## Lecture 8

# HTML Tables & Frames

## Objective

**This lecture provides an insight into the following:**

- Basic structure of Tables
- Aligning Table Elements within the cell
- Spanning of Rows and Columns
- Enhancements by the Netscape Navigator to Tables
- Using Frames in a Web Page

# Lecture - 8



# Lab Unit

## 8.1 Snap Shot

Tables can be used for more than just displaying a table of data. Table can also be used as a formatting tool. The data in a table can be text or images. Because many HTML documents rely less on text than their printed equivalents, Web-based tables have become an important way to structure documents. Frames allow the programmer to divide the browser window into sections, each section acting independently of each other. These two concepts are discussed in this lecture

## 8.2 HTML Tables

A table represents information in a tabular way, like a spreadsheet: distributed across a grid of rows and columns. In printed documents, tables commonly serve a subordinate function, illustrating some point described by an accompanying text. Tables still perform this illustrative function in HTML documents.

### Basic Table Structure

Like other block elements, a table is marked at its beginning with a <TABLE> tag and its end with a </TABLE>. A table can contain a variety of row and column definitions within. At its simplest level, a table consists of the <TABLE> tag and one row with one cell:

```
<TABLE>
        <TR> <!-- table row -->
                <TD>Content <!-- table data -->
</TABLE>
```

In a word, this simple table is called "pointless." However, there are a few things to note. First, only the <TABLE> tag needs to be closed with </TABLE>; otherwise, the browser doesn't know when to stop with table formatting. Second, the table row (<TR>) and table data (<TD>) cells can accept their respective closing tags, but they are not required. A new row or data cell automatically marks the end of the previous row or data, except in legacy versions of some browsers.

The <TABLE> tag has several attributes, which are given in the following syntax.

```
<TABLE [ BACKGROUND = path of an image file ]
        [ BGCOLOR  =  color name | #RRGGBB ]
        [ BORDER = pixels ]
        [ BORDERCOLOR = color name | #RRGGBB ]>
```

The BACKGROUND and BGCOLOR are same as in the <BODY> tag. The BORDER attribute sets the width of the border around table in pixels. Generally 0 is assumed to this attribute, i.e no border will appear. The last attribute BORDERCOLOR specifies a color for the table's border. The following listing builds a simple table that appears as in Figure 8.1.

```
<HTML>
<HEAD>
<TITLE>
Simple Table
</TITLE>
</HEAD>

<BODY>
<TABLE BORDER=3>
<TR>
<TD>1.1<TD>Cell 1.2<TD>Data Cell 1.3<TD>Very Big Data Cell 1.5
<TR>
<TD>2.1<TD>Cell 2.2<TD>Data Cell 2.3<TD>Very Big Data Cell 2.5
<TR>
<TD>3.1<TD>Cell 3.2<TD>Data Cell 3.3<TD>Very Big Data Cell 3.5
</TABLE>
</BODY>
</HTML>
```

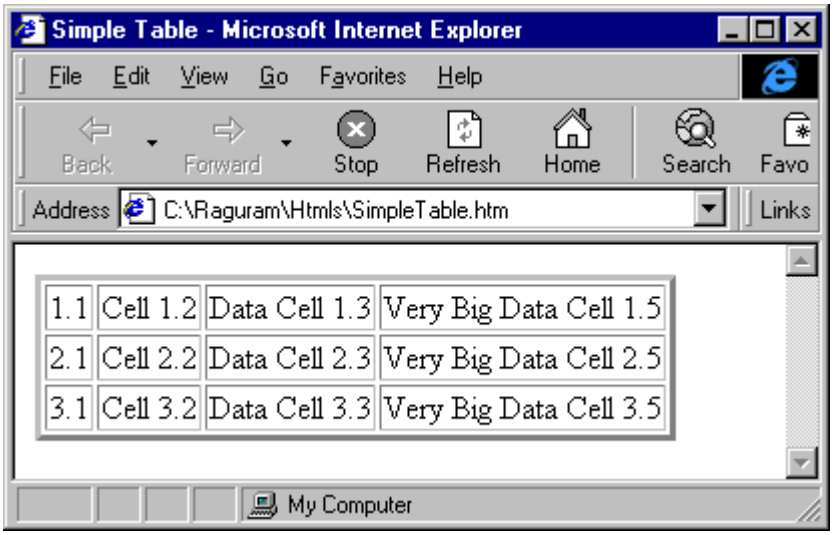



**Figure 8.1** *A simple table.*

## 8.3 Aligning Table Elements

### Cell Content Alignment: ALIGN and VALIGN

It is possible, through the use of the ALIGN and VALIGN attributes to align table elements within their cells in many different ways. These attributes can be applied in various combinations to the <CAPTION>, <TR>, <TH> and <TD> elements.

**Syntax**

<TD ALIGN = left | center | right  VALIGN = top | middle | bottom >

In these attributes, ALIGN controls horizontal justification and VALIGN controls the vertical justification.

Explaining the Table: <CAPTION>

An important element for a table is a caption to identify it. The <CAPTION> tag is to the table what the <TITLE> tag is to be <HEAD> of the document, except that it is used to specify whether the caption appears at the top or bottom of the table.

If included, a caption must appear immediately after the open <TABLE> tag before any table rows or cells are defined.

**Syntax**

<TABLE>
    <CAPTION [ ALIGN=alignment] >Caption Text</CAPTION>
    ...Table contents
</TABLE>

The ALIGN attribute is used slightly differently from other elements. In this case, it indicates where the caption is placed in relation to the table. Its possible values are **top, bottom, left, or right.**

***Beginning to Include Data: <TR>***

The beginning of a row is marked with a <TR> tag.

**Syntax**

<TR [ ALIGN = horizontalAlign ] [ VALIGN = verticalAlign ] >

<TR> marks the beginning of a row of cell definitions and can set default display settings for its cells. The ALIGN attribute sets the horizontal spacing for cells on the row and can be one of the following values: **left, right, or justify.**

The VALIGN attribute sets the vertical placement of information in the cell and takes one of these values: **top, middle, or bottom.**

***Individual Data Cells and Headings: <TD> and <TH>***

With the beginning of the row marked with <TR>, it's time to finally get down to the work of filling each cell. Two types of cell tags are used in a table. The first, <TH>, marks a header cell, which is similar to a heading tag on a Web page. Most browsers include a different font style for header cells to help emphasize their purpose to the user.

**Syntax**

<TABLE>
<TR>

<TH>*Cell Header*</TABLE>

The *Cell Header* is the cell's header content. Generally this tag (<TH>) is used in defining a table's first row.

The other tag is a data cell tag, <TD>, used for the body of the table.

**Syntax**

<TD [ROWSPAN = numRows] [COLSPAN=numCols]

[alignment]> cellContent

The attributes for <TD> are discussed in the following sections. A closing </TD> or </TH> is not required for individual cells, although many HTML designers and editing programs use it for clarity and compatibility with some older browsers.

## 8.4 Row and Column Spanning

### Cell Size Attributes: ROWSPAN and COLSPAN

The first set of attributes determines its capability to merge with an adjacent cell. The ROWSPAN and COLSPAN attributes are used to combine adjacent cells into larger cells. It's important to note that when these attributes are used, the adjacent cells aren't eliminated; they're just "hidden" while the acquiring cell uses their space.

**Syntax**

<TD [ROWSPAN = numRows] [COLSPAN=numCols]>

In this attribute, numRows is the number of rows, including the current cell, that are joined. Likewise, numCols is the number of columns joined together. The default for both values is 1.

Spanning is a little tricky to plan and carry out. For example, start with look at the following listing, which has a 3 X 3 table, which requires three rows with three cells each.

```
<HTML>
<HEAD>
<TITLE>
Cells Span
</TITLE>
</HEAD>
```

```
<BODY>
<CENTER>
<TABLE BORDER=3>

<TR>
        <TD>Cell 1.1 <TD>Cell 1.2 <TD>Cell 1.3
<TR>
        <TD>Cell 2.1 <TD>Cell 2.2 <TD>Cell 2.3
<TR>
        <TD>Cell 3.1 <TD>Cell 3.2 <TD>Cell 3.3

</TABLE>
</CENTER>
</BODY>
</HTML>
```

Now it is time to merge the cells Cell 1.1 and Cell 1.2. Because this method reaches across columns, it is called *column span*. To do this change the data tag of the Cell 1.1 in the above listing as given below:

```
<TD COLSPAN=2>Cell 1.1
```

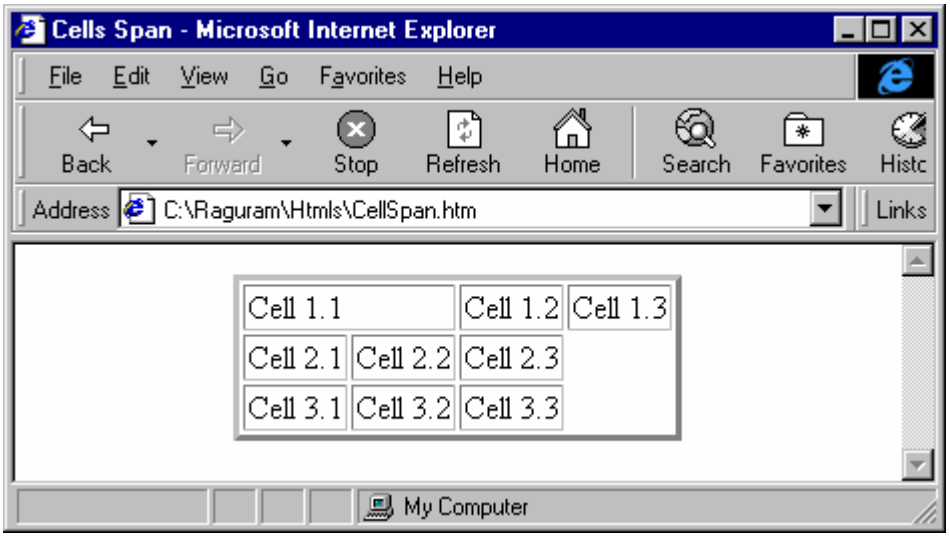



**Figure 8.2** *The revised table with Column span.*

This change produces document as in Figure 8.2.

