

Web Design

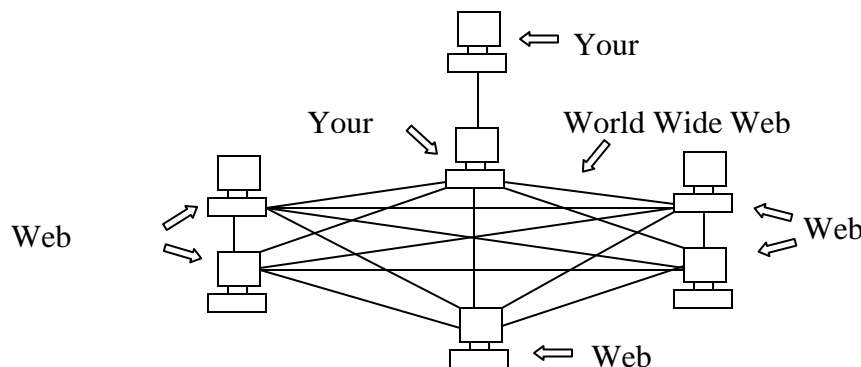
1

What is the World Wide Web?

1. Created by European physicists in the late 1980's to provide an easy way for scientists to communicate ideas.
2. It is part of the **Internet** which is a global computer network
3. Reasons why its popularity has increased:
 - Extremely quick communication medium
 - Documents are easily created
 - People in every part of the world can communicate with each other without the delays experienced with some telecommunications options

How It Works

1. Documents are created as **Web Pages**.
2. Web pages are assembled into a **Web Site**.
3. The web site is placed on a computer called a **Web Server**.
4. The site is given an address called a **URL** (Uniform Resource Locator).
5. To view the web site, you need a **Web Browser** which can:
 - Send a request to see a web site to its URL
 - Receive the data
 - Display the information on your computer screen
 - Examples of web browsers:
 1. Netscape Navigator
 2. Internet Explorer
6. Your computer is connected to the Internet through an **ISP** (Internet Service Provider) such as America On Line etc.
7. The browser sends a request to the web server containing the desired web site via its URL.
8. The web server recognizes your computer and sends the web pages to it in small pieces called **Packets**.
9. The packets travel the network cables connecting all of the web servers until they arrive at your computer.
10. Your browser collects these packets and assembles them into a viewable document.
11. If some packets are late in arriving, your browser sends another request to the web site's URL and they are sent again.

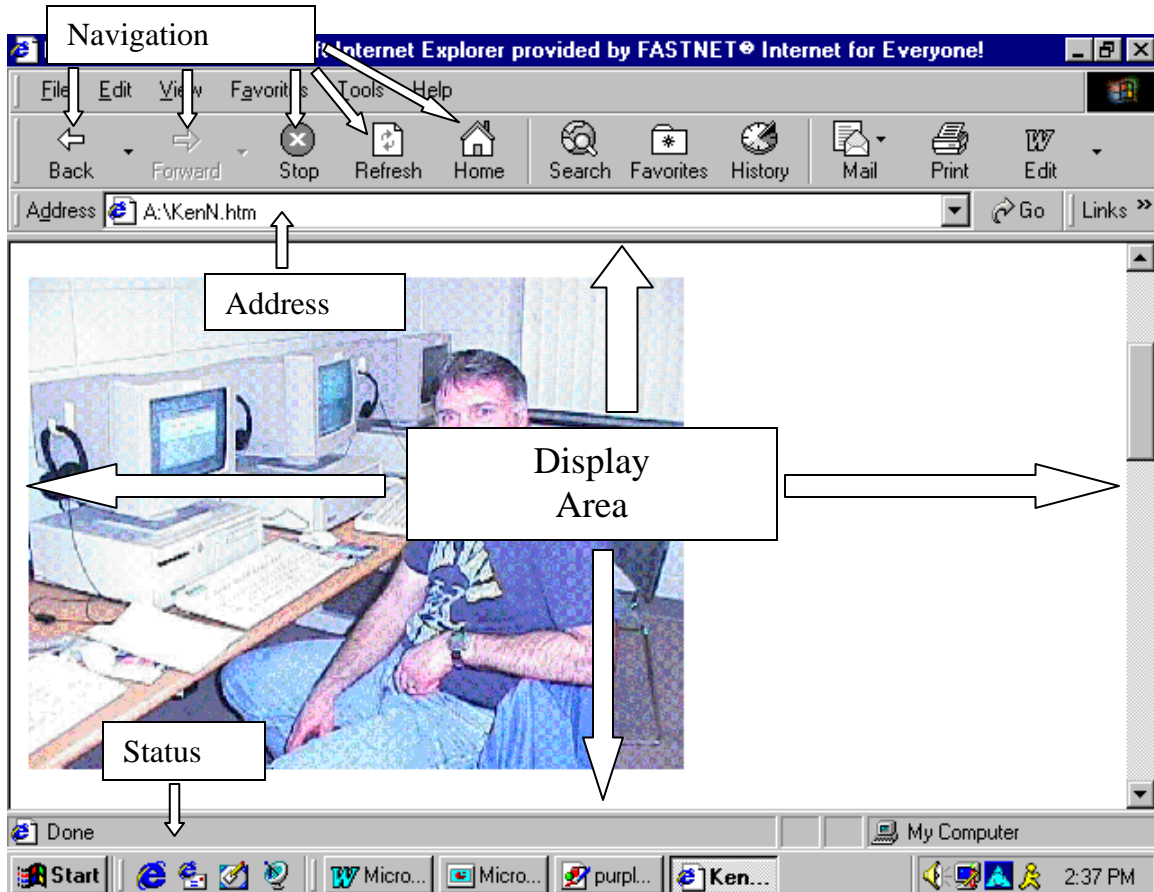


Web Design

2

The Web Browser

1. Allows you to skim through web sites and their pages using **Hyperlinks**.
 - Hyperlinks are pointers to specific pages of specific sites.
2. Provides the ability to move forward or backward through the pages you have already visited.



