

Chapter 5. HTML Forms

Table of Contents

Objectives.....	2
5.1 Introduction.....	2
5.1.1 Processing Forms	2
5.2 Creating Forms	3
5.2.1 Starting a Form.....	3
5.2.2 Single-line Text Entry	4
5.2.3 Multi-line Text Entry.....	6
5.2.4 Radio Buttons	7
5.2.5 Check Boxes.....	9
5.2.6 Menu Buttons and Scrolling Lists	10
5.2.7 Submit and Reset Buttons	13
5.3 HTML5 form elements	14
5.3.1 Email address field	14
5.3.2 Search.....	14
5.3.3 Url	15
5.3.4 Autofocus	15
5.3.5 Dates and Times	15
5.3.6 Color.....	15
5.3.7 Numbers as Spinboxes	15
5.3.8 Numbers as Sliders	16
5.4 Using Forms.....	16
5.5 Additional Content and Activities	17
5.5.1 Supplementary information on HTML forms.....	17
5.5.2 Additional Activity — Tabular Layout of a Complex Form	17
5.6 Review Questions	18
5.7 Discussions and Answers.....	20
5.7.1 Discussion of Activity 1	20
5.7.2 Discussion of Activity 2	20
5.7.3 Discussion of Activity 3	21
5.7.4 Discussion of Activity 4	23
5.7.5 Discussion of Activity 5	25
5.7.6 Discussion of Activity 6	25
5.7.7 Discussion of Activity 7	28
5.7.8 Discussion of Activity 9	29
5.7.9 Discussion of Additional Activity	31
5.7.10 Answer to Review Question 1	32
5.7.11 Answer to Review Question 2	33
5.7.12 Answer to Review Question 3	33
5.7.13 Answer to Review Question 4	33
5.7.14 Answer to Review Question 5	33
5.7.15 Answer to Review Question 6	33
5.7.16 Answer to Review Question 7	33
5.7.17 Answer to Review Question 8	33
5.7.18 Answer to Exercise 1	34
5.7.19 Answer to Exercise 2	34
5.7.20 Answer to Exercise 3	34

Objectives

At the end of this chapter you will be able to:

- Create forms with basic elements such as text boxes and buttons:
- Create forms using HTML5 elements such as form validation and email address fields.

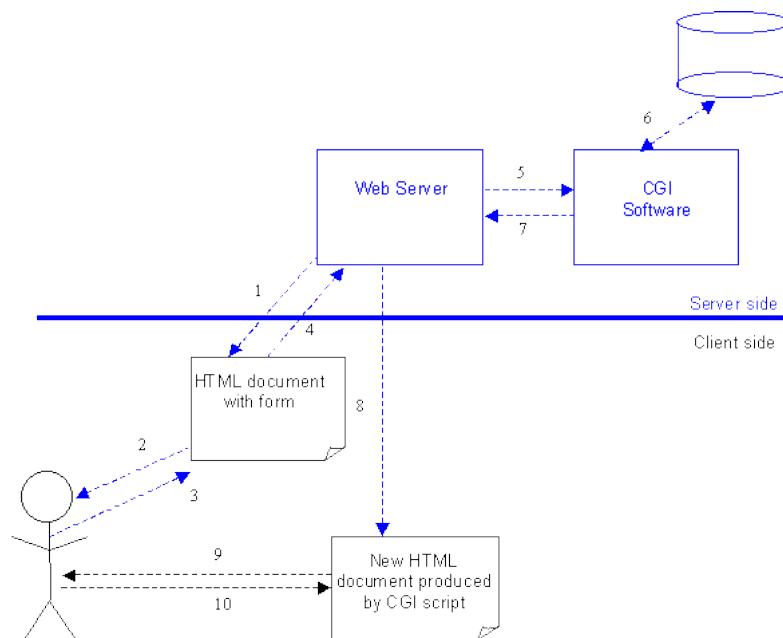
5.1 Introduction

Forms are best learnt using a hands on approach. To become proficient with HTML forms you need to create many, sorting out the problematic nuances as you go along. Therefore, the main content of the unit is a series of sections: the first is a short introduction to HTML forms; the second discusses each form element, and involves some textbook study. (You may find it more convenient to postpone activities until you have covered all the form elements).

This introduction covers the main form elements. It also explains the process that occurs when a form is submitted. The main elements of forms are: Text fields; Password fields; Text areas; Radio buttons; Check boxes; Menu buttons and scrolling lists; Submit and reset buttons; and file picker. HTML5 defines a number of new input types that can be used in forms. Examples are Email address fields; web address fields; numbers as spin boxes and sliders; date pickers; search boxes; color pickers; form validation; and required fields. We will look at some of these in this chapter.

5.1.1 Processing Forms

Although forms could simply be used to display information, HTML provides them in order to supply a way for the user to interact with a Web server. The most widely used method to process the data submitted through a form is to send it to server-side software typically written in a scripting language, although any programming language can be used. The figure below outlines the kind of processing that takes place.



1. The user retrieves a document containing a form from a Web server.
2. The user reads the Web page and interacts with the form it contains.
3. Submitting the form sends the form data to the server for processing.
4. The Web server passes the data to a CGI programme.
5. The CGI software may use database information or store data in a server-side database.

6. The CGI software may generate a new Web page for the server to return to the user.
7. The user reads the new Web document and may interact with it.

Typically, form data is sent to a server (or to an email address) as a sequence of pairs, each pair being made up of a name and an associated value. The method that this data uses to arrive at its destination depends on the data encoding. Normally the pairs will be sent as binary-encoded characters, making them straightforward to process by software, and easy to read by humans. For example, an on-line store selling used computer parts might use a form when ordering second-hand disk drives; the form would send to the server for processing information identifying the manufacturer, the model name, and maybe quote price thus:

```
manufacturer=syquest&model=e2135&price=45
```

This text represents a sequence of three name/value pairs. The names are **manufacturer**, **model** and **price**, and their associated values are *syquest*, *e2135* and *45*. There is nothing special about the names chosen or the way values are written, except that what is sent depends entirely on what the CGI software expects. If it expected **maker**, **item**, and **cost**, then the data from submitting the form would have to be:

```
maker=syquest&item=e2135&cost=45
```

Quite simply, whatever the processing software expects determines what the HTML form must provide. Often the same person or team develops both form and CGI software, so this is usually of little concern.

Because of the standard way in which the server-side software that process form data is supplied with data, such software is usually referred to as a Common Gateway Interface (CGI) script. Quite often CGI scripts on Unix servers are written in a language called Perl, but languages such as Python are becoming popular; when complex or fast processing is required, C, C++ or Java may be used.

To avoid server side programming when developing forms, and to avoid depending on scripts that may require considerable study, we will mostly use a different method of processing form information: email. In fact, it is very useful to submit form data to an email address, particularly in situations when the data should be seen by a human before being processed by software.

Review Questions

Do Review Questions 1-2.

5.2 Creating Forms

This section explores the main elements found on HTML forms. This is done in a manner matching the way many people develop forms — a little bit at a time. The discussion of each form element involves both reading from your textbook as well as a practical activity. (Some of the activities will take considerable time.) You may prefer to postpone doing Activities 1-7 until you have reached the end of the section on submit and reset buttons.

5.2.1 Starting a Form

All forms start with the `<FORM>` tag and end with `</FORM>`. All other form objects go between these two tags.

The form tag has two main properties: **METHOD** and **ACTION**.

METHOD refers to **post** or **get**. The **post** attribute will send the information from the form as a text document. The **get** attribute is used mostly with search engines, and will not be discussed. We will generally set **METHOD="post"**.

ACTION usually specifies the location of the CGI script that will process the form data. We are not using CGI

HTML Forms

scripts, and are instead setting this attribute to an imaginary email address (which causes the form data to be emailed to that address).

```
ACTION="mailto:put.your@email.address.here"
```

Putting these together gives us:

```
<FORM METHOD="post" ACTION="mailto:put.your@email.address.here"></FORM>
```

To Do

Read about Forms in your textbooks.

Activity 1: Starting a Form

This is the first step in creating a form. You may build on this activity in later ones by using the document you create here as a starting point (or you may prefer to save each different form with a name corresponding to the activity). First, create a new HTML document called form.htm.

Enter the normal <HEAD> and <BODY> tags. Then include the following <FORM> tag:

```
<FORM METHOD="post" ACTION="mailto:put.your@email.address.here">
```

Remember to close the form with the </FORM> tag. Press the return key a few times to move it down the page, as you will be entering further code.

Finally, view your work in a Web browser, but be warned: at the moment there is little to show!

Read Discussion of Activity 1 at the end of the Unit.

5.2.2 Single-line Text Entry

A single-line text entry allows the user to input a small amount of text, like a name or an email address.

Please input your name:

This is achieved with the following:

```
Please input your name: <INPUT TYPE="text" SIZE="40" MAXLENGTH="30"  
NAME="personal-name">
```

The tag has the following elements:

<INPUT> is the tag used for most of the form objects.

TYPE="text" sets the object to a single-line text field.

Size="40" sets the field to show 40 characters.

MAXLENGTH="30" means that only a total of 30 characters can be typed in this field.

NAME="personal-name" sets the text field's name to be personal-name (this information is part of the form data sent on for further processing). The name is required to identify the value data which will be associated with it. For example, if the text box was for an email address or telephone number, we might set this attribute

HTML Forms

with a more suggestive value, e.g. `NAME="email"` or `NAME="tel-no"`. The easiest way to choose the name is simply to use the purpose of the field.

VALUE=... Another attribute, **VALUE**, can be used to place an initial text in the field. As might be expected, if this text is left unchanged it becomes the value associated with the **NAME** when the form is submitted. Putting an initial value in the field is usually not required, but can be used as a prompt. For example, the following HTML produces the figure that comes after it:

```
Name: <INPUT TYPE="text" NAME="name" SIZE="35"
      VALUE="---please type here---">
```

Name:

Do not confuse the use of 'name' when we refer to the **NAME** attribute of an `<INPUT>` tag with the possibility of using **name** as the text field's actual name, or 'Name' being used in a text field that prompts the user for their own name (as in the example immediately above).

Exercise 1

After the following HTML form has been rendered by a browser, a user enters their age. The form is subsequently submitted for processing to a CGI script. Write down the name/value pair that is sent to the server-side software. Solution can be found at the end of the chapter.



A browser window showing a form with a text field labeled "Age:" containing the number "39".

