Ajax and JavaScript Recipes

Dojo plugin is deprecated

The Dojo plugin will be deprecated on Struts 2.1

Prevent options from loading when page loads

Common

All examples on this page assume the following JSP fragment is on the same page as the example.

```
<%@ taglib prefix="s" uri="/struts-tags" %>
<%@ taglib prefix="sx" uri="/struts-dojo-tags" %>

<head>
    <sx:head />
</head>

<s:url id="url" value="/MyAction.action" />
```

Requests

Request is triggered by a topic

```
<s:submit value="Make Request" onclick="dojo.event.topic.publish('/request')" />
<sx:bind listenTopics="/request" href="%{#url}" />
```

Attached to an event

```
<s:submit value="Make Request" id="submit" />
<sx:bind sources="submit" events="onclick" href="%{#url}" />
```

Attached to an event on multiple sources

```
<s:submit value="Make Request" id="submit0" />
<s:submit value="Make Request" id="submit1" />
<sx:bind sources="submit0,submit1" events="onclick" href="%{#url}" />
```

Attached to multiple events on multiple sources

```
<s:textarea id="area0" />
<s:textarea id="area1" />
<sx:bind sources="area0,area1" events="onfocus,onchange" href="%{#url}" />
```

Update target element with content returned from url

```
<s:div id="div" />

<!-- With a bind tag -->
<s:submit value="Make Request" id="submit" />
<sx:bind targets="div" sources="submit" events="onclick" href="%{#url}" />

<!-- With a submit tag -->
<sx:submit targets="div" value="Make Request" href="%{#url}" />

<!-- With an anchor tag -->
<sx:a targets="div" value="Make Request" href="%{#url}" />
```

Update multiple target elements with content returned from url

```
<s:div id="div0" />
<s:div id="div1" />

<!-- With a bind tag -->
<s:submit value="Make Request" id="submit" />
<sx:bind targets="div0,div1" sources="submit" events="onclick" href="%{#url}" />

<!-- With a submit tag -->
<sx:submit targets="div0,div1" href="%{#url}" />

<!-- With an anchor tag -->
<sx:a targets="div0,div1" href="%{#url}" />
```

Show indicator while request is in progress

```
<img id="indicator" src="${pageContext.request.contextPath}/images/indicator.gif" style="display:none" />

<!-- With a bind tag -->
<s:submit value="Make Request" id="submit" />
<sx:bind indicator="indicator" sources="submit" events="onclick" href="%{#url}" />

<!-- With a submit tag -->
<sx:submit indicator="indicator" href="%{#url}" />

<!-- With an anchor tag -->
<sx:a indicator="indicator" href="%{#url}" />
```

Highlight content of target with blue color, for 2 seconds

```
<s:div id="div" />

<!-- With a bind tag -->
<s:submit value="Make Request" id="submit" />
<sx:bind highlightColor="blue" highlightDuration="2000" targets="div" sources="submit" events="onclick" href="%
{#url}" />

<!-- With a submit tag -->
<sx:submit highlightColor="blue" highlightDuration="2000" targets="div" href="%{#url}" />

<!-- With an anchor tag -->
<sx:a highlightColor="blue" highlightDuration="2000" targets="div" href="%{#url}" />
```

Execute JavaScript in the returned content

```
<s:div id="div" />

<!-- With a bind tag -->
<s:submit value="Make Request" id="submit" />
<sx:bind executeScripts="true" targets="div" sources="submit" events="onclick" href="%{#url}" />

<!-- With a submit tag -->
<sx:submit executeScripts="true" targets="div" href="%{#url}" />

<!-- With an anchor tag -->
<sx:a executeScripts="true" targets="div" href="%{#url}" />
```

Publish a topic before the request

```
<script type="text/javascript">
dojo.event.topic.subscribe("/before", function(event, widget){
   alert('inside a topic event. before request');
   //event: event object
   //widget: widget that published the topic
});
</script>

<!-- With a bind tag -->
<s:submit value="Make Request" id="submit" />
<sx:bind beforeNotifyTopics="/before" sources="submit" events="onclick" href="%{#url}" />

<!-- With a submit tag -->
<sx:submit beforeNotifyTopics="/before" href="%{#url}" />

<!-- With an anchor tag -->
<sx:a beforeNotifyTopics="/before" href="%{#url}" />
```

