Developed by Walter Blady for use in Adobe GoLive - Base compatibility: Netscape 4, Internet Explorer 4.

VerifyForm.Action

This powerful Form Checker will verify that applicants have properly filled in up to 15 text fields before submitting the form.

Description:

Drop the VerifyForm Action into the GoLive Modules > JScripts > Actions folder, then put the VFList.js, VFProfan.js and VFMess.js files in a suitable place within your site.

VerifyForm is used to verify entries in up to 15 **Text fields** on a standard HTML form. There are 18 verification categories - E-mail format, Exact match, Numbers only etc., 4 field modifiers - to Lower & upper case etc. There are 2 special purpose options - **Pop-up list:** when users make a selection, they are re-directed to an associated URL after the form is submitted. **Profanity filter:** Checks all the text fields and areas of a form for black-listed words, phrases or numbers. There is also an option to guard against more than one submission of the same form.

If any field entry fails to meet your verification criteria, VerifyForm will open a small window displaying a detailed list of all the entry errors, and the form will not be submitted. You can also set fields that are Required.

Cautions:

Most form checkers are server side scripts and not client side scripts. Since this form checker is embedded on the HTML page, anyone can see or change the value of the fields if they edit the HTML code. Chances are most people won't, but using this Action to check for sensitive entries like passwords and the like can be a bit risky.

VerifyForm can not be used to verify Radio Buttons, Check Boxes or Selection (pop-up) Lists. It can only be used to verify Text fields and Text Areas.

There might be a problem using **CS-Western (MacRoman)** font encoding with Netscape. Select **CS-Western (Latin1)** Font Encoding for your page.

Figure 1

How to use it:

VerifyForm can be linked to either a **Form Tag**, a **Submit Button** or some **Hyper Text**. This example will link it to the Form Tag.

1 - Drag a **Form tag** onto your page. Click on it and give your form a **name**, then fill out the remainder of the Form Inspector box (*Figure* 1).

The form's **Action** can be set to either a server side CGI script which parses the form data and sends it to you via e-mail, or it can be an e-mail address (see the caution on using e-mail addresses later in these instructions).

2 - Select the **Encoding** and the **Method** you need**.** You can leave the Target field blank for this kind of form.

3 - If you want to submit the form by E-mail, enter **text/plain** as the Encoding and **Post** as the Method. The form data will be delivered to you as **Label=data** pairs instead of the usual long line of encoded text.

4 - Design your form and enter all the fields. Remember the names that you give to your form and its fields because they will be used in the VerifyForm.Action.

VerifyForm.Action

Setting up the Submit Button:

With GoLive 5 you can set up either a **regular Submit button**, a **graphic button** that you've designed yourself or a **form input image**. GoLive 4 must be set up with a graphic button.

Graphic button for GoLive 4:

1 - Place an **Input Image** where you would like the submit button to go in your form. An Input Image is the same as a regular image except it has the **Is form** box checked for you.

2 - Click on the **Input Image Icon** then select the **Basic** tab. Link the Input Image to whatever graphic you want to use for your submit button (*Figure* 2).

3 - Select the **More** tab and check the **Is Form** box and enter a name

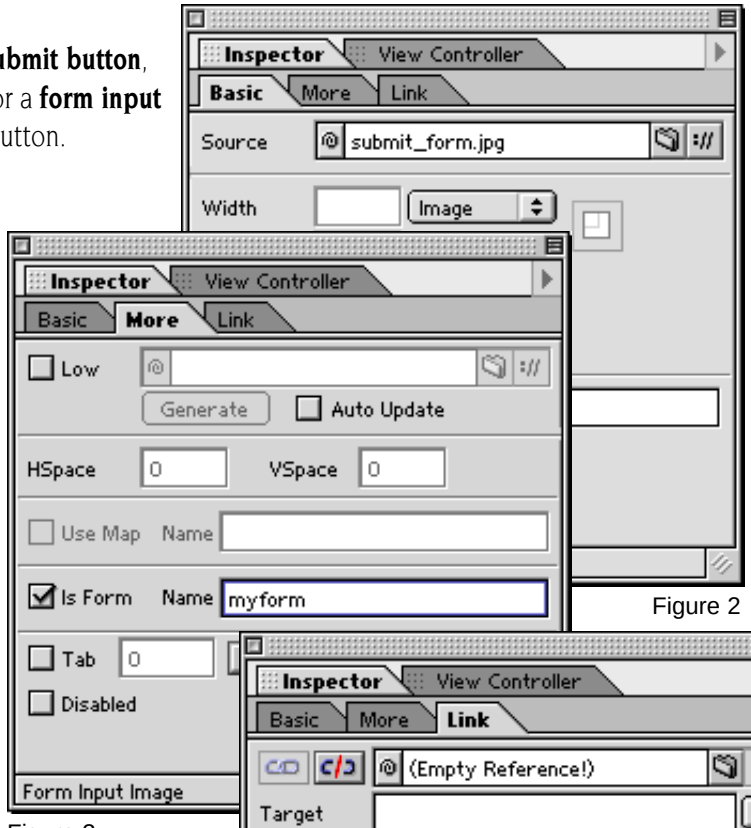

Figure 2

Figure 3

Linking VerifyForm in GoLive 4:

4 - Click on the submit button graphic then select the **Link** tab and click on the URL link button (*Figure* 4). You don't have to enter anything in the URL field. Selecting a link here allows you to attach an **Event** to the image under the Actions tab.

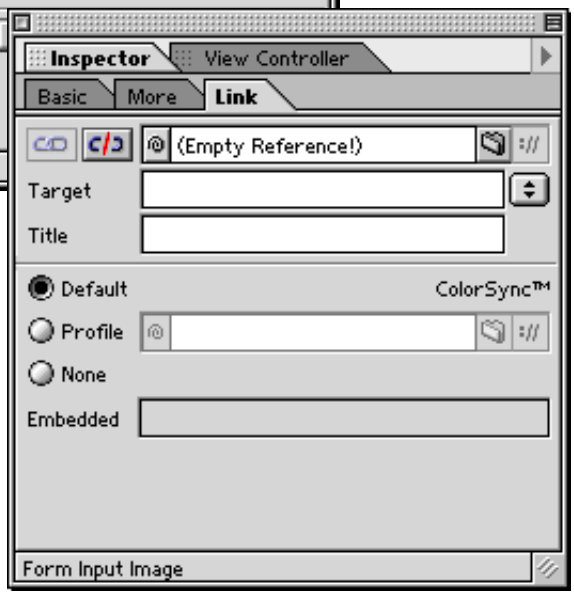

Figure 4

5 - Select the **Actions** tab then select the VerifyForm Action as a **Mouse Click** Event (Figure 5).

