HTML Tutorial 3 –Forms

Why use a Form?

Basic HTML provides a host of features which allow the designer to define the way in which text and images are to be displayed on a page. But this simple collection of commands has no way of allowing the website owner to collect information from the visitor. Such an operation can prove to be very useful - especially if the site is a commercial one - and it's helpful to collect details on visitors who wish to be sent detailed information at a later date. On a more personal level it may be desirable to collect email addresses of visitors so they may be contacted when the site details are updated.

In order to help collect such information, HTML provides us with a series of features collectively known as form elements. From text boxes which allow details such as names to be entered to user-defined selections and drop-down boxes, forms provide a highly flexible way to gather the information which can be emailed to a particular individual or sent to a server for more specialised processing.

We will examine the various form controls and discuss ways in which the data they gather can be passed on.

The <FORM> tag

This container pair tells the browser that whatever is contained between the <FORM> and </FORM> tags is a set of commands which are designed specifically for the collection of information through a web page which will be passed to the page owner.

The FORM definition may contain a number of attributes which tell the browser exactly what to do with the data once it has been entered.

These attributes are:

ACTION
The action attribute defines which procedure will take place when the data is being sent. Usually ACTION takes one of two values - the URL of a suitable server script which will accept the form data and process it or a mailto: declaration which gives the email address of the recipient of the form information.

METHOD
METHOD defines the way in which the data from the Form is physically transmitted over the Internet. The Method accepts one of two attributes:

POST
This approach is the most commonly used and simply sends the form information to the relevant CCI script or email address.

GET
Using the GET attribute forces the browser to pass the form information by appending it to the URL specified in the GET command. Each portion of information is separated from the rest by means of the '?' character.

Whilst there are several situations where this method of sending data is most suited, it can be a tricky beast to deal with for any number of reasons, and so it's probably best to avoid it.

Input tags for Forms

Once the <FORM> definition has been created it is necessary to place some commands within it to accept the desired data.

The <INPUT> tag is used for this purpose and collects data typed from the keyboard. There are a number of other ways to collect information and we shall examine those later.
.
INPUT can take several forms but unlike the <FORM> tag itself is not a container and does not carry a </INPUT> partner tag.

The format of the tag is:

```
<INPUT  NAME = "aname"  TYPE = "(see below)">
```

Each input tag should have a unique NAME associated with it. Failing to associate such a name can result in unpredictable results and incorrect data being returned.

TYPE=TEXT
The text type displays an input box which is one line deep. It carries several sub-attributes to define the way in which data will be accepted:

SIZE
The length (in characters) of the input box on-screen.

VALUE
The initial text which will be displayed in the textbox. This is useful if it is necessary to set a default value for a particular textbox.

MAXLENGTH
The maximum length of the input (in characters). This attribute is very useful for instances where only a certain number of characters are required - telephone numbers for instance.

TYPE=PASSWORD
The PASSWORD attribute is identical to its TEXT cousin with the exception that when the user types information into the associated onscreen box a star character '*' is

displayed for each ordinary character entered. This can be useful for providing access to restricted areas of a website or providing special user identification in a form.

TYPE=TEXTAREA
The TEXTAREA expands on the TEXT tag, providing an extended box which allows users to enter data over several lines.

TEXTAREA is a container pair and any normal text between the <TEXTAREA> and </TEXTAREA> will be placed into the on-screen box as the default entry.

Attributes for <TEXTAREA> are:

ROWS
Specified in characters this value tells the browser how many lines high to make the input box. Users can type as many lines of text as they like, but if the number entered exceeds the number specified by ROWS, a scroll bar will appear within the box to indicate the depth of the entry.

COLS
Specified in characters this value tells the browser how wide to make the TEXTAREA. If lines entered by users exceed the value specified by COLS some browsers will provide scrollbars in a similar way to that described above whilst others will simply wrap the text within the TEXTAREA.

In addition to the text entry tags there are also a number of INPUT commands which will allow choices and selections to be made:

TYPE=CHECKBOX
The checkbox type performs a task similar to the multiple selection boxes often found on questionnaires. Checkboxes can be used to allow those filling in the form to select just one item from a list or multiple items.

In its simplest form the CHECKBOX allows users the choice of 'on' or 'off - useful for including those boxes where visitors can tick a box to request further information or leave it empty if the info is not required.

To produce this type of box use a variant of the following code:

```
<INPUT  TYPE = CHECKBOX  NAME = "moreinfo"  VALUE = "sendit">
```

Using this code the checkbox, which is named moreinfo, will send a value of 'sendit' if the user clicks in the box. If the box remains unchecked no value will be sent.

To create a list of multiple items where several can be selected it is necessary to create several instances of the CHECKBOX tag, where each instance contains the same NAME but differing values.

