ColdFusion XML form components

Table of Contents

Tree

Version 1.0

What this component does

Do you want to display a fancy DHTML tree in your ColdFusion XML forms? Then take a look at this component.

Some highlights:

- Microsoft™ XP® style (by default)
- skinning (through usage of a Style Sheet)
- single item highlighting/selection
- selected item is automatically expanded in the tree on initialization
- AJAX calls to show large trees
- ultra fast (one of the fastest tree components available)

Prerequisites

In order for this component to work, you'll need a copy of ColdFusion MX 7.0 (or higher). This tag has been tested and is working under Internet Explorer® 5.0 (and higher) and FireFox 1.5 (and higher). More browsers might be supported but are untested.

How to install this component

You should first locate the `CFIDE` directory on your server. This is the place where ColdFusion stores its components and administrator pages. You'll find this directory most probably in your web server's root. If you don't have access to this directory (e.g. in sharing host environments), you cannot use this component!

Once you've found the correct location, open the subdirectory `scripts` followed by the subdirectory `xsl`. You will now see a file called `default.xsl`. Make a copy of this file and rename it to `ezform.xsl`.

IMPORTANT: don't overwrite `ezform.xsl` if it already exists. This probably means you've purchased other XML components from the same author before.

Now open `ezform.xsl` in a text editor and add the following lines just before the comment string `include cfform javascript generation`:

```
<!-- include tree -->
<xsl:include href="ezformtree.xsl" />
```

Copy the files contained in the ZIP file you have been emailed to the `CFIDE` directory as follows:

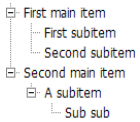
Copy:	**To:**
`ezformtree.xls`	`CFIDE/scripts/xsl`
`ezformtree.css`	`CFIDE/scripts/css`
`ezformtreeminus.gif`	`CFIDE/scripts/css`
`ezformtreeminuslast.gif`	`CFIDE/scripts/css`
`ezformtreenode.gif`	`CFIDE/scripts/css`
`ezformtreenodelast.gif`	`CFIDE/scripts/css`
`ezformtreeplus.gif`	`CFIDE/scripts/css`
`ezformtreeplusfirst.gif`	`CFIDE/scripts/css`
`ezformtreepluslast.gif`	`CFIDE/scripts/css`
`ezformprogress.gif`	`CFIDE/scripts/css`
`ezformtree.js`	`CFIDE/scripts`

Congratulations, the tree component is now installed and recognized by ColdFusion!

Using the tree in your CFFORM

If you want to show a tree, you must use the tags `cftree` and `cftreeitem`. Because of limitations in ColdFusion, not all parameters you'll find in the CFML Reference Guide are supported by this DHTML variant.

The following CFML example shows a basic tree, which looks like this:




```
<cfform
action="targetcfm.cfm"
method="POST"
format="XML"
skin="EZFORM"
name="test">

<cftree
name="tree1"
format="XML"
border="No"
delimiter=""
width=300
height=200>

<cftreeitem
value="1"
display="First main item">

<cftreeitem
value="2"
display="Second main item">
```

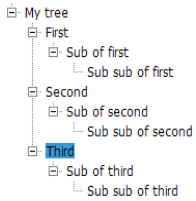


```
<cftreeitem
value="3"
display="First subitem"
parent="1">

<cftreeitem
value="4"
display="Second subitem"
parent="1">

<cftreeitem
value="5"
display="A subitem"
parent="2">

<cftreeitem
value="6"
display="Sub sub"
parent="5">

</cftree>

<cfinput
type="Submit"
name="Submit"
value="Submit">

</cfform>
```

The `cftree` tag accepts the following parameters: `name` (which will become the hidden form field name containing the value selected in the tree), `border` (can either be `Yes` or `No`), `width` and `height` (the dimensions of the tree on the screen) and `delimiter` (which is used to pass the selected value in the tree because the attribute `value` is not supported by cftree). Don't forget to add `format="XML"`, which is a required attribute.

Each `cftreeitem` tag adds a leaf to the tree. Supported parameters are `value` (the value to pass to the form), `display` (what is shown in the tree) and optionally `parent` (a pointer to the value of the parent leaf in the tree).