To Do

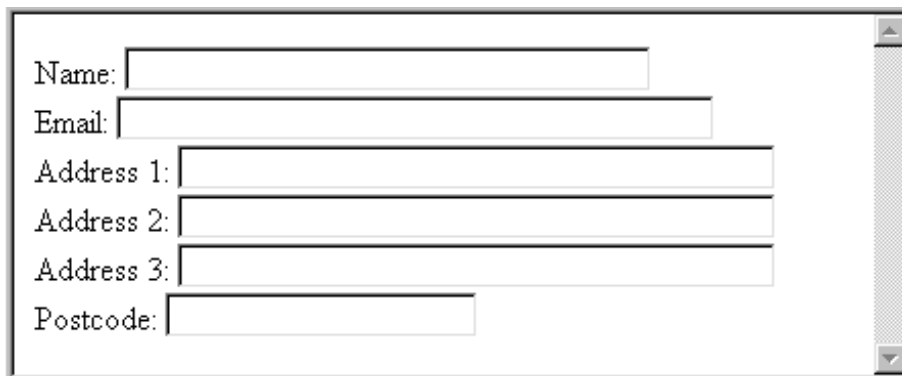
From your textbooks and Internet resources, read about the types of text that you can set for a text-entry, such as password and hidden types. Look at the Additional Content and Activities Section for some on-line resources.

Activity 2: Single-line Text Entry

In the HTML document you created before (form.htm), or a new one (as you prefer), create

1. A text field for your name that is 35 characters long.
2. A text field for your email that is 40 characters long.
3. Three text fields for your address, each 40 characters long, and an additional smaller field for your postal code.

Test your code to ensure that it produces something as shown below. You may find it convenient to implement and test each part of the form separately.

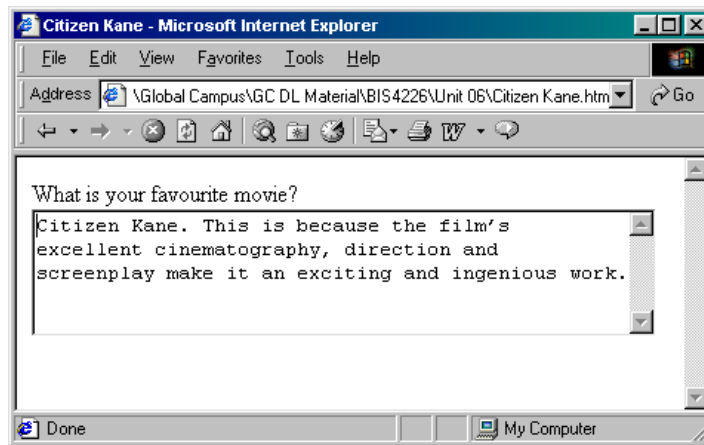


A browser window showing a form with six text fields: Name, Email, Address 1, Address 2, Address 3, and Postcode.

Read Discussion of Activity 2 at the end of the Unit.

5.2.3 Multi-line Text Entry

Although it is possible to type as much text as needed into a single line text field, it does become more practical to enter large amounts of text into a multiple-line input field. HTML calls these fields' text areas, as in the example below:



This is achieved with using the `<TEXTAREA></TEXTAREA>` tags. They create a scrollable text area for larger amount of text:

```
<TEXTAREA NAME="movie-comments" COLS="50" ROWS="5"></TEXTAREA>
```

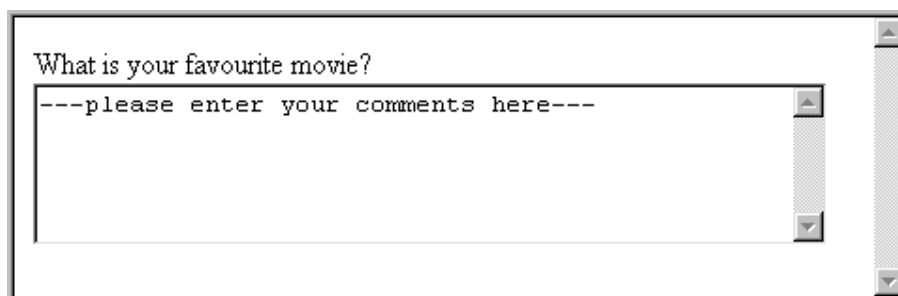
`NAME="movie-comments"` supplies the text field with the given label. Here, because the example allows the user to write about movies, we have named the form element "**movie-comments**".

`COLS="50"` specifies the width, in characters, of the text area. Here 50 characters have been specified.

`ROWS="5"` specifies the height of the text area in rows. This examples specifies five columns. Note that the text that should appear in the multi-line field is placed between the `<TEXTAREA>` and `</TEXTAREA>` tags. So, for example, users could be prompted to input their comments on a movie thus:

```
<TEXTAREA NAME="movie-comments" COLS="50" ROWS="5">
---please enter your comments here--- </TEXTAREA>
```

This would produce the following:



No horizontal scroll bars are present in the above text area examples (just as there rarely are any in a Web browser). This is because, by default, text-wrapping is on. Wrapping is a feature that causes words that cannot fit within a window pane to be moved to the following line or row. Web browsers wrap text (and most other elements) so that when a window is resized, text is redistributed over subsequent lines. A **WRAP** attribute is available, and the default value is `WRAP="ON"`. You can change wrapping to off, which will cause a horizontal scroll bar to appear at the bottom edge of the text area.

Exercise 2

Change the HTML for the movie opinion text area so that the text does not wrap and a horizontal scroll bar

is provided, as in the figure below:

What is your favourite movie?
 en Kane. This is because the film's excellent cine:

Solution can be found at the end of the Unit.

To Do

Read about Text Area in your textbooks.

Activity 3: Multi-line Text Entry

In an HTML document do the following:

1. Replace the address text fields with one text area large enough to hold the whole address. Use the facilities of the `<INPUT>` and `<TEXTAREA>` tags to prompt the user by including placeholder information in the text fields and text area.
2. Now make the layout of the form more aesthetically pleasing by placing the form in a two-column table, with field labels (e.g. 'Name:', 'Email:') in the left hand column, and the widgets in the right-hand column.
3. Think of three questions relating to the Internet that you might like to ask a user if you were a market researcher trying to gather user information. Create the additional text areas in your form to ask these questions.

Read Discussion of Activity 3 at the end of the chapter.

5.2.4 Radio Buttons

Radio buttons are often used on questionnaires to indicate a person's opinion, or their likes and dislikes. They can also be used for 'yes' or 'no' responses. Radio buttons should be used when only one answer from a set of possible answers may be chosen. This is best illustrated by example:

Example 1

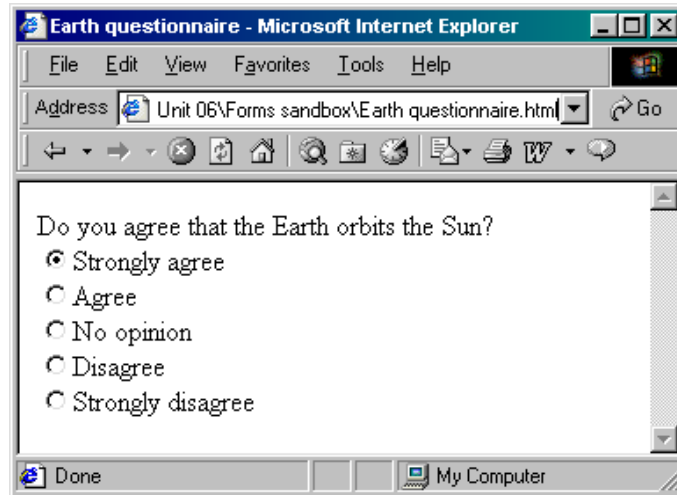
Do you like chocolate? Yes No

This is achieved with:

```
Do you like chocolate?
<input type="radio" name="chocolate" value="yes">Yes
<input type="radio" name="chocolate" value="no">No
```

Notice that the *same* name — chocolate — has been used for each element. This is necessary because when the form is submitted only the value of the currently selected radio button will be submitted with the given name. The software processing the data will *either* receive **chocolate=yes** or **chocolate=no**, which allows for straightforward processing.

Example 2



This is achieved with:

```
Do you agree that the Earth orbits the Sun?<br>
<INPUT TYPE="radio" NAME="orbits" VALUE="strongly_agree">Strongly agree<BR>
<INPUT TYPE="radio" NAME="orbits" VALUE="agree">Agree<BR>
<INPUT TYPE="radio" NAME="orbits" VALUE="no_opinion">No opinion<BR>
<INPUT TYPE="radio" NAME="orbits" VALUE="disagree">Disagree<BR>
<INPUT TYPE="radio" NAME="orbits" VALUE="strongly_disagree">Strongly
disagree<BR>
```

The tag and its attributes are as follows.

<input> is the tag used for most of the form objects.

type="radio" sets the object to a radio button.

name="orbits" labels the entire set of radio buttons. This makes identifying the data easier. For example if the radio button was for the question 'Do you own an automobile?', you might set **name="automobile"**

VALUE=... With a set of radio buttons for one question, it is not enough to provide a name only. We need to give each radio button a value so that the form-processing software (CGI script or email) can determine which radio button has been selected. This is where the **value** attribute comes in. Each of the values above is set to the answer for that radio button. This information is sent with the data when the form is submitted. Hence, the form in the above figure would submit **orbits=strongly_agree**.

CHECKED This has not been used in the above example. If it were included, the button is set as if it had been clicked. This is useful if one choice should be made the default.

To Do

Read about Radio Buttons in your textbooks.

Activity 4: Radio Buttons

In an HTML document, add these numbered questions and create radio buttons for them.

1. Do you like playing computer games? Yes / No

HTML Forms

2. How much did you enjoy the 'Star Wars' Trilogy? I enjoyed it / I did not enjoy it / I have not seen it
3. Do you have an email address? Yes / No
4. Chocolate is delicious? strongly agree / agree / neutral / disagree / strongly disagree
5. Then create four more questions of your own choice that use radio buttons.

