ExceptionHandling

If you use the 1.0.3 release (or a very recent daily build), the behavior of handling exceptions thrown by an action event is now programmable, in several different ways.

By default, exceptions thrown from lifecycle callback methods (init, destroy, etc.), or from the action method connected to a command link or command button, will still be accumulated.

You can change this by defining a class that implements org.apache.shale.view.ExceptionHandler and storing it in application scope under the key defined by FacesConstants.EXCEPTION_HANDLER (the literal value is "org$apache$shale$view$EXCEPTION_HANDLER").

During a phase listener that is invoked after the Invoke Application phase (i.e. just after your command action has returned a logical outcome), it is checked whether there have been any exceptions accumulated by the default handler above.

If so, it can optionally do a RequestDispatcher.forward() call to the context relative path of an error display page for your application. You can configure such a path in web.xml with a context init parameter like this:

```
<context-param>
    <param-name>org.apache.shale.view.EXCEPTION_DISPATCH_PATH</param-name>
    <param-value>/exception-viewer.faces</param-value>
</context-param>
```

This page will be displayed instead of the one that would normally be selected by your navigation rules.

In addition, it will receive request attributes containing interesting facts about the error that occurred, just like an error page you provide to your servlet container as an exception handler. The most interesting one is an attribute named "javax.servlet.error.exception", which will receive a Shale ApplicationException that lists all the exceptions that have occurred for this request.

Can I turn off Shale's exception handling?

No. The exception handler is locked together with the view controller functionality for two reasons:

- Without it, Shale could not fulfill its contract to call destroy() if init() was ever called, when the application throws an exception.
- JSF implementations tend to eat exceptions anyway (for a similar sort of reason – to guarantee that they can call the afterPhase() listeners if beforePhase() was called).

However, you can replace the default exception handler with one of your own. (See above.)

References

- http://mail-archives.apache.org/mod_mbox/shale-user/200609.mbox/%3cf8b39ace0609031622n2ff712e8mf780bc018a820aa0@mail.gmail.com%3e
- http://www.nabble.com/How-to-turn-of-Shale-ExeptionHandling--t2371987.html