PortalPageLabels

Portal Page Labels

The Cocoon Portal operates by processing events. The default implementation of the portal exposes these events in the links returned to the client browser. This creates several issues in developing websites:

1. The events are different depending on the page being viewed. Thus, having a link to event 1 has different meanings depending on what the last request was.
2. Because events are relative to the last request the browser back button cannot work properly.
3. Since events don't exist before a request has been made it is not possible to maintain bookmarks to a site. This has been partially addressed by the Bookmark action (see below).

The bookmark action helps with the third item, but it requires its own configuration file separate from the portal layout and requires redirects to work.

Portal page labels are a different approach to solving all the problems listed above. Through the addition of some new components, configured in cocoon.xconf, events for all the navigation items defined in the layout portal.xml are generated when the user first accesses the portal. These events are associated with keys constructed using the names specified in the layout portal.xml. When a link is requested for a navigation item the link will contain a reference to that key. Thus, the generated links will be consistent for as long as the portal.xml file is. Also, because all the navigation events are generated at the first request, bookmarked links will work.

For example, if portal.xml contains:

```
<composite-layout name="tab" id="maintab">
  <named-item name="Main">
    <coplet-layout name="coplet">
      <coplet-instance-data>Portal-Demo-1</coplet-instance-data>
    </coplet-layout>
  </named-item>
  <named-item name="TabDemo1">
    <composite-layout name="tab">
      <named-item name="One">
        <coplet-layout name="coplet">
          <coplet-instance-data>CZ Weblog-1</coplet-instance-data>
        </coplet-layout>
      </named-item>
      <named-item name="Two">
        <coplet-layout name="coplet">
          <coplet-instance-data>ML Weblog-1</coplet-instance-data>
        </coplet-layout>
      </named-item>
    </composite-layout>
  </named-item>

  <named-item name="TabDemo2">
    <composite-layout name="linktab">
      <named-item name="One">
        <coplet-layout name="coplet">
          <coplet-instance-data>CZ Weblog-1</coplet-instance-data>
        </coplet-layout>
      </named-item>
      <named-item name="Two">
        <coplet-layout name="coplet">
          <coplet-instance-data>ML Weblog-1</coplet-instance-data>
        </coplet-layout>
      </named-item>
    </composite-layout>
  </named-item>
</composite-layout>
```

The generated links would be:

```
portal?pageLabel=Main
portal?pageLabel=TabDemo1
portal?pageLabel=TabDemo1.One
portal?pageLabel=TabDemo1.Two
portal?pageLabel=TabDemo2
portal?pageLabel=TabDemo2.One
portal?pageLabel=TabDemo2.Two
```

Another feature of interest is that all non-navigation events still will be appended to the generated link, but the link will also contain the page label the event is for. In other words, events are no longer relative to the last request, but to the last request for a specific page label. Thus, the back button now can be used to return to a prior page as the events for that page will still exist. When the back button will cause a change to a different page than what is currently selected, the appropriate events will occur to cause a switch to that page and any other events attached to the link will be ignored.

Getting the Page Label components

Support for Page Labels was added to version 2.1.6 and in the 2.2 development branch.

Configuring Page Labels

1. Build Cocoon.
2. Modify cocoon.xconf:
    a. Add an aspect definition for P'***ageLabelEventAspect to the list of the list of aspects in the component with the role named org.apache.cocoon.portal.event.aspect.E***'ventAspectSelector.
    b. Remove the action-counter aspect from the list of aspects in the event manager.
    c. Add the page-label aspect to the list of aspects in the event manager. The list of aspects should now look like:

```
<event-aspects>
  <aspect type="frame"/>
  <aspect type="link"/>
  <aspect type="full-screen-coplet"/>
  <aspect type="page-label"/>
  <aspect type="request-parameter"/>
</event-aspects>
```

1. Change the class for the L'***inkService from D***'efaultLinkService to P'*_ageLabelLinkService.
2. Change the class for the E_**'ventConverter from D'***efaultEventConverter to P***'ageLabelEventConverter.
3. Add a component definition for the P'*_ageLabelManager