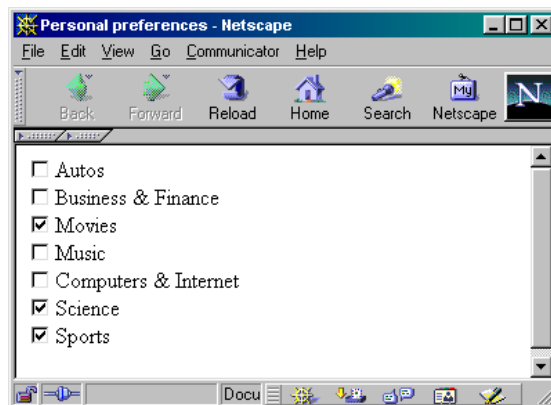
(While working on this activity, you might like to think how the form elements might interact with list structures.)

Read Discussion of Activity 4 at the end of the chapter.

5.2.5 Check Boxes

Checkboxes are one of the simplest objects that can be placed on a form. These input elements can only be selected and de-selected. Unlike radio buttons, more than one can be selected at a time.

For example, when signing up for a free e-mail account with GMail [<http://www.gmail.com>] or Hotmail [<http://www.hotmail.com>], a user may well have to fill in a series of forms. One of them is often an interests form, and looks something like this:



```
<INPUT TYPE="checkbox" NAME="autos" VALUE="yes">Autos<BR>
<INPUT TYPE="checkbox" NAME="business"
VALUE="yes">Business & Finance<BR>
<INPUT TYPE="checkbox" NAME="movies"
VALUE="yes">Movies<BR>
<INPUT TYPE="checkbox" NAME="music" VALUE="yes">Music<BR>
<INPUT TYPE="checkbox" NAME="computing"
VALUE="yes">Computers & Internet<BR>
<INPUT TYPE="checkbox" NAME="science"
VALUE="yes">Science<BR>
<INPUT TYPE="checkbox" NAME="sports"
VALUE="yes">Sports<BR>
```

`<INPUT>` is the tag used for most of the form objects. `type="checkbox"` sets the object to a checkbox. `name` is used to supply a name to the checkbox.

`VALUE="yes"` if the item is checked, this is the value that will be associated with the name

HTML Forms

when the form is submitted for processing. Hence, in the above example, **yes** will be associated with each of the name values **movies**, **science** and **sports**.

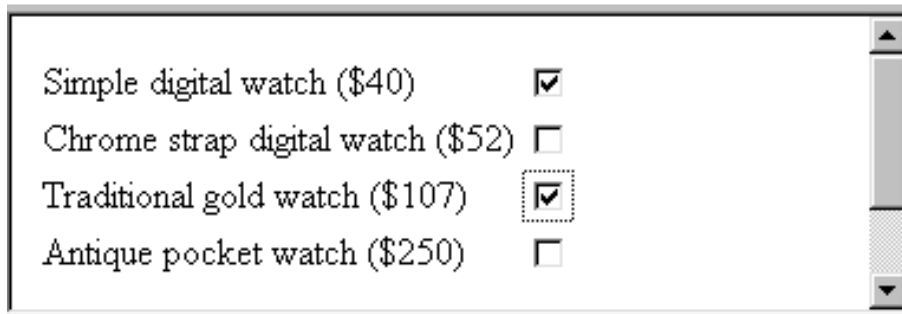
CHECKED This has not been used in the above examples. If it were included as an attribute of the tag, the check box would be set as if it had been clicked by the user. This is useful if one or more options are to be offered as defaults.

To Do

Read about Check Box in your textbooks.

Activity 5: Check Boxes

Imagine you are developing a website that sells various types of watch. You want to allow customers to select what they want to buy with a set of check boxes, as in the figure below.



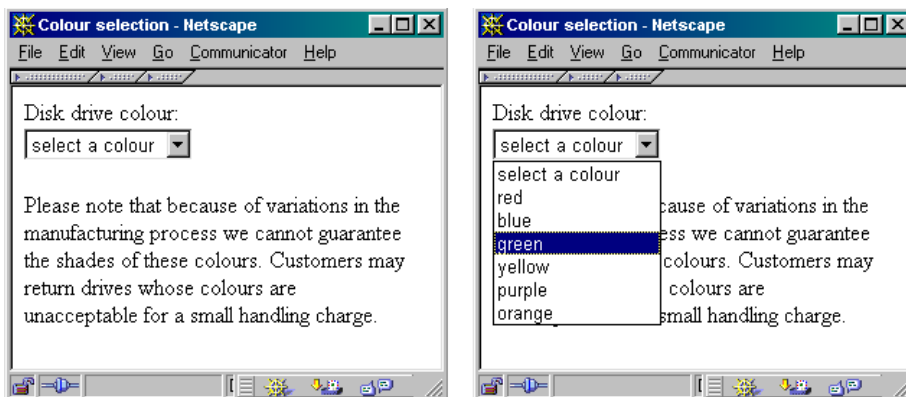
Simple digital watch (\$40)	<input checked="" type="checkbox"/>
Chrome strap digital watch (\$52)	<input type="checkbox"/>
Traditional gold watch (\$107)	<input checked="" type="checkbox"/>
Antique pocket watch (\$250)	<input type="checkbox"/>

Write an HTML form to achieve this (note the tabular layout). Remember that a server-side script may be expecting the prices of the checked items, so these values will need to be transmitted for processing.

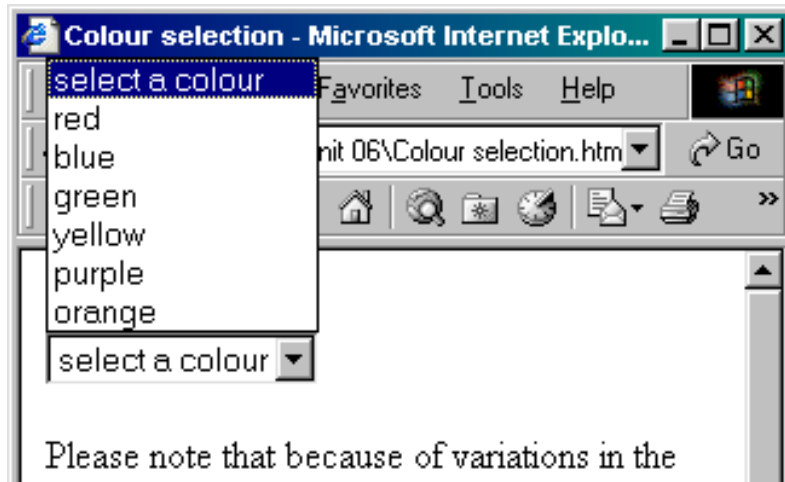
Read Discussion of Activity 5 at the end of the chapter.

5.2.6 Menu Buttons and Scrolling Lists

Menu buttons and scrolling lists are useful when you have multiple options to choose from. For example, the following shows a menu button (sometimes imprecisely called a pull-down or drop-down menu). Clicking on the 'select a colour' option on the button displays all the other options in the button's menu. Pulling down to the required option will set the value for this element (for subsequent transmission).



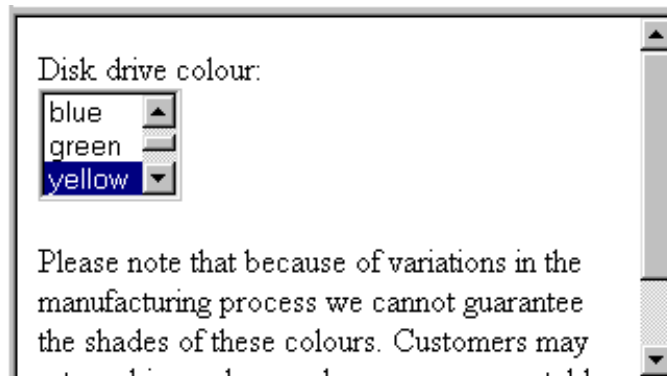
If the menu button is near the bottom of a window, the menu pops up:



Notice that, by convention, the first option of a menu button is usually not an option at all, but a prompt to suggest an action to the user. Hence, in the previous example, 'select a colour' is not an option but a prompt. However, if the form containing this menu button were submitted for processing, the value 'select a colour' would be associated with the named attribute.

Scrolling lists, otherwise known as selection lists, are similar to menu buttons, but they usually display more than one of the available options at a time. They rarely show all options, and the user is required to scroll in order to view them all. If all the options are displayed, no scroll bar is included with the list, and it may not be obvious to the user that they should select an option.

The above example could be redesigned using a scrolling list. The figures below show the three options, and all six options (excluding the prompt):



One clear benefit of these two input elements are that they do not take up as much space as, say, a list of radio buttons. Like radio buttons, they are also used to select one choice one from a set of mutually exclusive options.

The HTML for the menu button and scrolling list (the three-option version) are given below:

HTML Forms

Disk drive colour: <SELECT NAME="colour"> <OPTION>select a colour</OPTION> <OPTION>red</OPTION> <OPTION>blue</OPTION> <OPTION>green</OPTION> <OPTION>yellow</OPTION> <OPTION>purple</OPTION> <OPTION>orange</OPTION> </SELECT>	Disk drive colour: <SELECT NAME="colour" SIZE="3" MULTIPLE> <OPTION>red</OPTION> <OPTION>blue</OPTION> <OPTION>green</OPTION> <OPTION>yellow</OPTION> <OPTION>purple</OPTION> <OPTION>orange</OPTION> </SELECT>
<P>Please note that because of variations in the manufacturing process we cannot guarantee the shades of these colours. Customers may return drives whose colours are unacceptable for a small handling charge.</P>	<P>Please note that because of variations in the manufacturing process we cannot guarantee the shades of these colours. Customers may return drives whose colours are unacceptable for a small handling charge.</P>

<SELECT></SELECT> are used for both menu buttons and scrolling lists (the <INPUT> tag is not used).

name="colour" specifies the data name to be transmitted on form submission.

<OPTION></OPTION> are used to specify the option we want to appear in both types of menus.

size="3" This attribute sets the number of options that the scrolling list shows. In the above example, the number of options has been set to 3.

MULTIPLE This attribute guarantees that the <SELECT> tag results in a scrolling list rather than a menu button. It is not needed if **SIZE** is set to be greater than 1. However, if **MULTIPLE** is omitted and **SIZE=1**, or is omitted, the tag produces a menu button and not a scrolling list.

SELECTED This attribute can be included in an <OPTION> tag; it has not been used in the above examples. If it were included, the option is selected by default.

Exercise 3

A small Egyptian airline flies internally to Egypt while also offering flights to Kuwait, Lebanon, Bahrain, and Oman. Their new website is being developed to promote their business, especially the external services. The website should provide a list of destination options to users. Bearing in mind that most of the company's business is internal and that it wants to advertise its flights to other countries, select an appropriate selection mechanism and arrangement of options. Write the HTML.