This approach allows any of the boxes to be ticked and will send the VALUEs of all those selected when the form is submitted.

The following code segment will produce a multiple selection item:

Please send me more information on:
Floppy Disks <INPUT TYPE=CHECKBOX NAME ="request"
VALUE = "floppy disks">
Hard Disks <INPUT TYPE=CHECKBOX NAME = "request"
VALUE = “hard disks">
CD ROMS <INPUT TYPE=CHECKBOX NAME=”request”
VALUE = "CD ROMS">
Zip Disks <INPUT TYPE=CHECKBOX NAME= "request”
VALUE-"zip disks">

Giving one of the CHECKBOX definitions the additional attribute
CHECKED = TRUE will display that CHECKBOX with an initial tick (showing it's the default option).

TYPE=RADIO
It is not always desirable to allow multiple selections from a list and sometimes it's better to have them make one, and only one, selection.

The radio button allows us to do this (it's called a radio button as it mimics the operation of push button radios where only one frequency button could be selected).

As with the CHECKBOX, each radio button definition for a selection set should have the same NAME and different VALUEs. Adding CHECKED=TRUE to one of the items in the list will set it as the default.

The following code fragment demonstrates the use of the radio button:

<INPUT TYPE=RADIO NAME = "moreinfo" VALUE = "yes"
CHECKED="YES">
<INPUT TYPE=RADIO NAME = "moreinfo" VALUE = "no">

SELECTions
The <SELECT> tag embodies the best parts from the previous two forms of list, allowing one or several selections to be made from a single list.

The SELECT container pair holds a number of instances of the <OPTION> tag - a one line tag which contains a text item.

This text item will appear as part of the list when it is displayed on-screen and also provides the text which will be returned to the server or email address when the form is submitted.

Adding the attribute SELECTED to an OPTION tag will result in the item being highlighted as the default.

If the MULTIPLE attribute is added to the <SELECT> tag visitors may select more than one item from the list which will be displayed in a box with a scrollbar. Omitting MULTIPLE will result in a drop-down box from which only one option can be selected.

The following code fragment illustrates the way in which the SELECT tag is used:

```
What's your favourite colour? <P>
<SELECT>
        <OPTION>Orange
        <OPTION SELECTED>Red
        <OPTION>Yellow
        <OPTION> Purple
        <OPTION >Green
<SELECT>
```

TYPE=HIDDEN
Just to prove that every conceivable situation has been catered for, HTML provides us with the HIDDEN type, which is invisible to the user and cannot be changed by on-screen input.

Because of the way HTML, Forms and World Wide Web servers communicate it is often impossible to detect if Forms have been filled completed in a particular way.

By combining HIDDEN data with CGI (Common Gateway Interface) scripts it is possible to track such information and even to introduce new, hidden information which allows the way the forms appear and control to be dictated.

TYPE=SUBMIT and TYPE=RESET
These two types mark the final elements in a <FORM> which allow the user, once data has been entered, to either send it or to clear it completely and re-start.

The buttons take the format:

```
<INPUT  TYPE= SUBMIT  VALUE="Enter details">
<INPUT  TYPE= RESET  VALUE="Start again">
```

The VALUE attribute determines what text will appear on each button.

In order to make the SUBMIT button more attractive it is possible to use TYPE=IMAGE in its place. It does the same job - when it's clicked it sends the form data - but it allows a graphic image to be used in place of the rather drab grey button SUBMIT presents us with. An example of this tag is:

```
<INPUT TYPE="IMAGE" SRC="mygrafic.gif" ALIGN="LEFT">
```

Forms - An Example

```
<FORM METHOD=”POST” ACTION="mailto:mymail@ukonline.co.uk">

Please enter your details-<P>

Name: <INPUT TYPE="text" SIZE="40" NAME= “name" >                    (1)

<p>
Address: <TEXTAREA NAME="address" ROWS="3" COLS=                     (2)
“40"></TEXTAREA>

Age: 0-10<INPUT TYPE="radio" checked NAME="age”
value="0-10">
        l-20<INPUT TYPE="radio" NAME= “age”                          (3)
value="l1-20">
        21-30<INPUT TYPE= “radio" NAME="age"
value= “21-30">
        31-40<INPUT TYPE="radio" NAME= "age"
value= “31-40">
        41-50<INPUT TYPE= radio NAME=”age"
value='41-50'>

<p>
Your Interests: <br>
        < INPUT TYPE= “checkbox" NAME=”Interests"                    (4)
VALUE = "Gardening" >Gardening<br>
        <INPUT TYPE="checkbox" NAME= "Interests"
VALUE= “Motoring">Motoring<br>
        <INPUT TYPE="checkbox" NAME=”Interests"
VALUE= "Computing"> Computing<br>

<br>
Your earnings:
        <SELECT NAME="eamings" size="1">                             (5)
        <OPTION> £0-5,000
        <OPTION> £5,001-£10,000
        <OPTION >£10,001 -£20,000
        <OPTION>£20,001-£30,000
        <OPTION>£30,001-£40,000
</SELECT><p>

<INPUT TYPE="submit" name="Submit"
<INPUT TYPE="reset" name="Reset Form" value="Reset" >                (6)
```

