# InputSuggestAjax

## InputSuggestAjax

### Overview

Autocomplete textfield through an Ajax call. The component is a 1:1 conversion of the Dojo ComboBox widget.

### Usage

The simplest usage is to just provide a backend method, which returns the suggested items:

```
<s:inputSuggestAjax suggestedItemsMethod="#{bean.getSuggestItems}" value="#{bean.selectedValue}"/>
```

There is also the possibility to shorten the number of suggested items:

```
<s:inputSuggestAjax suggestedItemsMethod="#{bean.getSuggestItems}"
                    maxSuggestedItems="10" value="#{bean.selectedValue}"/>
```

### Backend

```
public List getSuggestItems(String prefix) {
  List li = new ArrayList();

  li.add(prefix+1);
  li.add(prefix+2);
  li.add(prefix+3);

  return li;
}
```

Or alternatively, in case of using "maxSuggestedItems":

```
public List getSuggestItems(String prefix, Integer maxSize) {
...
}
```

### Advanced Usage

If there is the need to provide a kind of value/label functionality like using a SelectItem in a SelectOneMenu, you have to use the attribute "itemLabelMethod". When using the component like this, the "suggestedItemsMethod" has to return a list of special objects instead of simple Strings. A converter which handles the conversion of a special object into a specific value and vice versa should also be mapped to the component. Have a look:

```
<s:inputSuggestAjax suggestedItemsMethod="#{bean.getAddresses}"
                    itemLabelMethod="#{bean.getAddressLabel}"
                    value="#{bean.choosenAddress}">
      <f:converter converterId="inputSuggestAjaxConverter" />
</s:inputSuggestAjax>
```

In this case, the "suggestedItemsMethod" returns a list of Address objects, which retain data such as streetNumber, state, streetName, zip and so on. The "itemLabelMethod" can be something like this:

```
public String getAddressLabel(Object adress)
{
    if (adress instanceof Address)
    {
        Address a = (Address) adress;
        return a.getCity() + "," + a.getStreetName() + "," + a.getState();
    }
    else
    {
        return adress.toString();
    }
 }
```

As a consequence, all suggested items are displayed as a readable concatenation of city, streetname and state of each address. In order to put all the pieces together, the converter has to do some work:

```
public class InputSuggestAjaxConverter
        implements Converter
{
    public Object getAsObject(FacesContext context,
                              UIComponent component,
                              String value) throws ConverterException
    {
        //search all the addresses and return the address object
        //which equals the posted value streetNumber
    }

     /**
     * All suggested address objects are iterated and for
     * each address the converter is called and a unique value is calculated
     */
    public String getAsString(FacesContext context,
                              UIComponent component,
                              Object value) throws ConverterException
    {
        if (value instanceof Address)
        {
            Address address = (Address) value;

            return Integer.toString(address.getStreetNumber());
        }
        else if (value instanceof String)
        {
            return (String) value;
        }
        else return null;
    }
}
```

The bounded "value" of the component should than hold the address object which was choosen on client as the label string.

## Styling

All styling should be done like it is explained in the following section. Styling through the "style" or "styleClass" attribute will have no affects.

InputSuggestAjax provides a default style defined by Dojo. However, if you want to style the component your own way, you have to override Dojo`s default style-classes by prefixing the body element.

Following the original Dojo css styles:

```
.dojoComboBoxOuter {
        border: 0px !important;
        margin: 0px !important;
        padding: 0px !important;
        background: transparent !important;
        white-space: nowrap !important;
```

```
}

.dojoComboBox {
        border: 1px inset #afafaf;
        margin: 0px;
        padding: 0px;
        vertical-align: middle !important;
        float: none !important;
        position: static !important;
        display: inline !important;
}

/* the input box */
input.dojoComboBox {
        border-right-width: 0px !important;
        margin-right: 0px !important;
        padding-right: 0px !important;
}

/* the down arrow */
img.dojoComboBox {
        border-left-width: 0px !important;
        padding-left: 0px !important;
        margin-left: 0px !important;
}

/* IE vertical-alignment calculations can be off by +-1 but
these margins are collapsed away */
.dj_ie img.dojoComboBox {
        margin-top: 1px;
        margin-bottom: 1px;
}

/* the drop down */
.dojoComboBoxOptions {
        font-family: Verdana, Helvetica, Garamond, sans-serif;
        /* font-size: 0.7em; */
        background-color: white;
        border: 1px solid #afafaf;
        position: absolute;
        z-index: 1000;
        overflow: auto;
        cursor: default;
}

.dojoComboBoxItem {
        padding-left: 2px;
        padding-top: 2px;
        margin: 0px;
}

.dojoComboBoxItemEven {
        background-color: #f4f4f4;
}

.dojoComboBoxItemOdd {
        background-color: white;
}

.dojoComboBoxItemHighlight {
        background-color: #63709A;
        color: white;
}
```

For example, if you want to customize the list of suggested items, just add this to your own css-file:

```
body .dojoComboBoxItem {
        padding-left: 5px;
        padding-top: 5px;
        margin: 0px;
}
```

Or, if the whole input field should be affected:

```
body .dojoComboBox {
        border: 1px solid red;
        width:300px;
        margin: 0px;
        padding: 0px;
        vertical-align: middle !important;
        float: none !important;
        position: static !important;
        display: inline !important;
}
```

## Facelets

For using thin InputSuggestAjax-component in facelets you need to have the proper tag definition in your sandbox-taglib.xml ( http://wiki.java.net/bin/view/Projects/FaceletsTaglibsMyfacesSandbox ). The mentioned handler-class must be added to your project.

The source for the handlers class can be found at: http://wiki.java.net/bin/view/Projects/InputSuggestAjaxComponentHandler