A solution can be found at the end of the Unit.

To Do

Read about creating menus with the <SELECT> tag in your textbooks.

Activity 6: Menu Buttons and Scrolling Lists

In an HTML document:

1. Create menu buttons for the following:

- Title: Mr / Mrs / Miss/ Ms / Dr
- Age: under 20 (<19) / 20—29 / 30—39 / 40—49 / 50—59 / over 60 (60>)
- Gender: Male / Female
- Status: Employed / Unemployed / Student

2. Create scrolling menus to enter a date of birth using the following:

- Day: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31

HTML Forms

- Month: January, February, March, April, May, June, July, August, September, October, November, December
- Year: 1960, 1961, 1962, 1963, 1964, 1965, 1966, 1967, 1968, 1969, 1970, 1971, 1972, 1973, 1974, 1975, 1976, 1977, 1978, 1979, 1980, 1981, 1982, 1983, 1984, 1985, 1986, 1987, 1988, 1989, 1991, 1992, 1993, 1994, 1995, 1996

Read Discussion of Activity 6 at the end of the chapter.

5.2.7 Submit and Reset Buttons

Submit and reset buttons have simple uses: they allow the user to send the form data onwards for processing, or to clear the form fields of all entered information.

For example: a user has ordered a large pizza on-line. They have entered four possible topping selections. To send the data for processing, the user clicks on the button labelled 'Order the pizza'. If the user were to make an error and wish to restart, clicking the button labelled 'Clear to restart' will remove the text from the multi-line text area and (as it happens) reset the choice of pizza size to 'Medium'.

We only use fresh ingredients and some pizza toppings may not be available.
We will supply three from your list.
Please list preferred toppings in order:

Pizza size:

Small
 Medium
 Large

The new elements in this example are the buttons labelled 'Order the pizza' and 'Clear to restart', which are submit and reset buttons respectively. When a submit button is clicked, the Web browser looks at the <form> tag to see how the data should be processed. It will either be sent as an email, as shown below, or sent for processing by a server-side script. Here is HTML for the above example:

```
<FORM METHOD="post" ACTION="mailto:pizza-orders@medpizzas.com.eg">
<P>We only use fresh ingredients and some pizza toppings
may not be available. We will supply three from your list.</P> Please
list preferred toppings in order:<BR>
<TEXTAREA NAME="toppings" COLS="40" ROWS="5" WRAP=OFF></TEXTAREA><br>
Pizza size:<BR>
<INPUT TYPE="radio" NAME="pizza_size" VALUE="small">Small<BR>
<INPUT TYPE="radio" NAME="pizza_size" VALUE="medium" CHECKED>Medium<BR>
<INPUT TYPE="radio" NAME="pizza_size" VALUE="large">Large<BR>
<INPUT TYPE="submit" VALUE="Order the pizza">
<INPUT TYPE="reset" VALUE="Clear to restart">
</FORM>
```

Buttons can be customised to perform other functions, such as navigation, by using JavaScript (developers mostly use images and text links for navigation, however) See the Extension section.

A submit button may, in fact, be replaced with an image. For instance, the above example can be

modified as follows:



The <INPUT> tag for this submit button uses **image** as the value of the **TYPE** attribute and adds a **SRC** attribute to specify the image to be used, as in:

```
<INPUT TYPE="image" SRC="go-pizza-button.gif">
```

Clicking on an image used as if it were a submit button has exactly the same effect as a standard submit button, in that the form data is dispatched for processing.

Note that there is no corresponding facility to replace a reset button with an image.

To Do

Read about Submit and Reset Buttons in your textbooks.

Activity 7: Submit and Reset Buttons

Implement the check box example on personal interests and add submit and reset buttons. Test the reset button by checking several options and then clearing them by clicking Reset. To test the submit button, set the form (using the **ACTION** attribute) to send email to a real address.

Test how the submit button works when using the default encoding (see your textbook). You should receive a binary file attachment at the email address in the **ACTION** attribute. You can open this with a text editor (like Notepad).

Change the <FORM> tag to include the encoding **ENCTYPE="text/plain"** and resubmit the form. Examine the email to see how its contents have changed.

Read Discussion of Activity 7 at the end of the Unit.

5.3 HTML5 form elements

Create a new HTML file and practice the HTML5 elements in this section.

5.3.1 Email address field

The first of these new input types is for email addresses. You can allow the input field to access multiple inputs by using the **multiple** attribute as shown below. For <input type="email">: separate each email with a comma. Try typing an email address without the @ field and notice that the browser performs a simple validation.

```
<form>
  <input type="email" multiple>
  <input type="submit" value="Go">
</form>
```

5.3.2 Search

Search functions are performed on popular websites such as Google, e-commerce sites as well as personal blogs. It is probably the most common action performed on the Web every day. With HTML5 you can create a simple search action by using:

```
<form >
  <input type="search" name="search">
</form>
```

When you start typing in the search bar you will notice that you may get some suggestions that you can click on, just like you would on Google. Notice also the 'x' on the right which you can use to clear your search results, like you would on Safari browser.

5.3.3 Url

The url input type is for web addresses. You can use the multiple attribute to enter more than one URL. Like type="email", a browser will carry out simple validation on these fields and present an error message on form submission. This is likely to include looking for forward slashes, periods, and spaces, and possibly detecting a valid top-level domain (such as .com or .co.uk). Use the url input type like so:

```
<input type="url" name="url">
```

5.3.4 Autofocus

If you wish to have one of the field on you form automatically selected when a user loads the webpage, you would give that field the autofocus attribute. Assume that you want to mark the 'url' field from 5.33 with autofocus, you would use the markup:

```
<input type="url" name="url" autofocus>
```

5.3.5 Dates and Times

Usually, date pickers are constructed using JavaScript library. HTML5 enables this functionality possible within the browser. Use the markup:

```
<input id="dob" name="dob" type="date">
```

To only show the month and year, use the markup:

```
<input id="expiry" name="expiry" type="month" required>
```

To show the time, use the markup:

```
<input id="time" name="time" type="time">
```

5.3.6 Color

The color input type is pretty self-explanatory: it allows the user to select a color and returns the hex value for that color. It is anticipated that users will either be able to type the value or select from a color picker, which will either be native to the operating system or a browser's own implementation. If you are using Chrome you will be able to pick a color from a color picker. To use the color tag, use the markup:

```
<input id="color" name="color" type="color">
```

5.3.7 Numbers as Spinboxes

Using HTML5 you can create an input field that accepts minimum and maximum numbers and then you can be able to scroll through the numbers and pick one. Use the following markup:

```
<input type="number" min="0" max="10" step="1" value="0">
```

- type="number" means that this is a number field.

HTML Forms

- `min="0"` specifies the minimum acceptable value for this field.
- `max="10"` is the maximum acceptable value.
- `step="1"`, combined with the min value, defines the acceptable numbers in the range: 0, 1, 2, and so on, up to the maxvalue.
- `value="0"` is the default value that displays before selecting another number.

5.3.8 Numbers as Sliders

By changing the input type from 'number' to 'range' you can change the spinboxes to a slider. Use the markup:

```
<input type="range" min="0" max="10" step="1" value="0">
```

5.4 Using Forms

It is time to consider how organizations can make use of forms on their websites. This primarily involves investigating and discussing how commercial enterprises on the Web currently make use of forms.

Activity 8: How websites use Forms

For this Activity you will need to take a look at a number of websites:

- Amazon Books [<http://www.amazon.com>]
- Loot Free Ads [<http://www.loot.com>]
- Google [<http://www.google.com>]
- Ebay on-line auctions [<http://www.ebay.com>]
- Any of the computer manufacturers (e.g. Dell [<http://www.dell.com>])

Make a note of the following:

- The ways in which websites make use of forms.
- The different types of data that the website are trying to gather. Are there any distinct differences?

To finish this section, you will develop two extensive Web pages that make use of forms.

Activity 9: Job Application Form

Create an on-line job application form. The application form is for a computer company called SpeedyPC who are advertising for computer programmers. Your form's action should post the application to your email address.

Your application form must have the following elements:

- Position applied for (autofocus), name, nationality, date of birth (selected from an auto picker), address (in a text area), telephone number and email (required).
- Educational history and qualifications.
- Work experience/employment/training in terms of employer history and number of years of experience selected from a slider. Set maximum years of experience to 10 years.
- Personal statement.
- Two referees including names, occupation, relationship, address, telephone.

Read discussion at the end of the chapter.

5.5 Additional Content and Activities

5.5.1 Supplementary information on HTML forms

Spend 10-20 minutes reviewing the tutorials at one or both of these sites:

- Bare Bones Guide to HTML [<http://werbach.com/barebones/barebone.html>]
- Yale C/AIM Web Style Guide [<http://info.med.yale.edu/caim/manual/contents.html>]

5.5.2 Additional Activity — Tabular Layout of a Complex Form

Create a form for processing orders for boxes that looks like those in the figures below. The order form is for a company called 'Land of Boxes'. Their product line consists of only five items:

1. The EconoBox: Cost R5
2. The Standard Box: Cost R10
3. The Premium Box: Cost R15
4. The Deluxe Box: Cost R20
5. The Super Deluxe Box: Cost R30

The order form should include the following elements:

1. Contact details of purchaser, including name, address, telephone number and email.
2. Product details including price, product name, product number and product quantity.
3. Method of payment including credit card type, card number and expiry date.
4. Billing address and delivery address if they are different.
5. The final layout should look as shown in the next two figures. You will need to use tables or CSS styles to achieve this layout.

HTML Forms

Land Of Boxes Order Form

Name:

Email:

Address:

Postcode:

Product	Part Number	Quantity	Unit Price	Subtotal
EconoBox	LB100	<input type="text" value="3"/>	R5	<input type="text" value="R15"/>
Standard Box	LB200	<input type="text" value="4"/>	R10	<input type="text" value="R40"/>
Premium Box/TD>	LB300	<input type="text" value="1"/>	R15	<input type="text" value="R15"/>
Deluxe Box/TD>	LB400	<input type="text" value="0"/>	R20	<input type="text" value=""/>
Super Deluxe Box/TD>	LB500	<input type="text" value="1"/>	R30	<input type="text" value="R30"/>
Total:				<input type="text" value="R100"/>

Credit Card Details

Card Type:

Mastercard Card Number:

Visa Expiry date:

American Express

Diners Club

Delivery Address:

Read Discussion of Additional Activity at the end of the chapter.