3. Type the URL of the desired web site into the **Address Box** and press **Enter** to open a particular web page.
4. Click the **Back Button** to move to the previously visited page (if applicable).
5. Click the **Forward Button** to move to the web page visited after the currently opened page (if applicable).
6. Click the **Stop Button** to cease loading the web page currently being assembled by the browser.
7. Click the **Refresh Button** to return the display area to an updated version of the last opened web page.
8. Click the **Home Button** to open the default web page (that page that opens first when the browser is activated).
9. To load a different web page:
 - Highlight the current URL in the address box and type the new URL and ENTER.
 - Left click the **down arrow** at the right of the address box to view a list of previously visited sites. Left click on the one desired.

Web Design

3

Searching the Web

There are two types of tools available to search for information on the web.

Search Engine

Search engines gather information about web pages automatically and use this data base to assist the user in finding sites that match criteria entered in the search request.

AltaVista is an example of a search engine. It can be accessed by entering **www.altavista.com** in the address box and pressing ENTER.

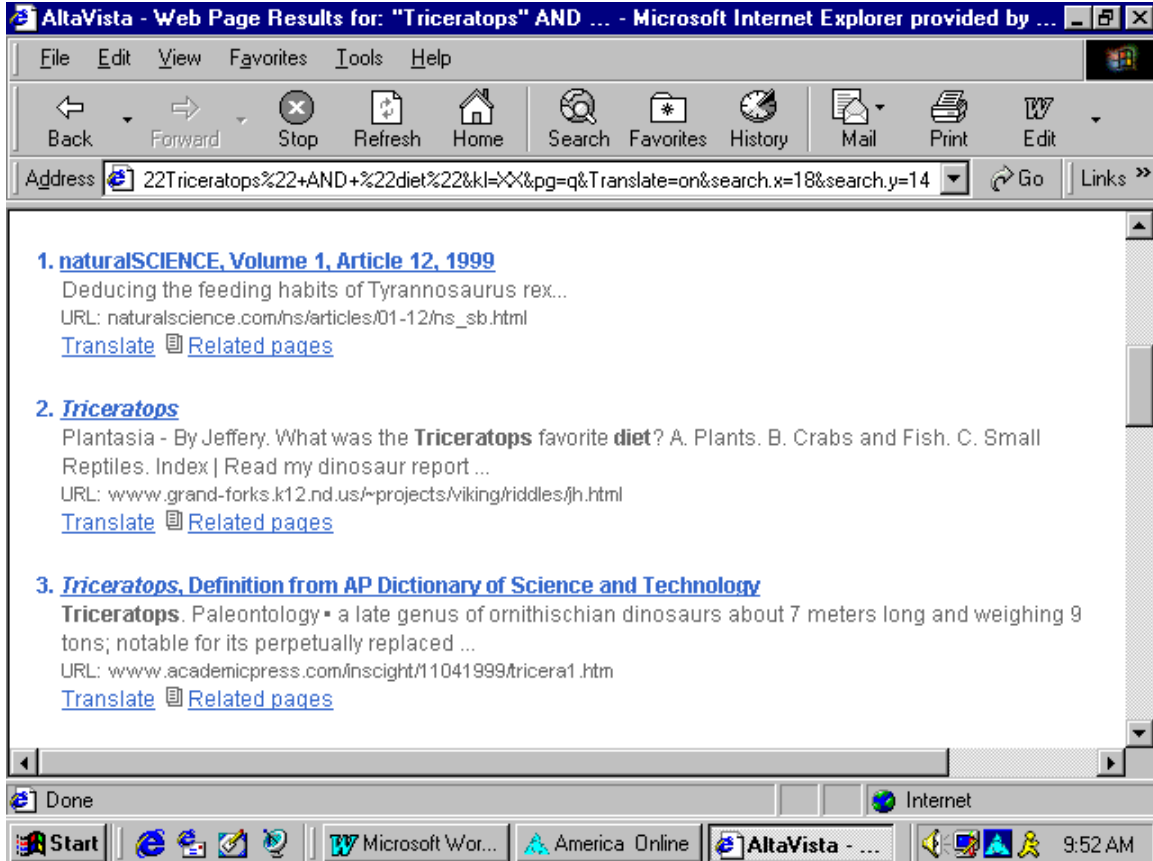


To find information about a subject:

- Type the **keywords** about your subject into the search box. For instance, to find information about the dinosaur Triceratops, type "Triceratops" in this box.
- Use **boolean logic** to limit your search. For instance, to find out what Triceratops ate, type (including the quotation marks) "**Triceratops**" AND "**diet**" in the box. The capitalized AND means that the words TRICERATOPS and DIET must appear in the web page in order to be listed as a possible source.
- After entering your search criteria, click the **Search** button.
(See a picture of the resulting screen on the next page.)