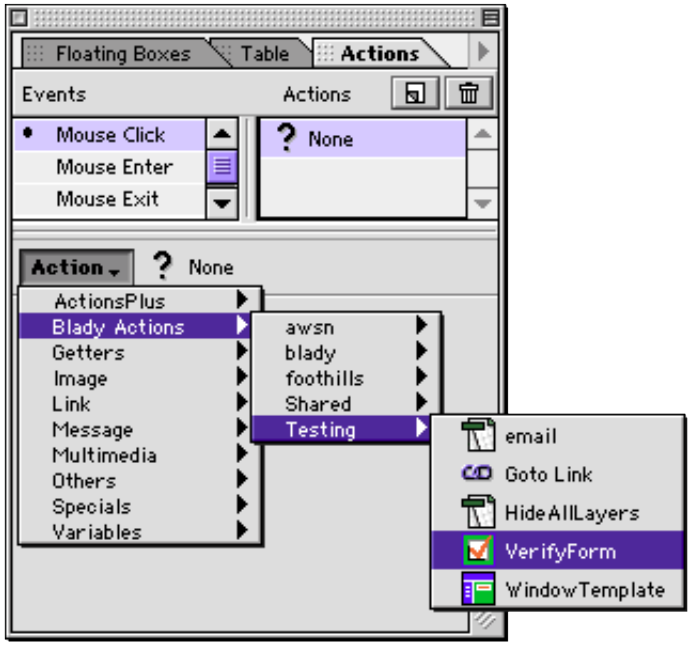

Figure 5

VerifyForm.Action

Linking VerifyForm in GoLive 5+:

VerifyForm can be linked to either the **form tag**, a **standard submit button**, a **graphic button** or a **form input image**.

Linking to the Form Tag:

1 - Click on the Form tag, select **Form Submit** then attach the VerifyForm Action to it (Figure 6).

Linking to the buttons or image:

1 - Decide what type of button you want to use then position it on your form.

2 - Click on the button, then attach the VerifyForm Action to it as a **Mouse Click** Event (Figure 7).

3 - You can give the button a name if you want. It doesn't affect the sending of the form.

NOTE:

See **Finalizing the Submit button setup** at the end of these instruction.

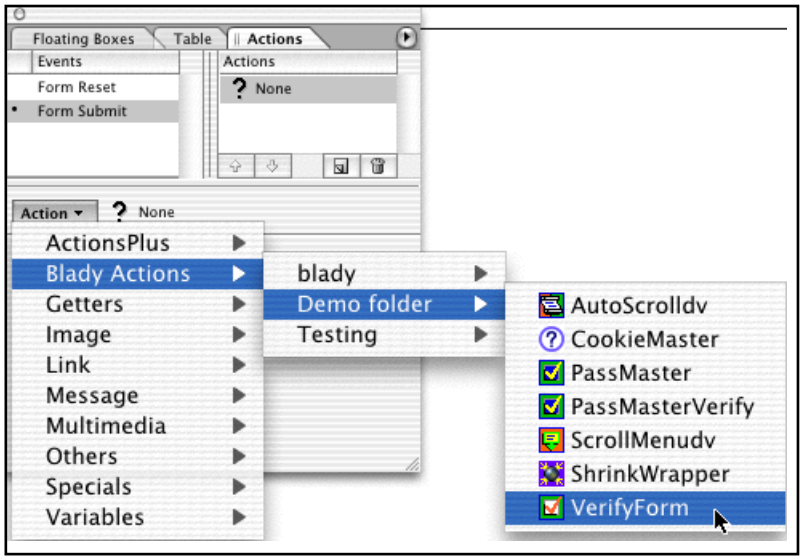

Figure 6

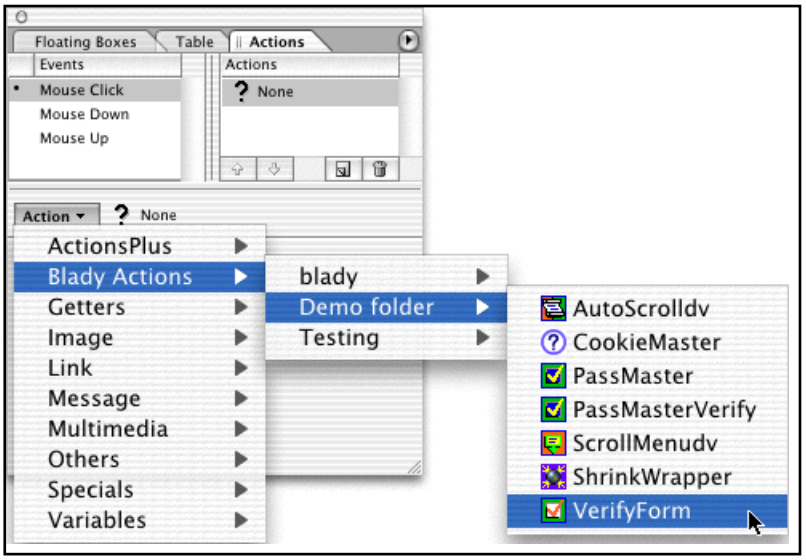

Figure 7

VerifyForm.Action

Setting up the VerifyForm Action:

1 - Enter the name of your form in the **Form name** field (Figure 8).

2 - If your form is in a **Floating Box**, then select the box name from the pull down menu.

3 - Enter the names of the **Text Fields** or **Text Areas** that you want to verify in the "Field name" boxes. These names must be exactly the same as the ones used on your form. All name are case sensitive, and **spaces** are **not** allowed.

4 - Enter **descriptive text** for each field in the "Field description" boxes. This text is used in the alert box if a field isn't properly filled in. **Apostrophes** and **Quotation marks** are not allowed.