5.6 Review Questions

1. Write down the main purpose of HTML forms. Answer at the end of the chapter.
2. What is CGI software?

HTML Forms

Answer at the end of the chapter.

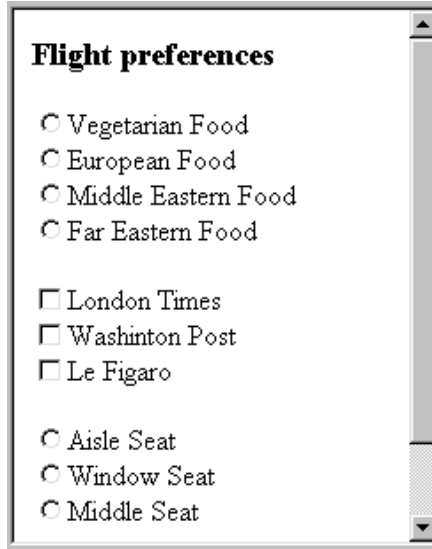
3. Is the size of the text a user can enter in a text field limited by the value of the **SIZE** attribute in the `<INPUT>` tag?

Answer at the end of the chapter.

4. What attribute must be included in a `<TEXTAREA>` tag to ensure a horizontal scroll bar is included in a multi-line text field?

Answer at the end of the chapter.

5. What is wrong with the following HTML? The code should allow the user to specify their seating arrangement, food, and preferred newspaper, for a flight with a particular airline.



The screenshot shows a web form titled "Flight preferences" with a scrollable area. It contains three sections of options:

- Food preferences: Four radio buttons for "Vegetarian Food", "European Food", "Middle Eastern Food", and "Far Eastern Food".
- Newspaper preferences: Three checkboxes for "London Times", "Washinton Post", and "Le Figaro".
- Seating preferences: Three radio buttons for "Aisle Seat", "Window Seat", and "Middle Seat".

```
<INPUT TYPE="radio" NAME="flight_prefs" VALUE="veggie">Vegetarian  
Food<BR>  
<INPUT TYPE="radio" NAME="flight_prefs"  
VALUE="euro_food">European Food<BR>  
<INPUT TYPE="radio" NAME="flight_prefs"  
VALUE="midEast_food">Middle Eastern Food<BR>  
<INPUT TYPE="radio" NAME="flight_prefs" VALUE="farEast_food">Far  
Eastern Food<BR><BR>  
<INPUT TYPE="checkbox" NAME="flight_prefs" VALUE="times">London  
Times<BR>  
<INPUT TYPE="checkbox" NAME="flight_prefs" VALUE="post">Washinton  
Post<BR>  
<INPUT TYPE="checkbox" NAME="flight_prefs" VALUE="figaro">Le  
Figaro<BR><BR>  
<INPUT TYPE="radio" NAME="flight_prefs" VALUE="aisle">Aisle  
Seat <BR>  
<INPUT TYPE="radio" NAME="flight_prefs" VALUE="window">Window  
Seat<BR>  
<INPUT TYPE="radio" NAME="flight_prefs" VALUE="middle">Middle  
Seat<BR><BR>
```

Answer at the end of the chapter.

6. How do you set the initial state of a check

box? Answer at the end of the chapter.

7. How do menu buttons and scrolling lists differ in how they save space on a Web

page? Answer at the end of the chapter.

8. Create a scrolling list that shows four items from the following list of personal computer processor types: Motorola 68000, Intel 8088, Intel Pentium MMX, Intel Pentium II, Intel Pentium III, Intel Celeron, PowerPC G3, PowerPC G4, AMD Athlon.

Answer at the end of the chapter

5.7 Discussions and Answers

5.7.1 Discussion of Activity 1

Your document will probably look something like this:

```
<HEAD>
<TITLE>Form example</TITLE>
</HEAD>
<BODY>
<FORM METHOD="post" ACTION="mailto:put.your@email.address.here">
</FORM>
</BODY>
```

You will notice that nothing shows when this form is rendered because it has none of the interactive elements that make forms useful.

5.7.2 Discussion of Activity 2

To produce the text fields one by one:

1. The name text field (below) and its HTML.

Name:

Name: `<INPUT TYPE="text" NAME="name" SIZE="35">
`

2. The email text field (below) and its HTML.

Email:

Email: `<INPUT TYPE="text" NAME="email" SIZE="40">
`

3. The three address fields and the post code field (below), with the HTML.

HTML Forms

Address 1:

Address 2:

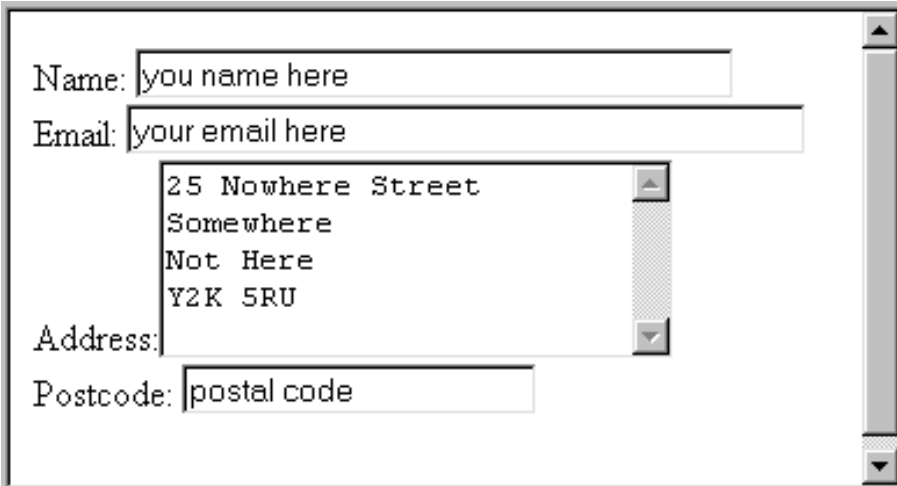
Address 3:

Postcode:

```
Address 1: <INPUT TYPE="text" NAME="address1" SIZE="40"><BR> Address
2: <INPUT TYPE="text" NAME="address2" SIZE="40"><BR> Address 3:
<INPUT TYPE="text" NAME="address3" SIZE="40"><BR> Postcode: <INPUT
TYPE="text" NAME="postcode" SIZE="20"><BR>
```

5.7.3 Discussion of Activity 3

1. The multi-line address text area and its HTML are as follows.



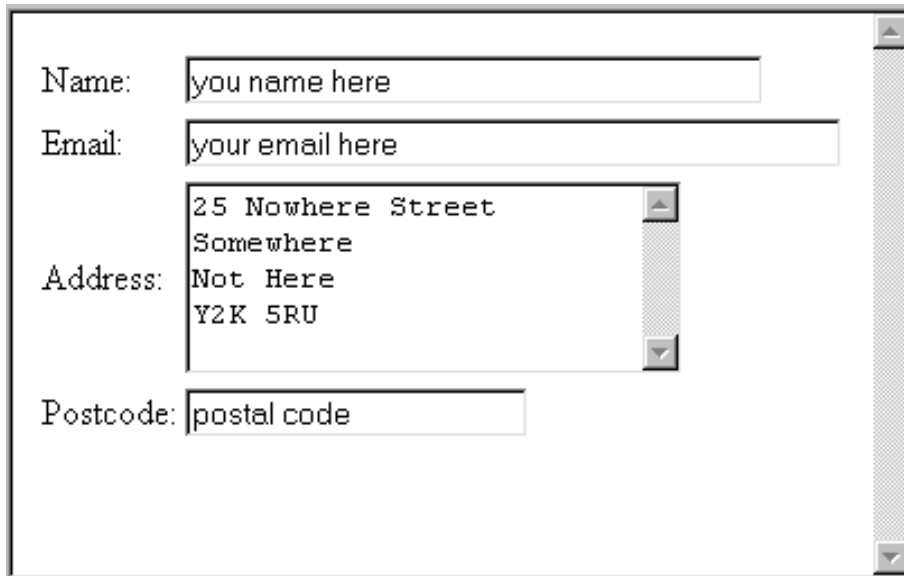
The screenshot shows a web form with the following fields and content:

- Name:
- Email:
- Address:
- Postcode:

```
Name: <INPUT TYPE="text" NAME="name" SIZE="35" VALUE="you name here"><BR>
Email: <INPUT TYPE="text" NAME="email" SIZE="40" VALUE="your email
here"><BR>
Address:<TEXTAREA NAME="textfield3" COLS="25" ROWS="5">
25 Nowhere Street Somewhere
Not Here Y2K
5RU
</TEXTAREA><BR>
Postcode: <INPUT TYPE="text" NAME="postcode" SIZE="20" VALUE="postal
code"><BR><BR>
```

