TilesLikeViewComposition

Required reading

Most of this is already documented on the project home page of Clay. I'll try to concentrate on this specialized trail.

Just to introduce you to the examples below: they use an HTML template as base layout for Tiles-like view composition based on replacement variables (called "symbols" in Clay). Following is the code of the "/layouts/basicLayout.html" (notice the difference in defining plain text replacement variables - `@title` - and template replacement references - `<span jsfid="clay" clayJsfid="@bodycontent">`).

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "DTD/xhtml1-strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
<title>@title</title>
...
<span jsfid="clay" clayJsfid="@bodycontent">Dummy body - notice the difference in defining plain text
replacement variables and template replacement references</span>
...
</html>
```

Tiles-like forwards to full HTML view compositions

Something very comfortable in Tiles (and other integration libraries, e.g. MyFaces' JspTilesViewHandler) is that you just have to forward to a certain view id to get the view composite defined in some configuration file. The page definition of the desired view composition is located ONLY in the configuration file - a real HTML or JSP template is not needed.

Classic full HTML view approach

With the full HTML view approach of Clay, this is not possible - you always have to have a physical file with the .html suffix representing the given view id. That means, for a forward to "/my_view_composition.html", you have to have a file named "my_view_composition.html" in your document root, even if you have defined a view composition in the corresponding clay-config.xml. Assuming you have the following components defined in your clay-config.xml:

```
<!-- base layout definition  -->
<component jsfid="basePage" extends="clay">
        <attributes>
                <set name="clayJsfid" value="/layouts/basicLayout.html" />
        </attributes>
        <symbols>
                <set name="@title" value="Default Title" />
                <set name="@bodycontent" value="space" />
                <set name="@footercontent" value="footerPanel" />
        </symbols>
</component>

<!-- page definition -->
<component jsfid="my_view_composition" extends="basePage">
        <symbols>
                <set name="@title" value="Test page for page composition" />
                <set name="@bodycontent" value="/content/page1.html" />
        </symbols>
</component>
```

You reference this view composition definition in a HTML template file "my_view_composition.html" with the following content:

```
<html jsfid="my_view_composition" allowBody="false">
        Dummy Body
</html>
```

For every view composition definition you have to create a "dummy" HTML template - not very clean.

Combining with full XML view facility

The (or one?) solution is provided by the full XML view capability of Clay. When using XML views, Clay allows the components resp. view compositions to be defined ONLY in the configuraton file. The defined "jsfid" attribute must reflect the real JSF view id, but that's no problem. The only thing that changes is the suffix of the JSF view id - it must be ".xml" or something different from the suffix defined for HTML views.

The following steps have to be taken to implement the solution:

1. Create a separate "clay-pages-config.xml" containing the view composition definitions. We'll re-use our example from above. The "basePage" definition is the same, only the "jsfid" of the view composition definition has changed:

```
...
<!-- page definition -->
<component jsfid="/my_view_composition.page" extends="basePage">
        <symbols>
                <set name="@title" value="Test page for page composition" />
                <set name="@bodycontent" value="/content/page1.html" />
        </symbols>
</component>
```

2. Change the default suffix for XML views in web.xml (As an example we use ".page" here to distinguish from the plain XML approach because we still use full HTML templates. Of course you can use the default suffix ".xml", but then you have to change the "jsfid" from above accordingly.)

```
...
<context-param>
        <param-name>org.apache.shale.clay.XML_TEMPLATE_SUFFIX</param-name>
        <param-value>.page</param-value>
</context-param>
...
```

3. Specify the configuration file containing the view composition definitions in web.xml

```
...
<context-param>
        <param-name>org.apache.shale.clay.FULLXML_CONFIG_FILES</param-name>
        <param-value>/WEB-INF/clay-pages-config.xml</param-value>
</context-param>
...
```

4. Add the new suffix to the FacesServlet mapping and possible other JSF filters (e.g. MyFaces' ExtensionsFilter)

```
...
<servlet-mapping>
        <servlet-name>FacesServlet</servlet-name>
        <url-pattern>*.page</url-pattern>
</servlet-mapping>
...
```