Instead of adding all leaves to the tree by using a multitude of `cftreeitem` commands, you can also point `cftree` to a database which already contains a level structure (not a tree structure!), like in the example below:




```
<cfset myQuery =
QueryNew("FirstLevel,SecondLevel,ThirdLevel")>

<cfset newRow = QueryAddRow(MyQuery, 3)>

<cfset temp = QuerySetCell(myQuery,
"FirstLevel","First", 1)>

<cfset temp = QuerySetCell(myQuery,
"SecondLevel","Sub of first", 1)>

<cfset temp = QuerySetCell(myQuery,
"ThirdLevel","Sub sub of first",1)>

<cfset temp = QuerySetCell(myQuery,
"FirstLevel","Second",2)>

<cfset temp = QuerySetCell(myQuery,
"SecondLevel","Sub of second",2)>
```



```
<cfset temp = QuerySetCell(myQuery,
"ThirdLevel","Sub sub of second",2)>

<cfset temp = QuerySetCell(myQuery,
"FirstLevel","Third",3)>

<cfset temp = QuerySetCell(myQuery,
"SecondLevel","Sub of third",3)>

<cfset temp = QuerySetCell(myQuery,
"ThirdLevel","Sub sub of third",3)>

<cfform
action="targetcfm.cfm"
method="POST"
format="XML"
skin="EZFORM"
name="test">

<cftree
name="tree1"
format="XML"
border="No"
delimiter="Third"
width=300
height=200>

<cftreeitem
query="myQuery"
value="FirstLevel,SecondLevel,ThirdLevel"
display="FirstLevel,SecondLevel,ThirdLevel"
queryasroot="My tree">

</cftree>
```

```
<cfinput
type="Submit"
name="Submit"
value="Submit">


</cfform>
```

You'll notice that this time `queryasroot` is used. If this parameter is `Yes`, the name of the query is added as the parent of the tree. If `queryasroot` is not `Yes` or `No`, that value is used in the tree instead of the query name.

Of course `QueryNew` is used in the example above to avoid having to create a database.

Using the AJAX version of the tree

If you want to show a large tree, you're better off using the AJAX version which only loads the part of the tree the user wants to display.

In order to activate AJAX calls, you have to change the `delimiter` parameter of the `cftree` tag and append the location of the ColdFusion template which will receive the calls.

An example will show you how it works:

```
<cfquery name="myquery"
datasource=" MYDATASOURCE">
select myseqid as id, mytitle as title, (select
count(*) from mytable where myparentid
=main.myseqid) as numberofchildren from mytable
main where myparentnumber=0
</cfquery>


<cfform
action="targetcfm.cfm"
method="POST"
format="XML"
skin="EZFORM"
name="test">


<cftree
name="tree1"
format="XML"
border="No"
delimiter="1\treevalues.cfm?selectedid="
width=300
height=200>


<cfoutput
query="myquery">
```

```
<cftreeitem
value="#id#"
display="#title#\#numberofchildren#">


</cfoutput>


</cftree>


<cfinput
type="Submit"
name="Submit"
value="Submit">


</cfform>
```

When the tree is called for the first time, you only need to pass the items at level 0 of the tree. This is done in the example above using the `where myparentid=0` clause in the `cfquery` tag. Furthermore, the SQL select command uses a sub-select to see if the current item has children or not. The number of children is passed to the main query as the field `numberofchildren`. The tree needs to know beforehand when there are children to render the graphics correctly.

Note that the `display` parameter of the `cftreeitem` tag now consists out of two parts, delimited with a `\` character. The first part is the title, the second part is the number of children this item has.

The `delimiter` parameter has two parts as well. The first part before the `\` is the current value (which may be left out), the second part is a ColdFusion template name. Each time the user clicks on a plus-node, the template mentioned in the `delimiter` parameter is called. In our example, this file is called `treevalues.cfm` and looks as follows:


```
<cfquery
name="myquery"
datasource="MYDATASOURCE">
select myseqid as id, mytitle as title, (select
count(*) from mytable where myparentid
=main.myseqid) as numberofchildren from mytable
main where myparentnumber=#url.selectedid#
</cfquery>

<cfsetting enablecfoutputonly="Yes">

<cfoutput
query="myquery">#title#\#id#\#numberofchildren##ch
r(9)#</cfoutput>
```

As you can see, the `CFQUERY` is 100% identical to the previous one, with the exception of `where myparentnumber=#url.selectedid#` (which fetches the records belonging to the `selectedid` passed through the url in the AJAX call).

The results of the query are returned as a line, delimited with a `chr(9)` character. Inside one line, three parameters are passed, and each parameter is delimited with a `\` sign. The first parameter is the title, the second the id, and the third the number of children the current record has.

The style sheet

The standard style sheet which comes with this tree is called `ezformtree.css`. It mimics the looks of a standard Microsoft™ XP ® tree. You are free to copy and modify this file to customize the look.

Because the functionality of the tree is closely related to this style sheet, only the following items may be modified:

`ul.tree ul`	Font properties of the tree.
`ul.tree a:link, a:visited, a:active`	Properties of an item.
`ul.tree a:hover, a:visited:hover`	Properties of an item when the cursor is above the item.
`.treeselected`	Properties of a selected item.

Upgrades, updates, bug fixes, new versions

Small updates or bug fix versions are released on a regular basis. They have a higher number after the point in the version number and are free for users who bought the component.

New versions (having a higher number before the point in the version number) aren't free, but current users can upgrade at a reduced price. More information on this can be found at our website.

Bugs? Questions?

Mail the author at **wim@dewijngaert.be** and he'll contact you ASAP.