2. The nicely laid out version and the table-based HTML follow:

HTML Forms



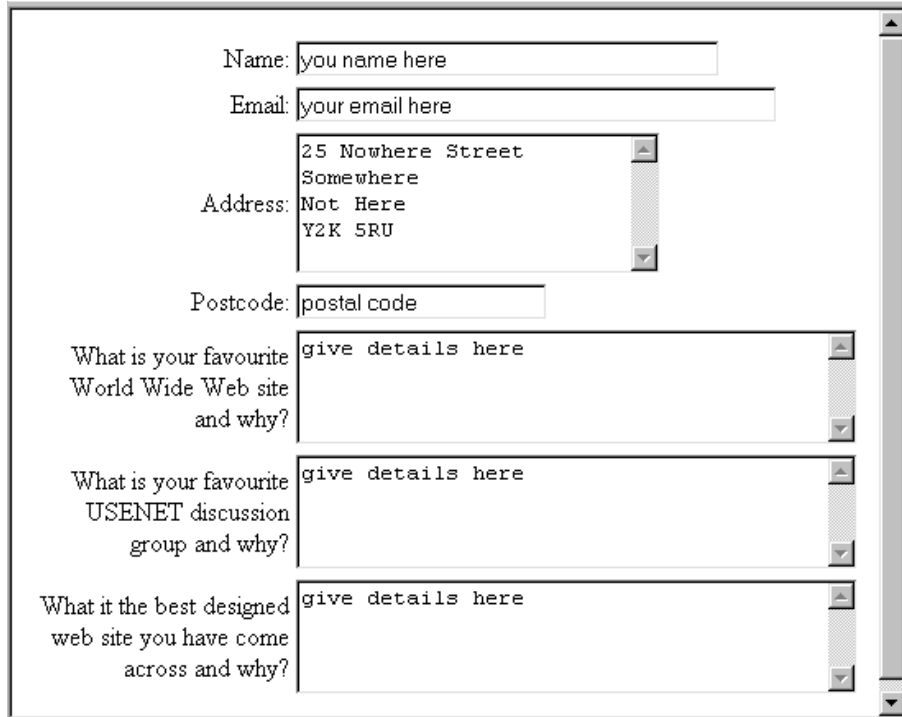
```
</FORM>
<TABLE>
<TR>

<TD>Name:</TD>
<TD><INPUT TYPE="text" NAME="name" SIZE="35" VALUE="you name here"></TD>
</TR>
<TR>
<TD>Email:</TD>
<TD><INPUT TYPE="text" NAME="email" SIZE="40" VALUE="your email
here"></TD>
</TR>
<TR>
<TD>Address:</TD>
<TD><TEXTAREA NAME="textfield3" COLS="25" ROWS="5">
25 Nowhere Street
Somewhere
Not Here
Y2K 5RU
</TEXTAREA></TD>
</TR>
<TR>
<TD>Postcode:</TD>
<TD><INPUT TYPE="text" NAME="postcode" SIZE="20" VALUE="postal
code"></TD>
</TR>
</TABLE>
</FORM>
```

Note that the cells have been indented to aid readability. Note that the initial value of the text area (between `<TEXTAREA>` and `</TEXTAREA>` tags) should not be indented unless the text, as rendered by the browser, should itself appear indented.

3. Adding three more text areas to the table structure is more complicated than adding them to a simple (non-table) form: adding a lot of text before the text areas separates them the labels ('Name:', 'Email:', etc.). Aligning the labels to the right brings them closer, as in the figure below. The extra HTML follows the figure:

HTML Forms



The screenshot shows a web form with the following fields:

- Name:
- Email:
- Address:
- Postcode:
- What is your favourite World Wide Web site and why?
- What is your favourite USENET discussion group and why?
- What is the best designed web site you have come across and why?

<TR>

```
<TDALIGN="right">What is your favourite World Wide Web site
and why?</TD>
<TD><TEXTAREA name="textfield4" cols="40" rows="4">give details
here</TEXTAREA></TD>
```

</TR>

<TR>

```
<TDALIGN="right">What is your favourite USENET discussion group
and why?</TD>
<TD><TEXTAREA name="textarea" cols="40" rows="4">give details
here</TEXTAREA></TD>
```

</TR>

<TR>

```
<TDALIGN="right">What is the best designed website you have
come across and why?</TD>
<TD><TEXTAREA name="textarea2" cols="40" rows="4">give details
here</TEXTAREA></TD>
```

</TR>

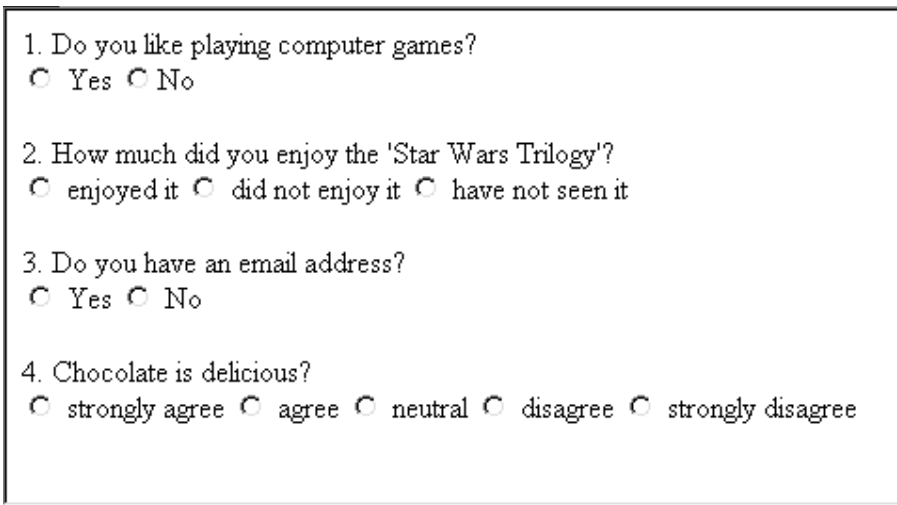
5.7.4 Discussion of Activity 4

The following HTML produces the required form (shown after the HTML).

```
<FORM METHOD="post" ACTION="mailto:name@xyz.ac.uk">
<p>1. Do you like playing computer games? <br>
<input type="radio" name="games" value="yes"> Yes
<input type="radio" name="games" value="no">No</p>
<p>2. How much did you enjoy the 'Star Wars Trilogy'? <br>
<input type="radio" name="starwars" value="eit"> enjoyed it
```

HTML Forms

```
<input type="radio" name="starwars" value="dneit"> did not enjoy it
<input type="radio" name="starwars" value="hnsit"> have not seen
it</p>
<p>3. Do you have an email address? <br>
<input type="radio" name="email" value="yes"> Yes
<input type="radio" name="eamil" value="yes"> No</p>
<p>4. Chocolate is delicious? <br>
<input type="radio" name="radiobutton" value="sagree"> strongly
agree
<input type="radio" name="radiobutton" value="agr"> agree
<input type="radio" name="radiobutton" value="neutral"> neutral
<input type="radio" name="radiobutton" value="disagree"> disagree
<input type="radio" name="radiobutton" value="sdisagree"> strongly
disagree</p>
</FORM>
```



1. Do you like playing computer games?
 Yes No

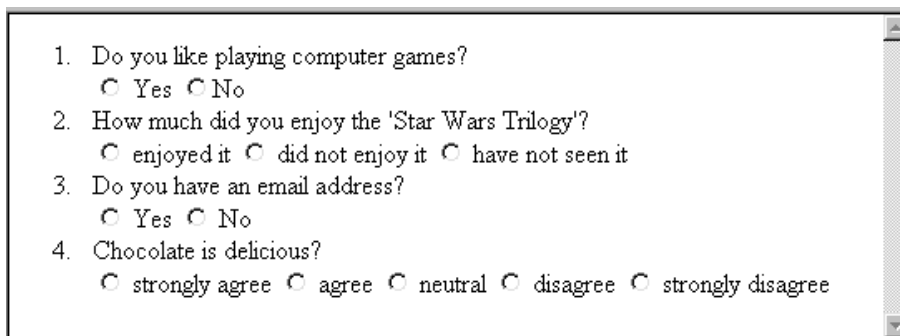
2. How much did you enjoy the 'Star Wars Trilogy'?
 enjoyed it did not enjoy it have not seen it

3. Do you have an email address?
 Yes No

4. Chocolate is delicious?
 strongly agree agree neutral disagree strongly disagree

Note that a different style of layout can be achieved by combining the form elements with the `` and `` tags. Here is a fragment of HTML and its associated layout.

```
<OL>
<li> Do you like playing computer games? <br>
<input type="radio" name="games" value="yes"> Yes
<input type="radio" name="games" value="no">No</li>
...
</OL>
```



1. Do you like playing computer games?
 Yes No

2. How much did you enjoy the 'Star Wars Trilogy'?
 enjoyed it did not enjoy it have not seen it

3. Do you have an email address?
 Yes No

4. Chocolate is delicious?
 strongly agree agree neutral disagree strongly disagree

Here is HTML code for extra questions. Note how the values have been abbreviated. The values are never seen by the user, but are for processing by software, typically a server-side script, and so do not need to be understood by the end users.

```
<p>1. Which of these foods do you prefer? <br>
```


HTML Forms

```
<input type="radio" name="food" value="italian">Italian
<input type="radio" name="food" value="chinese">Chinese
<input type="radio" name="food" value="indian">Indian </p>
<p>2. Which car do you prefer? <br>
<input type="radio" name="car" value="ferrari">Ferrari
<input type="radio" name="car" value="astin">Aston Martin
<input type="radio" name="car" value="lotus">Lotus
<input type="radio" name="car" value="jaguar">Jaguar</p>
<p>3. Do you like music? <br>
<input type="radio" name="music" value="yes">Yes
<input type="radio" name="music" value="no">No</p>
<p>4. I enjoy reading?<br>

<input type="radio" name="read" value="sa">strongly agree
<input type="radio" name="read" value="a">agree
<input type="radio" name="read" value="n">neutral
<input type="radio" name="read" value="d">disagree
<input type="radio" name="read" value="sd">strongly disagree</p>
```

5.7.5 Discussion of Activity 5

Here is the HTML. Note that summing the totals of the selected items requires server-side software. (This is one of the reasons that JavaScript is used to allow such calculations before a form is submitted; see Unit 16.)

```
<TABLE>
<TR>
<TD>Simple digital watch ($40)</TD>
<TD><INPUT TYPE="checkbox" VALUE="40"></TD>
</TR>
<TR>
<TD>Chrome strap digital watch ($52)</TD>
<TD><INPUT TYPE="checkbox" VALUE="52"></TD>
</TR>
<TR>
<TD>Traditional gold watch ($107)</TD>
<TD><INPUT TYPE="checkbox" VALUE="107"></TD>
</TR>
<TR>
<TD>Antique pocket watch ($250)</TD>
<TD><INPUT TYPE="checkbox" VALUE="250"></TD>
</TR>
</TABLE>
```

5.7.6 Discussion of Activity 6

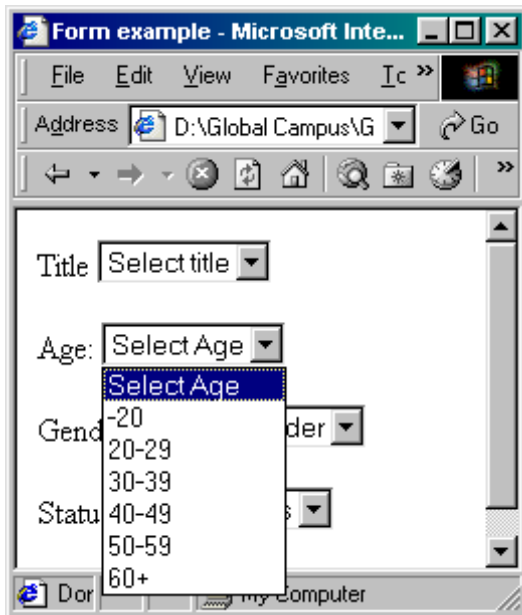
1. The HTML and the output for Title, Age, Gender, employment Status.

```
<FORM METHOD="post" ACTION=mailto:name@xyz.ac.uk>
Title
<SELECT NAME="title">
<OPTION>Select title</OPTION>
<OPTION>Mr</OPTION>
<OPTION>Mrs</OPTION>
<OPTION>Miss</OPTION>
<OPTION>Ms</OPTION>
<OPTION>Dr</OPTION>
```