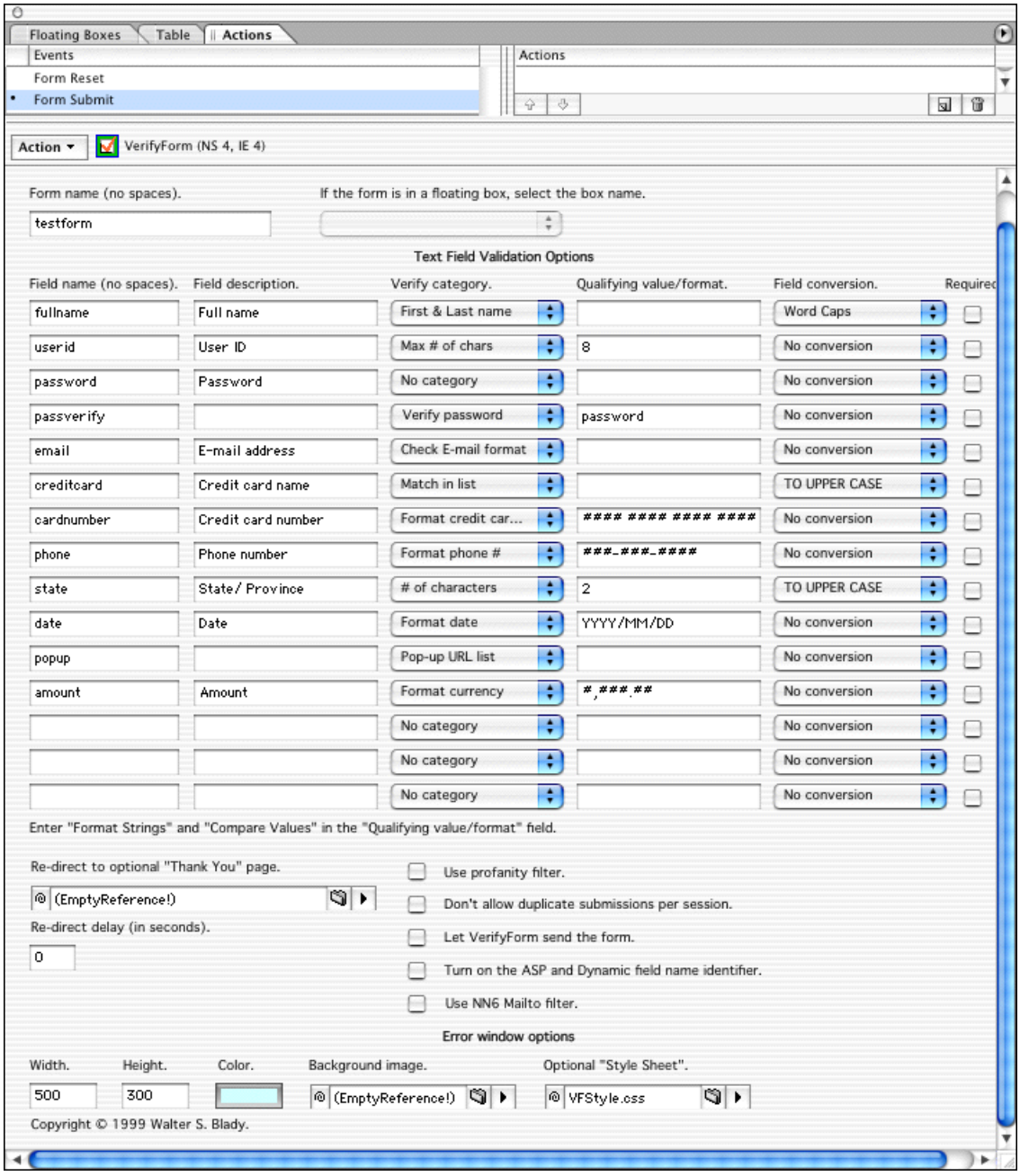

Figure 8

VerifyForm.Action

5 - Select the **criteria** for checking each field from the "Verify category" pull down menus beside each field. If you don't want to check the field, select the **No category** option.

6 - Some of the Verify categories need a value entered in the Qualifying Value/Format field. Enter a **qualifying value** or **format** string as required (format strings are explained further on).

7 - Select an optional **field modifier**. If you don't want to modify the field, select the **No conversion** option.

8 - Check the **Required** box of any field that must be filled in by the applicant.

9 - Complete all the fields you want to verify.

Verify category options:

No category -
VerifyForm will ignore any field that has this option selected.

2nd field filled in also -
Checks that this field and a companion field have something entered in them. The companion field name is entered in the "Qualifying value/format" column. This option can be used, for example, to ensure that two related but non-required fields get filled in together.

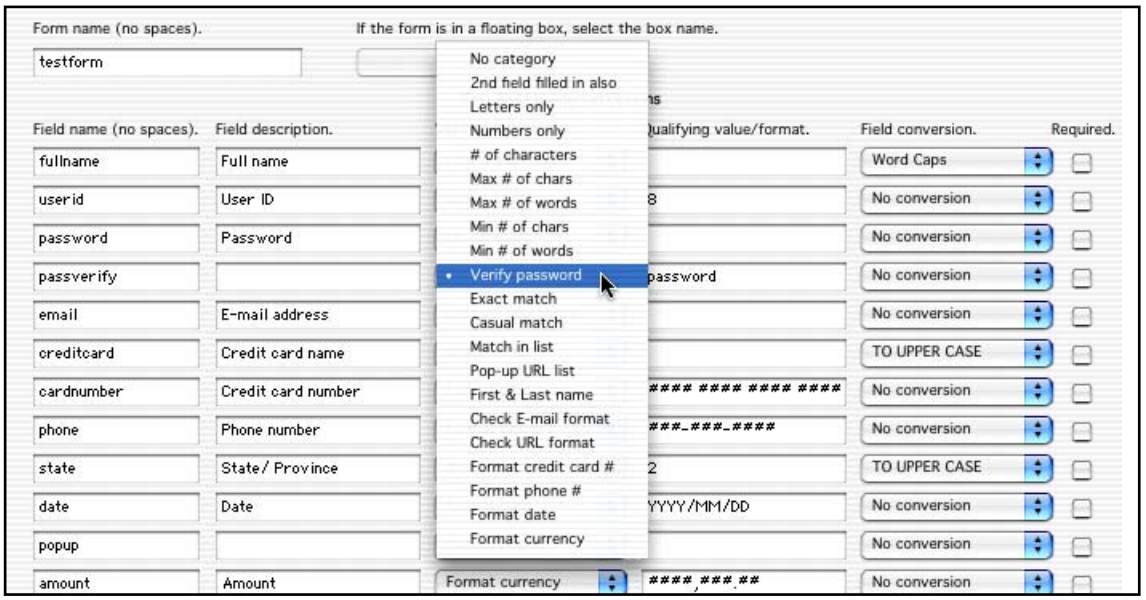

Figure 9

Letters only -
Checks that only alphabetic characters have been entered.

Numbers only -
Checks that only numbers have been entered (spaces are not allowed).

