Java Server Pages (JSP) and JavaBeans

Dr. Awais Majeed

Outline

1. Few more concepts

2. JavaBeans

Implicit Objects

- These are the objects in JSP which are automatically available in JSP
- These are provided by JSP container to the developer to be used in servlets and JavaBeans
- These objects are:
  request, response, Session, Out, PageContext, Application, Config, Page

Request Object

- The class or interface name of the request object is `http.httpservletrequest`
- The object `request` is of type `Javax.servlet.http.httpservletrequest`.
- The request object gives information related to the request (parameters and headers) and the client

Implicit Objects-II

Request Object

- Getting the parameters
  - request object provides different methods to access the parameters in the client's request
  - `request.getParameter("par-name")` returns a String value of the requested parameter
  - In certain cases, a parameter may have more than 1 values
  - For example the parameter `http://localhost:8080/my_page.jsp?aaa&aaa=4&aaa=7` has three values.
  - Use `request.getParameterValues("par-name")` to get an array of String values
- Other methods are getRemoteAddress, getRemoteHost, getLocale()

Implicit Objects-III

Session Object

- The term session refers to all the interactions that a user has with a server from the moment the user accesses the first page of an application to the moment he closes his browser (or the session expires, because the browser remained inactive for too long).
- The class or the interface name of the object Session is `http.HttpSession`.
- The object Session is of type `Javax.servlet.http.httpsession`.
- When Tomcat receives an HTTP request, it checks whether the request contains a cookie named JSESSIONID.
- If it is not present it creates a cookie with unique value and attaches it to the response

Implicit Objects-IV

Session Object ..

- session objects allows to JSP pages to store information related to each individual user
- `session.setAttribute(’’attribute-name’’, ’’value’’)` can be used to set a unique identifier identifying logged in user
- Other methods are: getAttribute(String name), getAttributeNames, isNew(), getCreationTime, getId, invalidate(), getLastAccessedTime, getMaxInactiveInterval, removeAttribute(String name), setAttribute(String, object)
- ***posssible attributes that can be used to identify a particular user ??***

out Object

- Similar to System.out in standard Java applications, writes the standard output
- The standard output for a JSP page is the HTML page, so it writes to the HTML page produced
- in scriptlet you can write it as `<%out.print(’’abc’’);%>` similar to `<%=’’abc’’%>`

JSP Standard Actions-I

- Tomcat executes directives elements while translating a page, it executes action elements when processing a client's HTTP request.
- The purpose of JSP actions is to specify activities to be performed when a page is requested
- Actions can operate on objects and have an effect on each response
- They normally take the following form:
    - `<jsp:action-name action-attribute-list/>`

JSP Standard Actions-II

useBean action

- The `useBean` action declares a new JSP scripting variable and associates it with a Java object.
  - The syntax is:
    ```
    <jsp:useBean id="objectName"
    scope="page|request|session|application"
    class="packagename.className"/>
    ```
  - The useBean action can actually instantiate new objects, rather than just declare and make accessible objects that are already defined

JSP Standard Actions-III

- Once a Javabean is instantiated, one might want to set the properties of the bean
- This is done by using the 'setProperty' action
- A bean property is nothing else than an attribute of the bean class for which the standard get and put methods have been defined Example
  `<jsp:setProperty property="*" name="customer"/>`
  where all properties of bean object 'customer' are set from the parameter values passed during the request process in which each parameter has the same name as the bean properties. Other way to define a particular value is:
  `<jsp:setProperty name="obj" property="i" value="22"/>`
- `action-attribute-list` is a sequence of one or more attribute-name="attribute-value" pairs.
- There are eight JSP standard actions (element, forward, getProperty, include, plugin, setProperty, text, and useBean)

JSP Standard Actions-IV

Actions: forward, include, and param

- To terminate execution of the current page and forward the request to another page, use this code:

```
<jsp:forward page="myOtherPage.jsp">
<jsp:param name="newParName" value="newParValue"/>
</jsp:forward>
```

- Use this code to execute another page and append its output to that of the current one:
  `<jsp:include page="myOtherPage.jsp"/>`
- In the previous example, we have created a new parameter 'newParName' which will be accessible to the new page by using the request method.

JavaBeans

- JavaBeans components are Java classes that can be easily reused and composed together into applications.
- Any Java class that follows certain design conventions can be a JavaBeans component
- A JavaBeans component property can be:
  - Read/write, read-only, or write-only
  - Simple, which means it contains a single value, or indexed, which means it represents an array of values
  - Most often declared as private

JavaBeans-II

- The property must simply be accessible using public methods that conform to certain conventions:
  - There is an empty constructor in each JavaBean
  - For each readable property, the bean must have a method of the form `public PropertyClass getProperty() { ... }`
  - For each writable property, the bean must have a method of the form `public setProperty(PropertyClass pc) { ... }`
- To use a bean in a JSP page, one needs an instance(object) of that class that can be created by using <jsp:useBean> action.
- Properties can be set using the `<jsp:setProperty>` action as discussed earlier.
- The values of the bean properties can be accessed by using `<jsp:getProperty name=''myobj'' property=''myprop''>` action