Notice the bottom two rows of cells in Figure 8.2. This reveals that spanning cells isn't quite simple as just adding the span attributes. Here's what happened: when the browser encountered COLSPAN = 2; it knew Cell 1.1 needed to take up the space of two cells, and it provided the space accordingly. Then, it continued across the row and finished adding the next two cells. The result was space for a total of four cells (two joined and two individuals).

At the next row tag, the browser started a new row by adding three cells. The browser didn't encounter a fourth <TD> tag, even though there was room in the table for one. Instead of adding a cell that the user didn't specify, it simply filled in with blank space. The final result is a 3 X 4 table with the bottom two rows only partially defined.

This feature of cell spanning can cause many headaches when working with tables. For each cell span, the corresponding cell definition is to be removed.

Remove the cell definition **<TD> Cell 1.2** in the previous listing and view it in the browser. It will appear as in Figure 8.3.

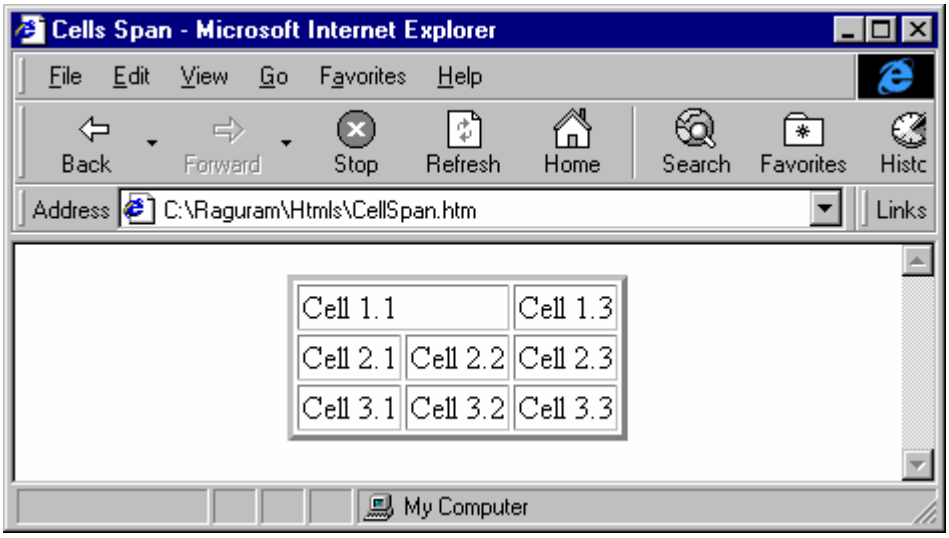



**Figure 8.3** *Table's appearance after the adjoining cell removal.*

The same rules also apply for the ROWSPAN attribute, except cells below the originating tag should be removed. For example, in the following listing three adjoining cell tags are removed. It's also possible to combine the two attributes for other effects (see Figure 8.4).

```
<HTML>
<HEAD>
<TITLE>
Cells Span
</TITLE>
</HEAD>

<BODY>
<CENTER>
<TABLE BORDER=3>

<TR>
        <TD ROWSPAN=2 COLSPAN=2>Cell 1.1  <TD>Cell 1.3
<TR>
        <TD>Cell 2.3
<TR>
        <TD>Cell 3.1 <TD>Cell 3.2 <TD>Cell 3.3
```

```
</TABLE>
</CENTER>
</BODY>
</HTML>
```

This produces the table as in Figure 8.4.


**Figures 8.4** *Table in which four adjoining cells are combined together.*

When spanning in two directions at once, it is required to delete the all the data cells in the path of the merge, which means removing cells from the right, left, and diagonal directions.

## 8.5 Netscape Table Enhancements

Netscape Navigator has introduced several enhancements to HTML tables to increase the degree of control. This allows HTML authors to have design how their documents are displayed. The Netscape table enhancements are as follows

WIDTH attribute – This enables the author to specify the width of the table, either in pixels or as a percentage of the width of the browser window

HEIGHT attribute – This enables the author to specify the height of the table, either in pixels or as a percentage of the height of the browser

BORDER attribute – This attribute exists in the draft HTML 3.0 specification and puts a border around the table, and in that respect is supported by most Web browsers with table support. The enhancement also enables it to be used as a numerical attribute, BORDER=<num>, which makes the border <num> pixels wide

It should be noted here that, when using the Netscape BORDER=<num> table enhancement, it is possible to specify a table with no borders by including BORDER=0 in the <TABLE> element. While this will give a borderless table when viewed with Netscape Navigator, Web browsers that do not support this enhancement will ignore this "=0" and display the table with a border. So, to use a borderless table that will work on all browsers that support tables, include the <TABLE> element without specifying a BORDER attribute at all.

CELLPADDING and CELLSPACING – These numerical attributes include extra space within each cell in the table and/or within the borders of the table. If the border is not being displayed, they are equivalent.

Nested Tables – Netscape Navigator enables tables to be included as elements within other tables.

### Using the STYLE attribute

The STYLE attribute can also be used (as used in <P> tag) for table elements to format the table, rows and columns independently. The following listing shows how to use this attribute.

```
<HTML>
<HEAD>
<TITLE>
Projects of Cyber World
</TITLE>
</HEAD>

<BODY>
<CENTER>
<H2>Department of Projects, Cyber World</H2>

<HR>
We have developed awesome projects, as this table shows. <BR>

<TABLE Cols=3 Border Bordercolor="Black" Style="color:green;background-color:Yellow">
<CAPTION Style="font-weight:bold;font-size:15">Project Details</CAPTION>

<TR>
<TD Rowspan=2 Style="text-align:center;font-weight:bold;background-
color:Goldenrod">Projects</TD>
```

```
<TD Colspan=2 Style="text-align:center;font-weight:bold;background-
color:Goldenrod">Team</TD>
</TR>

<TR Style = "text-align:center;font-weight:bold">
<TD Style = "background-color:Goldenrod">Leader</TD>
<TD Style = "background-color:Goldenrod">Programmers</TD>
<TR>

<TR>
<TD>Computer Based Training Development Platform  </TD>
<TD>Saravana Kumar</TD>
<TD>Kalathi, Antony</TD>
</TR>

<TR>
<TD>Net Train</TD>
<TD>Balaji</TD>
<TD>Kannan, Ramanathan</TD>
</TR>

<TR>
<TD>256 Color Graphics Library</TD>
<TD>Raguram</TD>
<TD>Ganesh Sudhakar, Mythili</TD>
</TR>

</TABLE>
</CENTER>

</BODY>
</HTML>
```

This listing produces a table as in Figure 8.5.


**Figure 8.5** *Appearance of the Table due to the STYLE attribute.*

## 8.6 Frames in HTML

Though tables, images and hyperlinks add attraction to Web pages, these alone are not enough to design pretty Web pages. Frames come handy in many ways to design web pages. It enables to design sophisticated user interfaces. A framed document divides a browser window into multiple panes, or smaller window frames. Each frame may contain a different document.

The benefits of this approach are obvious. Users can view information in on frame while keeping another frame open for reference, instead of moving back and forth between pages. The contents of one frame can be manipulated, or linked to the contents of another. This allows designers to build sophisticated interfaces. For example, one frame can contain links that produce a result in another frame. A Figure for such an interface is given in the next section.

### Overview of Frames

A frame is an independent scrolling region, or window, of a Web page. Every Web page may be divided up into many individual frames, which can even be nested within other frames. Fixed screen sizes limit how many frames can realistically be

used at once. Frames provide navigation facilities. Figure 8.6 provides a visual overview of the components of a framed document.

