

CS320 Web and Internet Programming

Working with Multiple JSP Pages

Chengyu Sun
California State University, Los Angeles

Overview

- ◆ Include and include
- ◆ Forward and redirect
- ◆ Scopes again
- ◆ Error pages

<%@ include>

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

- ◆ For all intent and purposes, the included file is simply part of the host page
- ◆ "Inclusion" happens at the translation time
- ◆ Example:
 - HostPage.jsp and i1

Path to the Included File

- ◆ Relative path (relative to who??)
 - e.g. `<%@ include file="page1.jsp" %>`
- ◆ Absolute path (absolute from where??)
 - e.g. `<%@ include file="/page1.jsp" %>`
- ◆ Hide pages under *WEB-INF*

```
<%@ include file="/WEB-INF/header.jsp" />
```

<jsp:include>

```
<jsp:include page="path_to_page" />
```

- ◆ Inclusion happens at request time
 - *Output* of JSP page
 - *Output* of servlet
 - Static resources
 - ◆ HTML pages
 - ◆ Plain text files

Requests and Parameters

- ◆ Access to request and parameters??
- ◆ Add additional parameters
 - header.jsp in Heather

```
<jsp:include page="/WEB-INF/header.jsp">  
  <jsp:param name="title" value="Student Home"/>  
  <jsp:param name="name" value="${stu.fname} ${stu.lname}"/>  
  <jsp:param name="p1" value="${sys.quarterYear}"/>  
  <jsp:param name="p2" value="${sys.date}"/>  
</jsp:include>
```

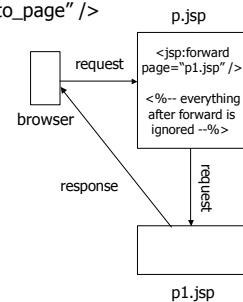
<%@ include> vs. <jsp:include>

- | | |
|--|---|
| <ul style="list-style-type: none"> ◆ <%@ include> <ul style="list-style-type: none"> ▪ content of the page ▪ translation time <ul style="list-style-type: none"> • better performance • may cause maintenance problem ▪ become part of the host page ▪ Don't use it | <ul style="list-style-type: none"> ◆ <jsp:include> <ul style="list-style-type: none"> ▪ output of the page ▪ request time <ul style="list-style-type: none"> • better flexibility • no maintenance problem ▪ become part of the HTTP response produced by the host page <ul style="list-style-type: none"> • cannot set host page response status code, header etc. ▪ Use it |
|--|---|

<jsp:forward>

<jsp:forward page="path_to_page" />

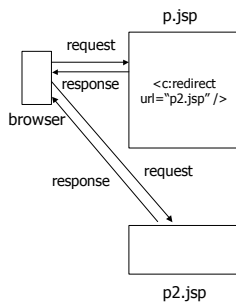
- ◆ forward the same request to another page
- ◆ Add more parameters to the request using <jsp:param>
- ◆ terminates the execution of the current page
- ◆ Don't use it (see MVC)



<c:redirect>

<c:redirect url="url" />

- ◆ redirect the browser to another URL
- ◆ the browser sends a new request to the URL
- ◆ terminates the execution of the current page??



Scoped Variables

- ◆ Sharing data among multiple JSP pages using *scoped variables*
- ◆ Scoped variables: data associated with one of the scopes
 - <name, value> pairs

Access Scoped Variables in Servlets

- ◆ application scope – ServletContext
 - getAttribute(String name)
 - setAttribute(String name, Object value)
- ◆ session scope – HttpSession
 - getAttribute(String name)
 - setAttribute(String name, Object value)
- ◆ request scope – HttpServletRequest
 - getAttribute(String name)
 - setAttribute(String name, Object value)
- ◆ page scope – PageContext
 - getAttribute(String name)
 - setAttribute(String name, Object value)

Access Scoped Variables in JSPs

```
<c:set var="v1" value="abc" scope="page" />
<c:set var="v1" value="bcd" scope="request" />
<c:set var="v2" value="hello" scope="session" />
```

$\${v1}$, or $\${pageScope.v1}$

```
<c:remove var="v2" />
<c:remove var="v1" scope="request" />
```

Scopes and Data Sharing

	page	request	session	application
<%@ include>				
<jsp:include>				
<jsp:forward>				
<c:redirect>				

Error Page

- ◆ In a production system
 - Hide runtime exception details from users
 - Make the web application look more professional

Error page:

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

Use error page:

```
<%@ page errorPage="path_to_errorPage" %>
```

Error Data

- ◆ Access details about the exception

```
${pageContext.errorData.requestURI}  
${pageContext.errorData.servletName}  
${pageContext.errorData.StatusCode}  
${pageContext.errorData.Throwable}
```

javax.servlet.jsp.ErrorData

- ◆ getRequestURI()
- ◆ getServletName()
- ◆ getStatusCode()
- ◆ getThrowable()

java.lang.Throwable

- ◆ getMessage()
- ◆ getStackTrace()

Alternative to Error Page

- ◆ Handle exceptions within the JSP page

```
<c:catch var="e">  
  <%-- some code that may throw an exception --%>  
</c:catch>
```

```
<c:if test="${! empty e}">  
  <%-- an exception happened. --%>  
  <%-- do something about it --%>  
</c:if>
```

Preparing for the Midterm ...

- ◆ Read all the notes and sample code
- ◆ Write a bean with an int property, a boolean property, a String property, and a collection property
- ◆ Write a simple example for each of the JSTL Core tags
 - Try use your bean in the examples
 - Try use EL in the examples
 - Make sure the examples work on the CS server

... Preparing for the Midterm

- ◆ Put all your code samples on the CS server (or somewhere handy)
- ◆ Disclaimer: you may still not do well in the midterm after all the preparation (programming can not be learned in 3 days)