Adding and Updating Records with cf_sebForm

Posted At : November 26, 2007 6:00 AM | Posted By : Steve
Related Categories: sebtags

I frequently have to build forms to add and edit data. After a few years of programming, this became a source of come tedium for me. I would build two forms almost exactly alike (one to add and one to edit). If the form needed to change, I would have to make the change twice. I would also have to write any validation twice for each form - once for the friendly client-side JavaScript validation and again for the serious server-side validation.

All told, this represented a fair bit of tedium in my work life. I decided that there had to be a better way. I looked at solutions that existed already in the ColdFusion universe, but I couldn't find anything that I liked. Then I came across some custom tags for forms that I client of mine built. Although his implementation didn't work as I wanted, I knew that he was on the right path.

Within a few weeks, I had the first working cf_sebForm custom tag set built. I refined these tags over the next few years (and am still doing so). In that time, the custom tags have maintained backward compatibility almost to the first version that I created.

Here is a basic example of an add/edit form:

```
<cfparam name="URL.id" default="0">

<cf_sebForm
   forward="basic-list.cfm"
   pkfield="RecordID"
   recordid="#URL.id#"
   CFC_Component="#Application.Basics#"
   CFC_GetMethod="getRecord"
   CFC_Method="saveRecord"
   >
   <cf_sebField fieldname="RecordVal" label="Value" required="true">
   <cf_sebField fieldname="RecordDescrip" label="Description" type="textarea">
   <cf_sebField type="Submit/Cancel" label="Save,Cancel">
</cf_sebForm>
```

Before continuing on, I suggest that you read on **getting data to cf_sebForm**. I generally use URL.id to identify the record, allowing me to use the following code (copied from the **demonstration site**):

```
<cf_sebForm
   forward="basic-list.cfm"
   pkfield="RecordID"
   CFC_Component="#Application.Basics#"
   CFC_GetMethod="getRecord"
   CFC_Method="saveRecord"
   >
   <cf_sebField fieldname="RecordVal" label="Value" required="true">
   <cf_sebField fieldname="RecordDescrip" label="Description" type="textarea">
   <cf_sebField type="Submit/Cancel" label="Save,Cancel">
</cf_sebForm>
```

This form will act as an edit form is URL.id is given and represents an existing record. Otherwise it will act as an add form.

Because the "RecordVal" field is required, cf_sebForm writes JavaScript (using the qForms JSAPI) to ensure that it is filled in when the form is submitted. On the server-side, the cf_sebForm custom tag checks to see if "RecordVal" is filled out as well. If it isn't, it returns the user to the form (with all of the fields populated with the data the user filled in) and displays a user-friendly error indicating that

"RecordVal" is required.

Any error displayed to the user (via JavaScript or server-side validation) will use the field label instead of the field name.

Assuming that the form passes validation, then cf_sebForm will call the method indicated in the "CFC_Method" argument, passing each field as a separate argument. An argument named the value of the "pkfield" attribute will also be passed in to the method with the value of the "recordid" attribute (which defaults to URL.id) of cf_sebForm (assuming a recordset was returned). If I wanted to pass in other arguments, I could pass in a structure of additional arguments to an optional "CFC_MethodArgs" attribute of cf_sebForm.

The method in the CFC_Method attribute would need to handle the logic to add or edit a record. Once I have all of this done, I have the basics of adding and updating records completed.

In upcoming entries, I will cover more validation options (including a way to validate on business logic without duplicating code or breaking encapsulation) and customization or styling and display.

The cf_sebForm custom tags are part of the **sebtags custom tag** set which is open source and free for any use.