SENG2050 – Introduction to Web Engineering

Lecture 4a: More on JSP

Overview

- Last week:
  - Introduction to JSP – a better way of generating HTML
  - Introduction to Java Beans
- This week:
  - JSP Actions
  - JSP Directives
  - More on Java Beans
  - Demo

JSP Actions

- Dynamic behavior at a given point
  - Coordinate JSP with Java Beans
  - Import classes
  - Include external files
  - Forward requests to another JSP or servlet
- Have the form **<jsp:*action* … />**

JSP Actions

- jsp:useBean
- jsp:setProperty
- jsp:getProperty
- jsp:include
- jsp:forward

jsp:useBean …….Review

- **`jsp:useBean`** associates a Bean with the current JSP
  - **`<jsp:useBean id="`*`beanId`*`" class="`*`beanpackage.BeanClass`*`" />`**
  - Associates a bean of the given class with the given id name (instance variable), accessible from later in the JSP
  - A new Bean is created if no Bean exists with the same ***beanId*** and ***scope***

jsp:useBean …….Review

- **`<jsp:useBean`** … **`scope="`*`scope`*`" />`**
  - **`scope="page"`** – the bean can be used within the page only
  - **`scope="request"`** – the bean can be used in any JSP that is processing the same request
  - **`scope="session"`** – the bean can be used in any page that participates in the client session
  - **`scope="application"`** – the bean can be used in any page in the current application
  - **Why?**

jsp:useBean

- **<jsp:useBean … scope="*scope*" />**
  - **scope="page"** – once-off values
    - Search parameters/results
  - **scope="request"** – once-off values but can be passed to other JSP or servlets that are dealing with the same request
    - Search parameters/results
  - **scope="session"** – values that are seen by, used by, and associated to a single user until they log off
    - Shopping cart
  - **scope="application"** – values that all users of the application can see
    - Current active users

jsp:useBean

- **<jsp:useBean … type="*JavaClass*" />**
  - Forces the Bean object to have the given type, instead of the type in the **class** attribute
  - Used to type a Bean as one of its superclasses or interfaces – polymorphic Beans!
- **<jsp:useBean … beanName="*beanFile*" />**
  - Replaces the **class** attribute
  - Can create a Bean from a serialized object
  - Persistent Beans, even if the server crashes!

jsp:setProperty …….Review

- For most tags, the values of **name** and **value** are computed when the JSP is compiled into a servlet
  - Not so in **jsp:setProperty** – **name** and **value** are evaluated at request time.
  - You can embed JSP expressions:
  - **<jsp:setProperty name="stringBean" property="message" value='<%= request.getParameter("message") %>' />**
  - Note the single-quotes of **value='…'** – avoids conflict with double-quotes of **"message"**

jsp:setProperty …….Review

- The value passed to **jsp:setProperty** must match the type of the Bean's corresponding **setXxxxx** method
  - If this isn't **String**, then you have to explicitly convert the result of **request.getParameter()**
  - Requires knowing the Java types of each Bean method ☹
  - Requires embedding Java code in your JSP ☹

jsp:setProperty …….Review

- **`jsp:setProperty`** can do this automatically
  - **`<jsp:setProperty name="`*beanId*`" property="`*propertyName*`" param="`*parameterName*`" />`**
  - Automatically converts parameter **`String`** into any of the Java built-in types (**`int`**, **`long`**, **`byte`**, **`float`**, **`double`**, **`boolean`**, **`char`**) and equivalent objects (**`Integer`**, **`Long`**, **`Byte`**, **`Float`**, **`Double`**, **`Boolean`**, **`Character`**)

jsp:setProperty …….Review

- **`jsp:setProperty`** can also set all properties from their corresponding parameters in one go
  - **`<jsp:setProperty name="`*beanId*`" property="*" />`**
  - Sets every property for which a parameter is passed
  - If parameter is not passed, then property is not set – missing parameters are not set to null
  - Property names and parameter names must match *exactly*