After completing these tasks cocoon.xconf should contain the following:

```
<!-- Event Aspect configuration -->
<component class="org.apache.cocoon.components.ExtendedComponentSelector"
           role="org.apache.cocoon.portal.event.aspect.EventAspectSelector">
  <aspect class="org.apache.cocoon.portal.event.aspect.impl.ActionCounterEventAspect"
          name="action-counter"/>
  <aspect class="org.apache.cocoon.portal.event.aspect.impl.RequestParameterEventAspect"
          name="request-parameter"/>
  <aspect class="org.apache.cocoon.portal.event.aspect.impl.FrameEventAspect"
          name="frame"/>
  <aspect class="org.apache.cocoon.portal.event.aspect.impl.LinkEventAspect"
          name="link"/>
  <aspect class="org.apache.cocoon.portal.event.aspect.impl.PageLabelEventAspect"
          name="page-label"/>
  <aspect class="org.apache.cocoon.portal.event.aspect.impl.FullScreenCopletEventAspect"
          name="full-screen-coplet"/>
  <!-- This aspect sets headers on the response that tell the client to not cache the response: -->
  <aspect class="org.apache.cocoon.portal.event.aspect.impl.NoClientCachingEventAspect"
          name="no-client-caching"/>
</component>
<component class="org.apache.cocoon.portal.event.impl.DefaultEventManager"
           logger="portal" role="org.apache.cocoon.portal.event.EventManager">
  <event-aspects>
    <aspect type="frame"/>
    <aspect type="link"/>
    <aspect type="full-screen-coplet"/>
    <aspect type="page-label"/>
    <aspect type="request-parameter"/>
  </event-aspects>
  <!-- add a new instance of each class as a subscriber: -->
  <subscriber-classes>
    <class name="org.apache.cocoon.portal.event.subscriber.impl.DefaultChangeAspectDataEventSubscriber"/>
    <class name="org.apache.cocoon.portal.event.subscriber.impl.DefaultJXPathEventSubscriber"/>
    <class name="org.apache.cocoon.portal.event.subscriber.impl.DefaulCopletDataEventSubscriber"/>
  </subscriber-classes>
  <!-- add each component as a subscriber (the component should be thread safe): -->
  <subscriber-roles>
    <!-- <role name="AVALON-ROLE"/> -->
  </subscriber-roles>
</component>

<component class="org.apache.cocoon.portal.impl.PageLabelLinkService" logger="portal"
            role="org.apache.cocoon.portal.LinkService"/>

<component class="org.apache.cocoon.portal.impl.PageLabelManager" logger="portal"
           role="org.apache.cocoon.portal.impl.PageLabelManager">
   <nonStickyTabs>true</nonStickyTabs>
</component>
<component class="org.apache.cocoon.portal.event.impl.PageLabelEventConverter"
           logger="portal" role="org.apache.cocoon.portal.event.EventConverter">
</component>
```

The P_*'ageLabelManager can be configured. The child elements it supports are:

parameterName - Identifies the request parameter to use to identify the page label. The default value is “pageLabel”.

aspectName - Specifies the aspect name to use when creating events. This must be the same value used when configuring the T'***abbedContentAspect. It is used along with the layout name and the layout hashcode to construct the key to use when saving the value that identifies which item has been selected in a layout. The default value is “tab”. The specified name must not conflict with any other keys stored using the same layout. It is possible that individual T***'abbedContentAspects can be configured with different aspect names. However, since the P'***ageLabelManager can only be configured with a single aspectName, all T***'abbedContentAspect instances must be configured with the same value.

nonStickyTabs - Specifies whether sticky tabs are used. This is described in the following section. The default value is “false”.

marshallEvents - Specifies whether event marshalling is to occur (currently only applies to JSR-168 portlet url events). If set to true marshalling will occur. See the section on JSR-168 Portlet URLs below.

Sticky Tabs

The Cocoon Portal supports pages with sub-navigation items on them. If a sub-navigation item is selected, the default behavior of the portal whenever the main navigation for the page is selected is to return to the last sub-navigation item that was selected. The Page Label Manager allows this behavior to be changed so that whenever a main navigation tab is selected the first sub-navigation item is displayed. This is accomplished by setting the nonStickyTabs element to true in the Page Label Manager's configuration.

JSR-168 Portlet URLs

JSR-168 defines two kinds of URLs; Action URLs and Render URLs. Pluto processes Action URLs by issuing a redirect. Of course, this causes a second invocation of the Portal for the same initial request. The implication of this is that any events created for the initial request are gone when control is returned to the browser. Thus, simply refreshing the screen can cause unexpected behavior. The only good solution to solve this is not to return event ids to the browser for Portlet URLs. Support to do this has been added in version 2.1.7.

Configuration

1. Portal P'*_ageLabels must be configured as described above.
2. Modify cocoon.xconf.
    a. Add an aspect definition for C_**'onvertableEventAspect to the list of the list of aspects in the component with the role named org.apache.cocoon.portal.event.aspect.E'**_ventAspectSelector and type "convertable".
    b. Add the convertable aspect to the list of aspects in the event manager.
    c. Add a paramter defintion with name "parameter-name" and value "cocoon-portal-event,url". The list of aspects should now look like:

```
  <event-aspects>
    <aspect type="convertable"/>
    <aspect type="frame"/>
    <aspect type="link"/>
    <aspect type="full-screen-coplet"/>
    <aspect type="page-label"/>
    <aspect type="request-parameter">
      <parameter name="parameter-name" value="cocoon-portal-event,url"/>
    </aspect>
  </event-aspects>
```

1. Add a new component using class org.apache.cocoon.components.E_**'xtendedComponentSelector and role org.apache.cocoon.portal.event.C'**_onvertableEventFactorySelector. Within this component define factories as:

```
<component class="org.apache.cocoon.components.ExtendedComponentSelector" role="org.apache.cocoon.portal.event.
ConvertableEventFactorySelector">
  <factory name="cocoon-portal-fs" class="org.apache.cocoon.portal.event.impl.FullScreenCopletEventFactory"/>
  <factory name="url" class="org.apache.cocoon.portal.pluto.PortletURLProviderFactory"/>
</component>
```

1. Configure the P_*'ageLabelManager to enable event marshalling by specifying:

```
  <marshallEvents>true</marshallEvents>
```

Note

In release 2.1.7 the sample portal contains the configuration listed in the preceding sections. However, most of the features are disabled by default. Comments have been placed in the configuration where needed to indicate how to enable these features.