HTML Forms

```
</SELECT><BR><BR>
Age :
<SELECT NAME="age">
<OPTION>Select Age</OPTION>
<OPTION>-19</OPTION>
<OPTION>20-29</OPTION>
<OPTION>30-39</OPTION>
<OPTION>40-49</OPTION>
<OPTION>50-59</OPTION>
<OPTION>60+</OPTION>

</SELECT><BR><BR>
Gender :
<SELECT NAME="gender">
<OPTION>Select Gender</OPTION>
<OPTION>Male</OPTION>
<OPTION>Female</OPTION>
</SELECT><BR><BR>
Status :
<SELECT NAME="status">
<OPTION>Select Status</OPTION>
<OPTION>Employed</OPTION>
<OPTION>Unemployed</OPTION>
<OPTION>Student</OPTION>
</SELECT><BR><BR>
</FORM>
```



2. The HTML and the output for Title, Age, Gender, employment Status

```
<FORM METHOD="post" ACTION="mailto:name@xyz.ac.uk">
Day :

<SELECT name="occupation" size="3">
<OPTION SELECTED>--Select Day--</OPTION>
<OPTION>1st</OPTION>
<OPTION>2nd</OPTION>
<OPTION>3rd</OPTION>
<OPTION>4th</OPTION>
<OPTION>5th</OPTION>
<OPTION>6th</OPTION>
```

HTML Forms

```
<OPTION>7th</OPTION>
<OPTION>8th</OPTION>
<OPTION>9th</OPTION>
<OPTION>10th</OPTION>
<OPTION>11th</OPTION>
<OPTION>12th</OPTION>
<OPTION>13th</OPTION>
<OPTION>14th</OPTION>
<OPTION>15th</OPTION>
<OPTION>16th</OPTION>
<OPTION>17th</OPTION>
<OPTION>18th</OPTION>
<OPTION>19th</OPTION>
<OPTION>20th</OPTION>
<OPTION>21st</OPTION>
<OPTION>22nd</OPTION>
<OPTION>23rd</OPTION>
<OPTION>24th</OPTION>
<OPTION>25th</OPTION>
<OPTION>26th</OPTION>
<OPTION>27th</OPTION>
<OPTION>28th</OPTION>
<OPTION>29th</OPTION>
<OPTION>30th</OPTION>
<OPTION>31st</OPTION>
```

```
</SELECT><BR><BR>
```

Month:

```
<SELECT name="select" size="3">
```

```
<OPTION SELECTED>--Select Month---</OPTION>
<OPTION>January</OPTION>
<OPTION>February</OPTION>
<OPTION>March</OPTION>
<OPTION>April</OPTION>
<OPTION>May</OPTION>
<OPTION>June</OPTION>
<OPTION>July</OPTION>
<OPTION>August</OPTION>
<OPTION>September</OPTION>
<OPTION>October</OPTION>
<OPTION>November</OPTION>
<OPTION>December</OPTION>
```

```
</SELECT><BR><BR>
```

Year:

```
<SELECT name="select2" size="3">
```

```
<OPTION SELECTED>--Select Year---</OPTION>
<OPTION>1960</OPTION>
<OPTION>1961</OPTION>
<OPTION>1962</OPTION>
<OPTION>1963</OPTION>
<OPTION>1964</OPTION>
<OPTION>1965</OPTION>
<OPTION>1966</OPTION>
<OPTION>1967</OPTION>
<OPTION>1968</OPTION>
<OPTION>1969</OPTION>
```

```

<OPTION>1970</OPTION>
<OPTION>1971</OPTION>
<OPTION>1972</OPTION>
<OPTION>1973</OPTION>
<OPTION>1974</OPTION>
<OPTION>1975</OPTION>
<OPTION>1976</OPTION>
<OPTION>1977</OPTION>
<OPTION>1978</OPTION>
<OPTION>1979</OPTION>
<OPTION>1980</OPTION>
<OPTION>1982</OPTION>
<OPTION>1983</OPTION>
<OPTION>1984</OPTION>
<OPTION>1985</OPTION>
<OPTION>1986</OPTION>
<OPTION>1987</OPTION>
<OPTION>1988</OPTION>
<OPTION>1989</OPTION>
<OPTION>1990</OPTION>
<OPTION>1991</OPTION>
<OPTION>1992</OPTION>
<OPTION>1993</OPTION>
<OPTION>1994</OPTION>
<OPTION>1995</OPTION>
<OPTION>1996</OPTION>

</SELECT><BR><BR>

</FORM>

```

The image shows a screenshot of a web browser window displaying a form with three dropdown menus. The first dropdown is labeled 'Day' and has '1st' selected. The second dropdown is labeled 'Month' and has 'January' selected. The third dropdown is labeled 'Year' and has '1961' selected. The form is contained within a rectangular frame with a vertical scrollbar on the right side.

5.7.7 Discussion of Activity 7

The HTML code is as follows:

```

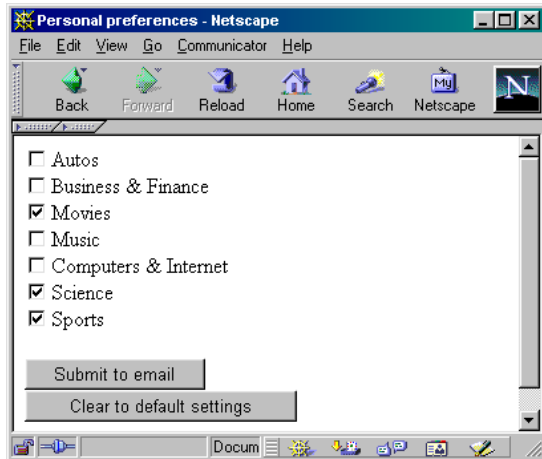
<FORM METHOD="post" ACTION="mailto:your-email@your.isp">
<INPUT TYPE="checkbox" NAME="autos" VALUE="yes">Autos<BR>
<INPUT TYPE="checkbox" NAME="b" VALUE="yes">Business & Finance<BR>
<INPUT TYPE="checkbox" NAME="movies" VALUE="yes">Movies<BR>
<INPUT TYPE="checkbox" NAME="music" VALUE="yes">Music<BR>
<INPUT TYPE="checkbox" NAME="comps" VALUE="yes">Computers & Internet
<BR>
<INPUT TYPE="checkbox" NAME="science" VALUE="yes">Science<BR>

```

HTML Forms

```
<INPUT TYPE="checkbox" NAME="sports" VALUE="yes">Sports <BR><BR>
<INPUT TYPE="submit" VALUE="Submit to email"><BR>
<INPUT TYPE="reset" VALUE="Clear to default settings">
</FORM>
</BODY>
```

For example, say the boxes labelled 'Movies', 'Science' and 'Sports', were checked, as in the figure below.



With default encoding you would receive an attached file in your email containing:

```
movies=yes&science=yes&sports=yes
```

This can be verified with a text editor.

With **ENCTYPE="text/plain"** you would receive the following in your email:

```
movies=yes
science=yes
sports=yes
```

5.7.8 Discussion of Activity 9

Sample HTML and output are given below. Give the form a more presentable layout using either tables or CSS.

```
<!DOCTYPE html>
<html>
<head>
<title>
Job application form
</title>
</head>

<BODY>
  <div class="formLayout">
    <H1>SpeedyPC Job Application Form</H1>
    <HR><BR>
    <FORM METHOD="post" ACTION=mailto:chaombogho@gmail.com>

      <B>Position Applied for:</B> <INPUT type="text" name="position"
size="30" autofocus>

      Job reference code: <INPUT type="text" name="ref" size="15"
bgcolor = "yellow"><BR>

    <HR><BR>
```

HTML Forms

```
First Name:
<INPUT type="text" name="firstname" size="30"
maxlength="30"><BR> Family Name:
<INPUT type="text" name="familyname" size="30"><BR>

Nationality:
<INPUT type="text" name="nationality" size="30"><BR>

Date of Birth:
<INPUT id="dob" name="dob" type = "date"><BR>

Address:<BR>
<TEXTAREA name="address" cols="30" rows="5"></TEXTAREA>
<BR>

Telephone:
<INPUT type="text" name="phone" size="30" maxlength="30">
Email:
<INPUT type="text" name="email" size="30" maxlength="30"
required /><BR>

<HR><BR>

<B>Educational History</B><BR>

School/College, Course Studied, Qualifications Received, Grades:
<TEXTAREA name="eduhistory" rows="6" cols="70"></TEXTAREA>

<B>Employment History</B><BR>

Employer, Date, Position/responsibilities:
<TEXTAREA name="emphistory" rows="6" cols="70"></TEXTAREA><BR>

<B>Personal Statement:</B><BR>
<TEXTAREA name="statement" rows="6"
cols="70"></TEXTAREA><BR><BR>
<HR><BR>

<B>YEARS OF WORK EXPERIENCE:</B><BR>
<input type="range" min="0" max="10" step="1" value="0">
<HR><BR>

<B>First Referee Details:</B><BR><BR> First name:
<INPUT type="text" name="firstname1" size="30"> Family name:
<INPUT type="text" name="surname1" size="30"><BR> Title:
<INPUT type="text" name="title" size="8"> Occupation:
<INPUT type="text" name="occl" size="30"><BR> Relationship:
<INPUT type="text" name="rell" size="20"><BR> Address:<BR>
<TEXTAREA name="address1" rows="4" cols="30"></TEXTAREA><BR>
Telephone:
<INPUT type="text" name="phone1" size="30"> Email:
<INPUT type="text" name="email1" size="30"><BR><BR>

<B>Second Referee Details:</B><BR><BR> First name:
<INPUT type="text" name="firstname1" size="30"> Family name:
<INPUT type="text" name="surname1" size="30"><BR> Title:
<INPUT type="text" name="title" size="8"> Occupation:
<INPUT type="text" name="occl" size="30"><BR> Relationship:
<INPUT type="text" name="rell" size="20"><BR> Address:<BR>
<TEXTAREA name="address1" rows="4" cols="30"></TEXTAREA><BR>
Telephone:
<INPUT type="text" name="phone1" size="30"> Email:
<INPUT type="text" name="email1" size="30"><BR><BR>
<HR><BR>
<INPUT type="submit" name="Submit" value="Submit">
```