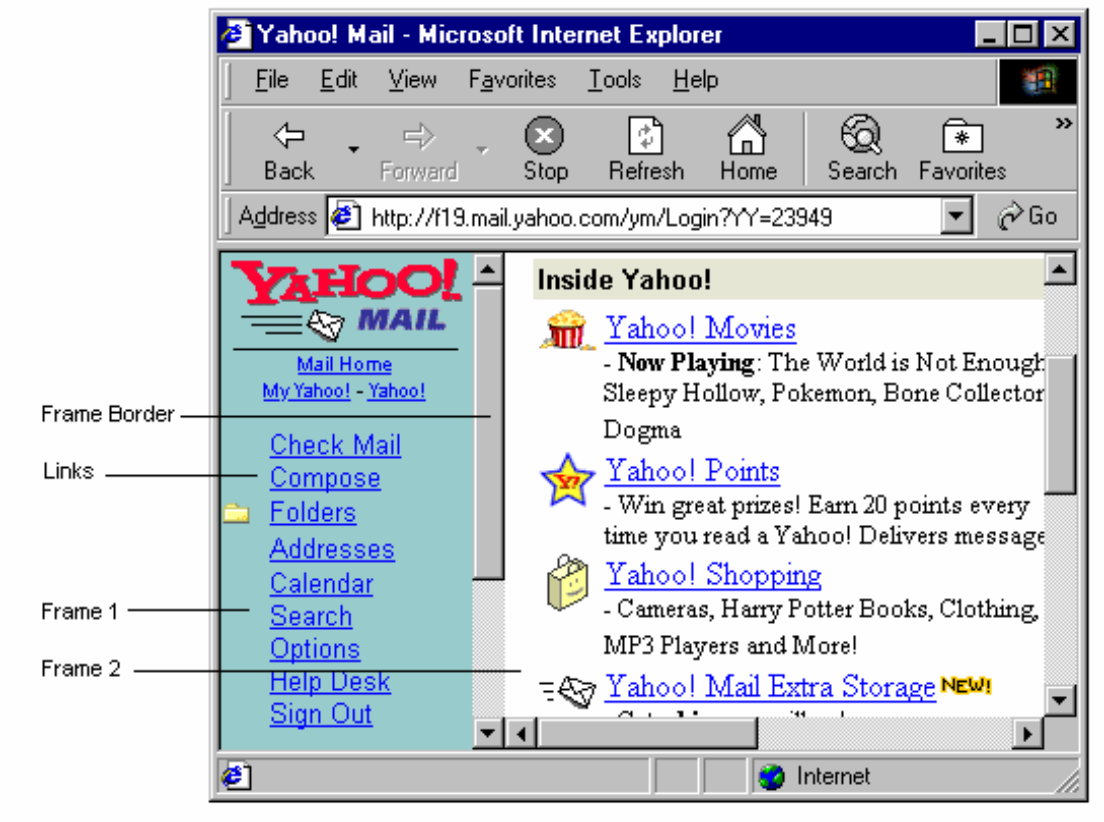



**Figure 8.6** *Frame road map.*

## 8.7 Frameset Container

The first point to remember is that a framed document is composed of several documents. To illustrate, a page with two frames will actually involve three files:

- The framing document that defines the framing relationship
- The file that contains the contents of frame one
- The file that contains the contents of frame two

A simple two-frame document is shown in Figure 8.7.

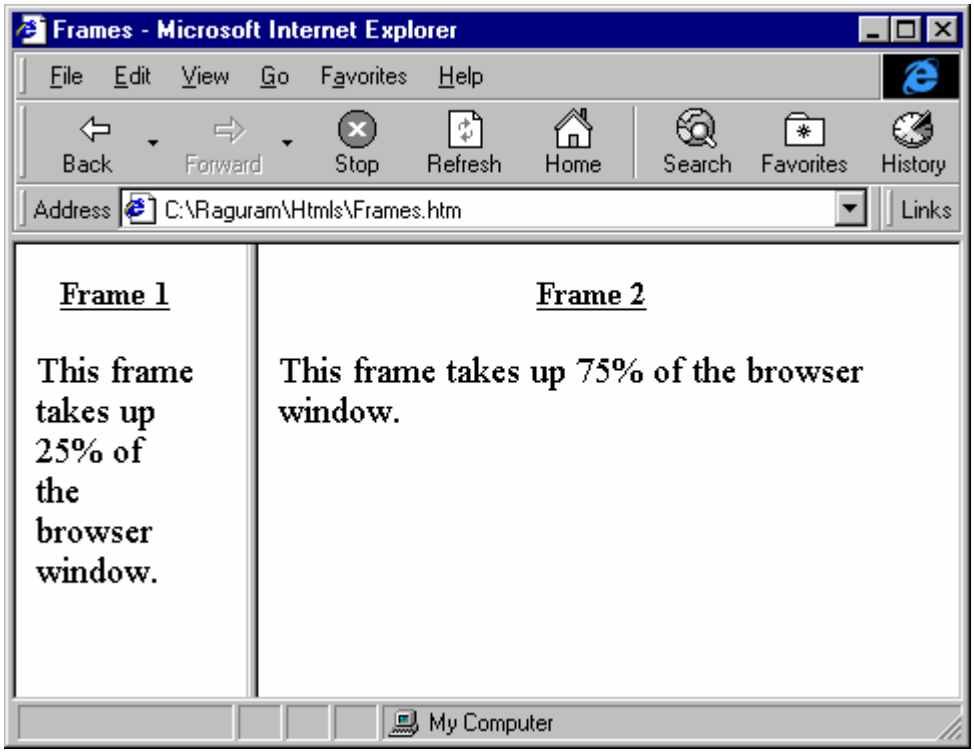



**Figure 8.7** *A framed document.*

The first frame, on the left, takes up around 25 percent of the browser window and the right frame takes up the other 75 percent of the browser window. For the purpose of this example, the left frame is named as Frame1 and the right frame as Frame2.

To specify a framing document, the <FRAMESET> element is used in the HTML file instead of the <BODY> element. This element defines the set of frames that make up the document.

**Syntax**

```
<FRAMESET
        [ BORDER = pixels ]
        [ BORDERCOLOR = color ]
        [ COLS = columns width ]
        [ FRAMEBORDER = NO | YES ]
        [ FRAMESPACING = pixels ]
        [ ROWS = row heights ]
        [ TITLE = advisory text ]

<FRAME>....<FRAME>
</FRAMESET>
```

The attributes of the <FRAMESET> element are described below:

**Border** This attribute set the width in pixels of frame borders within the frame set. Settings BORDER=0 eliminates all frame borders.

**Bordercolor** This attribute sets the color for frame borders within the frame set using either a named color or color specified in the hexadecimal #RRGGBB format.

**Cols** This attribute contains a comma-delimited list, which specifies the number and size of columns contained with a set of frames. List items indicate columns, left to right. Column size is specified in three formats, which may be mixed. A column can be assigned a fixed width in pixels. It can also be assigned a percentage of the available width, such as 50 percent. Last , a column can be set to expand to fill the available space by setting the value to *, which act as a wildcard.

**Frameborder** This attribute controls whether or not frame borders should be displayed. Netscape supports YES and NO values. Microsoft uses 1 and 0 as well as YES and NO.

**Framespacing** This attribute indicates the space between frame in pixels.

**Rows** This attribute contains a comma-delimited list that specifies the number and size of rows contained within a set of frames. The number of entries in the list indicates the number of rows. Row size is specified with the same formats used for columns.

### The Frame Tag

Once the frame layout is specified with the <FRAMESET> element, the contents of each frame must be specified using the <FRAME> element in the order that the frames were defined in the ROW or COL attribute. In the case of <FRAMESET COLS = "25%, 75%">, the contents of the first <FRAME> element encountered is loaded in the 25% column, and the contents of the second <FRAME> element in the 75% column.

### Syntax

```
<FRAME
        [ NAME = string ]
        [ NORESIZE ]
        [ SCROLLING = YES | NO | AUTO ]
        [ SRC = URL of frame contents ] >
```

The attributes of the <FRAME> element are described below:

**Name** This attribute assigns the frame a name so that it can be the target destination of hyperlinks.

**Noresize** This attributes overrides the default ability to resize frame and gives the frame a fixed size.

**Scrolling** This attribute determines if the frame has scroll bars. YES value forces scroll bar, a NO value prohibits them, and an AUTO lets the browser decide. When not specified, the default value of AUTO is used.

**Src** This attribute contains the URL of the content to be displayed in the frame. If absent nothing will be loaded in the frame.

The code that displays the document as in Figure 8.7 is given below:

```
<HTML>
<HEAD>
<TITLE>Frames</TITLE>
</HEAD>
<FRAMESET Cols = 25%,75% >
        <FRAME Name = "Frame1" Src = "Frame1.htm" >
        <FRAME Name = "Frame2" Src = "Frame2.htm">

        <NOFRAMES><P>This document uses frames. Please follow this link to a
         <A HREF = "noframes.htm">noframes</A> version.
        </NOFRAMES>

</FRAMESET>
</HTML>
```

The above listing uses the <NOFRAMES>...</NOFRAMES> element within the frame set. This element provides information to be displayed in browsers that do not support frames. Although this approach seems like a good idea, the page author may need to maintain both frame and no frame version of a site in order to accommodate different browsers.

## More about Frames (*For self learning*)

### Frame Targeting

When using frames it may be desirable to make the links in one frame target another frame. This way, when a user activates a link in one framed document, the requested page loads in another frame.