Publish a topic after the request

```
<script type="text/javascript">
dojo.event.topic.subscribe("/after", function(data, request, widget){
   alert('inside a topic event. after request');
   //data : text returned from request
   //request: XMLHttpRequest object
   //widget: widget that published the topic
});
</script>

<!-- With a bind tag -->
<s:submit value="Make Request" id="submit" />
<sx:bind afterNotifyTopics="/after" sources="submit" events="onclick" href="%{#url}" />

<!-- With a submit tag -->
<sx:submit afterNotifyTopics="/after" href="%{#url}" />

<!-- With an anchor tag -->
<sx:a afterNotifyTopics="/after" href="%{#url}" />
```

Publish a topic on error

```
<script type="text/javascript">
dojo.event.topic.subscribe("/error", function(error, request, widget){
   alert('inside a topic event. on error');
   //error : error object (error.message has the error message)
   //request: XMLHttpRequest object
   //widget: widget that published the topic
});
</script>

<!-- With a bind tag -->
<s:submit value="Make Request" id="submit" />
<sx:bind errorNotifyTopics="/error" sources="submit" events="onclick" href="%{#url}" />

<!-- With a submit tag -->
<sx:submit errorNotifyTopics="/error" href="%{#url}" />

<!-- With an anchor tag -->
<sx:a errorNotifyTopics="/error" href="%{#url}" />
```

Show a fixed error message on error

```
<div id="div" />

<!-- With a bind tag -->
<s:submit value="Make Request" id="submit" />
<sx:bind errorText="Error Loading" targets="div" sources="submit" events="onclick" href="%{#url}" />

<!-- With a submit tag -->
<sx:submit errorText="Error Loading" targets="div" href="%{#url}" />

<!-- With an anchor tag -->
<sx:a errorText="Error Loading" targets="div" href="%{#url}" />
```

Prevent a request

```
<script type="text/javascript">
dojo.event.topic.subscribe("/before", function(event, widget){
   alert('I will stop this request');
   event.cancel = true;
});
</script>


<!-- With a bind tag -->
<s:submit value="Make Request" id="submit" />
<sx:bind beforeNotifyTopics="/before" sources="submit" events="onclick" href="%{#url}" />

<!-- With a submit tag -->
<sx:submit beforeNotifyTopics="/before" href="%{#url}" />

<!-- With an anchor tag -->
<sx:a beforeNotifyTopics="/before" href="%{#url}" />
```

Submit a form (plain form)

```
<form id="form">
  <input type=textbox name="data">
</form>

<!-- With a bind tag -->
<s:submit value="Make Request" id="submit" />
<sx:bind formId="form" sources="submit" events="onclick" href="%{#url}" />

<!-- With a submit tag -->
<sx:submit formId="form" href="%{#url}" />

<!-- With an anchor tag -->
<sx:a formId="form" href="%{#url}" />
```

Submit a form (using s:form tag)

```
<!-- With a submit tag -->
<s:form namespace="/mynamespace" action="MyAction">
  <input type=textbox name="data">
  <sx:submit />
</s:form>

<!-- With an anchor tag -->
<s:form namespace="/mynamespace" action="MyAction">
  <input type=textbox name="data">
  <sx:a />
</s:form>
```

Div

Loads its content after page is loaded

```
<sx:div href="%{#url}">
  Initial Content
</sx:div>
```

Reloads content when topic is published

```
<sx:div href="%{#url}" listenTopics="/refresh">
  Initial Content
</sx:div>

<s:submit value="Refresh" onclick="dojo.event.topic.publish('/refresh')" />
```

Updates its content every 2 seconds, shows indicator while loading content

```
<img id="indicator" src="${pageContext.request.contextPath}/images/indicator.gif" style="display:none"/>
<sx:div href="%{#url}" updateFreq="2000">
  Initial Content
</sx:div>
```

Loads its content after a delay of 2 seconds

```
<sx:div href="%{#url}" delay="2000">
  Initial Content
</sx:div>
```

Show some text while content is loaded

```
<sx:div href="%{#url}" loadingText="reloading" showLoadingText="true">
  Initial Content
</sx:div>
```

Fixed error message

```
<sx:div href="noaction" errorText="Error loading content">
  Initial Content
</sx:div>
```

Execute JavaScript in the returned content

```
<sx:div href="%{#url}" executeScripts="true">
  Initial Content
</sx:div>
```

Control refresh timer using topics

```
<sx:div href="%{#url}"
        listenTopics="/refresh"
        startTimerListenTopics="/startTimer"
        stopTimerListenTopics="/stopTimer"
        updateFreq="3000">
  Initial Content
</sx:div>

<s:submit value="Refresh" onclick="dojo.event.topic.publish('/refresh')" />
<s:submit value="Start refresh timer" onclick="dojo.event.topic.publish('/startTimer')" />
<s:submit value="Stop refresh timer" onclick="dojo.event.topic.publish('/stopTimer')" />
```