Web Design

4

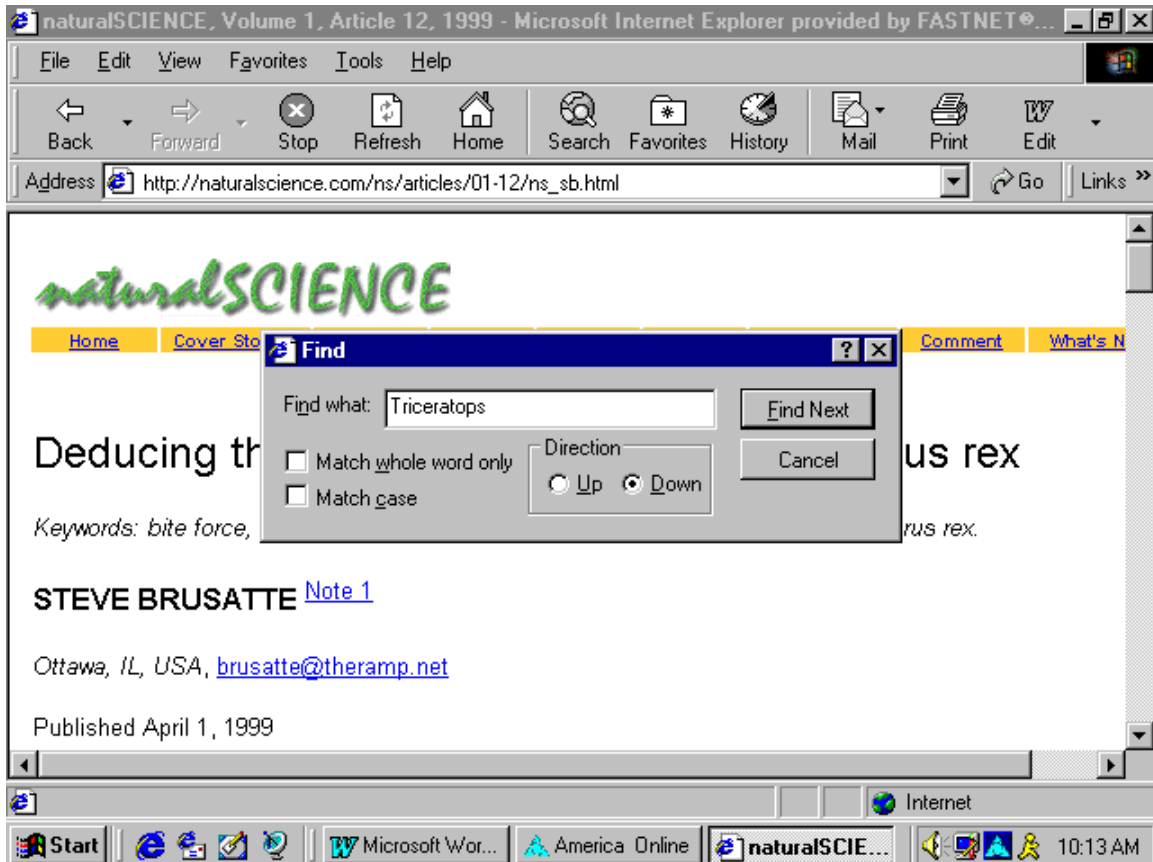


- Web pages are listed that coincide with your criteria. Any colored text (usually in blue) is a **hyperlink** to that web page.
- To read the information, simply left click on the hyperlink and the information will be loaded from the web by your browser.
- To locate information about your subject, you can use the **Edit - Find** options from the main menu.

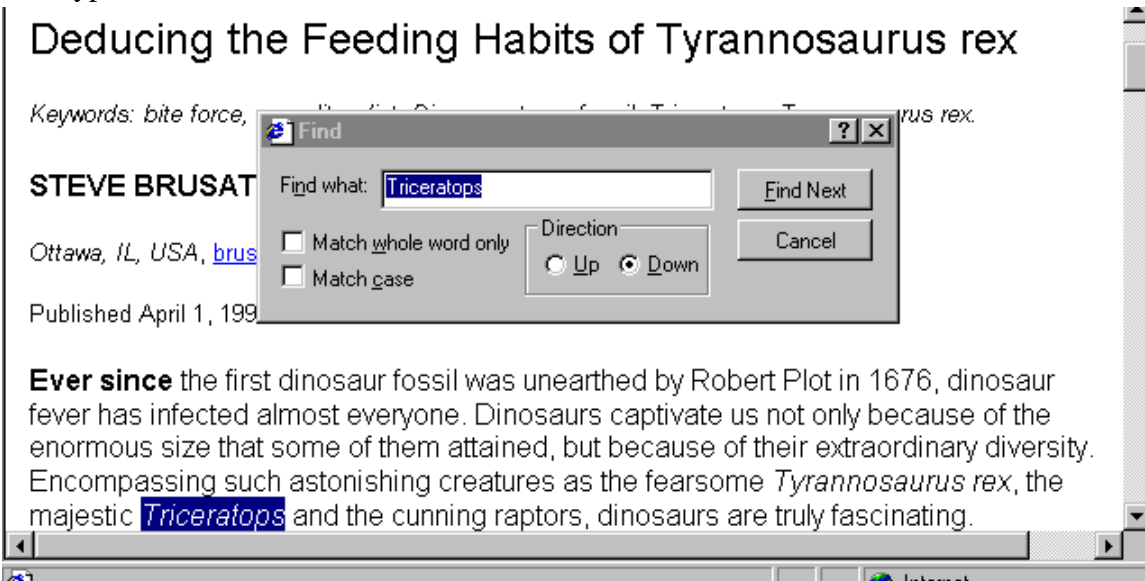


Web Design

5



- Type a search word in the **Find What** box and left click on **Find Next**.



- By continually clicking on **Find Next** you will see every instance of the word in the page. (**Note:** By noting the title of this document and reading some of the text, it is clear that "Triceratops AND diet" are used to indicate that Triceratops was on the diet list of Tyrannosaurus Rex. We will have to check another web page in our list.)

Web Design

6

Web Index

Web indexes are used the same way as search engines except for the fact that Web indexes list only those web pages that they have been asked to include. When a person creates a web site, he/she must ask the Web index to list the site. **Yahoo** is a popular web index and can be accessed at **www.yahoo.com**.

The Speed of the Web

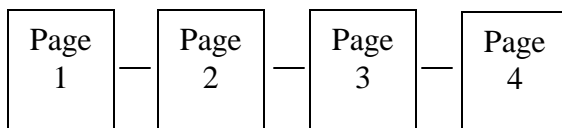
1. Web packets travel at the speed of electricity and travel, therefore, very quickly.
2. **The readers of your web site will want your pages to load quickly.** So if the packets travel as quickly as electricity, why do some pages take so long to download?
 - The size of the pages.
 1. Do they contain a great deal of graphics?
 2. Are they extremely large with a great deal of text?
 - The speed of the user's Internet connection.
 1. How fast is the user's modem?
 2. Does his/her phone line have some restrictions that limit the speed?
 - The speed of the user's web server at his/her ISP.
 - The amount of traffic on the Internet at the time.
3. You can only control the first factor. **The design of your web pages.**
4. You will need to **strike a balance** between the use of **eye catching graphics** to entice more people to visit your site and **limiting the size** of the site in order to ensure quick downloading of the site.
5. Many people will tire of slow downloading sites and choose instead to click the **Stop Button** and visit another site that downloads quicker.
6. You need to make your site **interesting and informative** without overdoing graphics and excessive text.