The first part of link targeting is to ensure frame naming by setting the **NAME** attribute in the **<FRAME>** element to a unique name. The next part of linking is to use the **TARGET** attribute in the <A> element to set the target for the anchor. For example, a link like,

<A HREF = "http//www.excite.com" **TARGET =** "Frame2">

would load the site specified by the HREF attribute into the window called "Frame2", if there is such a frame. If the target specified by the name does not exist, the link loads over the window it is in. Some particular values for the TARGET attribute have special meaning; these are summarized in the following Table.

| Value | Meaning |
|---|---|
| _new | Load the page into a new, generally unnamed, window. |
| _self | Load the page over the current frame. |
| _parent | Load the link over the parent frame. |
| _top | Load the link over all the frames in the window. |

Reserved **TARGET** values.

The **_parent** value is not often encountered, because it is only useful when frames are nested to great degree. The **_parent** value makes it possible to overwrite the parent frame that contains the nested frame.

The following listing, which is contained in the file Photos.htm, constructs nested frames. First it splits the window into two rows by setting the **ROWS = 2**. The first row displays the contents of the file **Title.htm,** which displays the title "Photo Exhibition". This row can't be resized as its corresponding <FRAME> element has the **NORESIZE** attribute. The second row is further divided into two columns by using yet another <FRAMESET> element which has **COLS = 2**. The first column displays the contents of the file **Links.htm**, which contains links to photos. When a link is activated, an HTML file that contains its corresponding photo is targeted to the second column. This column is named as "Photo" and the links in the Links.htm file use this name for their **TARGET** attribute.

```
<HTML>
<HEAD>
<TITLE>Photos</TITLE>
</HEAD>
<FRAMESET ROWS = 20%,* >
        <FRAME Name = "Title" Src = "Title.htm" NORESIZE>
        <FRAMESET COLS = 20%,* >
                <FRAME Name = "Links" Src = "Links.htm" >
                <FRAME Name = "Photo">
        </FRAMESET>
</FRAMESET>
</HTML>
```

The listing of the Title.htm is given below.

```
<HTML>
<BODY BGCOLOR = "silver">
<H2 ALIGN=Center><U>Photo Exhibition</U></A>
</BODY>
</HTML>
```

The listing of the Links.htm is given below:

```
<HTML>
<BODY BACKGROUND="C:\Shared\Images\backgrd.gif">
<Center><B>Click any one of the following:</B></center>
<BR>
```

```
    <A HREF = "Desert.htm" TARGET="Photo"> Desert </A> <BR>
    <A HREF = "Sea.htm" TARGET="Photo"> Sea </A> <BR>
    <A HREF = "Clouds.htm" TARGET="Photo"> Clouds </A> <BR>
<A HREF = "Mountains.htm" TARGET="Photo"> Mountains </A>

    </BODY>
    </HTML>
```

The listing of the Desert.htm is given below:

```
<HTML>
<BODY>
<IMG ALIGN=Center SRC= “Desert.jpg” >
</BODY>
</HTML>
```

The HTML files for the other links are similar to the Dersert.htm with one exception. It will contain different image source.

When the Photos.htm is viewed in the browser it will appear as in Figure 8.8.


**Figure 8.8** *A typical framed document.*

## Floating Frames: <IFRAME>

Up until this point, all the frames shown have been attached to the sides of the browser (left, right, top, or bottom). Another form of frame, called a *floating frame,* is introduced by Microsoft. The idea of the floating frame is to create an inline framed region. And in this region text or image can be flowed around it. An inline frame is defined by the <IFRAME> element and may occur anywhere within the <BODY> of an HTML document.

**Syntax**

<IFRAME
[ ALIG N = LEFT | RIGHT | TOP | BOTTOM ]
[ BORDER = pixels ]
[ BORDERCOLOR = color ]
[ FRAMEBORDER = YES | NO ]
[ HEIGHT = pixels or percentage ]
[ NAME = frame name ]
[ NORESIZE ]
[ SRC = URL of frame contents ]
[ WIDTH = pixels or percentage ] >
*... Contents for the browser that do not support floating frames*

</FRAME>

The major attributes to set for the <IFRAME> element include SRC, HEIGHT, and WIDTH. The SRC is set to the URL of the file or image to load, while the HEIGHT and WIDTH are set to the pixel or percentage value of the screen that the floating frame region should consume. The other attributes are same as in <FRAMESET> element.

Note that unlike the <FRAME> element the <IFRAME> comes with the close tag </IFRAME>. <IFRAME> and </IFRAME> should contain any HTML markup code and text that should be displayed in browsers that do not support floating frames.

The following listing gives example to use <IFRAME> element to display a big image, which appears as in Figure 8.9.

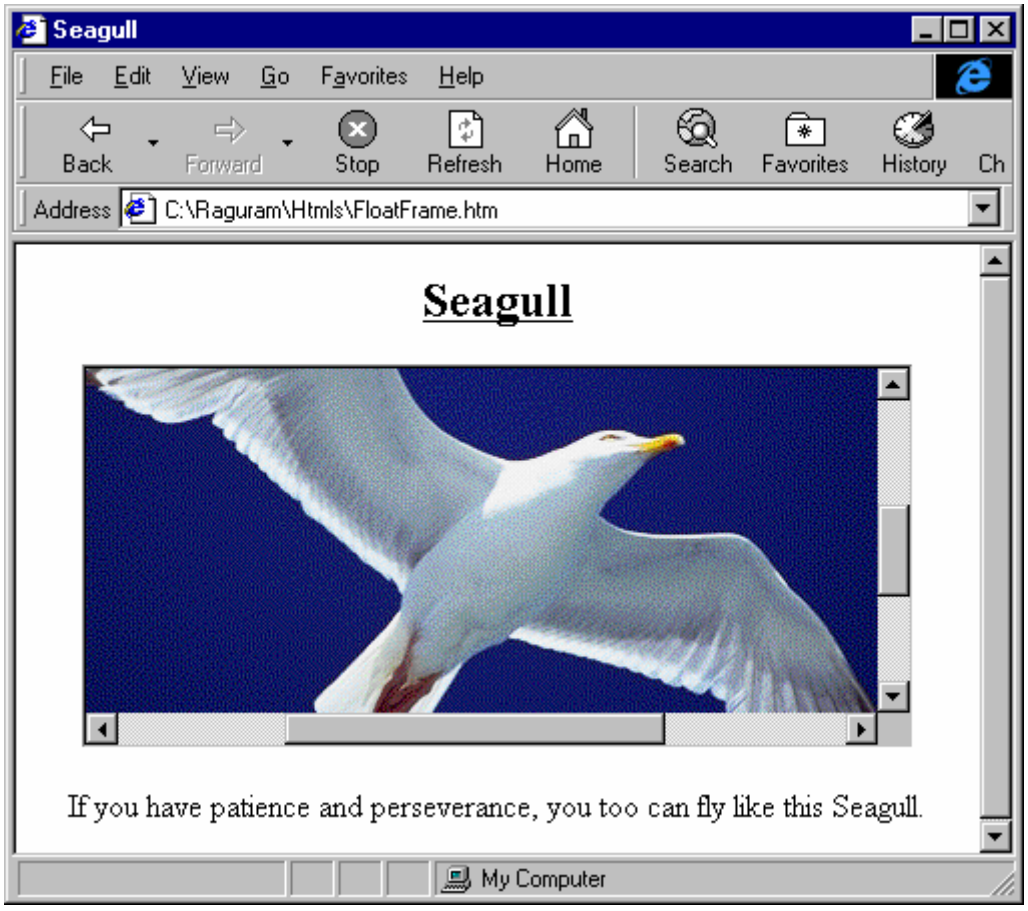



**Figure 8.9** *Floating frame containing an Image.*

If the URL of an HTML file is assigned to the **SRC** attribute of the IFRAME element, the HTML file will be displayed in side the floating frame.

## 8.8 Short Summary

- A table represents information in a tabular way, like a spreadsheet: distributed across a grid of rows and columns.
- Cell Content Alignment can be performed using ALIGN and VALIGN
- The ROWSPAN and COLSPAN attributes are used to combine adjacent cells into larger cells.
- A framed document divides a browser window into multiple panes, or smaller window frames.
- A frame is an independent scrolling region, or window, of a Web page.
- To specify a framing document, the <FRAMESET> element is used instead of the <BODY> element.

## 8.9 Brain Storm

1. Is it possible to create tables without border? Using border attribute is it possible to have table without border?

2. Is it possible to align the table in the center of the browser window

   a. Yes  b. No

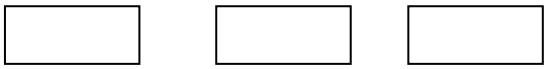
   If Yes what is the attribute?

3. Is it possible to align the contents of a particular row inside a table without affecting other Row's contents

   a. Yes  b. No

4. Is it possible to get a table structure like the following

   Table 1  Table 2  Table 3



   a. Yes  b. No

5. What is the need for a frame?
6. How many body tags are possible inside a frame set?
7. Which of the following browsers support frame?
   a. Internet Explorer 3.0
   b. Netscape Navigator 4.0
   c. Internet Explorer 4.0
   d. Netscape Navigator 3.0
   e. Mosaic
8. What is the significance of noresize attribute?
9. What is the difference between Yes and auto in a scroll attribute?
10. What is a floating frame?
11. Is it possible to have more than one floating frame in an HTML document?
12. What will happen if a browser does not support frames?
13. What is the use of no frames?
14. When parent value can be used for the target attribute, where the result is placed?

