**“WorkAround” para o problema da necessidade de dois clicks para acionar a ação de CommandButton, CommandLink**

Descrição do Problema

This problem is known as [JSF spec issue 790](http://java.net/jira/browse/JAVASERVERFACES_SPEC_PUBLIC-790). If you ajax-render some content which in turn contains a <h:form>, then its view state will get lost, which causes that the 1st action inside that form won't invoke anything. On the ajax response of the first action, the form will get new view state and thus any subsequent actions will succeed.

This is scheduled to be fixed for the upcoming JSF 2.2. But right now with JSF 2.0/2.1 you have to introduce a workaround. You first need to give the <h:form> a fixed ID which is yourFormId.

Fonte:

<http://stackoverflow.com/questions/10094615/a4jcommandbutton-action-is-only-invoked-on-second-click>

**Utilização do framework Omnifaces:**

Fonte:   
<http://showcase.omnifaces.org/scripts/FixViewState>

## - FixViewState

ui-buttonui-button

Available since OmniFaces 1.0

This script fixes the JSF view state if necessary. In Mojarra, it get lost on certain forms during certain ajax updates (e.g. [updating content which in turn contains another form](http://balusc.omnifaces.org/2011/09/communication-in-jsf-20.html" \l "AjaxRenderingOfContentWhichContainsAnotherForm)). Normally, the jsf.js script adds the JSF view state back to the updated form, but it does that only for the enclosing form, not for other forms when they are not explicitly specified in the render/update attribute. When this script is loaded after the standard jsf.js script containing standard JSF ajax API, then it will be automatically applied on all JSF ajax requests.

<h:outputScript library="javax.faces" name="jsf.js" target="head" />

<h:outputScript library="omnifaces" name="fixviewstate.js" target="head" />

This script also recognizes jQuery ajax API as used by some component libraries such as PrimeFaces. If jQuery is present, then it will also be automatically applied on all jQuery ajax requests.

<h:outputScript library="primefaces" name="jquery/jquery.js" target="head" />

<h:outputScript library="omnifaces" name="fixviewstate.js" target="head" />

Explicit declaration of jsf.js or jquery.js via <h:outputScript> is not strictly necessary. You only need to make sure that the <h:outputScript> on fixviewstate.js is being placed inside the <h:body> so that you can ensure that it's loaded after the JSF and/or jQuery script.

<h:head>

...

</h:head>

<h:body>

<h:outputScript library="omnifaces" name="fixviewstate.js" target="head" />

...

</h:body>

In case your JSF component library doesn't utilize standard JSF nor jQuery ajax API, but a proprietary one, and exposes the same "missing view state" problem, then you can still apply this script manually during the "complete" event of the ajax request whereby the concrete XMLHttpRequest instance is available as some argument as follows:

function someOncompleteCallbackFunction(xhr) {

OmniFaces.FixViewState.apply(xhr.responseXML);

});

This was scheduled to be fixed in JSF 2.2 spec, however it was postponed to JSF 2.3. We can't wait any longer, so it's been added to OmniFaces. Note that this fix is not necessary for MyFaces as they have internally already fixed it for long in their jsf.js.

In the below demo, try to subsequently press the both buttons in turn as long until a faces message shows up. When the fixviewstate.js is not loaded, you'll notice that after the first time, you need to press the other button twice in order to see it. This problem is fixed by loading the fixviewstate.js.