Date and Time picker

Date picker

```
<sx:datetimepicker name="picker" />
```

Time picker

```
<sx:datetimepicker type="time" name="picker" />
```

Set value from an String

```
<sx:datetimepicker value="%{'2007-01-01'}" name="picker" />
```

Set value from stack (value must evaluate to either a Date, Calendar, or an String that can be parsed using the formats defined in SimpleDateFormat, and RFC 3339)

```
<sx:datetimepicker value="date" name="picker" />
```

Set/Get value using JavaScript

```
<script type="text/javascript">
  function setValue() {
     var picker = dojo.widget.byId("picker");

     //string value
     picker.setValue('2007-01-01');

     //Date value
     picker.setValue(new Date());
  }

  function showValue() {
     var picker = dojo.widget.byId("picker");

     //string value
     var stringValue = picker.getValue();
     alert(stringValue);

     //date value
     var dateValue = picker.getDate();
     alert(dateValue);
  }
</script>

<sx:datetimepicker id="picker" />
```

Style the textbox

```
<sx:datetimepicker id="picker" cssStye="background:red" cssClass="someclass"/>
```

Publish topic when value changes

```
<script type="text/javascript">
  dojo.event.topic.subscribe("/value", function(text, date, widget){
      alert('value changed');
      //textEntered: String entered in the textbox
      //date: JavaScript Date object with the value selected
      //widget: widget that published the topic
  });
</script>

<sx:datetimepicker label="Order Date" valueNotifyTopics="/value"/>
```

Use other locales.

Locales must be specified in the sx:head tag.

```
<sx:head extraLocales="en-us,nl-nl,de-de" />

<sx:datetimepicker label="In German" name="dddp7" value="%{'2006-06-28'}" language="de-de" />
<sx:datetimepicker label="In Dutch"  name="dddp8" value="%{'2006-06-28'}" language="nl-nl" />
```

Tabbed Panel

Local Tabs

```
<sx:tabbedpanel id="tabContainer">
   <sx:div label="Tab 1" >
       Local Tab 1
   </sx:div>
   <sx:div label="Tab 2" >
       Local Tab 2
   </sx:div>
</sx:tabbedpanel>
```

Local and remote tabs

```
<sx:tabbedpanel id="tabContainer">
   <sx:div label="Local Tab 1" >
       Tab 1
   </sx:div>
   <sx:div label="Remote Tab 2" href="%{#url}">
       Remote Tab 2
   </sx:div>
</sx:tabbedpanel>
```

Fixed size (size does not adjust to current tab)

```
<sx:tabbedpanel cssStyle="width: 500px; height: 300px;" doLayout="true" id="tabContainer">
   <sx:div label="Tab 1" >
       Local Tab 1
   </sx:div>
   <sx:div label="Tab 2" >
       Local Tab 2
   </sx:div>
</sx:tabbedpanel>
```

Do not load tab 2 when page loads (it will be loaded when selected)

```
<sx:tabbedpanel id="tabContainer">
   <sx:div label="Remote Tab 1" href="%{#url}">
       Remote Tab 1
   </sx:div>
   <sx:div label="Remote Tab 2" href="%{#url}" preload="false">
       Remote Tab 1
   </sx:div>
</sx:tabbedpanel>
```

Reload tabs content when selected

```
<sx:tabbedpanel id="tabContainer">
   <sx:div label="Remote Tab 1" href="%{#url}" refreshOnShow="true">
       Remote Tab 1
   </sx:div>
   <sx:div label="Remote Tab 2" href="%{#url}" refreshOnShow="true">
       Remote Tab 2
   </sx:div>
</sx:tabbedpanel>
```

Disabled tabs

```
<sx:tabbedpanel id="tabContainer">
   <sx:div label="Tab 1" >
       Local Tab 1
   </sx:div>
   <sx:div label="Tab 2" disabled="true">
       Local Tab 2
   </sx:div>
</sx:tabbedpanel>
```