# of characters -
Checks that the number of characters entered matches the value in the "Qualifying value" column.

Max # of Characters -
Checks that the number of characters entered is not over the value in the "Qualifying value" column.

Max # of words -
Checks that the number of words entered is not over the value in the "Qualifying value" column.

Min # of Characters -
Checks that the number of characters entered is equal to or greater than the value in the "Qualifying value" column.

Min # of words -
Checks that the number of words entered is equal to or greater than the value in the "Qualifying value" column.

Verify password -
Many forms ask applicants to enter their password choice twice to help confirm the password. This option will try to match the user's entry with contents of the field name entered in the "Qualifying value" column.

Exact match -
Checks that the user's entry exactly matches the value you entered in the "Qualifying value" column.

Casual match -
Checks that the value you entered in the "Qualifying value" column is found in the user's entry. This match is not case sensitive.

Casual Match in External List -
The first thing you do is create a list of values in the **VFList.js** file. These values can be single words, word strings, numbers or a combination of both. Each value in this list is entered after the word **return** and must be enclosed in double quotes (Figure 10). The example list shows the values **"visa"**, **"master card"** and **"american express"**. The final value must be the word **"<end>"** enclosed on angle brackets.

The External list matching process can be used in two ways:

1 - Check that the user's entry is found somewhere in the list of values. The match can be partial and is not case

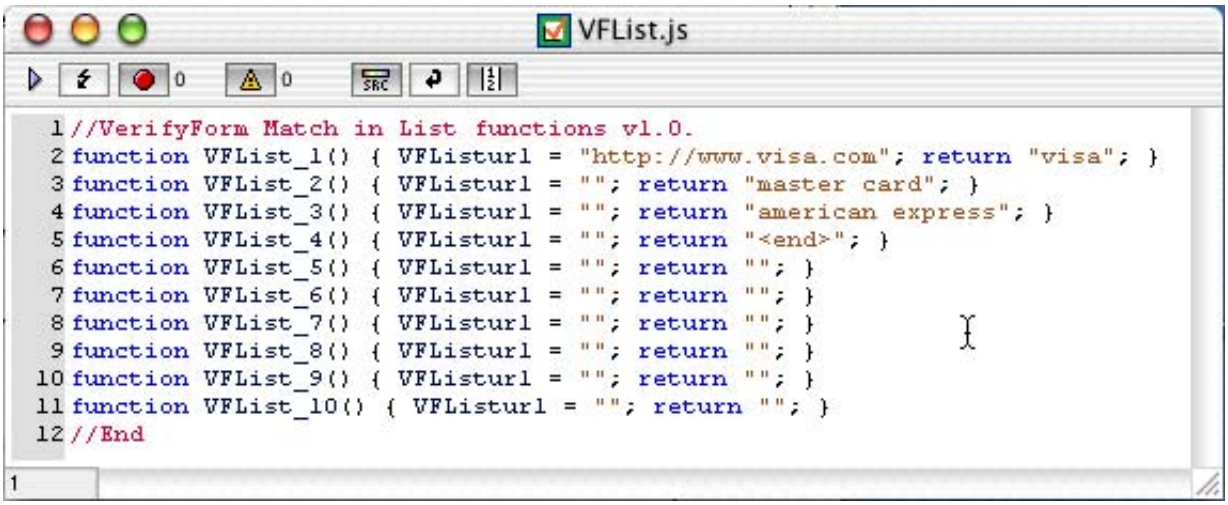

Figure 10

sensitive.

2 - Check that one of the values in the list is found somewhere in the user's entry. (reverse of the above). The match can be partial and is not case sensitive. To use this second method, enter the word **"list"** in the "Qualifying value" column.

You can also enter an optional URL for each entry in your list. After the form is successfully submitted, the browser will jump to this URL. (*See* **URL *loading order***).

NOTE:
You only have to link to the **VFList.js** file if you use the **External list** option.

VerifyForm.Action

How to set up the VFList.js file.

1 - Open the **VFList.js** file in either GoLive or a text editor.

2 - Enter your list in the **return column** (Figure 10). Enter optional URLs for each entry.

If you have to add more lines, make sure the numerical order of the **Function numbers** are sequential.

3 - Enter the word **"<end>"** enclosed in angle brackets as the last entry in your list.

4 - Save and close the file.

5 - Put the VFList.js file in a suitable place in your website.

6 - Drag a **Script** icon from the **Head** palette to the head section of your page (Figure 11).

7 - Check the **Source box** under the Inspector's tab then select the VFList.js file as the source file.

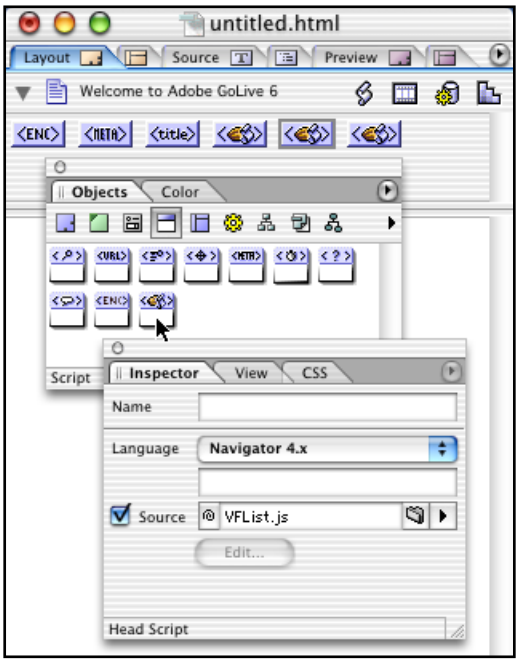

Figure 11

NOTE:
The VFList.js file can be renamed to whatever you like but you can only use one of these files per page.

Pop-up URL list -
This option is a little different. You can set up a **pop-up menu** on your form and assign a URL to each menu option (Figure 12).

It is important to set the first item in the pop-up list to **No topic** with a **blank** value, otherwise VerifyForm will automatically jump to the first URL in the list, ignoring the URLs in the **VFList.js** file and the URL at the bottom of the VerifyForm Inspector's window.

When the form has been successfully submitted VerifyForm will jump to the selected URL (*See* **URL *loading order***).

