Scope Interceptor

{snippet:id=description|javadoc=true|url=org.apache.struts2.interceptor.ScopeInterceptor}

Parameters

{snippet:id=parameters|javadoc=true|url=org.apache.struts2.interceptor.ScopeInterceptor}

Extending the Interceptor

{snippet:id=extending|javadoc=true|url=org.apache.struts2.interceptor.ScopeInterceptor}

Examples

{snippet:id=example|lang=xml|javadoc=true|url=org.apache.struts2.interceptor.ScopeInterceptor}

Some more examples

The scope interceptor can be used to pass arbitrary objects from one action ActionA to another other ActionB, provided you have a getter in ActionA and and a similar setter in actionB. Also, you should use a key parameter to make sure you tell ASF/WW which action gets which objects. This allows you to mix several actions with several scopes, without running the risk of getting wrong objects.

xml <action name="scopea" class="com.mevipro.test.action.ScopeActionA"> <result name="success" type="dispatcher">/jsp/test.jsp</result> <interceptor-ref name="basicStack"/> <interceptor-ref name="scope"> <param name="key">funky</param> <param name="session">person</param> <param name="autoCreateSession">true</param> </interceptor-ref> </action> <action name="scopeb" class="com.mevipro.test.action.ScopeActionB"> <result name="success" type="dispatcher">/jsp/test.jsp</result> <interceptor-ref name="scope"> <param name="key">funky</param> <param name="session">person</param> <param name="autoCreateSession">true</param> </interceptor-ref> <interceptor-ref name="basicStack"/> </action>

Don't forget: you'll need at least a getPerson() getter in ScopeActionA and a setPerson(Person person) setter in ScopeActionB, Ã¡nd you need to make sure you specify the key (you don't need this if you only use one action, as in the example above). Without the key, the scope interceptor will store your variables, but won't set them on the other action.