JSP + Java Bean Demo

jsp:include

- `<jsp:include page="`*`url`*`" flush="true" />`
  - Include output from the resource at the relative *url* into the output stream of the JSP at this point
    - Why?
  - The output is regenerated for each request
    - Has access to request object
    - Can generate dynamic content
  - The `page` attribute can be a JSP expression

jsp:include

- `<jsp:include page="`*`url`*`" flush="true" />`
  - Include output from the resource at the relative *url* into the output stream of the JSP at this point
    - Why – useful for common page elements like navigation, headers, and footers
  - The output is regenerated for each request
    - Has access to request object
    - Can generate dynamic content
  - The `page` attribute can be a JSP expression

jsp:include

- `<jsp:param name="`*`name`*`" value="`*`value`*`" />`
  - Used to pass parameters to the included resource
  - ```
    <jsp:include page="url" flush="true" >
      <jsp:param name="name" value="value" />
      …
      <jsp:param name="name" value="value" />
    </jsp:include>
    ```
  - The `value` attribute can be a JSP expression
    ```

jsp:forward

- `<jsp:forward page="`*url*`" />`
  - Passes responsibility for the request to the relative *url*
  - Preserves the `request` implicit object
  - Shares Beans with `scope="session"`
  - The `page` attribute can be a JSP expression

jsp:forward

- `<jsp:forward page="`*url*`" />`
  - Handled in the server
  - The client sees it as the same URL
  - Server treats it as a single request
- Versus `response.sendRedirect(`*url*`)`
  - Handled by the client
  - Client sees the new URL
  - Server treats it as two requests

JSP Directives

- Control global properties of the page
  - Class imports
  - Customising the servlet class
  - Setting content-type
- `<%@ directive ... %>`
- `<jsp:directive.directive ... />`
- Directives:
  - `include`
  - `page`
  - `taglib`

page Directives

- `<%@ page session="boolean" %>`
  - `session="true"` (the default) – bind to an existing session object, or create a new one
  - `session="false"` – do not create or bind a session; throws an exception if you try to use the `session` implicit object

page Directives

- `<%@ page buffer="`*sizeInKB*`" %>`
- `<%@ page buffer="none" %>`
- `<%@ page autoflush="`*boolean*`" %>`
  - Control buffering of the output writer
  - If `buffer="none"` ensure all HTTP headers are set before generating HTML
  - You cannot use `buffer="none"` with `autoflush="false"`

page Directives

- `<%@ page extends="`*Class*`" %>`
  - The class that the servlet object extends
- `<%@ page info="`*message string*`" %>`
  - Defines the string that will be returned by the servlet's `getServletInfo()` method
- `<%@ page language="java" %>`
  - The programming language used in the page
  - At present, Java is the only valid choice

page Directives

- `<%@ page errorPage="`*`url`*`.jsp" %>`
  - Placed in a JSP, sets a relative URL to jump to when an uncaught exception occurs
- `<%@ page isErrorPage="true" %>`
  - Indicate that the JSP is an error page
  - The exception which caused the error is available in the implicit object `exception`

include Directive

- `<%@ include file="`*`url`*`" %>`
  - Pastes the source text of the relative URL into the JSP at this point – like a C `#include`
  - This happens only when the JSP is converted into a Java servlet
  - If the included file changes, then you have to **force** the servlet to be regenerated – most JSP servers will not do this automatically
  - Useful for including headers and footers

Notes on include

- A mechanism for abstraction in JSP development
  - Breaking a large and complex JSP into sub-pages (sub-modules/sub-routines/components)
- Which form to use?
  - Core Servlets: Use directive only if you need JSP constructs
  - Hans Bergsten: Use directive if the included page rarely changes; use action if it is decided at run-time

Notes on include

- **include** directive is more efficient
- **include** directive is less secure
  - Included page can contain malicious JSP
- **include** action cannot share page variables (use Beans or sessions)
- **include** action cannot set headers or cookies

Directives demo