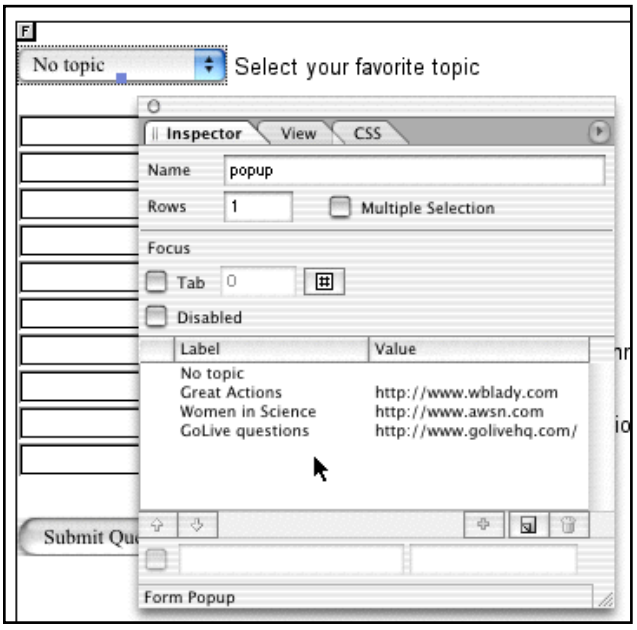

Figure 12

First & Last name -
This option checks that the user has entered either two or three words only. No single character initials are allowed - with or without periods. Some last names are comprised of two words eg: **Eric von Stienmetz, Bork van Muisen** etc. For this reason a check for first and last name looks for an entry of two or three words.

Check E-mail format -
Check that the entry conforms to conventional E-mail formats.

Check URL format -
Checks that the URL begins with either **http://** or **ftp://**, and that there are no spaces in the URL.

VerifyForm.Action

Format Credit card number -
This option will reformat the applicant's entry to match the **card number format** entered in the "Qualifying value" column. To enter a format, type Hash Marks where numbers should be, and separate groups of numbers with a **delimiter character** or a **space**. Eg. (#### #### #### #### or ################).

Format Phone number -
This option will reformat the applicant's entry to match the **phone number format** entered in the "Qualifying value" column. To enter a format, type Hash Marks where numbers should be.
Eg. (###-###-#### or (###) ###-####).

Format Date -
This option will reformat the applicant's entry to match the **date format** entered in the "Qualifying value" column. To enter a format, type "Y", "M" and "D" where the year, month and day should be, and any delimiters between the date segments. Eg. (**YYYY/MM/DD** or **YYYY,MM,DD**). This option does not check the date validity of the numbers entered. The formatting will either add or remove leading zeros for days and months, depending on how many characters you put in the format field.

Format Currency -
This option will reformat the applicant's entry to match the **currency format** entered in the "Qualifying value" column. **Decimals** and **Commas** are interchangeable. **Spaces** may be used as a Thousands delimiter.
The format strings are:

Thousand and decimal - `#,###.##`	Thousand and no decimal - `#,###`
Decimal and no thousand - `####.##`	No thousand and no decimal - `####`

If you omit the decimal, VerifyForm will trim the decimal portion of the applicant's entry. If you include the decimal and the applicant doesn't, VerifyForm will add zeros for the decimal portion.

Field modifier options:

No conversion -
VerifyForm will ignore any field that has this option selected.

TO UPPER CASE -
VerifyForm will convert all characters in the field to UPPER CASE

to lower case -
VerifyForm will convert all characters in the field to lower case.

Word Caps -
VerifyForm will convert the first letter of every word to Upper Case and the rest to lower case.

Sentence caps -
VerifyForm will convert the first letter of every sentence to Upper Case and the rest to lower case.

This option simply checks the text for either a **Period & space**, **Question mark** or an **Exclamation mark**, and then capitalizes the next alphabetic character it finds. It will also capitalize the letter **"i"** if it has a space on both sides, or a space on the left and an apostrophe on the right. All the rest is converted to lower case.

NOTE:
VerifyForm strips leading and trailing spaces from each entry, applies any optional field modifier, then inserts the string back into the form field before any validation is done.

VerifyForm.Action

URL loading order:

There are three ways to jump to a URL after VerifyForm has successfully submitted your form.

1 - Load a URL from the **Pop-up** list.
2 - Load a URL from a match made in the **VFList.js** file.
3 - Load the URL at the bottom of the **VerifyForm set up window**.

Obviously VerifyForm can't jump to all three URLs after the form has been submitted. The order of URL precedence is: the Pop-up list, the VFList.js file and finally the URL at the bottom of the VerifyForm set up window.

Sometimes a browser will still be in the process of submitting the form when the VerifyForm tries to jump to the URL. This sometimes stops the new page from loading. Unfortunately there is no way to check if a form has been submitted, so there is a **time delay** option you can use. Experiment with delays until you find one that works for you. The time delay is entered at the bottom of the VerifyForm set up window. This delay applies to all three URL set ups.

There are 6 additional options at the bottom of VerifyForm.

***Re-direct to optional "Thank you" page*:**
If you want to send the applicant to a URL, like a Thank you page, after the form has been submitted, enter the URL. This option doesn't work quite like a server side script. VerifyForm will immediately go to the new URL as soon as the applicant clicks the submit button. However, some browsers will post an E-mail warning before the form is actually sent. If the user clicks Cancel, they might get confused at the Thank you page, thinking their form has been submitted. *See* **URL *loading order*** *for more info on jumping to a new page.*

***Use Profanity filter*:**
This filter can be used to catch any unwanted words, phrases or numbers entered in your form, and warn the user that they have made an inappropriate entry.

***Don't allow duplicate submissions per session*:**
Some users might click the Submit button more than once. Check the **"Don't allow duplicate submissions..."** box to allow only one submission per session. The form can be sent again if the user refreshes the page.

***Let VerifyForm send the form*:**
VerifyForm can be linked to any hyper text instead of to the form tag or a submit button. If you decide to trigger the Action this way, check this box. If you don't, your form won't be sent.

Turn on the ASP and Dynamic field name identifier.
If your form is set up as an **Active Server Page** or you are using **Dynamic Forms**, you must check the "Turn on ASP and Dynamic field name identifier" box at the bottom of VerifyForm's options window or the field names won't be recognized.

***Use NN6 mailto filter*:**
Sending a form directly to an e-mail address works with some browsers and not with others. For example NN6 and Opera5 MAC open the e-mail client instead of just sending the form, but don't include the form data. Go figure. IE5 for Mac & Win works fine.

The NN6 option is a quick workaround that will gather all the filled in fields of your form as **label:data** pairs, open your e-mail client and strip the data into the body of the e-mail. The filter only engages if you use a **mailto** action and the script detects a NN6 or Opera5 browser. No other browsers are affected. The best answer to this problem is to contact your ISP and ask if they have a user's **CGI form handler** you can use.

