UsingPortletModes

How to extend MyFaces portlets to use modes

While JSF has good support for portlets, the JSF 1.2 spec does not address the use of portlet modes in a JSF application. The JSR-168 portlet spec specifies that a portlet container (a portal) must at least support VIEW, EDIT, and HELP modes. By default, modes are ignored in MyFaces and all requests are treated the same regardeless of the current mode.

At its lowest level, the only difference in the modes is in what is returned when someone calls PortletRequest.getPortletMode(). javax.portlet.GenericPortlet provides a concrete Portlet class that dispatches to different render methods based on which mode is active. These methods are doView(), doEdit(), and doHelp(). Because MyFacesGenericPortlet extends javax.portlet.GenericPortlet, these three methods can be overridden to provide custom rendering behavior depending on the current mode.

If you want to provide custom rendering behavior based on the mode then you will want to "jump" to a new view whenever the mode changes. Because JSF does not know anything about modes, you can not use normal JSF navigation. However, JSF spec section 2.1.1 provides a way to do this navigation by treating the "jump" as a non-faces request needing a faces response. The MyFacesGenericPortlet method nonFacesRequest() implements the sequence of events specified in 2.1.1 which makes this "jump" easy to accomplish.

Note 1: This technique is somewhat different from the one I presented at JavaOne. It is untested, but it should work and be cleaner. The fact is, there are several ways to get this to work depending on how your application is structured. You just need to understand how to use the MyFacesGenericPortlet methods nonFacesRequest() and facesRender(). Any way you cut it, this will be a bit of a hack until mode support is added to the JSF specification.

Note 2: This will not work with MyFaces 1.0.9. Until the next release, you will need to build MyFaces from head in order to have the latest MyFacesGenericPortlet changes.

Step 1: Extend MyFacesGenericPortlet

```
public class MyOwnPortlet extends MyFacesGenericPortlet
```

Step 2: Detect that the mode has changed

```
// override the render() method from GenericPortlet to detect a
// mode change
public void render(RenderRequest request, RenderResponse response)
            throws PortletException, IOException {
   PortletSession session = request.getPortletSession();
   PortletMode mode = (PortletMode)session.getAttribute("CurrentPortletMode");

   if (mode == null) {
      mode = request.getPortletMode();
   }

   if (!mode.equals(request.getPortletMode())) {
      request.setAttribute("isPortletModeChanged", Boolean.TRUE);
   } else {
      request.setAttribute("isPortletModeChanged", Boolean.FALSE);
   }

   session.setAttribute("CurrentPortletMode", request.getPortletMode());
   super.render(request, response);
}
```

Step 3: Jump to a new view if the mode changed to Edit, otherwise continue normally

```
protected void doEdit(RenderRequest request, RenderResponse response)
            throws PortletException, IOException {

   Boolean isPortletModeChanged = (Boolean)request.getAttribute("isPortletModeChanged");
   if (isPortletModeChanged.booleanValue()) {
      setPortletRequestFlag(request);
      nonFacesRequest(request, response, "/MainEditPage.jsp");
      return;
   }

   facesRender(request, response);
}
```

Step 4: Specify this Portlet class in portlet.xml

```
<portlet-class>my.package.MyOwnPortlet</portlet-class>
```