The form works in the following way:

1 Creates a single-line textbox 40 characters wide which will accept the users name.

2 A multi-line textbox 40 characters wide by 3 lines high which will accept the users address.

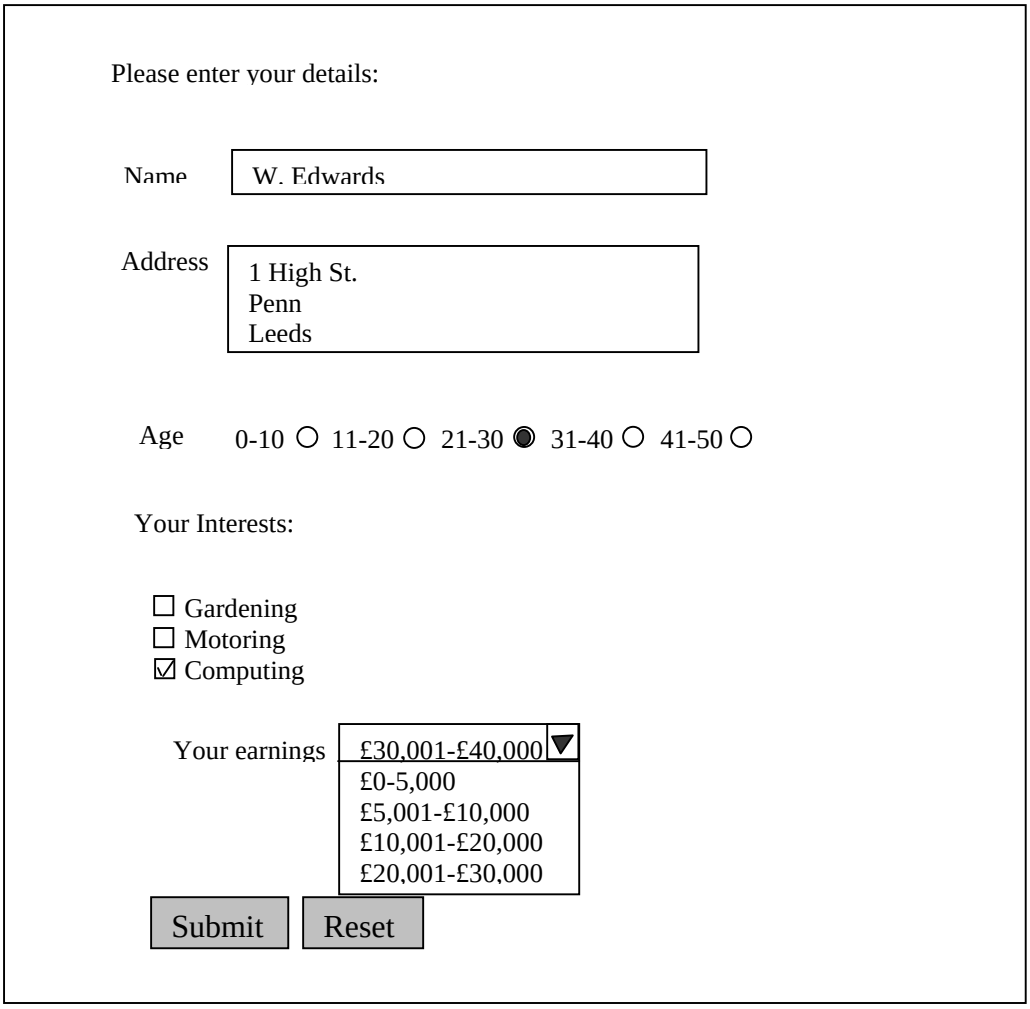
3 This radio button set allows the user to define his or her age. Note that the first box (0-11 years) contains the attribute CHECKED which represents the default value.

4 Checkboxes which allow the user to select several interest areas.

5 A SELECTion which presents a drop-down box from which the user can select only one value. If the <SELECT> tag had contained the MULTIPLE attribute several values could have been selected.

6 The SUBMIT and RESET buttons are defined here. Failing to include these will render the form useless as there is no way to tell the browser when the form filling has been completed.

When the web page is accessed it will look something like this:



Exercise

Add a new page to your Computing web site so that a prospective student can request more information about a course by completing a form. Typical information should include: Name, Address, Telephone number, Course interested in receiving more information about, etc.