NetUI JSP Overview

Table of contents

1. Introduction

NetUI adds three tag libraries to normal JSP usage to assist with the binding of the JSPs to the controller class. These tag libraries add a number of features such as HTML form controls, data grids, and trees. For a detailed overview of the tag libraries and their usage see the NetUI Tag Library Overview. The tag library API documentation describes all of the details.

The tag libraries provided by NetUI are:

- `<netui:xxx>` -- This is the primary tag library supporting HTML, including the form controls. In addition, it contains the Tree support.
- `<netui-data:xxx>` -- This tag library provides binding to relational data, providing the Data Grid.
- `<netui-template:xxx>` -- This tag library provides a very simple templating facility, allowing common elements such as headers, footers, etc.

NetUI also makes extensive use of ***data binding expressions*** to bind the JSPs to data in the controller class. For a detailed explanation of databinding expressions see Databinding: Passing Data Between Controller Classes and JSPs

2. Starting a NetUI JSP

NetUI JSPs that act as stand alone HTML pages, can use the following template as the basis for a new page. Using this template will insure that the structure of the HTML page is valid and that all of the features in the tag library are enabled.

Simple NetUI JSP Template

```
<%@ page language="java" contentType="text/html;charset=UTF-8"%>
<%@ taglib prefix="netui"
uri="http://beehive.apache.org/netui/tags-html-1.0"%>
<netui:html>
    <head>
        <title>Page Title</title>
        <netui:base/>
    </head>
    <netui:body>
    </netui:body>
</netui:html>
```

As with Page Flow Controllers, a certain amount of common boilerplate text is required in each page. The first two lines should set the content-type, the encoding, and import the base NetUI tag library. The `taglib` binds the NetUI tags to the `netui` prefix.

```
<%@ page language="java" contentType="text/html;charset=UTF-8"%>
```

```
<%@ taglib prefix="netui"
uri="http://beehive.apache.org/netui/tags-html-1.0"%>
```

After the common prolog, the JSP can be written like most any other JSPs. The NetUI HTML tags can be configured to support both the HTML 4.01 spec and XHTML. See the HTML to NetUI Tag topic to see how the NetUI tags relate to HTML. In addition to the prolog and using the HTML tags to represent the HTML structure, `<netui:base>` should be present within the `<head>` element. This will ensure that relative URIs are resolved correctly.

The resulting general form of a NetUI JSP is as follows:

```
<%@ page language="java" contentType="text/html;charset=UTF-8"%>
<%@ taglib uri="http://beehive.apache.org/netui/tags-html-1.0"
prefix="netui"%>
<netui:html>
  <head>
    <title>...</title>
    <netui:base/>
  </head>
  <netui:body>
    ..
    ..
    ..
  </netui:body>
</netui:html>
```

3. Simple Linking

The `<netui:anchor>` tag replaces the normal `<a>` HTML anchor tag. A plain HTML `<a>` links directly from one URL to another and doesn't provid the controller an opportunity to perform any conditional logic. Using the `<netui:anchor>` tag you can specify an action in a Page Flow controller which will run when the link is pressed.

While it may seem silly to use NetUI tags for simple *constant forward methods*, the advantage is that if a page gets renamed or you wish to change the flow through an application, the destination only needs to be changed once, within the controller. Otherwise, you may have to edit a handful of JSPs manually changing the URLs inside normal `<a>` tags.

In addition, the <netui:anchor> tag supports many additional features such as submitting forms and popup windows.

3.1. Linking Example

Initially, we will examine simple linking through a controller to another JSP using the Page Flow controller introduced earlier.

implementation page flow

If your application changes and you desire to show a terms-of-service before allowing login, you can simply alter the `login()` controller method to send a user to `terms_of_service.jsp` before further sending him to the actual login screen. Instead of specifying the URL to another page using the HTML <a> tag, we use the <netui:anchor> and specify the name of the action on the controller to call.

Instead of using `<a>`

```
<a href="login.jsp">Login!</a>
```

Use `<netui:anchor>`

```
<netui:anchor action="login">Login!</netui:anchor>
```

When the link is displayed on-screen, clicking it will cause control to go through the `Controller`'s `login()` method, which will return the correct forward to select the actual next page to display.

If you want to add the term-of-service page, you can simply modify the action in the Page Flow controller to go to the newly created JSP. The advantage is that the flow of control is stored in the Page Flow controller and not in the pages themselves.

4. Handling Forms

To post data from forms to Page Flow controller's action handling the form post, the `<netui:form>` container tag, along with specialized NetUI form control tags that replace the normal form elements are used within a JSP. Similar to how `<netui:anchor>` replaces normal HTML `<a>` tags, the `<netui:form>` tag replaces the typical HTML `<form>` tag. In addition, the NetUI form control tags replace the typical HTML form controls.

4.1. Form Example

Instead of using `<form>`

```
<form action="LoginServlet" method="POST">
```

Use `<netui:form>`

```
<netui:form action="processLogin" method="POST">
```

The other tags typically used with a `<form>` also have replacements from the NetUI tag library.

For the `processLogin(...)` form-processing method, the matching JSP form would be:

```
<netui:form action="processLogin" method="POST">
    <netui:textBox dataSource="actionForm.username" size="20"/>
```

```
    <netui:textBox dataSource="actionForm.password" size="20"
password="true"/>
    <netui:button  type="submit" value="Login"/>
</netui:form>
```

When the user submits the form by clicking upon the `Login` button, an instance of the `LoginForm` subclass of `FormData` is created and passed to the `processLogin(LoginForm form)` method of the controller class.

5. Next...

Next, learn about how to compile and package up a complete web project.

- NetUI Project Model