Contents of Your Web Site

Organizing the Data

Sequential Organization

In a sequential organization, each page is placed in the site in the order in which they should be viewed.



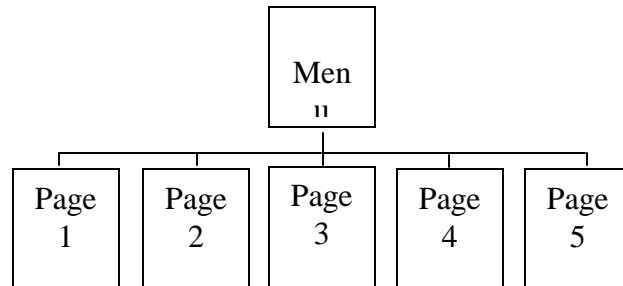
Each page should contain a **navigation button** to allow the viewer to move to the next page or the previous page.

Web Design

7

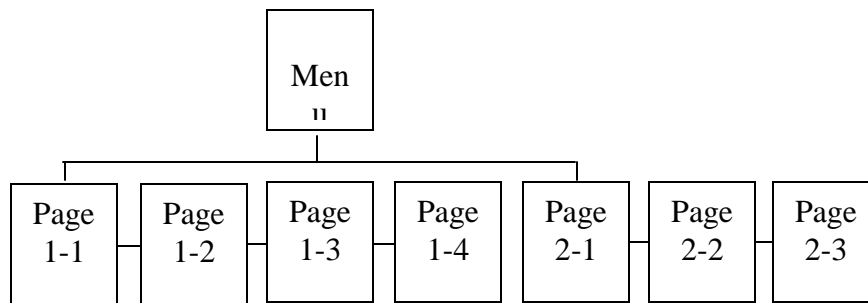
Hierarchical Organization

The first page of the site is a **menu** containing a link that points to each page. The user is allowed to choose any of the pages based upon the information contained on each. In this type of organization, the order in which the information is presented is not important.



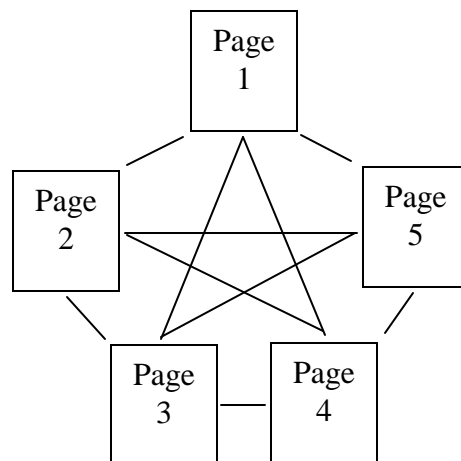
Combination Sequential and Hierarchical Organization

In the combination organization, part of the site (the first page covering a particular subject, for instance) can be accessed via a menu and each subsequent page within that subject can be accessed sequentially.



Web Organization

In the web organization, any page can be accessed from any other page in the site.



What Should Each Page Include?

1. Title
 - Identify the **web site**
 - Identify the **page** within the site
 - Important because some users may not enter your site from the home page
2. Navigation Links
 - **At least** a link to the **home page**
 - **Next** and **Previous** navigation buttons if the site has a **sequential organization**
3. Author and Copyright Information
 - Needed on every page because users may not always enter your site via the home page

What Should Every Site Include?

1. Home Page
 - Entry point to the site
 - Make it **attractive** and **informative**
 - Include an attractive **title**
 - Include a **site menu** to give the user a quick idea of the material contained in your site.
 - The site menu may be in the form of an **image map** where clicking on a particular portion of an image will open an appropriate page dealing with the graphic representation just clicked upon.
 - Optional items that may be included
 - Any **new content** recently added to the site
 - The **date** the site was **last updated**
 - Copyright notice
 - A **hit counter** to let the user know how many visitors there have been
2. Optional Cover Page
 - **Displays briefly** before the home page appears.
 - Usually contains a **flashy graphic**
 - Usually appears for about **ten seconds**
 - **Don't let it take more than a few seconds to load.**
3. Optional Site Map
 - Used if your site contains a large number of pages
 - Is a menu providing links to areas of your site
4. Contact Information
 - Usually linked to your email address
 - Gives users an opportunity to relay opinions or suggestions about your site to you
5. Optional Help Page
 - Used if the site contains more than a few pages to explain how to navigate the site
6. FAQ (Frequently Asked Questions)
 - Affords an easy way to answer users' questions about the information in the site
7. Related Links
 - Links to other web sites that offer additional information about the subject(s) covered in your site

