# HTML Forms for Business on the Web

Adding a simple form to your home page can increase user interaction and produce valuable information about your customers.

In October 1995, we looked at how to create a basic World Wide Web home page using the Hypertext Markup Language (HTML). Some of the more advanced capabilities of HTML can be applied to create a home page suitable for business needs. After all, being able to create a home page where interested parties can read your information is fine, but enabling visitors to provide feedback and interact with your home page—and ultimately your company—is of even greater value to your business.

We will look at some of the methods for creating Web pages to allow the user to enter and subsequently capture information for applications such as requests for information, being added to your mailing list, surveys, orders or just about anything else for which you want the user to provide you with information.

A standard Web page is by nature static. That is, the person reading your Web page is using Web browser software to interpret the HTML file you have created and placed on your Web server. The HTML document is simply called up on the server and delivered in a static, read-only fashion to the user.

To create dynamic Web pages for two-way communications that possess the capability to capture and store information, it's necessary to use additional HTML elements to design a form. Typically this is done by creating a script on the server, which is then executed by the Web server, which in turn creates the HTML document on the fly and delivers it to the user. Web scripts can be created in almost any programming language capable of reading from the standard input (keyboard), writing to the standard output (terminal) and accessing environment variables. Probably the most commonly used language for this is Perl, though C works well also.

Because the Web was originally designed as a one-way communications medium, a new protocol was invented that combines a number of elements to make two-way communications possible. The Common Gateway Interface (CGI) protocol is necessary for Web server and remote client to communicate with each other. The CGI protocol oversees the communication and transmission of data between the client on the Web server, the Web server itself and the script.

## Creating a Form

We're going to create a simple entry form to collect information for mailing lists. In the process, we will ask a few simple questions to gather some pertinent information about the individual submitting the information request, as well as the specific product he/she is interested in. We are going to include standard fields like name, address, company name, phone number, fax number and e-mail address, along with some check boxes and a free-form area in which the user may enter other comments or suggestions that are not included on the standard form. If I were actually to publish this home page, I would add other HTML elements to the page, like underlines or solid bars to break up the presentation of the information on the page. (See *UniForum Monthly*, Oct. 1995, for information on using some of these elements.)

As we proceed, please refer to the numbered listing for our home page form document. We'll break it down to each of the new HTML elements or markup tags used to create the form. If you have access to Web browser software and have not created a home page, you can use the following to create an HTML document and see how the page would appear if published on the Web. For clarity here, HTML tags are displayed in uppercase letters. In reality, HTML doesn't care if it is upper- or lowercase. However, uppercase markup tags are easier to locate during editing.

In the first eight lines, we declare this as an HTML document. We also declare (and later undeclare) the initial header, the document title and a reverse hypertext link to this site's Webmaster, so the reader provides feedback on this particular Web page to whoever "made" it; this is generally a useful feature. The remainder includes a reference to a GIF file for an image and secondary header, Information Request, declared and undeclared with <H1> and </H1>.

Line 9 is the prelude to the interactive form section of the document, where the individual viewing this page will provide the information we want in order to be added to our mailing list. To use the necessary HTML markup tags for creating forms, though, we have to declare the portion of the document that contains the forms tags we will be using. Let's use the <FORM> and </FORM> tag pair to encapsulate the section of the document containing the interactive portion of the form. The FORM tag with the ACTION attribute

By Glenn K. Schulke

in line 11 indicates what will be done with the form once completed.

Lines 12 and 13 use the real backbone of HTML forms, the INPUT tag or element. Additional attributes can be assigned to INPUT, including ALIGN for alignment left, middle or right; CHECKED to automatically check a checkbox without input from the user; MAXLENGTH to indicate the maximum length of an input field; NAME to assign a variable name to the INPUT field; SIZE to specify the size of the input box (if the MAXLENGTH is greater than the SIZE, the text will scroll in the box) and TYPE. The most commonly used values for the TYPE attribute are *text* for textual information, *password* for password entry, *checkbox* for a box to be checked off, *radio* for a radio button, *submit* to submit the form for processing and *reset* to clear all entries on the form and reset any checkboxes, radio buttons, and so on to their original default values.

In these lines we use two checkboxes

## Creating a Simple Home Page Form

```
1)  <HTML>
2)  <HEAD>
3)  <TITLE> XYZ Inc.'s Request for Additional Information</TITLE>
4)  <LINK REV="made" HREF=" webmaster@xyz.com">
5)  </HEAD>
6)  <BODY>
7)  <IMG SRC="/gifs/xyxlogo.gif" ALT="XYZ Inc.">
8)  <H1>Information Request</H1>
9)  Please enter your information request and contact information and it will
10) be emailed to XYZ Inc. staff.<P>
11) <FORM ACTION="/htbin/email/info@xyz.com">
12) <INPUT TYPE="checkbox" NAME="general" CHECKED> General Information<BR>
13) <INPUT TYPE="checkbox" NAME="new"> New Product Information<BR>
14) <I>What new product information would you like?</I><BR>
15) <SELECT NAME="INFO" SIZE="2" MULTIPLE>
16) <OPTION SELECTED>Standard Gizmos
17) <OPTION SELECTED>High Performance Gizmos
18) <OPTION SELECTED>Plasma Processor Gizmos
19) <OPTION SELECTED>Liquid Cooled Plasma Gizmos
20) </SELECT>
21) </P>
22) <INPUT NAME="fname" SIZE="30"> First Name<BR>
23) <INPUT NAME="lname" SIZE="30"> Last Name <BR>
24) <INPUT NAME="address1" SIZE="30"> Address 1 <BR>
25) <INPUT NAME="address2" SIZE="30"> Address 2 <BR>
26) <INPUT NAME="city" SIZE="20"> City <BR>
27) <INPUT NAME="state" SIZE="2"> State <BR>
28) <INPUT NAME="zip" SIZE="10"> Zipcode <BR>
29) <INPUT NAME="email" SIZE="30"> Your email address <BR>
30) <INPUT NAME="subject" SIZE="30"> Subject <BR>
31) <INPUT NAME="phone" SIZE="30"> Phone <BR>
32) <INPUT NAME="fax" SIZE="30"> FAX<BR>
33) <P>
34) Comments: <BR>
35) <TEXTAREA NAME="text" ROWS=8 COLS=60>
36) </TEXTAREA>
37) <INPUT TYPE="submit" VALUE="Submit">
38) <INPUT TYPE="reset" VALUE="Reset"> <P>
39) </FORM>
40) <HR>
41) <ADDRESS>&lt;info@xyz.com&gt;</ADDRESS>
42) <A HREF="/Welcome.htm"><IMG SRC="/gifs/xyzhome.gif" ALT="HOME"></A>
43) </BODY>
44) </HTML>
```

