Forms

Forms are a very important part of HTML. It is a way of soliciting information from customers or students.

A form requests the server to run a program. Thus, a form is associated with a program on the server. Forms are used to collect data. It does not do any processing of the data. Thus, the only way for one to get a form to do anything useful is to send the data gathered by the form to a program on the server. The program on the server will then take the information gathered by the form and process it. The server-side, data-processing aspects of forms are not part of the HTML or XHTML standards; they are defined by the server's software.

<form=options>
  items in list
</form>

Creates a simple form. It defines the form.

<form action="url" method="method" enctype="type">
  
  • ACTION is the only required attribute. The URL is executed when the form is submitted.
  • METHOD is either GET or POST. GET tacks all data onto the url when it calls the program. This is the default. It is not generally used. It can sometimes result in your form data being truncated before the program ever receives it. POST is a more reliable delivery method. It is the preferred method.
  • ENCTYPE is included for completeness. It only has one value: "x-www-form-encoded". This is the default, so it can be ignored until it sprouts more options in future versions of HTML.

Following the ACTION element is the URL. For example, <form action="/cgi_test.cgi"> will display the responses that each client responded to the questions on the form. cgi_test.cgi is a compiled c-program that simply displays the responses. cgi_test.cgi is in the same directory as the form. If not, a more detailed description of the URL would be necessary. The processing program can be written in other languages, for example, PERL, JAVA, Active Server Page (ASP), or PHP. PHP is a widely-used general-purpose scripting language that is especially suited for web development and can be embedded into HTML. If you are interested in learning more about PHP try the following web site: http://www.php.net/. Microsoft adds some other choices such as WEBBOT. The default is to save to file WEBBOT. You can also mail to WEBBOT.

Form Input Elements:

ALL of the following Form Input Elements must be used within the <form></form> tags. Using separate <form></form> tags for each of them, creates separate forms. Thus, one usually will
use all of the tags in one form. Input elements include text, radio buttons, checkbox, text area, reset buttons, and submit buttons.

**Text**

```html
<input type="text" size=number maxlength=number name="field_name" value="field_value">
```

A box will appear on the page with the designated size, measured in characters. NAME defines the name of the data. It is required. One would type the text into the box. For example,

```html
<input type="text" size=32 maxlength=32 name="yourname" value="field_value">
```

creates a box of 32 characters to receive one's name as input. VALUE allows for a default value to be displayed. The MAXLENGTH tag restricts the number of characters that one can type in the space. Other attributes include SIZE, NAME, and VALUE. SIZE is the length of the display window to be seen on the form and it is indicated in pixels.

**Radio Button**

```html
<input type="radio" name="field_name" value="field_value">
```

One way of assuring easy and accurate data entry is to allow the user to select only one choice from a provided list. One way of providing a list is by displaying radio buttons for the client to select one's choice. For example,

```html
<input type="radio" name="sport" value="baseball">Baseball
<input type="radio" name="sport" value="football">Football
<input type="radio" name="sport" value="soccer">Soccer
<input type="radio" name="sport" value="hockey" checked>Hockey
```

allows one to check the appropriate button. Items in the list are grouped together by sharing a common NAME. Items can be displayed using list tags or line breaks. VALUE is the default value to be displayed. The values in this case are "baseball", "football", and "soccer", By including the word CHECKED at the end of the INPUT tag just before the >, a default value, which automatically appears as selected, is made.

**Check Box**

```html
<input type="checkbox" name="field_name" value="field_value">
```

This tag creates check boxes for the client to select one's choice. For example,

```html
<input type="checkbox" name="size" value="small">Small
<input type="checkbox" name="size" value="medium">Medium
<input type="checkbox" name="size" value="large" checked>Large
```
This allows one to check the appropriate box and one can select more than one checkbox at a time. Items in the list are grouped together by sharing a common NAME. Items can be displayed using list tags or line breaks. VALUE is the default value to be displayed. The values in this case are "small", "medium", and "large". By including the word CHECKED at the end of the INPUT tag just before the >, a default value, which automatically appears as selected, is made. One does not have to mark one as CHECKED. This allows the manufacturer to indicate that no selection by the individual will result in this case a "large" t-shirt being provided for example.

**Pull Down Menus and Scroll Boxes**

TYPE can also have a value of SELECT. SELECT provides people with a drop down menu or scroll box to select from. With a long list of items to select from, this is the preferred way of soliciting information. Sometimes it is a message displayed rather than one of the choices. This is appropriate otherwise the default choice might be chosen more than it should be chosen.

```html
<select name="field_name" size="number">
  <option selected> value
  <option> value
  <option> value
  ...
  <option> value
</select>
```

Pull-down menus and scroll boxes can be assigned a default value using the OPTION SELECTED tag. NAME attribute is required. It defines the name of the data. SIZE attribute determines how many items will be displayed. If SIZE is omitted or set to 1, then choices will appear as a pop-up menu. If SIZE is set to 2 or more, then the choices appear as a scroll box. MULTIPLE attribute will allow more than one selection to be made. It will force a scroll box. VALUE attribute is the value assigned to a choice. For example,

```html
<select name="color" size="1">
  <option selected> blue
  <option> red
  <option> green
  <option> yellow
  <option> pink
  <option> orange
</select>
```

**Text Area**

Sometimes, it is very helpful to solicit comments back from customers or students. Here a larger area of several lines is needed so that the person can expound upon their thoughts.

```html
<textarea name="field_name" rows=number cols=number>
</textarea>
```
This displays a window or text area on the form, with the indicated number of rows and columns for one to type a response into. For example,

```html
<textarea name="comments" rows=5 cols=40>
</textarea>
```

provides a window of 40 columns by 5 rows for one to type in information.

**Reset Button**

```html
<input type="reset" value="value">
```

This push button is a pre-set displayed value that resets the form to the original choices, usually blank or default checks, so that one may start over.

**Submit Button**

```html
<input type="submit" value="value">
```

This push button is a pre-set displayed value that allows one to click the button when the form is completed in order to submit the form for processing. The data is sent to the server for processing.

One of the variations for the submit button is as follows:

```html
<INPUT TYPE="image" NAME="sub" SRC="go.gif">
```

This uses the image as the submit button on the form. In other words, when the image is clicked on, the form is sent for processing.

**Fieldsets and Legends**

A fieldset is used to group similar information on a form. The `<legend>` tag within the fieldset tag is optional. Using the `<legend>` tag creates a legend for the fieldset, which is the text that appears in the grouping borders.

The code might be as follows:

```html
<fieldset><legend align="left">Required Information</legend>

</fieldset>
```