Enable/Disable tabs using JavaScript

```
<script type="text/javascript">
   function enableTab(param) {
      var tabContainer = dojo.widget.byId('tabContainer');
      tabContainer.enableTab(param);
   }


   function disableTab(param) {
      var tabContainer = dojo.widget.byId('tabContainer');
      tabContainer.disableTab(param);
   }
</script>

<sx:tabbedpanel id="tabContainer" id="tabContainer">
   <sx:div id="tab1" label="Tab 1">
       Local Tab 1
   </sx:div>
   <sx:div id="tab2" label="Tab 2" disabled="true">
       Local Tab 2
   </sx:div>
</sx:tabbedpanel>

<!-- By Tab Index -->
<input type="button" onclick="enableTab(1)" value="Enable Tab 2 using Index" />
<input type="button" onclick="disableTab(1)" value="Disable Tab 2 using Index" />

<!-- By Tab Id -->
<input type="button" onclick="enableTab('tab2')" value="Enable Tab 2 using Id" />
<input type="button" onclick="disableTab('tab2')" value="Disable Tab 2 using Id" />

<!-- By Widget -->
<input type="button" onclick="enableTab(dojo.widget.byId('tab2'))" value="Enable Tab 2 using widget" />
<input type="button" onclick="disableTab(dojo.widget.byId('tab2'))" value="Disable Tab 2 using widget" />
```

Set Tab labels position to bottom (can be: top, right, bottom, left)

```
<sx:tabbedpanel labelposition="bottom" id="tabContainer">
   <sx:div label="Tab 1" >
       Local Tab 1
   </sx:div>
   <sx:div label="Tab 2" >
       Local Tab 2
   </sx:div>
</sx:tabbedpanel>
```

Allow tab 2 to be removed(closed)

```
<sx:tabbedpanel id="tabContainer">
   <sx:div label="Tab 1" >
       Local Tab 1
   </sx:div>
   <sx:div label="Tab 2"  closable="true">
       Local Tab 2
   </sx:div>
</sx:tabbedpanel>
```

Publish topics when tab is selected

```
<script type="text/javascript">
   dojo.event.topic.subscribe('/before', function(event, tab, tabContainer) {
      alert("Before selecting tab");
   });

   dojo.event.topic.subscribe('/after', function(tab, tabContainer) {
      alert("After tab was selected");
   });
</script>
<sx:tabbedpanel beforeSelectTabNotifyTopics="/before" afterSelectTabNotifyTopics="/after" id="tabContainer">
   <sx:div label="Tab 1">
       Local Tab 1
   </sx:div>
   <sx:div label="Tab 2">
       Local Tab 2
   </sx:div>
</sx:tabbedpanel>
```

Select tab using JavaScript

```
<script type="text/javascript">
   function selectTab(id) {
     var tabContainer = dojo.widget.byId("tabContainer");
     tabContainer.selectTab(id);
   }
</script>
<sx:tabbedpanel id="tabContainer">
   <sx:div label="Tab 1" id="tab1">
       Local Tab 1
   </sx:div>
   <sx:div label="Tab 2" id="tab2">
       Local Tab 2
   </sx:div>
</sx:tabbedpanel>

<input type="button" onclick="selectTab('tab1')" value="Select tab 1" />
<input type="button" onclick="selectTab('tab2')" value="Select tab 2" />
```

Prevent tab 2 from being selected

```
<script type="text/javascript">
   dojo.event.topic.subscribe('/before', function(event, tab, tabContainer) {
      event.cancel = tab.widgetId == "tab2" ;
   });
</script>
<sx:tabbedpanel beforeSelectTabNotifyTopics="/before" id="tabContainer">
   <sx:div id="tab1" label="Tab 1">
       Local Tab 1
   </sx:div>
   <sx:div id="tab2" label="Tab 2">
       Local Tab 2
   </sx:div>
</sx:tabbedpanel>
```

Customize template css path (Dojo widget template css)

```
<sx:tabbedpanel templateCssPath="%{#cssUrl}" id="tabContainer">
   <sx:div id="tab1" label="Tab 1">
       Local Tab 1
   </sx:div>
   <sx:div id="tab2" label="Tab 2">
       Local Tab 2
   </sx:div>
</sx:tabbedpanel>
```

Autocompleter

Fixed list

```
<sx:autocompleter list="{'apple','banana','grape','pear'}" />
```

Set initial value

```
<sx:autocompleter list="{'apple','banana','grape','pear'}" value="apple"/>
```

Force valid option (restore option when focus is lost)

```
<sx:autocompleter list="{'apple','banana','grape','pear'}" forceValidOption="true"/>
```

Using the JSON plugin to generate the values (one of the possible ways)

The action

AutocompleterExample.java

```
public class AutocompleterExample extends ActionSupport {

   public Map<String, String> getOptions() {
      Map<String,String> options = new HashMap<String,String>();
      options.put("Florida", "FL");
      options.put("Alabama", "AL");

      return options;
   }
}
```

The mapping:

struts.xml

```
<struts>
...
   <package name="autocompleter" namespace="/autocompleter" extends="json-default">
       <action name="getStates" class="AutocompleterExample">
           <result type="json">
               <param name="root">options</param></result>
       </action>
   </package>
...
</struts>
```

The JSP (fragment):

```
<s:url id="optionsUrl" namespace="/autocompleter" action="getStates" />

<sx:autocompleter href="%{#optionsUrl}" />
```

Example action

When a form containing an autocompleter is submitted, two values will be submitted for each autocompleter, one for the selected value, and one for its associated key.