## Lab Unit (2 *Real Time Hours*)

This Exercise will make you understand completely about Table and Links. The web page you are about to create is for LIFE STYLE -Shopping complex.

Heading LIFE STYLE - SUPER MALL

**The contents -**

*LIFE STYLE is a chain of International shopping mall Estd. in 1980 in CHENNAI. Promoted by Business tycoons from Singapore with an investment of nearly 100 crores. In the very first Year of its launch LIFE STYLE has captured a major share in the market and they have opened 20 Branches across 5 countries.*

*The Details about their Branches:*

LIFE STYLE 's INTERNATIONAL MARKET

| COUNTRY | PLACE |
|---|---|
| ***INDIA*** | ***CHENNAI*** |
| | ***BANGALORE*** |
| | ***HYDERABAD*** |
| ***AUSTRALIA*** | ***SYDNEY*** |
| | ***MELBOURNE*** |
| ***CANADA*** | ***TORONTO*** |
| ***AMERICA*** | ***NEW YORK*** |

All the leading Corporate are is doing business of their products through LIFE STYLE.

The following are available for the customers in LIFE STYLE

FOOD & BEVERAGES

CLOTHING

AUDIO / VIDEO

VARIETY ITEMS

The listed items are HYPERLINKS to other pages.

Create corresponding web page, which gives more detailed info about the selected link items.

For example if you click AUDIO/VIDEO link it takes you to…/audiovideo.htm

| *Customer's Choice* | | |
|---|---|---|
| **BRAND NAME** | ***PRODUCT DESCRIPTION*** | ***PRICE*** |
| *AIWA* | DVD PLAYER | *$19000* |
| *PANASONIC* | *VCD HI-FI MINI PLAYER* | *$12,800* |
| *THOMSON* | *21" FFST T.V* | *$8000* |
| | *29" HOME THEATRE SYSTEM* | *$14,980* |
| | *WALKMAN* | *$20* |
| *LG* | *GOLDEN EYE* | *$13,777* |

Design the Web Pages according to your requirement. Use all the possible TAG options.

## Lecture 9

# HTML Forms

## Objective

**This lecture provides an insight into the following:**

- Using Forms to create interactive Web pages
- The Form elements
- What are dynamic documents
- Including background graphics and color to our Web page
- Playing background sound in our Web page
- Marquees as a tool for scrolling

# Lecture - 9



# Lab unit

## 9.1 Snap Shot

It is very important to note that, most Web pages do more than just rendering information. E-commerce, the emerging technology is done through the Web. HTML offers Forms to design Web pages that can be used for order processing on a retail site or they can be set up to get customer feedback.

## 9.2 HTML Forms

Forms are constructed in the HTML documents by using the <FORM> element. This element contains several other elements, called controls that have a variety of methods for gathering information. When a form is completed and submitted, the information in its active controls is passed to a program that takes whatever action the form has been designed to perform. Each element in the form has both a name and a value, thus the data that's passed for processing is in the form of name/value pairs.

The processing of data is done by scripting languages or CGI program. CGI programs are written in the languages - Perl, Java and C. This chapter doesn't deal with the data processing but shows the construction of Forms using its controls.

### The FORM element

The FORM element has three main attributes – Name, Action and method.

### Syntax

```
<FORM
        [ NAME = string ]
        [ ACTION = URL ]
        [ METHOD = GET | POST ] >
        … Form elements
</FORM>
```

The NAME attribute gives a name to the Form. The ACTION attribute gets an URL that specifies the address of the program used for processing the data, as in the following example:

```
<FORM ACTION = "http:/www.cybershopping.com/getmoney.pl"
   METHOD = "POST">
……
</FORM>
```

The METHOD attribute can have either GET or POST as its value; GET is the default value choice. It submits the name/value pairs to the URL specified in the METHOD attribute as an appendage to the URL itself; POST, on the other hand, sends them as a separate section following the HTTP header. This separate section is called the

*entity body*. But the last two attributes are required only when dealing with Active Server Pages.

## 9.3 The <Input> Tag

The INPUT Element is the most critical to using forms. It's entirely possible to build and entire form using no other elements due to the variety of widgets available through the **TYPE** attribute (see Figure 9.1).

### Syntax

<INPUT [ TYPE = text | password | checkbox | radio | submit | reset | image | button ]
[ NAME = control name ]
[ VALUE = control value ]
[ CHECKED ]
[ DISABLED ]
[ READONLY ]
[ SIZE = control width ]
[ MAXLENGTH = word length ]
[ SRC = URL ]
[ ALIGN = left | center | right ]
[ TABINDEX = tab number ] >

Acceptable values for the TYPE attribute are listed in Table given below

| Form Controls | Values for the TYPE attribute |
|---|---|
| Custom push button | Button |
| Off/On check box | Checkbox |
| Graphic files | Image |
| Masked text entry | Password |
| Radio buttons | Radio |
| Reset button | Reset |
| Submit button | Submit |
| Text entry boxes | Text |

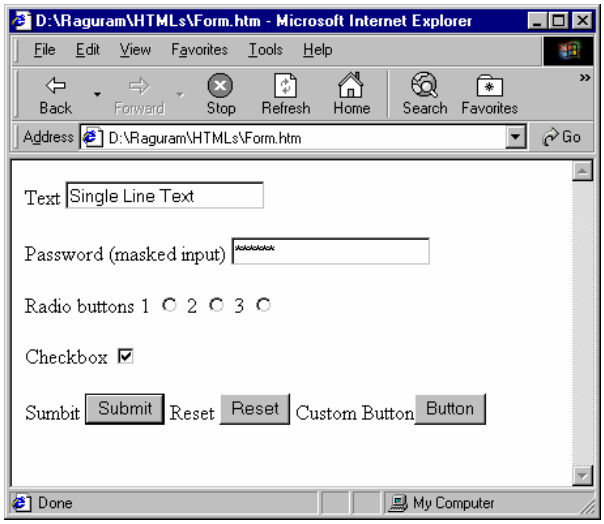



**Figure 9.1** *Form built with only INPUT element.*

The following sections examine each type of button in more detail.

### The Button value

Before HTML 4, the only buttons available were the Submit and Reset buttons whose meaning and actions were predetermined. The button input type, however, has no default function; its function is defined by the author in a script. A typical button declaration looks like this:

<INPUT TYPE= "Button" NAME= "button_01" VALUE = "Click Me">

Each control is assigned a unique identifier with the NAME attribute; without this identifier, it would be impossible to tell which control was assigned what value. The VALUE attribute's text is displayed on the button.

### The Reset Value

The reset value for the type attribute creates a button with only one purpose: all form entries are cleared to their default entries or left blank if no default is specified. It's declared with the following line:

<INPUT TYPE= "Reset">

### The Submit Value

The Submit value of the TYPE attribute creates a button that, like the Reset button, has only a single purpose. In this case, it's to send the name/value pairs of the active form elements to the URL specified in the FORM declaration.

Strangely enough, the HTML specification enables multiple Submit buttons to exist. The Submit button is declared with the following code:

<INPUT TYPE= "Submit">

### The CheckBox Value

The Checkbox value of the TYPE attribute is a Boolean input device; it's either off or on. It looks like a hollow box that, when selected is filled with a check mark to indicate its active state. A check box is extremely versatile and can be used in several different ways. The basic code for declaring a check box is as follows:

<INPUT TYPE= "Check" NAME = "choice1" VALUE = "Cricket" CHECKED >

The **CHECKED** attribute is optional; it causes the check box to be "on" (that is, filled with a check mark) when the form is first created. Of course, if a user clicks on the filled check box, it switches to its empty state. The CHECKED attribute doesn't establish a permanent state; it just sets the Default State of the check box.

To enable user to make multiple selections on the same topic, several check boxes can be used with the same NAME and a different VALUE. For example, to request different choices on hobby, the code will be something like this:

```
<H3> Choose your hobbies </H3>
<FORM>
<INPUT TYPE= "Checkbox" NAME= "HOBBY" VALUE = "Games">Playing Games <BR>
<INPUT TYPE= "Checkbox" NAME= "HOBBY" VALUE = "Songs">Hearing Songs <BR>
<INPUT TYPE= "Checkbox" NAME= "HOBBY" VALUE = "Books">Reading Books <BR>
<INPUT TYPE= "Checkbox" NAME= "HOBBY" VALUE = "Shopping" CHECKED> Shopping
</FORM>
```

The result of this code is shown in Figure 9.2.

