ShaleViewController

Overview

The Shale ViewController functionality allows a managed bean to get callbacks on a per-view (ie per-page) basis, in contrast to the normal callbacks invoked per-component.

Configuration

The jsf configuration files included in the shale-core jarfile automatically registers the necessary shale objects with the JSF implementation, so no changes need to be made to web.xml; just placing the shale-core jarfile in the webapp's WEB-INF/lib directory is sufficient. (NOTE – as of version 1.0.4, this logic will be factored out into the new shale-view jarfile, and this refactoring exists in current nightly buids.)

The ViewHandler class implements the JSF standard ViewHandler class. It is installed as a wrapper around the ViewHandler implementation provided by the underlying JSF implementation. It is this class that invokes the setPostBack method on the ViewController.

In order to tell shale's view controller framework which bean to call for each page, it is necessary to define a ViewControllerMapper strategy which creates a managed bean name from the value returned by UIViewRoot.getViewId(), ie from the portion of the page url between the webapp name and the query params. An instance of DefaultViewControllerMapper is used by default; see the documentation for that class for details of the mapping. The managed bean with that name is expected to implement the ViewController interface (but see comments on Tiger below).

It is possible to override the default ViewControllerMapper simply by defining an application-scoped managed bean with name "org\$apache\$shale\$view\$VIEW_MAPPER" that implements the ViewControllerMapper interface.

Subviews

In the Shale 1.0.3 release (latest at the current date) there is no support for callbacks for subviews. This means that even if a page is composed of multiple other pages (via jsp:include or other mechanism), all page-related callbacks must go through a single managed bean which introduces undesirable coupling.

The trunk code as at 2006-11-07 contains some code that appears to support a separate ViewController managed bean for each f:subview in the page by overriding the renderer for subviews to invoke preprocess on processDecodes and prerender on encodeBegin. (NOTE - as of version 1.0.4, subview lifecycle events will be fully supported, and this support exists in current nightly builds.)

Tiger

If the Shale Tiger libraries are present, then a few more features are available. Tiger is specifically for java 1.5 and allows the use of annotations on ordinary managed beans to mark the view-controller callback methods rather than requiring the managed bean to implement the ViewController interface.

Implementation Classes

- org.apache.shale.view.faces.ViewController defines the callbacks available; the managed bean needs to implement this interface in order to get callbacks. Alternately, concrete subclass AbstractViewController can be subclassed.
- org.apache.shale.view.faces.LifecycleListener invokes the init and destroy callbacks.
- ViewViewHandler wraps the standard ViewHandler and whenever a view is created (eg restore or forward) it determines which managed bean corresponds to this view and caches this in the current request.
- ViewPhaseListener invokes the preprocess/prerender methods on the ViewController managed bean located earlier by the ViewViewHan dler.