PortalEngineBookmarks

Q: How does the bookmarks feature work?

A: You have a file, typically called bookmarks.xml, which contains definitions for a number of different portal events. When you use the portal-bookmark action, it decodes the parameters you have passed in for each event, and generates a URL which will create the events. An example: lets say you have two nested tab layouts, and you want to switch to the first tab on both of them and then pass a value to one of the coplets. You would create a bookmarks file like this:

```
<bookmarks>
  <events>
    <event type="jxpath" id="top">
      <targettype>layout</targettype>
      <targetid>maintab</targetid>
      <path>aspectDatas/tab</path>
    </event>

    <event type="jxpath" id="inner">
      <targettype>layout</targettype>
      <targetid>leftnav-home</targetid>
      <path>aspectDatas/tab</path>
    </event>

    <event type="jxpath" id="coplet1">
      <targettype>coplet</targettype>
      <targetid>Coplet-1</targetid>
      <path>attributes/appid</path>
    </event>
  </events>
</bookmarks>
```

The significance of each part is:

`<event type="jxpath" id="top">` Create a jxpath event (the only type you can create at the moment) with an id of `top`.

`<targettype>layout</targettype>` The target for the event is a `layout` (the other alternative is `coplet`).

`<targetid>maintab</targetid>`: The layout you are targeting is called `maintab`. This refers to an entry in your portal layout xml file:

```
<composite-layout name="toptab" id="maintab">
  <named-item name="navbar.home">
    <composite-layout name="leftnav" id="leftnav-home">
    [...]
```

`<path>aspectDatas/tab</path>` set the value of the `aspectDatas/tab` property of the target layout (i.e. switch to that tab).

(Tab indexes for layouts are 0-based).

So you can use a URL like this:

`bookmark?top=0&inner=0&coplet1=123`

The bookmark matcher looks like this:

```
    <map:match pattern="bookmark">
        <map:act type="auth-protect">
            <map:parameter name="handler" value="portalhandler"/>
            <map:parameter name="application" value="portal"/>

            <map:act type="portal-bookmark">
                    <map:parameter name="portal-name" value="portal" />

                    <map:redirect-to uri="portal?{uri}"/>
            </map:act>
        </map:act>
    </map:match>
```

The URL is inspected by the portal-bookmark action, which creates a set of events which are available in the sitemap with the {uri} placeholder. So the matcher ends by redirecting to that URL.

Q: Great, but the URLs I have to use are still pretty big. Can I make it so that a url like `app?id=123` will switch to the right page?

A: Well, you could use a matcher.

```
<!-- Handle app?id=123 by redirecting to a
     bookmark URL which does the right thing -->
<map:match pattern="app">
  <map:redirect-to uri="bookmark?top=0&amp;inner=0&amp;coplet1={request-param:id}"/>
</map:match>
```

That does the trick, but of course your user then has to wait for the initial redirect to the `bookmark` url, then the redirect from that one to the `portlet` url.

Q: OK, so how do I make it so that `app?id=123` just returns the page without needing the two redirects?

A: Here's one way - you handle it all in the sitemap, and just serve up the resulting page to the user.

```
<map:match pattern="app">
  <map:redirect-to uri="cocoon:/bookmark?top=0&amp;inner=0&amp;coplet1={request-param:id}"/>
</map:match>
```

Using that technique, the user is left with `app?id=123` in the URL bar of their browser, which looks a bit nicer than the great big list of attributes that you get after the redirect.

Back to PortalEngine

JonEvans