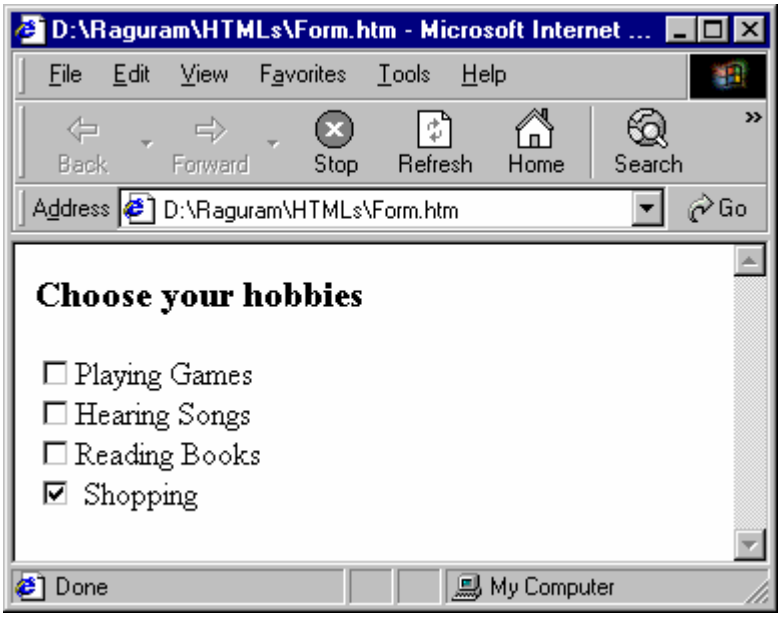



**Figure 9.2** *A Form with multiple check boxes.*

### The Radio Value

The radio value of the TYPE attribute is very similar to the Checkbox type. The difference between the two is that radio buttons are mutually exclusive meaning that selecting one turns the others off. So only the final selection is sent along with the other form data when the form is submitted. Example of designing radio buttons is given below:

```
<H3> Choose your pet </H3>
<FORM>
<INPUT TYPE= "Radio" NAME= "Pet" VALUE = "Dog" CHECKED>Dog <BR>
<INPUT TYPE= "Radio" NAME= "Pet" VALUE = "Cat">Cat <BR>
<INPUT TYPE= "Radio" NAME= "Pet" VALUE = "Dove">Dove <BR>
</FORM>
```

The result of this code is shown in Figure 9.3.

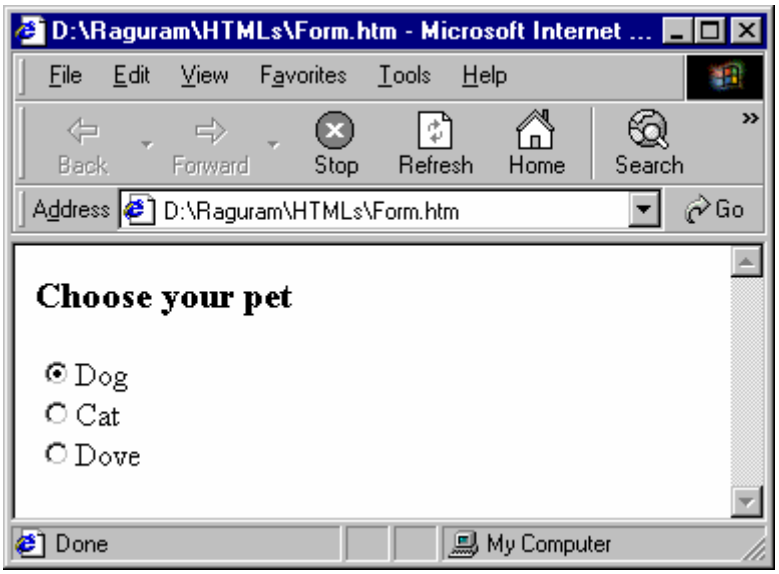



**Figure 9.3** *A set of radio buttons.*

### The Text Value

It asks for input in the form of a single-line, typed response of a given length. The code for adding text response for a user's first name is as follows:

```
<INPUT TYPE= "Text" NAME= "FirstName">
```

A number of text responses could be strung together to create a meaningful form, as shown in the following code and illustrated in Figure 9.4.

```
<H3> Your Address Please </H3>
<FORM>
Name : <INPUT TYPE= "Text" NAME= "Name" SIZE = "50"> <BR>
```

Address: <INPUT TYPE= "Text" NAME= "Address" SIZE = "40">
Phone: <INPUT TYPE= "Text" NAME= "Phone" SIZE = "20"> <BR>
</FORM>

The size attribute sets the length of the text box in characters. It is also possible to set the maximum number of characters that can be entered by using the MAXLENGTH attribute. The code for setting MAXLENGTH IS looks like this:

<INPUT TYPE= "Text" NAME= "Name" SIZE = "50"
MAXLENGTH = "50">

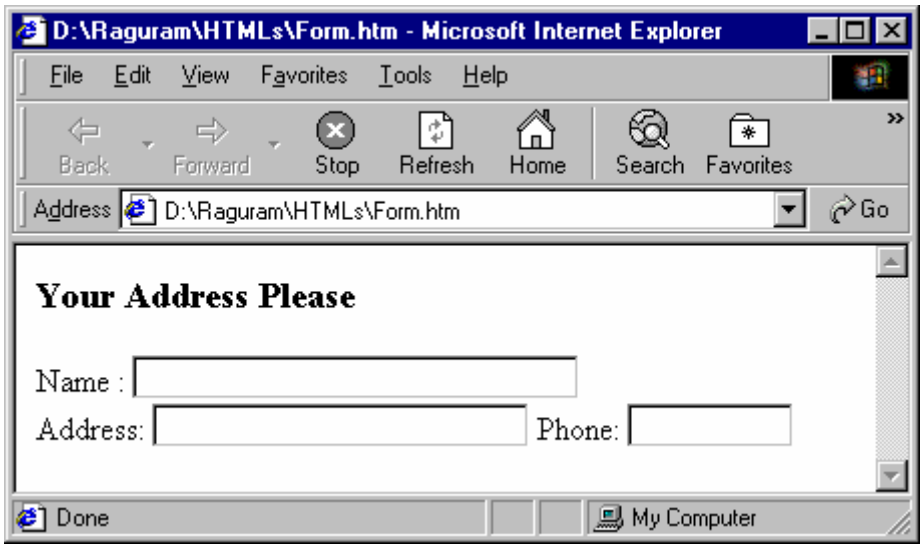



**Figure 9.4** *Text boxes used to gather visitors input.*

Text can be locked against change by the READONLY attribute. The value of such text is included with the values of other active elements when the form is submitted. Read-only elements are capable of receiving focus and are included in tabbing navigation. This kind of text enables the user to see what's being sent.

### The Password Value

The Password value of the TYPE attribute is exactly like the TEXT attribute, except that the visible response on the form is masked with an asterisk or similar character so that no one looking over the shoulder of the user can read what is typed. Here's an example of the code for setting Password:

<INPUT TYPE= "Password" NAME= "Secret" SIZE= "20">

## 9.4 Dynamic Documents

Server push and client pull are Netscape 2.0's innovative new techniques for automatically updating Web pages. We shall now consider a few uses for these "dynamic documents"

- Creation of automated slide shows by loading sequence of pages at timed intervals

- Can automatically advance viewer to another site. This is useful, especially if our Web site moves to a new URL. "Slipper" pages can be created, which is displayed but disappear after a few seconds. We might use this feature for a "teaser" that shows users what is available at a site but that would not stay put unless the user enters passwords.
- Creation of auto surf documents that advance viewers to a new random page every few seconds.

Server push applications keep a connection open from the server to the browser and "push" a stream of data at the browser. Server push is often used for data-driven applications such as animating icons, though it is rapidly being replaced for such purposes by Java applets.

Client pull, on the other hand, is implemented using the <META> tag and a browser capable of performing it, such as Netscape 2.0. Compared to server push, client pull is relatively easy to set up. Though simple in its implementation and somewhat obscure in syntax - client pull is a powerful new Netscape addition to HTML

## 9.5 Background Graphics and Color

We can include background graphics and color to our Web page, which makes it attractive for any viewer. The <BODY> tag allows two extensions to define tileable background graphics and custom background colors for Web pages. Though these extensions originated in Netscape, most browsers now support these features.

<BODY BACKGROUND = "flower.gif" BGCOLOR = "#FFFFFF">

Here the BACKGROUND attribute fills the background of the browser window with the specified graphic image file. If the image is smaller than the page, it is tiled to fill the window. We also have an alternative to using hexadecimal color values.

*Example:*

<BODY BGCOLOR="BLUE"> gives a blue screen.

## 9.6 Microsoft Internet Explorer Extensions

### Background Sound

BGSOUND tag of the Microsoft Internet Explorer aids us to create pages with background sounds. When a page with this tag is loaded, the sound plays automatically, and a LOOP attribute controls the number of times it is to be played. Sounds can consist of .wav or .au sampled sound files or MIDI (.mid) music files.

<BGSOUND SRC = "xmascarol.wav">

The SRC attribute contains the URL of the source file to be played. The sound "xmascarol.wav" in the preceeding example plays once when the page is loaded. Here are two examples that play more than once using the LOOP attribute:

<BGSOUND SRC="xmascarol.wav" LOOP=5>

<BGSOUND SRC="xmascarol.wav" LOOP=INFINITE>

The first example plays five times; the second plays until we leave the page. LOOP can have any integer value or be INFINITE; -1 is the same as specifying INFINITE.