VerifyForm.Action

How to set up the Profanity Filter:

This filter can be used to catch any unwanted words, phrases or numbers entered in your form, and warn the user that they have made an inappropriate entry.

1 - Check the **"Use profanity filter"** box.

2 - Open the **VFProfan.js** file in either GoLive or a text editor.

3 - Enter your list of taboo words and phrases (in yellow) (Figure 13). Each word or phrase you enter must be separated by a **comma**.

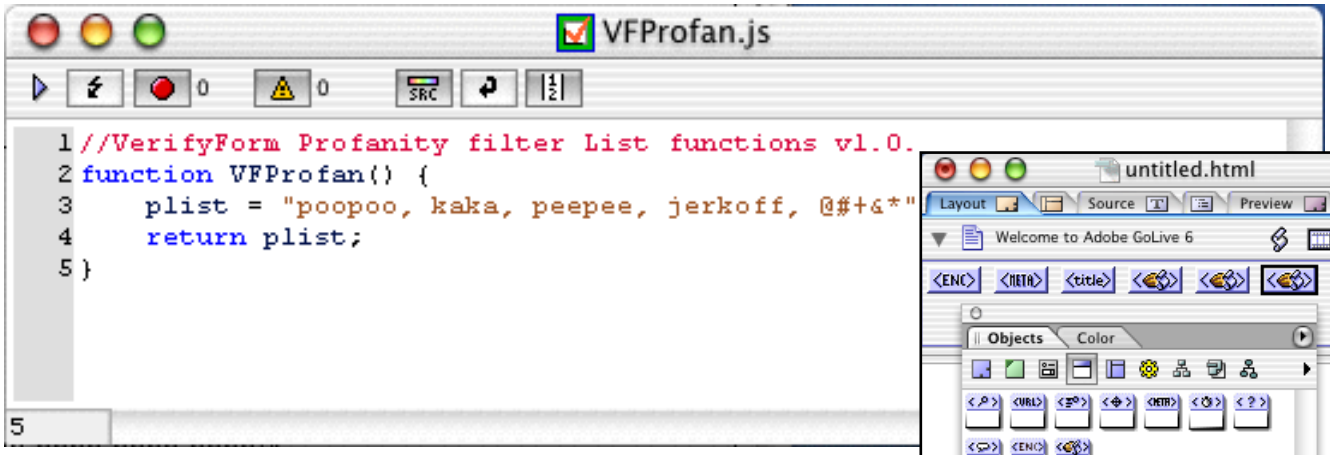

Figure 13

4 - Save and close the file.

5 - Put the VFProfan.js file into a suitable folder in your website.

6 - Drag a **Script icon** from the **Head** palette to the head section of your page (Figure 14).

7 - Check the **Source box** under the Inspector's tab then select the VFProfan.js file as the source file.

NOTE:
The VFProfan.js file can be renamed to whatever you want but you can only use one of these files per page.

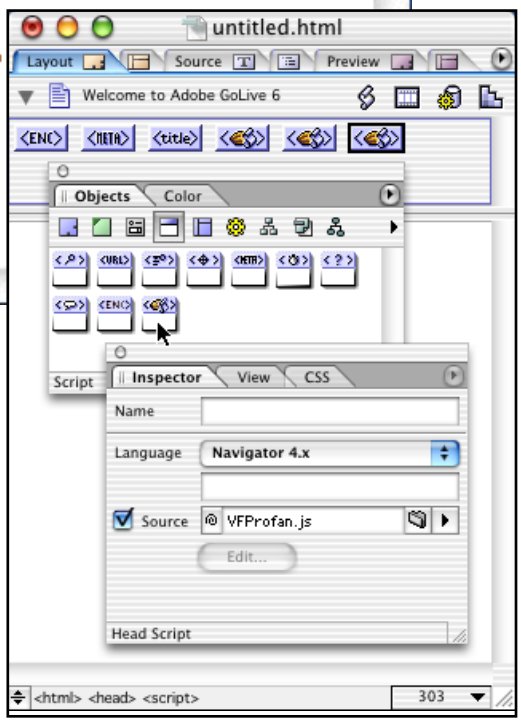

Figure 14

VerifyForm.Action

Error messages:

If the applicant fails to satisfy any of the criteria, VerifyForm will open a new window and display a detailed list of all the entry errors (Figure 15), and the form will not be submitted.

NOTE:
VerifyForm will also not send a form if all the fields are blank.

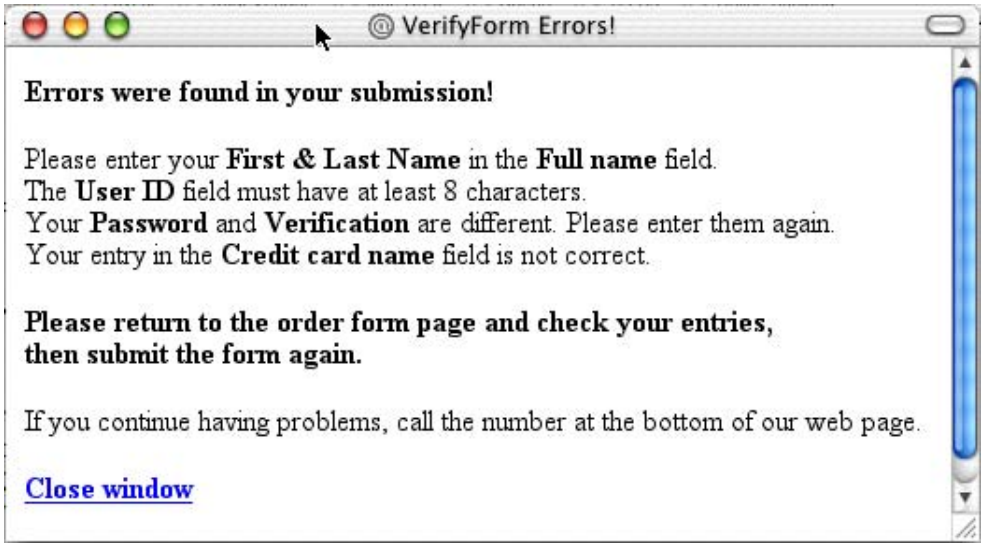

Figure 15

Editing VerifyForm's Alert messages:

All the displayed messages in VerifyForm are contained in an editable Javascript file called **VFMess.js**. This file must be linked to your page before you can use VerifyForm.

1 - Put the VFMess.js file in a suitable place in your website.

2 - Drag a Head Script icon to the head section of your page (Figure 16).