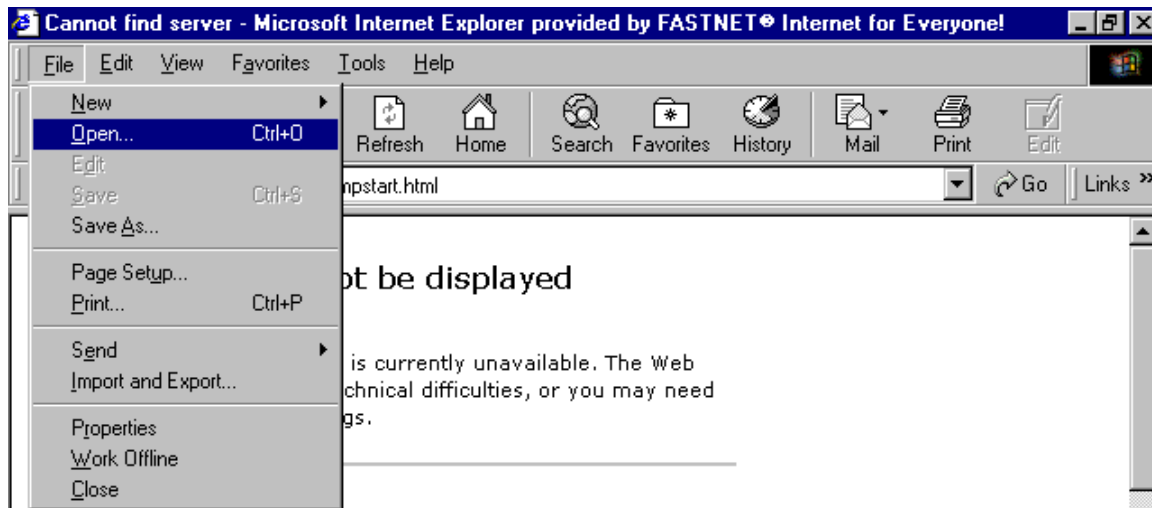
Creating a Web Page

HTML Basics

1. Stands for **H**yper**T**ext **M**arkup **L**anguage
2. Web pages are text files that are formulated using HTML code.
3. These text files can be created using any standard text editor ranging from Notepad through MsWord or Corel WordPerfect.

Creating an HTML Document In Notepad

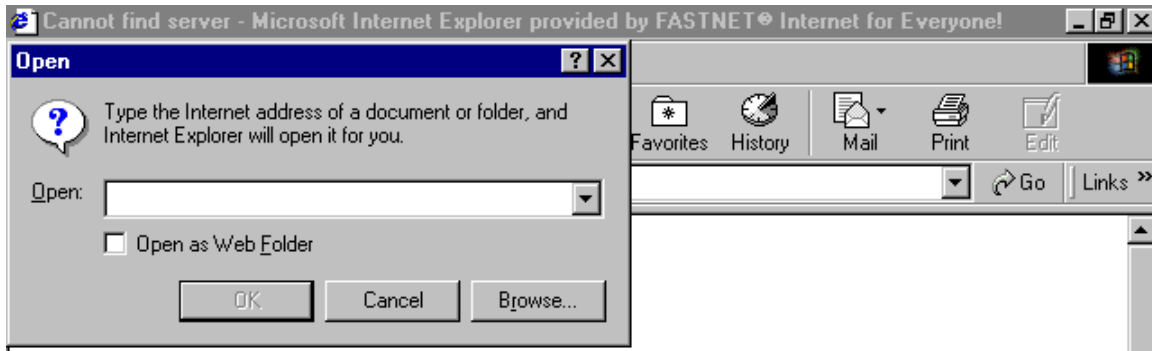
1. Open Notepad
 - Start - Programs - Accessories - Notepad
2. Create the HTML code within the Notepad document.
3. When finished, save the document as an HTML file.
 - File - Save As
 - Choose the drive and/or folder in which to save the file by choosing it in the **Save In** window.
 - In the **Save As Type** window, choose the **All Files (*.*)** option.
 - In the **File Name** window, type the name of the file followed by a **.htm** extension.
 - For example, an html file named "MyPage" would be named **MyPage.htm**.
4. To open the file in your browser:
 - Open your browser
 - Choose the **File - Open** option from the main menu.



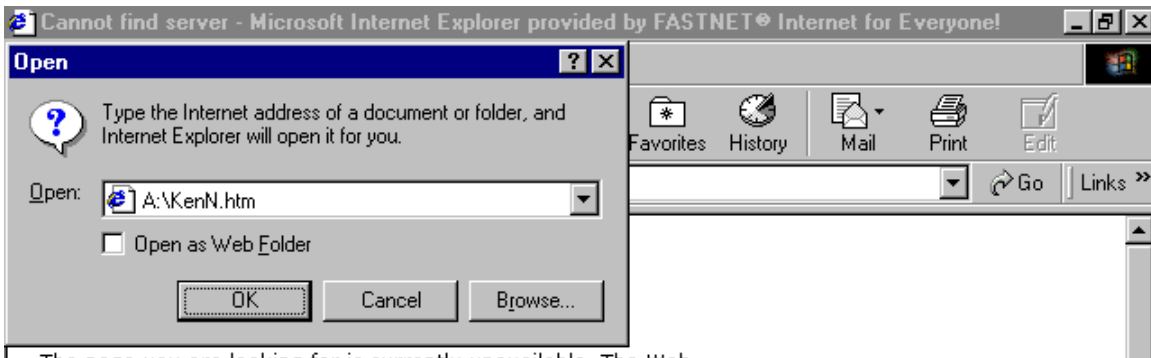
Web Design

10

- Use the **Browse** option which allows you to choose the drive and folder from which to load the page.



- After choosing the desired file, click on the **Open** option.
- Its name will then appear in the **Open** window.



- Click on **OK** and the document will be displayed in the browser display area.

Basics of HTML

1. The text entered in an HTML document is interpreted by the browser through the use of **HTML tags**.
 - All tags begin with a "less than" symbol <
 - Followed by the text and characters that tell the browser what to do
 - Closed with a "greater than" symbol >
 - Example: The title tag would appear as <Title> or <Title> or <title> (It is not case sensitive)
 - Most HTML tags appear in pairs where the first (start) tag indicates the beginning of the tag and the second (end) tag indicates the end of the tag.
 - The end of the tag is indicated with a **front slash** /
 - Example: The title tags would appear as <Title>My Web Page</Title>
2. There are three basic parts to an HTML document:
 - The **HTML** portion is the outer shell that encompasses the entire document.
 - The **Head** portion is at the beginning and contains title information etc. This text will be displayed at the top of the browser's window, not in the display area.
 - The **Body** portion is the part that contains the text and images for the page.


Web Design

11

Note: If a tag is started within any one of these three portions, it MUST END INSIDE THAT SAME PORTION.