HTML Forms

```
<INPUT type="reset" name="reset" value="Reset">

</FORM>
</div>
</BODY>

</html>
```

5.7.9 Discussion of Additional Activity

Your HTML should look something like this:

```
For example, say you were to check the boxes labelled 'Movies',
'Science' and 'Sports', as in the figure below.
<HEAD>
<TITLE>Land Of Boxes Form</TITLE>
</HEAD>
<H1><FONT SIZE="5" COLOR="coral">Land Of Boxes Order
Form</FONT></H1>
<BODY>
<FORM METHOD="post" ACTION="mailto:name@xyz.co.za"
<TABLE>
<TR>
<TD ALIGN="right">Name:</TD>
<TD><INPUT TYPE="text" NAME="name" SIZE="35"></TD>
</TR>
<TR>
<TD ALIGN="right">Email:</TD>
<TD><INPUT TYPE="text" NAME="email" SIZE="35" VALUE=""></TD>
</TR>
<TR>
<TD ALIGN="right">Address:</TD>
<TD><TEXTAREA NAME="address" COLS="30" ROWS="5"></TEXTAREA></TD>
</TR>
<TR>
<TD ALIGN="right">Postcode:</TD>
<TD><INPUT TYPE="text" NAME="postcode" SIZE="20"></TD>
</TR>
</TABLE><BR><BR>
<TABLE BORDER=1>
<TR>
<TD><B>Product</B></TD>
<TD><B>Part Number</B></TD>
<TD><B>Quantity</B></TD>
<TD><B>Unit Price</B></TD>
<TD><B>Subtotal</B></TD>
</TR>
<TR>
<TD>EconoBox</TD><TD>LB100</TD>
<TD><INPUT type="text" name="Qlb100" size="8"></TD><TD>R5</TD>
<TD><INPUT type="text" name="Sublb100" size="15" value="R"></TD>
</TR>
<TR>
<TD>Standard Box</TD><TD>LB200</TD>
<TD><INPUT type="text" name="Qlb200" size="8"></TD><TD>R10</TD>
<TD><INPUT type="text" name="Sublb200" size="15" value="R"></TD>
</TR>
<TR>
<TD>Premium Box</TD><TD>LB300</TD>
```

HTML Forms

```
<TD><INPUT type="text" name="Qlb300" size="8"></TD><TD>R15</TD>
<TD><INPUT type="text" name="Sublb300" size="15" value="R"></TD>
</TR>
<TR>
<TD>Deluxe Box</TD><TD>LB400</TD>
<TD><INPUT type="text" name="Qlb400" size="8"></TD><TD>R20</TD>
<TD><INPUT type="text" name="Sublb400" size="15" value="R"></TD>
</TR>
<TR>
<TD>Super Deluxe Box</TD><TD>LB500</TD>
<TD><INPUT type="text" name="Qlb500" size="8"></TD><TD>R30</TD>
<TD><INPUT type="text" name="Sublb500" size="15" value="R"></TD>
</TR>
<TR>
<TD COLSPAN=5 ALIGN="right">
Total: <INPUT type="text" name="total" size="15" value="R"></TD>
</TR>
</TABLE><BR><BR>
<B>Credit Card Details</B><BR>
<TABLE>
<TR>
<TD>Card Type:</TD>
</TR>
<TR>
<TD><INPUT type="radio" name="cardmake"
value="master">Mastercard</TD>
<TD>Card Number:</TD>
<TD><INPUT type="text" name="cardnumber" size="20"></TD>
</TR>

<TR>
<TD><INPUT type="radio" name="cardmake" value="visa">Visa</TD>
<TD>Expiry date:</TD>
<TD><INPUT type="text" name="expiry" size="10" maxlength="5"
value="mm/yy"></TD>
</TR>

<TR>
<TD><INPUT type="radio" name="cardmake" value="amex">American
Express</TD>
</TR>

<TR>

<TD><INPUT type="radio" name="cardmake" value="diners">Diners
Club</TD>
</TR>
</TABLE><BR><BR>

<TABLE>
<TR>
<TD ALIGN="right"><B>Delivery Address:</B></TD>
<TD><TEXTAREA NAME="delivery" COLS="25" ROWS="5"></TEXTAREA></TD>
</TR>
</TABLE><BR><BR>
<INPUT type="submit" name="submit" value="Send order">
<INPUT type="reset" name="reset" value="Clear form">
</FORM>
</BODY>
```

5.7.10 Answer to Review Question 1

HTML forms provide a way for the user to interact with a server.

5.7.11 Answer to Review Question 2

CGI software is software that adheres to the Common Gateway Interface protocol. A CGI programme is written for a special purpose and performs server-side processing of data submitted from a form.

5.7.12 Answer to Review Question 3

No, the **SIZE** attribute in an `<INPUT>` tag merely sets the size of the field that will be rendered by a browser. The

size of text is limited by the **MAXLENGTH** attribute.

5.7.13 Answer to Review Question 4

To ensure a horizontal scroll bar is included in a multi-line text field, you must add the `<TEXTAREA>` tag the attribute **WRAP=OFF**

5.7.14 Answer to Review Question 5

The problem here is that all the buttons have the name `flight_prefs`, which means that only one name/ value pair will be submitted where three are needed. Further, the first and last sets of radio buttons have unintentionally been made mutually exclusive, again because of the common name

5.7.15 Answer to Review Question 6

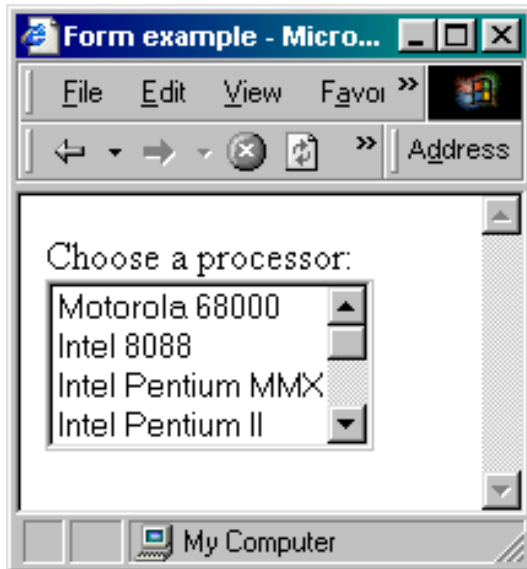
To initially set a check box, include the **CHECKED** attribute in the `<INPUT>` tag.

5.7.16 Answer to Review Question 7

Menu buttons require only one line when rendered. The list of options appears over (usually) whatever is below it on the page. Scrolling lists occupy as many lines as specified by the **SIZE** attribute of the `<SELECT>` tag; since all the options are rarely shown, fewer lines than options are usually used.

5.7.17 Answer to Review Question 8

Output and HTML are given below:



```
Choose a processor:<BR>
<SELECT name="processor" SIZE=4>
<OPTION>Motorola 68000</OPTION>
<OPTION>Intel 8088</OPTION>
<OPTION>Intel Pentium MMX</OPTION>
<OPTION>Intel Pentium II</OPTION>
<OPTION>Intel Pentium III</OPTION>
<OPTION>Intel Celeron</OPTION>
<OPTION>PowerPC G3</OPTION>
<OPTION>PowerPC G4</OPTION>
<OPTION>AMD Athlon</OPTION></SELECT>
```

5.7.18 Answer to Exercise 1

The following name/value pair will be sent to the CGI software. An ampersand (&) would precede or follow it if other pairs were sent.

```
age=39
```

Two layout tips are worth remembering:

1. Use tables to layout the form in an organised way (see Unit 4).
2. You can use the
 tag (no closing tag needed) to move text or form objects to the next line. You can use the <P></P> tags to space your work in paragraphs.

5.7.19 Answer to Exercise 2

Your HTML code should look something like this:

```
What is your favourite movie?<BR>
<TEXTAREA NAME="movie-comments" COLS="50" ROWS="5"
WRAP=OFF></TEXTAREA>
```

5.7.20 Answer to Exercise 3

There may be conflicting requirements here. It is not easy to reflect the fact that the usual destination for customers is within Egypt while promoting the others. For example, a scrolling list with Egypt at the top would appear to satisfy this requirement, but the other countries would not be visible. Breaking alphabetical ordering might distract some users:

```
Destination Country:<BR>
<SELECT NAME="colour" SIZE="3" MULTIPLE>
<OPTION>Egypt</OPTION>
<OPTION>Bahrain</OPTION>
<OPTION>Kuwait</OPTION>
<OPTION>Lebanon</OPTION>
<OPTION>Oman</OPTION>
</SELECT>
```

Alternatively, you could use alphabetical ordering to force the user to scroll through at least the options that precede the most likely destination, Egypt. But there would only be one option before Egypt, so it could be moved to last. To help users find Egypt, it can be pre-selected:

```
Destination Country:<BR>
<SELECT NAME="colour" SIZE="3" MULTIPLE>
<OPTION>Bahrain</OPTION>
<OPTION>Kuwait</OPTION>
<OPTION>Lebanon</OPTION>
<OPTION>Oman</OPTION>
<OPTION SELECTED>Egypt</OPTION>
</SELECT>
```

The final alternative is to use a menu button: it can preserve alphabetic ordering and shows all the other options:



Destination Country:

Bahrain

Bahrain

Egypt

Kuwait

Lebanon

Oman

```
Destination Country:<BR>
<SELECT NAME="colour">
<OPTION>Bahrain</OPTION>
<OPTION>Egypt</OPTION>
<OPTION>Kuwait</OPTION>
<OPTION>Lebanon</OPTION>
<OPTION>Oman</OPTION>
</SELECT>
```

Note that, when using a menu button, you should not pre-select Egypt if you want customers for internal flights to see the other options. Pre-selecting does exactly that: it would leave only Egypt visible on the button and most customers would not click on the button to reveal the other options in the menu.