Tiles Support

Table of contents

1. Introduction

Often, within your web application there are shared components such as a menu bar, banner or header, footer, and copyright. NetUI provides a couple of mechanisms to support layout management for assembling presentation pages from separate components. First, there are the NetUI Template Tags, a simple templating mechanism, defining a template page that contains the layout and information you want on each page within a site or page flow. The other option is to use the Tiles framework support built into NetUI.

Support for Tiles is achieved by incorporating the Tiles plugin and providing Tiles related "annotations" in the NetUI declarative programming model for the page flow "controller" class.

The following is a brief description of the annotations and the steps required to use the Tiles support in NetUI.

2. Declaring Tiles Definitions

First, create your Tiles definitions and the corresponding layout templates. Place your Tiles definition configuration file(s) in the WEB-INF directory. A Tiles definition file can also be located in the same directory as the page flow controller; however, if it is placed locally, you should avoid including it in the deployed webapp (or, if you are deploying your web project in place, you should add a Servlet Filter to prevent the file from being externally visible).

The definitions you declared should use webapp-relative paths to the JSP files that will be the template(s) for the layout, using <tiles:insert> tags from the tiles tag library, and the common component JSP files such as header, footer, menu, etc. used by the templates. The example definition below illustrates the use of a template JSP ("/layout/page.jsp") and common component files for the header, menu, and footer. The definitions that extend "defaultLayout" define the actual content that should be inserted as the content of the "body" component in the page.

/WEB-INF/tiles-defs.xml

```
<tiles-definitions>
    <definition name="defaultLayout" path="/layout/page.jsp">
        <put name="exampleTitle" value="Tiles" />
        <put name="header"       value="/layout/header.jsp"/>
        <put name="menu"         value="/menu.jsp"/>
        <put name="body"         value="/layout/body.jsp"/>
        <put name="footer"       value="/layout/footer.jsp"/>
    </definition>

    <definition name="flow1" extends="defaultLayout">
```

```
        <put name="exampleTitle" value="Tiles - Flow 1" />
        <put name="body"         value="/flow1/index.jsp" />
    </definition>

    <definition name="flow2" extends="defaultLayout">
        <put name="exampleTitle" value="Tiles - Flow 2" />
        <put name="body"         value="/flow2/index.jsp" />
    </definition>
    ...
</tiles-definitions>
```

To use the definitions in a page flow, add the `tilesDefinitionsConfigs` attribute to your `@Jpf.Controller` annotation. This attribute takes an array of paths to support the use of multiple Tiles-definitions files if desired. The following is an example of how the annotation and attribute would be written for a single configuration file named "tiles-defs.xml" and located in the WEB-INF directory.

```
@Jpf.Controller(
    tilesDefinitionsConfigs = { "/WEB-INF/tiles-defs.xml" },
    ...
)
```

3. Using Tiles Definitions as Forwards in a Page Flow Controller

For any `@Jpf.Forward` , `@Jpf.SimpleAction` , or `@Jpf.ConditionalForward` annotation you've defined in the actions of your controller, you can define the Tiles definition for the `@Jpf.Forward`. Rather than set the path attribute for a JSP, you set the attribute, `tilesDefinition`. This value of the attribute should be the name of a Tiles definition. Using the definitions defined above, the following is an example of a Forward annotation with the `tilesDefinition`.

```
@Jpf.Action(
    forwards = {
        @Jpf.Forward(
            name = "continue",
            tilesDefinition = "flow2")
    })
```

Review the NetUI samples for a concrete example of using Tiles support to share a common UI component across page flows.