## 9.7 Font Tag Enhancements

Like Netscape, Explorer adds the BASEFONT tag, as well as the FONT tag with COLOR and SIZE attributes. But it adds one more attribute FONT FACE, which is used to define the font, or typeface for the tagged text. Here's the generic format:

<FONT FACE="Times New Roman", "Courier New", "Arial">Hello!</FONT>

In this example, the message Hello! Would be displayed in the Times New Roman font, if available. If not, Courier New would be used, or Arial in case Courier New does not exist.

## 9.8 Scrolling Marquees

We may get the impression that many of Explorer's extensions to HTML are meant to compete directly with the HotJava demos being shown by SunMicrosystems. They typically display small animations and scrolling marquees. Explorer has added a new MARQUEE HTML tag just for creating the latter. A variety of special attributes exist for controlling Explorer marquees.

### Syntax

<MARQUEE> Scrolling Scrolling Scrolling </MARQUEE>

Any text within the <MARQUEE>…</MARQUEE> tags scroll sideways. The BEHAVIOR attribute lets us pick how the text moves. With a value of SCROLL, text scrolls in from one side and off the other. SLIDE scrolls in from one side and stops as soon as the text touches the opposite margin. ALTERNATE bounces text back and forth within the marquee.

Beyond the basics, MARQUEE offers an incredible amount of control over the way our marquee looks and acts.

Attributes of MARQUEE:

- BEHAVIOR=SCROLL|SLIDE|ALTERNATE: Defines how the text moves.
- SCROLL wraps, SLIDE stops and ALTERNATE ping-pongs.

- DIRECTION=LEFT|RIGHT: Specifies the direction in which the text moves to and not the direction it comes from.
- ALIGN=TOP|MIDDLE|BOTTOM: Determines whether surrounding text aligns with the TOP, MIDDLE or BOTTOM of the marquee.
- BGCOLOR= "color": Defines background color as a hexadecimal value ("#FFFFFF") or color name ("blue")
- HEIGHT=n|n%: Specifies marquee height in pixels (HEIGHT=50) or as a percentage of screen height (HEIGHT=33%)
- WIDTH=n|n%: Determines marquee width in pixels or a percentage of screen height (See HEIGHT)
- HSPACE=n: Defines left and right outside margins in pixels (HSPACE=35)
- VSPACE=n: Specifies top and bottom outside margins in pixels (See HSPACE)
- LOOP=n|INFINITE: Determines how may times a marquee loops.
- SROLLAMOUNT=n: Defines how far the marquee moves each step, in pixels
- SCROLLDELAY=n: Specifies the delay between moves in milliseconds.

Let us now consider an example, which uses most of these attributes. The output is given in figure 9.5

```
<HTML>
<HEAD>
<TITLE> World of Marquess</TITLE>
</HEAD>
<BODY>
<FONT SIZE =10>
<A HREF = "marq.html">
<MARQUEE BEHAVIOR=SCROLL DIRECTION=LEFT LOOP=INFINITE BGCOLOR= "blue" HEIGHT=250 WIDTH=85% HSPACE=20 VSPACE=10 BORDER=20 SCROLLDELAY=25 SCROLLAMOUNT=3 ALIGN=MIDDLE> This is the Wonderful World of Marquees! </MARQUEE></A></FONT>
</BODY>
</HTML>
```

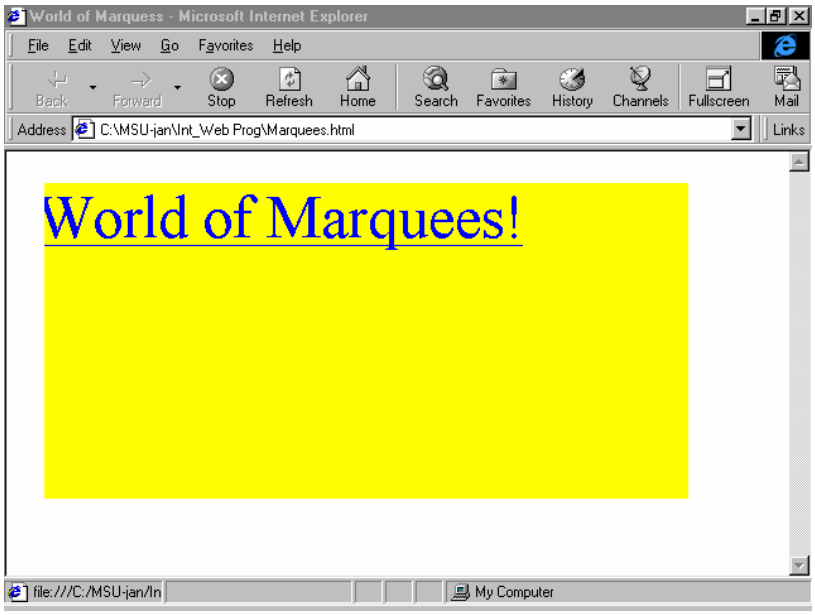



**Figure 9.5** *An Example for Scrolling Marquees.*

### Other form Attributes (for self learning)

Closely related to READONLY is the DISABLED attribute. When an element is disabled, its contents are not only unalterable, but unusable, so it's taken out of the tab order. When a form is submitted, the name/value pairs of disabled form elements are not included. If the disabled element is a button (whether custom, Reset, or Submit), the button can't be activated.

Although READONLY is limited to the **text** and **password** types within the INPUT element, DISABLED can be applied to any INPUT type, as well as to the TEXTAREA, SELECT, OPTION, LABEL and BUTTON elements.

### The BUTTON Element

The BUTTON element takes three possible values for the TYPE attribute: submit, reset and button. These values are all duplicates of their counterparts in the INPUT element.

### Syntax

<BUTTON
[ NAME = control name ]
[ VALUE = control value ]
[ TYPE = button | submit | reset ]
[ DISABLED ]
[ TABINDEX = tab number ] > Button Text </BUTTON>

### The SELECT and OPTION Elements

The SELECT element is used to create a list of choices either as a drop-down menu or a list box. Each of the choices in the list is an OPTION element.

### Syntax

```
<SELECT
        [ NAME = control name ]
        [ SIZE = control width ]
        [ MULTIPLE ]
        [ DISABLED ]
        [ TABINDEX = tab number ] >
        <OPTION
                [ SELECTED ]
                [ DISABLED ]
                [ VALUE = control value ] >

                ……..Text label or value
        </OPTION>
            ….. other option elements
</SELECT>
```

A selection list is created by adding one or more OPTION elements within the SELECT element, much like an HTML list:

```
<SELECT NAME = "Pets" >
        <OPTION> Dog </OPTION>
        <OPTION> Cat  </OPTION>
        <OPTION> Dove </OPTION>
</SELECT>
```

The end tags on the OPTION element are optional. If they're not used, the element automatically terminates at the beginning of the next OPTION element or at the end of the SELECT element that contains it.

The OPTION element can also have a specified VALUE assigned to it, but that's not required. If it's absent, the contents of the OPTION element will become the "VALUE" part of the name/value pair.

The result of the proceeding code is shown in Figure 9.6.


**Figure 9.6** *The SELECT element drop-listing its OPTIONS.*

Using SELECTED attribute with options causes the selected element to be highlighted. The user can accept it or deselect it at will, and more than one option can be selected.

By default, the SELECT element displays a drop-down menu in which only the first element is displayed (as in Figure 9.6). However, the SIZE attribute can be used to make more of the options visible. The following code illustrates this:

```
<FORM>
<SELECT NAME = "Pets" SIZE=5 >
        <OPTION> Playing Games </OPTION>
        <OPTION> Hearing Songs </OPTION>
        <OPTION> Reading Books </OPTION>
        <OPTION> Shopping </OPTION>
</SELECT>
</FORM>
```

When the SIZE attribute is specified the appearance of the menu will slightly be different. The menu is displayed as a simple list instead of as a drop down list (see Figure 9.7).

Users can select only a single choice from the menu, but using the MULTIPLE attribute will enable them to select a range of choices. Of course, they can still choose only one if that's their wish, but they can now choose any or all available options. If more than one option is chosen, multiple name/value pairs are sent when the form is submitted. Each has the same name, but a different value. The SELECT element that uses the MULTIPLE attribute is given below.

```
<FORM>
<SELECT NAME = "Pets" SIZE=5 MULTIPLE>
        <OPTION> Playing Games </OPTION>
        <OPTION> Hearing Songs </OPTION>
        <OPTION> Reading Books </OPTION>
        <OPTION> Shopping </OPTION>
</SELECT>
</FORM>
```

The result of this code is shown in Figure 9.7.


**Figure 9.7** *SELECT element enabling multiple selection.*

### The TEXTAREA Element

The TEXTAREA element is similar to the INPUUT element's text type. The difference is that users can type a larger section of text than they can with the text boxes. Instead of a single of line of text, there is a large window where multiple-lines responses can be typed.