3 - Check the Inspector's **Source** box then select the VFMess.js file.

4 - If you want to change the display text to a more appropriate language, you can edit the messages in this file with GoLive. Edit the text **between the quotes** (in yellow) only (Figure 17).

4 - When you're through, click the **Syntax checker** in the upper right corner of your page (looks like a small lightning bolt) to make sure you haven't made any mistakes. Always work on a backup copy.

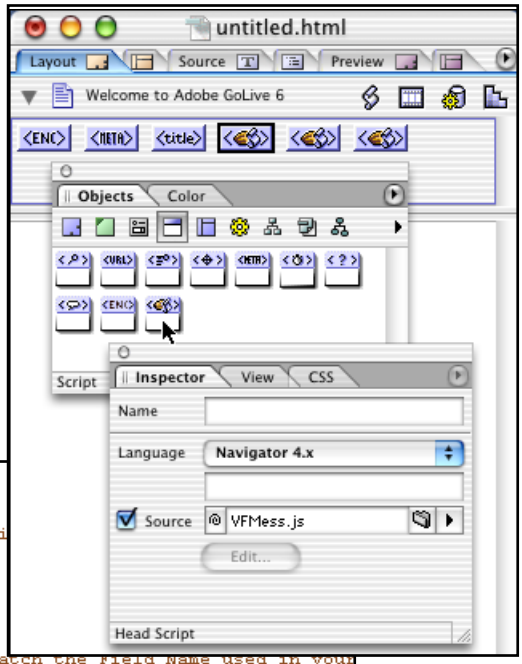

Figure 16

IMPORTANT:
You MUST include the VFMess.js file on every page where you use VerifyForm otherwise the script will produce errors. Watch your editing carefully and **only modify the words between the double quotes**.

```
1 //VerifyForm messages v1.0 - English.
2
3 //Error notice heading.
4 function VFmessErrorsFound() {return "<p><b>Errors were found in your submi
5
6 //Bad Form name message.
7 function VFmessFormName() {alert("The Form Name used in VerifyForm and the
8
9 //Bad Field name message.
10 function VFmessFieldName(action, i) {alert("'" + action[i] + "' does not match the Field Name used in your
11
12 //An entry is required message.
13 function VFmessRequired(action, i){return "Something must be entered in the <b>" + action[i+1] + "</b> fie
14
15 //Blank form message.
16 function VFmessBlank() {return "You have not entered any information. This form will not be sent.<br>";}
17
```

Figure 17

Error Window options:

There are options at the bottom of the VerifyForm options window that can be used to customize the the appearance of the Error Window (Figure 18).

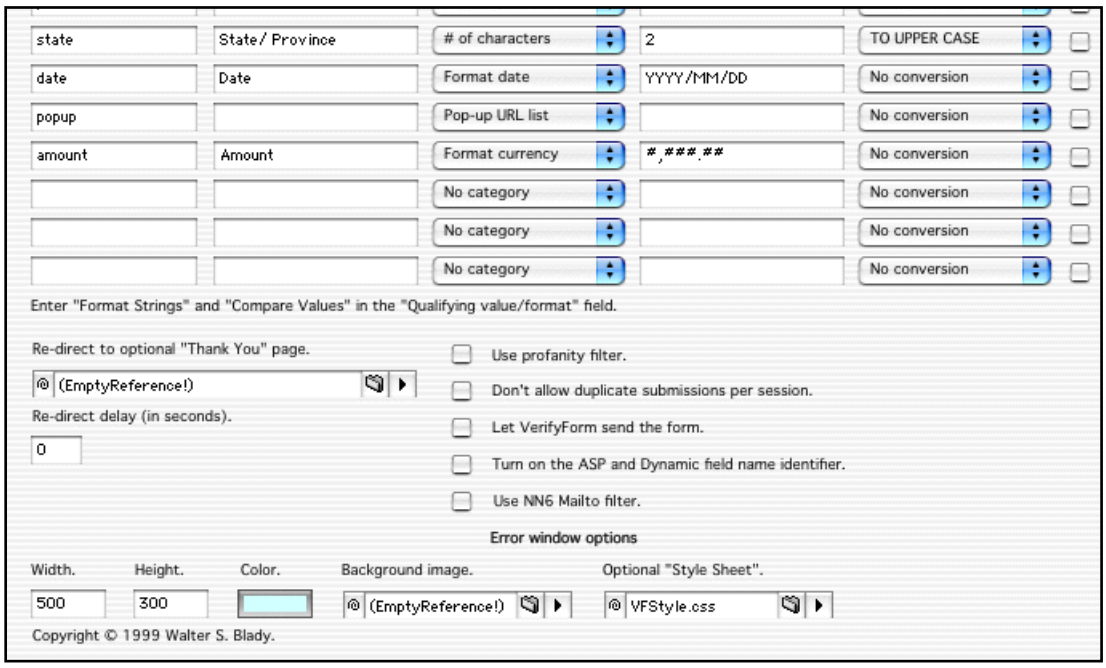

Figure 18

Width & Height:
Enter the width and height of the Error Window.

Color:
Select the background color for the window. Drag a color from GoLive's color picker to this color box.

Background image:
Select an optional background image for the window.

Optional "Style Sheet":
Select an optional **Style Sheet** for the error message text.

A simple style sheet called **VFStyle.css** comes with VerifyForm. This style sheet has only two text classes assigned to it called **"head"** and **"bodyText"**. The head class is used for the error message heading, and the bodyText class is used for everything else. You can change these classes to any font, color, point size etc. that suits your taste.

NOTE:
All error message text will be set to the browser default if you don't select a style sheet.

1 - Open the **VFStyle.css** file in GoLive and edit the **"head"** and **"bodyText"** text properties to your preferences.

2 - Save the revised file and put it with your site, then select it in VerifyForm.

VerifyForm.Action

Finalizing the Submit button setup:

The verification process performed by VerifyForm can be short circuited by some browsers if the user presses either the **Return** or **Enter** key while in a form field. Another problem that can happen is when the browser still submits the form even though the error message pops up. The problem isn't with VerifyForm, but with the way some browsers handle the **onSubmit()** event handler. There is a solution though.

This solution can be applied to either the standard Submit Button or an Input Image in GoLive 5+, and the graphic Submit Button in GoLive 4, but some source code modification is necessary. Even if you're unfamiliar or uncomfortable with modifying Javascript, the process is fairly simple if you watch the details, and is worth the results. Just follow these simple instruction.

Modifying the Graphic Submit Button in GoLive 4:

1 - Go to the **Source View** of your page.

