**Form Pulldown Menus Used as Navigation Elements**

In this section, I am going to talk about "submitting" a FORM to a JavaScript function, then extracting information from that form using dot syntax and a number of specialized keywords. To demonstrate this process, we are going to look at a FORM pulldown menu used as a navigational element.

Here is an [**example**](http://fog.ccsf.edu/~srubin/formpull1.html) of what I mean.

In an earlier module, we discussed creating form pulldown menus, and using the VALUE attribute of the individual OPTION tags to store URL information.

Example (abbreviated):

<option value="http://www.yahoo.com/">Yahoo</option>

For the purposes of this demonstration, I am going to name my SELECT tag "destList"; I do NOT need to name my FORM tag.  
  
Example:  
  
<form>  
<select name="destList">  
<option value="http://www.yahoo.com/">Yahoo</option>  
<option value="http://www.w3.org/">W3C</option>  
<option value="http://www.mozilla.org/">Mozilla</option>  
<option value="http:/fog.ccsf.org/~srubin">Steve Rubin</option>  
</select>  
</form>  
  
When you "submit" your form to a JavaScript function, you will NOT use the submit INPUT; instead, you will use the generic button INPUT. You will pass the value, this.form, as the argument to your function, which will, essentially, pass a complete copy of the form to the function.  
  
Note: the this keyword, and the value, this.form below. It turns out that, in this particular case, the this keyword refers to the button, and the form property of the button is really a special keyword referring to the entire form attached to the button. The phrase this.form, then, can be passed as an argument to a function; the reference encompasses the entire form enclosed by the FORM tag attached to the given form element.  
  
The following example presumes that the function I'm about to create is called goToNewPage().  
  
Example (abbreviated):  
  
<input type="button" value="Go" onclick="goToNewPage(this.form);">  
  
Example (in context):  
  
<form>  
<select name="destList">  
<option value="http://www.yahoo.com/">Yahoo</option>  
<option value="http://www.w3.org/">W3C</option>  
<option value="http://www.mozilla.org/">Mozilla</option>  
<option value="http://fog.ccsf.org/~srubin">Steve Rubin</option>  
</select>  
  
<input type="button" value="Go" onclick="goToNewPage(this.form);">  
</form>  
  
Now, I'm ready to create my function; the function will have one local argument variable, myForm, which will contain the reference to the form (passed using this.form as an argument from the event handler).  
  
Example:  
  
function goToNewPage(myForm) {  
// lines of code here...  
}  
  
In this function, I'm going to need to look at the destList SELECT tag within the form; this can be done using dot syntax.  
  
Example:  
  
myForm.destList  
  
Now, I'm going to need to look at the options[] array within the destList SELECT tag. All of the OPTION tags in a SELECT tag are stored in an options[] array. This, too, will be accessed using dot syntax.  
  
Example:  
  
myForm.destList.options[]  
  
What I really need to see, however, is the value property of the desired option in the options[] array. The value property of the option contains the URL string that we've placed in the VALUE attribute.  
  
Example:  
  
myForm.destList.options[].value  
  
The problem with this syntax is that we do not know WHICH option we want!  
  
In the options[] array, as you might expect, the FIRST option (in this case, the option for Yahoo) is stored in options[0], the SECOND option (the option for the W3C) is stored in options[1], etc. If I want to get the URL value for Mozilla, then, I would need to look at options[2] in this example.  
  
Example:  
  
myForm.destList.options[2].value  
  
The above example will return the string value, "http://www.mozilla.org/".  
  
Alright, this hard-coding is good enough, in its way, but HOW do I find out WHICH option the USER has selected?  
  
There is a special keyword of the SELECT tag, selectedIndex, which returns the number of the option which the user has selected.  
  
Example:  
  
myForm.destList.selectedIndex  
  
I need to put the above code INTO the array access operator ([]) for the options[] array to access the selected option.  
  
Example:  
  
myForm.destList.options[myForm.destList.selectedIndex].value  
  
I would then place this extracted information into some sort of local variable.  
  
Example:  
  
var mydest = myForm.destList.options[myForm.destList.selectedIndex].value;  
  
Once I have extracted the URL string value that I require, I will need to ASSIGN that value to the browser window's location object. The location property of the window instance of the Window object gives us access to the location bar, and will advance the page in the web browser when assigned to the desired URL.  
  
Example:  
  
window.location = "http://www.mozilla.org/";  
  
Example:  
  
var mydest = myForm.destList.options[myForm.destList.selectedIndex].value;  
window.location = mydest;  
  
Example (complete function):  
  
function goToNewPage(myForm) {  
var mydest = myForm.destList.options[myForm.destList.selectedIndex].value;  
window.location = mydest;  
}  
  
In the example cited earlier, there was also a pulldown menu which changed the page location WITHOUT hitting a "Go" button. This functionality was achieved using the onchange event handler, which is triggered when the state of a form element is changed by the user; the onchange event handler would be added to the SELECT tag for the pulldown menu.  
  
Example:  
  
<select name="destList" onchange="goToNewPage(this.form);">  
<option value="http://www.yahoo.com/">Yahoo</option>  
<option value="http://www.w3.org/">W3C</option>  
<option value="http://www.mozilla.org/">Mozilla</option>  
<option value="http://fog.ccsf.org/~srubin">Steve Rubin</option>  
</select>  
  
In the above example, the Yahoo option is pre-selected. If the user wishes to go to Yahoo, then, they will NOT be able to do so, because the onchange event handler is ONLY called when the state of the pulldown menu is CHANGED.  
  
To get around this problem, web programmers add some sort of blank option to the pulldown menu; this blank option usually tells the user what they ought to do with the pulldown menu.  
  
Example:  
  
<select name="destList" onchange="goToNewPage(this.form);">  
<option>Choose dest:</option>  
<option value="http://www.yahoo.com/">Yahoo</option>  
<option value="http://www.w3.org/">W3C</option>  
<option value="http://www.mozilla.org/">Mozilla</option>  
<option value="http://fog.ccsf.org/~srubin">Steve Rubin</option>  
</select>  
  
That's it! Here's the [example page](http://fog.ccsf.edu/~srubin/formpull1.html) again. I strongly urge you to try typing out this code for yourself, by hand, to get a feel for the process.