Right

```
<HTML>  
<HEAD>  
<TITLE>  
</TITLE>  
</HEAD>  
<BODY>  
</BODY>  
</HTML>
```



Wrong

```
<HTML>  
<HEAD>  
<TITLE>  
<BODY>  
</TITLE>  
</HEAD>  
</BODY>  
</HTML>
```



- The following is an example the HTML code for an extremely basic web page that includes the three basic portions of the code:

```
<HTML>
```

```
    <HEAD>
```

```
        <TITLE>This is the title</TITLE>
```

```
    </HEAD>
```

```
    <BODY>
```

This is the portion that is shown in the display area of the browser.

```
    </BODY>
```

```
</HTML>
```

Exercise 1 Creating Your First Web Page

Create a web page using **Notepad** that has the following attributes:

1. A title of **My First Web Page**
2. A body that says **This is my first web page.**
3. Use the above example as a pattern for your page.
4. Save your Notepad document in the location designated by your instructor as **Index.htm**
5. Open your browser and view the web page.
6. Stop here and await further instructions

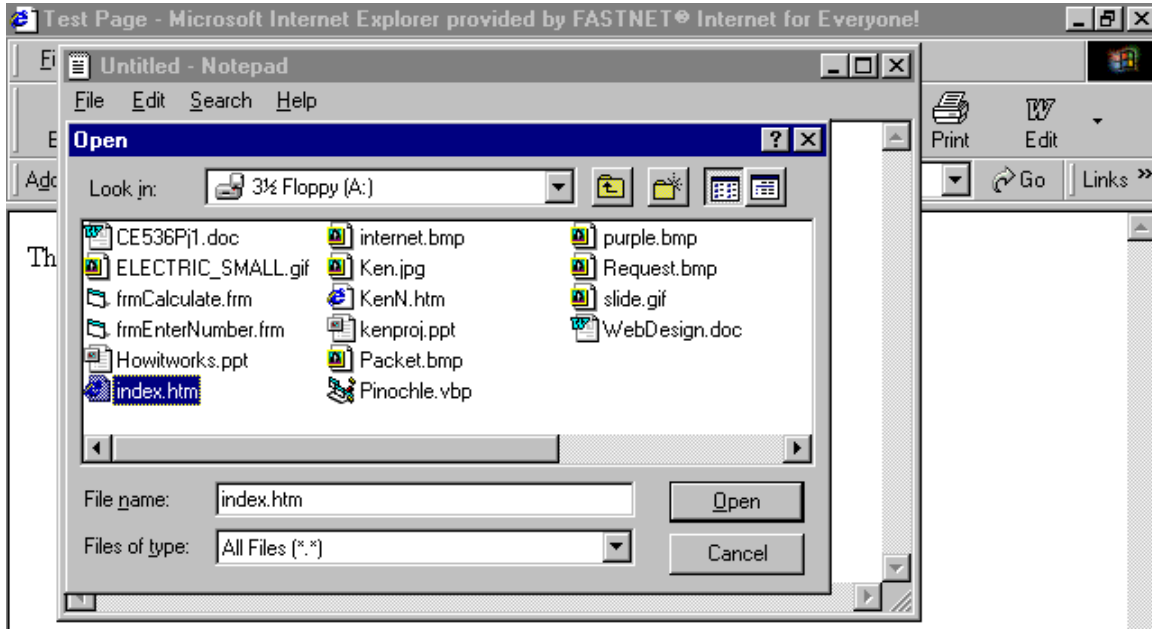
Note: We are saving the document as **Index.htm** because this is the first page a web server looks for when asked to find a file. By naming the main page Index.htm, you can be sure that, even if the file name is missing from the request, your main page will load.

Web Design

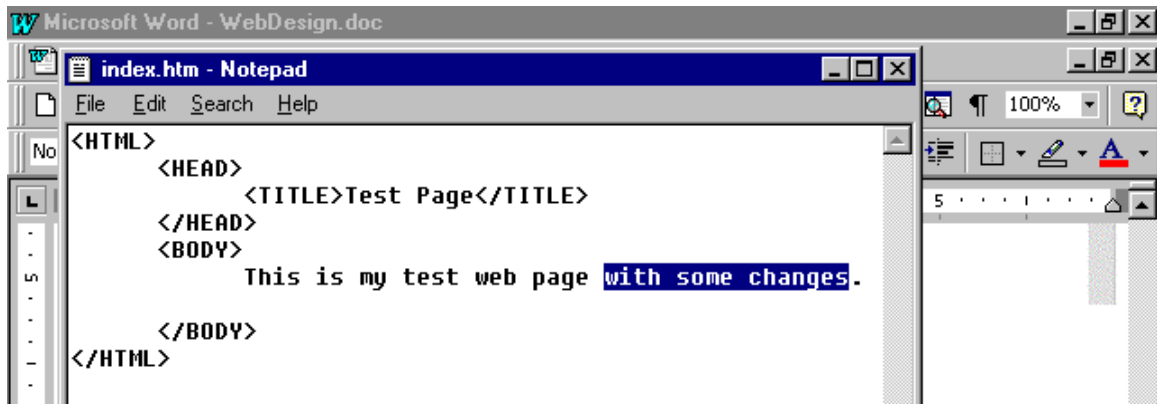
12

Editing an HTML Document

1. Open your web browser.
2. Open the desired web page
3. Open Notepad.
4. Open the source file for the web page on your browser
 - Left click **File - Open**
 - Choose **All Files** in the Files of Type box
 - Click the file containing the source code for the web page.
 - Click **Open**



5. Make the changes desired (highlighted below).

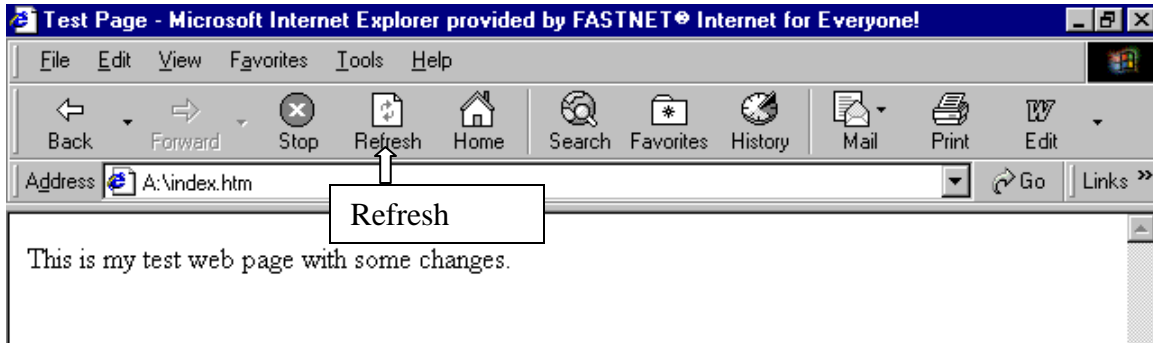


6. Save the changes.
 - Click on **File - Save**

Web Design

13

7. Click on **Refresh** (if using Internet Explorer) or **Reload** (if using Netscape) and the changes will appear in the web page.