2 - Figure 19 shows the **original source code**. Figure 20 show the same code with the modifications in red. Pay close attention to the position of the double quotes and semi-colons. Get these wrong and it won't work!

4 - Notice that the **"return CSClickReturn();"** in the onclick source line in Figure 19 has been removed.

CAUTION:
Every time you **edit** or **re-select** the VerifyForm Action, GoLive will put this line back into your source code... return CSClickReturn(); You'll have to strip it out every time you edit.

```
328     <body bgcolor="#ffffff">
329         <form name="testform" action="mailto:walter@wblady.com" method="post" enctype="text/plain">
330             <input type="text" name="fullname" size="24"> Full name (First &amp; Last)
331             <p><input type="text" name="userid" size="24"> User ID</p>
332             <p><input type="text" name="password" size="24"> Password</p>
333             <p><input type="text" name="email" size="24"> E-mail (user@domain.ext)</p>
334             <p><input type="text" name="cardname" size="24"> Card name</p>
335             <p><input type="text" name="cardnumber" size="24"> Credit card number (nnnn nnnn nnnn nnnn)
    </p>
336             <p><input type="text" name="phone" size="24"> Phone number (nnn-nnn-nnnn)</p>
337             <p><input type="text" name="state" size="24"> State/Prov<br>
338             </p>
339             <p> <a href="#" onclick="CSAction(new Array(/*CMP*/'B63D7D471')); return CSClickReturn();"
    csclick="B63D7D471"><input type="image" src="submit.jpg" align="absmiddle" name="testform" border="0">
    </a> <input type="reset"></p>
340         </form>
```

Figure 19

```
328     <body bgcolor="#ffffff">
329         <form name="testform" action="mailto:walter@wblady.com" method="post" enctype="text/plain"
    onSubmit="return CSAction(new Array(/*CMP*/'B63D7D471'));">
330             <input type="text" name="fullname" size="24"> Full name (First &amp; Last)
331             <p><input type="text" name="userid" size="24"> User ID</p>
332             <p><input type="text" name="password" size="24"> Password</p>
333             <p><input type="text" name="email" size="24"> E-mail (user@domain.ext)</p>
334             <p><input type="text" name="cardname" size="24"> Card name</p>
335             <p><input type="text" name="cardnumber" size="24"> Credit card number (nnnn nnnn nnnn nnnn)
    </p>
336             <p><input type="text" name="phone" size="24"> Phone number (nnn-nnn-nnnn)</p>
337             <p><input type="text" name="state" size="24"> State/Prov<br>
338             </p>
339             <p> <a href="#" onclick="return CSAction(new Array(/*CMP*/'B63D7D471'));" csclick=
    "B63D7D471"><input type="image" src="submit.jpg" align="absmiddle" name="testform" border="0"></a> <input
    type="reset"></p>
340         </form>
```

Figure 20

VerifyForm.Action

Modifying the source code for a Standard or Graphic Submit Button in GoLive 5+ (Standard button used here) - The modifications are simple in GoLive 5+. All you have to do is enter the word "**return**" in the appropriate place.

If you've attached VerifyForm to the **Form Tag**, then you add the word **return** after **onSubmit=** (Figure 21).

If you've attached VerifyForm to a **Submit Button** instead of the Form Tag, then you add the word **return** after **onclick=** (Figure 22). Do NOT add the word return when linking VerifyForm to anything other than the Form Tag or a Submit Button when using a mailto: action, because all browsers seem to handle it differently. You're on your own if you do.

CAUTION:
When you **edit** or **re-select** the VerifyForm Action, GoLive will remove the word **return**. You should watch for this and put it back.

If you're testing this page while in GoLive, remember to clear your browser's CACHE each time you change the source code.

```
543         <form action="mailto:walter@wblady.com" method="post" name="FormNamel" enctype=
    "text/plain" onsubmit="CSAction(new Array(/*CMP*/'B9826B880'));" cssubm="B9826B880">
544             <input type="text" name="namel" size="24" border="0">
545             <p><input type="submit" name="FormNamel" border="0"></p>
546         </form>
```

```
543         <form action="mailto:walter@wblady.com" method="post" name="FormNamel" enctype=
    "text/plain" onsubmit="return CSAction(new Array(/*CMP*/'B9826B880'));" cssubm="B9826B880"
544             <input type="text" name="namel" size="24" border="0">
545             <p><input type="submit" name="FormNamel" border="0"></p>
546         </form>
```

Figure 21

```
559         <form action="mailto:walter@wblady.com" method="post" name="FormName5">
560             <input type="text" name="name5" size="24" border="0">
561             <p><a onclick="CSAction(new Array(/*CMP*/'B98277A710'));return
    CSClickReturn();" href="#" csclick="B98277A710">submit</a></p>
562         </form>
```

```
559         <form action="mailto:walter@wblady.com" method="post" name="FormName5">
560             <input type="text" name="name5" size="24" border="0">
561             <p><a onclick="return CSAction(new Array(/*CMP*/'B98277A710'));return
    CSClickReturn();" href="#" csclick="B98277A710">submit</a></p>
562         </form>
```

Figure 22

VerifyForm.Action

A final check list before you save your page:

- Make sure that the **paths** to **VFMess.js**, **VFList.js** and **VFProfan.js** are correct.
- Check that the **Form Name** entered in VerifyForm is the same as the one used for the form.
- Make sure there are **no Spaces** in any of the Field names or the Form name.
- Make sure there are no **Single** or **Double Quotes** in any of the Field Description names.
- Make sure you haven't entered the names of **Check Boxes** or **Radio Buttons** in VerifyForm.
- Check that your **source code modifications** for the Submit Button are still intact.
- **Importing VerifyForm in an External Javascript file:** Check the top of the HTML page in **source view** and make sure the path to the CSScriptLib.js file points to the file on your server and NOT to the file on your hard drive before saving the page.

CAUTION:

If the browser is submitting your Form, even though VerifyForm has displayed error message, the problem is probably due to GoLive removing the word **"return"** from your source code. Make sure the word has not been removed.

Also, when an invalid Form is submitted without the error message appearing, it means that the browser has detected a Javascript error and has stopped running the VerifyForm script. The problem is most likely due to an editing error on either the **VFMess.js**, **VFList.js** or **VFProfan.js** files. Check your editing of these files very carefully.

Contact information:

Walter Blady
715 Varsity Estates Place N.W.
Calgary, AB T3B 4R1
Pho: (403) 286-2765
E-mail: walter@wblady.com
www.wblady.com