         HttpServletRequest request) {

  // Check for mandatory data
  ActionErrors errors = new ActionErrors();
  if (lastName == null || lastName.equals("")) {
    errors.add("Last Name", new ActionError("error.lastName"));
  }
  ...
  if (age == null || age.equals("")) {
    errors.add("Age", new ActionError("error.age"));
  }
  return errors;
}
```

- if returned errors is not empty the controller will return to the "input" page

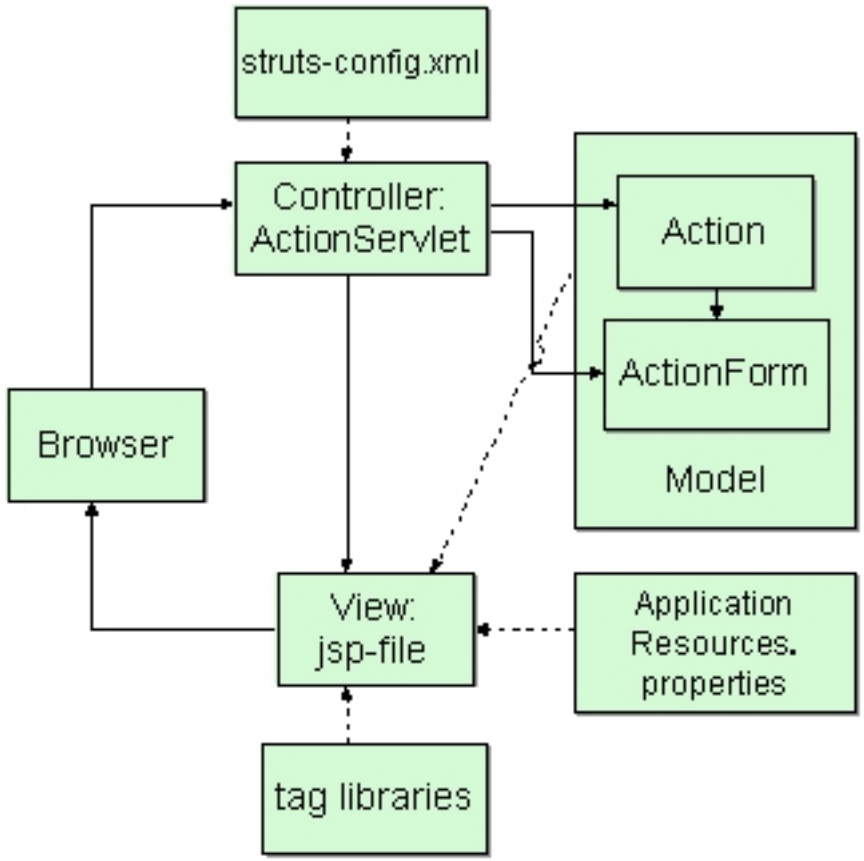
Localization

```
errors.header=<h4>Validation Error(s)</h4><ul>
errors.footer=</ul><hr>

error.lastName=<li>Enter your last name
error.address=<li>Enter your address
error.sex=<li>Enter your sex
error.age=<li>Enter your age
```

- text of error messages to be displayed by JSP
- specified in ApplicationResources.properties deployment file

Struts Components



Competitors

- Spring Framework
  - http://www.springframework.org
- Java Server Faces
  - http://java.sun.com/j2ee/javaserverfaces