HTML Code

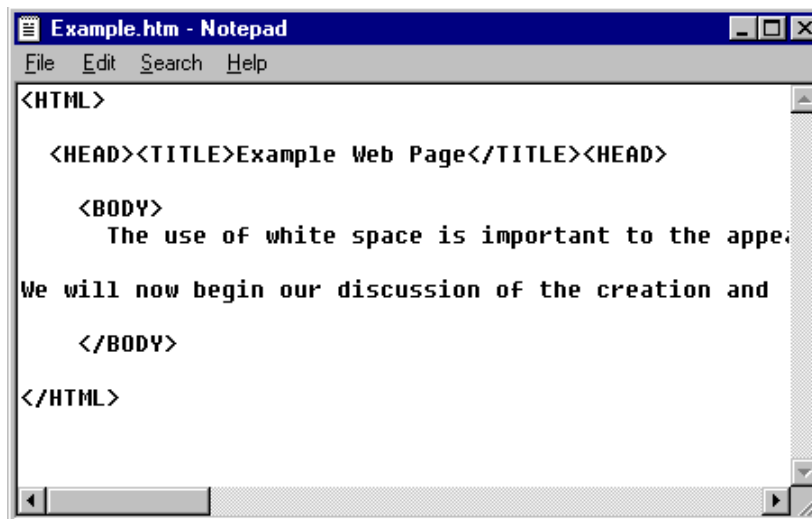
Adding White Space

When writing HTML code, it is important to remember that the browser will not add any spaces or skip lines unless told to do so. For instance, if the following code is written and copied to Notepad:

The use of white space is important to the appearance and visual presentation of any web page. Without carefully planning and adding white space, your web page will appear extremely plain and may be hard to read.

We will now begin our discussion of the creation and use of white space.

After it is pasted into Notepad, it will appear as follows:



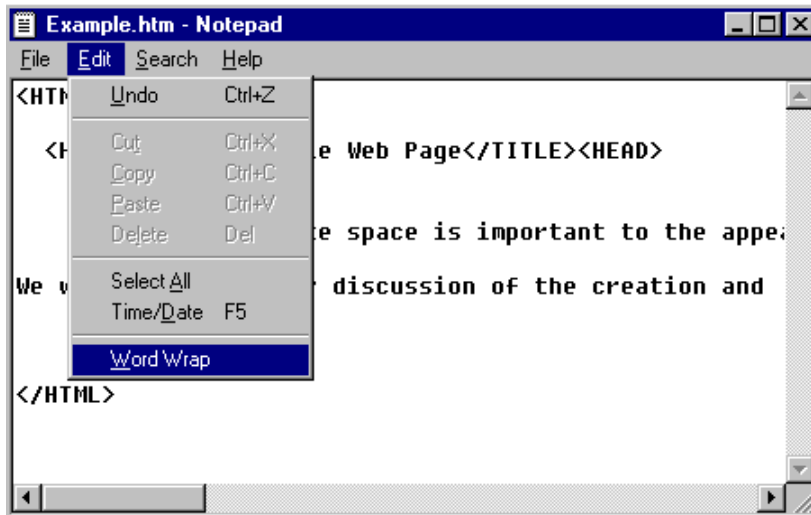
Notice how the text runs off to the right side of the window.

Web Design

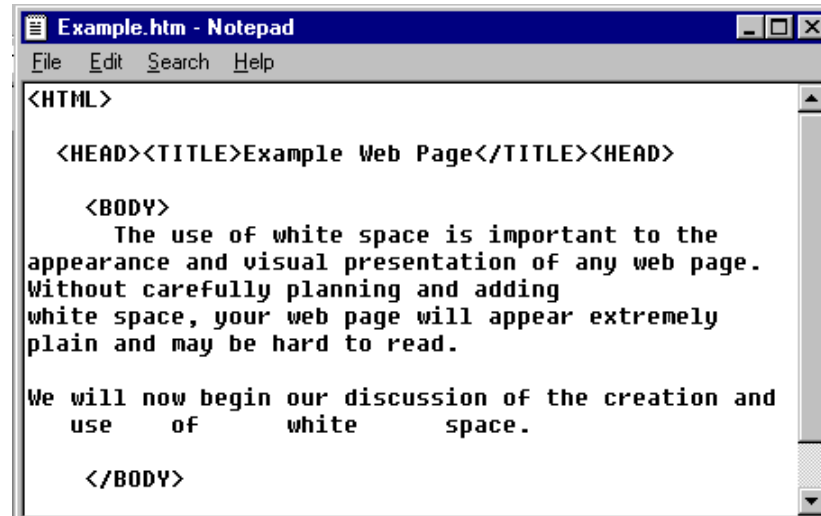
14

The text can be changed to that it will appear more like the written document by using **Word Wrap**.