to determine the type of information the user would like, general or new products. The first checkbox uses an additional tag called CHECKED, which will automatically present the box as being checked. By doing this, we arrange for the person responding to this page to be sent general information on the company without having to check the box.

## New Product Information

Lines 14 through 21 ask what information on our new products the user would like. If the user checks the New Product box, a pop-up list is presented next to the italicized line (the <I> and </I> tags), asking, "What new product information would you like?" Let's look at how to implement the pop-up list under HTML.

This list begins on line 15, with a net HTML tag called SELECT followed by four available options for the SELECT tag, as indicated with the OPTION SELECTED tags. However, I've added two more options to the SELECT tag: SIZE and MULTIPLE. Without these tags, the pop-up list would simply appear and allow the user to select only one of the items presented on the list. By using the SIZE option, a window pops up with the only the first two items from the list. However, a scroll bar is also displayed, allowing the user to scroll up and down through the list. In this example, the list of only four items could be displayed easily on the screen; with a longer list, this would become an important feature. So, other than a window with only two items and a scroll bar, SELECT is essentially unchanged.

With the addition of the MULTIPLE option comes a change in the behavior of the pop-up list. When MULTIPLE is included with the SELECT tag, the user can select more than one item from the list. By clicking the mouse on one of the options and then moving the mouse up or down the list, multiple options can be selected. We complete this section with the </P> or paragraph tag to create a new line.

We could also have used multiple radio buttons or checkboxes, but for larger lists that would get cumbersome. One drawback of using the SELECT tag, however, is developing the necessary code to parse multiple values with the same names into individual data elements. To my knowledge, HTML does not provide an elegant method of breaking down the data stream. As such, you will need to handle this in your back-end script or program.

## Capturing the User

Lines 22 through 33 use a series of named INPUT elements to capture the information we need to add a person to the mailing list. When using named INPUT elements, the value assigned to the NAME attribute is a key to the proper operation of your form, as it provides a unique identifier to the information to be contained by INPUT. Just as variables in a program should be unique, so should the named SELECT elements. If they are not, you end up with a routine that continuously writes new data over the old, negating any value to the data collected.

**To create dynamic Web pages that can capture and store information, it's necessary to design a form using additional HTML elements.**

The name assigned to the named INPUT is arbitrary and could be just about anything. I prefer to use names that are easy to remember and which could be used in developing a database schema for eventual importation into a database.

In each of the above INPUT elements we also define a SIZE attribute for each, along with assigning a name to be displayed to the user—First Name, Last Name, and so on. Finish each line using the <BR> element, which creates a new line and also white space by adding a blank line to the document, giving additional separation to each line for readability. Again, we complete this section with the </P> or paragraph tag to create a new line.

The final section of this document, lines 34 through 44, allows the user to input additional comments. To accomplish this, we use the <TEXTAREA> and </TEXTAREA> tags. As with the names assigned to INPUT elements above, we assign a unique name to TEXTAREA to capture the data.

We begin at line 34, with the text "Comments" displayed, followed by the <BR> tag. Next, the TEXTAREA tag defines our text variable, called simply "text," finishing with a ROW and COLS definition to display a box for the text that will be 8 lines long and 60 columns wide. (For future use, please note that, although you have defined a text box that is 8 lines by 60 columns, there is no limitation on how much text one can actually enter into the box. Therefore, when you are implementing your program or script to parse and capture the data, be sure to create a field to capture more data than the text box can.)

Next, lines 37 and 38 use the *submit* and *reset* input types to either clear and reset the forms default values or submit the form for processing. The form is now complete, so in the remaining lines 39 through 44, use the </FORM> tag and sign the form with your e-mail address. Note *&lt* and *&gt* at the beginning and end of the address. These are special insertion codes to insert specific characters into the document. If used, they are followed by a semicolon to indicate the end of the insertion code. In this case, the "less than" and "greater than" symbols will surround the address text.

We finish with a hypertext anchor and link back to the home page using an anchored image. To compete the page, we contain our original declarations of the document body and the document itself.

This is just a starter on using HTML forms for creating interactive forms. There is a great deal more you could do. Further, by using a shell script, C program or Perl routine, you can easily capture the information and ship it off to your database. IT

***Glenn K. Schulke*** *is president of Open Technologies, Inc., a systems integrator specializing in software integration services, located in Tempe, AZ. He can be reached at gschulke@aol.com.*