The action:

MyAction.java

```
public MyAction extends ActionSupport {
    private String optionsKey;
    private String options;

    ...
}
```

The JSP:

```
<s:form id="form">
  <sx:autocompleter name="options" label="Options" />
</s:form>
```

Set initial key and value

```
<s:url id="optionsUrl" namespace="/autocompleter" action="getStates" />

<sx:autocompleter href="%{#optionsUrl}" value="Florida" keyValue="FL"/>
```

Change default key name

The action:

MyAction.java

```
public MyAction extends ActionSupport {
    private String superKey;
    private String options;

    ...
}
```

The JSP:

```
<s:form id="form">
  <sx:autocompleter keyName="superKey" name="options" label="Options" />
</s:form>
```

JSON accepted

for this autocompleter:

```
<sx:autocompleter name="state" />
```

The following JSON will be accepted:

Map(recommended as it is the easiest one to generate)

```
{
    "Alabama" : "AL",
    "Alaska" : "AK"
}
```

Array of arrays

```
[
    ["Alabama", "AL"],
    ["Alaska", "AK"]
]
```

Array inside object, same name as field

```
{
    "state" : [
        ["Alabama","AL"],
        ["Alaska","AK"]
    ]
}
```

Map inside object, same name as field

```
{
    "state" : {
        "Alabama" : "Alabama",
        "Alaska" : "AK"
    }
}
```

Array inside object, field in response starts with the name of the autocompleter("state" in this example)

```
{
    "states" : [
        ["Alabama","AL"],
        ["Alaska","AK"]
    ]
}
```

No name match, use first array found, and hope for the best

```
{
    "Australopithecus" : [
       ["Alabama","AL"],
       ["Alaska","AK"]
    ]
}
```

Load characters while user types (when text size >= 3)

```
<sx:autocompleter href="%{#url}" loadOnTextChange="true" loadMinimumCount="3" />
```

Hide dropdown arrow

```
<sx:autocompleter href="%{#url}" showDownArrow="false" />
```

Limit options shown to 3

```
<sx:autocompleter href="%{#url}" resultsLimit="3" />
```

All matching options are shown

```
<sx:autocompleter href="%{#url}" resultsLimit="-1" />
```

Set dropdown height and width, in pixels

```
<sx:autocompleter href="%{#url}" dropdownHeight="180" dropdownWidth="200" />
```

Disable it when page is loaded

```
<sx:autocompleter href="%{#url}" disabled="true" />
```

Disable it/enable it using JavaScript

```
<script type="text/javascript">
  function enableit() {
     var autoCompleter = dojo.widget.byId("auto");
     autoCompleter.enable();
  }

  function disableit() {
     var autoCompleter = dojo.widget.byId("auto");
     autoCompleter.disable();
  }
</script>

<sx:autocompleter id="auto" href="%{#url}" />
```

Reload options when topic is published

```
<sx:autocompleter listenTopics="/reload" href="%{#url}" />
```

Submit form when options are loaded

```
<s:form id="form">
  <input type="textbox" name="data">
</s:form>

<sx:autocompleter formId="form" href="%{#url}"  />
```

Filter fields top be submitted when options are loaded (return true to include)

```
<script type="text/javascript">
  function filter(input) {
     return input.name == "data1";
  }
</script>

<s:form id="form">
  <input type="textbox" name="data0">
  <input type="textbox" name="data1">
</s:form>

<sx:autocompleter formId="form" formFilter="filter" href="%{#url}" />
```

Link two autocompleters, using topics

```
<form id="selectForm">
   <sx:autocompleter  name="select" list="{'fruits','colors'}"  valueNotifyTopics="/changed" />
</form>

<sx:autocompleter  href="%{#url}" formId="selectForm" listenTopics="/changed" />
```

Show options, but don't make suggestion (autocomple) in the textbox

```
<sx:autocompleter autoComplete="false" href="%{#url}" />
```

Prevent options from loading when page loads

```
<sx:autocompleter preload="false" href="%{#url}" />
```