### Syntax

```
<TEXTAREA
        [ NAME = control name ]
        [ ROWS = number of rows ]
        [ COLS = number of columns ]
        [ DISABLED ]
        [ READONLY ]
        [ TABINDEX = tab number ] >

...Text…
</TEXTAREA>
```

TEXTAREA is typically used for comments or "delivery" instructions – anything that requires more than a simple response. The dimensions of the window are specified with the ROWS and COLS attributes. Number of rows means number of lines and number of columns means number of characters per line. The following code illustrates the use of the TEXTAREA element; the results are shown in Figure 9.8.

```
<TEXTAREA NAME= "Comments" ROWS= "4" COLS = "15">
  This is the place to type your comments.
</TEXTAREA>
```


**Figure 9.8** *TEXTAREA in a Web page.*

TEXTAREA requires both start and end tags. As with the INPUT element's TEXT, TEXTAREA elements can use the READONLY attribute. If it's specified, then the user can't alter any initial content provided by the author.

### The LABEL Element

As the name implies, the LABEL element is the text that labels a control. Unlike normal text, however, the label and its associated control both share the same focus. In other words, if the label is clicked the effect will be same as though the control is clicked.

### Syntax

```
<LABEL
        [ FOR = control name ]
        [ DISABLED ] >
</LABEL>
```

Labels are associated with controls in one of two ways, either implicitly of explicitly. In implicit association method, the associated element is contained in the LABEL element as illustrated in Figure 9.9 and the following code:

```
<LABEL> Your nick name
            <INPUT TYPE= "Text" NAME= "NickName" SIZE=15>
</LABEL>
```

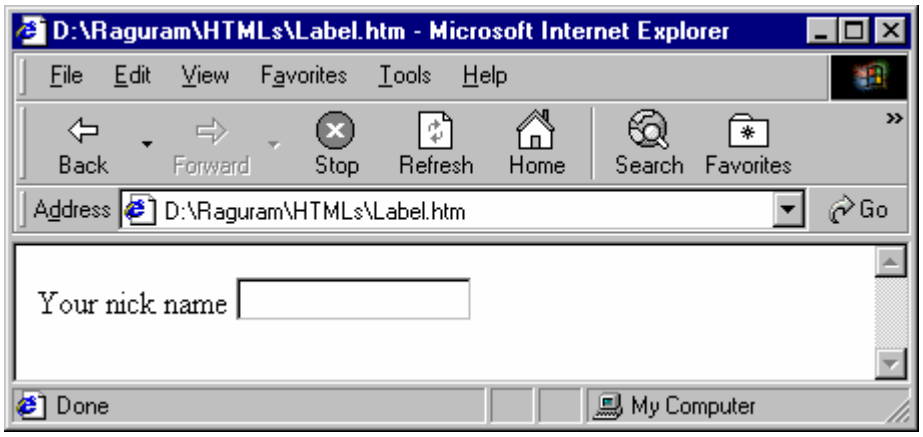



**Figure 9.9** *A Label associated with a text box.*

With an explicit association, the label is tied in by using the control element's ID attribute. The FOR attribute of the LABEL element is assigned the value of the control element's ID attribute, as shown in the following:

<LABEL **FOR= "NName"** > Your nick name </LABEL>

<INPUT TYPE= "Text" NAME= "NickName" SIZE=15 **ID= "NName"**>

The LABEL element must still come before the control for the display to look right; a logical association has nothing to do with visual placement.

### The FIELDSET and LEGEND Elements

The FIELDSET and LEGEND elements create border around grouped controls, just as panels do in standard programming, and legends are label that refer to overall field set.

### Syntax

<FIELDSET>
  <LEGEND>*Legend Text*</LEGEND>
  ....*Control elements*
</FIELDSET>

Both elements require start and end tags. The use of field sets and their legends is illustrated in Figure 9.10 and the following code:

```
<FIELDSET>
<LEGEND><B>Address Details</B></LEGEND>
<LABEL> Name :
<INPUT TYPE= "Text" NAME= "Name" SIZE = "30">
</LABEL> <BR>
<LABEL> Address:
<INPUT TYPE= "Text" NAME= "Address" SIZE = "15">
</LABEL>
```

```
<LABEL> Phone:
<INPUT TYPE= "Text" NAME= "Phone" SIZE = "10"> <BR>
</LABEL>
</FIELDSET>
```

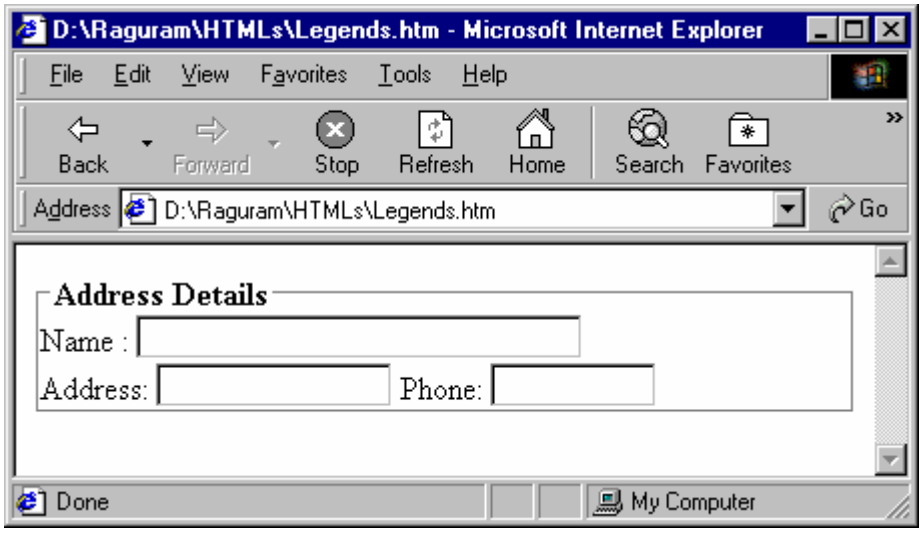



**Figure 9.10** *Result of FRAMESET element.*

There is no provision in the HTML specification for controlling the size of fields sets; they stretch across the entire screen, regardless of the space taken up by the controls they contain.

### Tab Navigation

The TABINDEX attribute is for navigating through a form by using the Tab key to move from one element to the next. This attribute is not required, if the form is to be navigated in the order that the elements appear, as is usually the case. If TABINDEX isn't specified, then that's the default behavior of the tab key.

However, if the navigation is required in some other order (such as vertically through adjacent control element), this attribute is useful. To set the tab order for a pair of text input boxes, the following code is used:

```
<INPUT TABINDEX= "1" TYPE= "Text" NAME = "FirstName">

<INPUT TABINDEX= "2" TYPE= "Text" NAME = "LastName">
```

If two elements have the same TABINDEX value, the navigation order is the order in which they appear.

It is not necessary for the assigned values to be sequential; all that's required is that, they be in ascending order corresponding to the navigation pattern. Therefore, the following code has the same effect as the previous example:

<INPUT TABINDEX= "10" TYPE= "Text" NAME = "FirstName">

<INPUT TABINDEX= "20" TYPE= "Text" NAME = "LastName">

In fact, it is good idea to give non-sequential number values to the TABINDEX attributes of successive elements. This technique gives room for the new elements. New elements can be inserted in the tab order, with out the need of reordering the tab index of other already existing elements.

The TABINDEX attribute is supported by the BUTTON, INPUT, SELECT, and TEXTAREA elements.

## 9.9 Short Summary

- Forms are constructed in the HTML documents by using the <FORM> element.
- Scripting languages or CGI program does the processing of data
- Server push and client pull are Netscape 2.0's innovative new techniques for automatically updating Web pages.
- The BACKGROUND attribute fills the background of the browser window with the specified graphic image file.
- Any text within the <MARQUEE>…</MARQUEE> tags scroll sideways.

## 9.10 Brain Storm

1. Discuss the result of a GET or POST value to the METHOD Attribute.
2. What is the significance of the NAME Attribute
3. Explain Server push and Client pull
4. How do you secure your information?
5. What are the different types in which a background graphic can be put in.
6. How can we restrict the background sound in our page?
7. MARQUEE offers an incredible amount of control over the way our marquee looks and acts - Discuss

**Lab Unit (2 *Real Time Hours*)**

**Using Tables design a web page of Yahoo mail options.**
**This is a web page designed by myself Mr./Ms. ……..on ……**

| Personalization | Mail Management | Delivery Services |
|---|---|---|
| Account Information | Block Addresses | Yahoo! Delivers |
| Change your password and edit user information. Update your profile. | Block addresses from which you don't want to receive mail. | Discounts and special offers direct to your Inbox. Personalize your interests! |
| Mail Preferences | Check Other (POP) Mail | Yahoo! Subscriptions |
| Customize your Inbox view. Change your outgoing name and address. Get a bigger mailbox! | Retrieve mail from all your other accounts into your Yahoo! Mailbox! | Have Yahoo! content delivered directly to your Inbox. |
| Signature | Filters | Reminders |
| Attach a custom signature to your outgoing messages. | Sort your incoming mail automatically into designated folders or to your mobile device. Filter out unsolicited email. | Set up Email, Messenger, or Pager reminders for any event! |
| Vacation Response | POP Access & Forwarding | |
| Send a custom, automatic message response when you are away. | Use Yahoo! as your permanent email address. Forward to another mail account, or download your Yahoo! messages to your POP3 mail client. | |

Click here to see my previous exercise