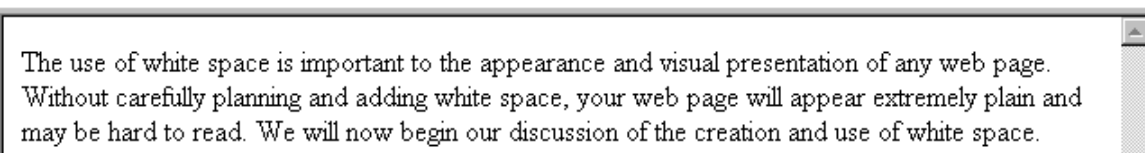
1. Click on **Edit - Word Wrap**



2. The text will then appear as below:



Notice the spaces inserted after the words "adding", "and", "use", "of", "white", and "space". Also, notice that we have inserted a blank line to signify a new paragraph immediately after the line "plain and may be hard to read." This is how it will appear in the browser:



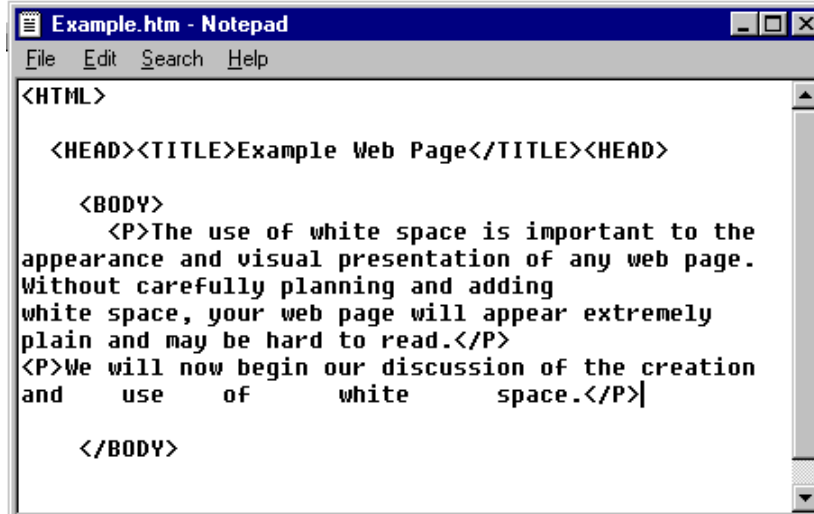
Notice that the only spaces included are those after periods. Otherwise, the only spaces inserted by the browser are the single spaces between words.

Web Design

15

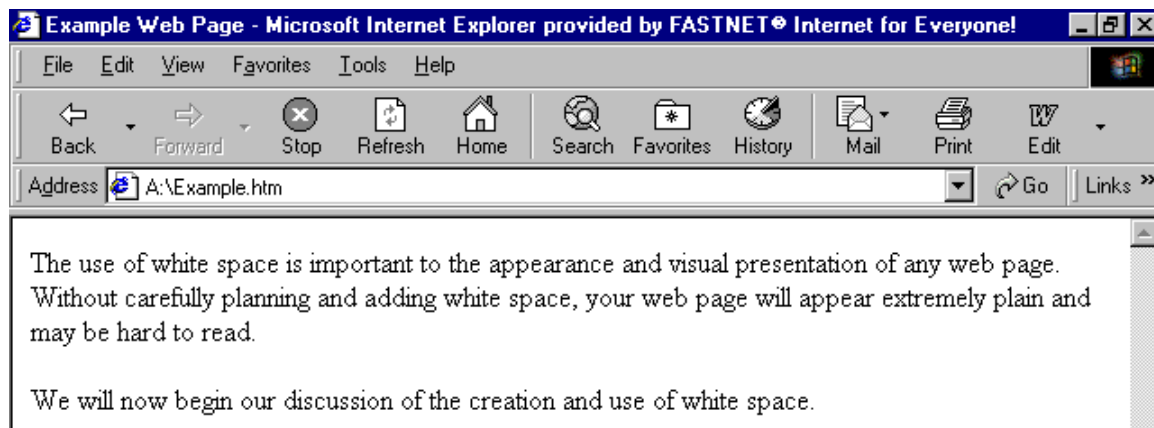
Paragraph Tags

1. `<P> ... </P>`
2. Will end the previous paragraph's line of text, insert a blank line, and start the next line of text on a new line.
3. Example:



```
Example.htm - Notepad
File Edit Search Help
<HTML>
  <HEAD><TITLE>Example Web Page</TITLE></HEAD>
  <BODY>
    <P>The use of white space is important to the
    appearance and visual presentation of any web page.
    Without carefully planning and adding
    white space, your web page will appear extremely
    plain and may be hard to read.</P>
    <P>We will now begin our discussion of the creation
    and use of white space.</P>
  </BODY>
```

Will result in the following web page:



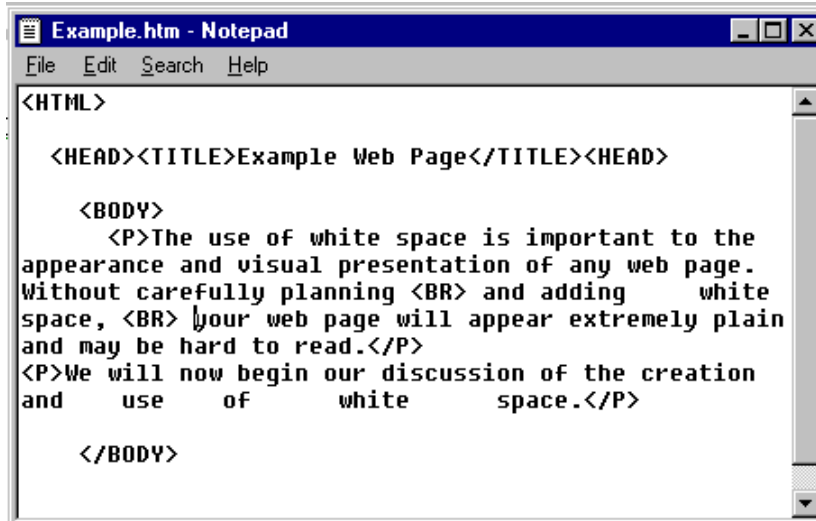
4. **Note:** The `</P>` tag is not always required. Just adding a `<P>` tag will do the same thing. However, some paragraphs may have special formatting and without adding the `</P>` tag, this formatting may carry over to the next paragraph. Therefore, it is a good practice to use the `</P>` tag in order to eliminate this problem in the future.

Web Design

16

Line Breaks

1. **
**
2. Will stop the text to the left and above the tag and start a new line of text **without adding a blank line.**
3. Example:



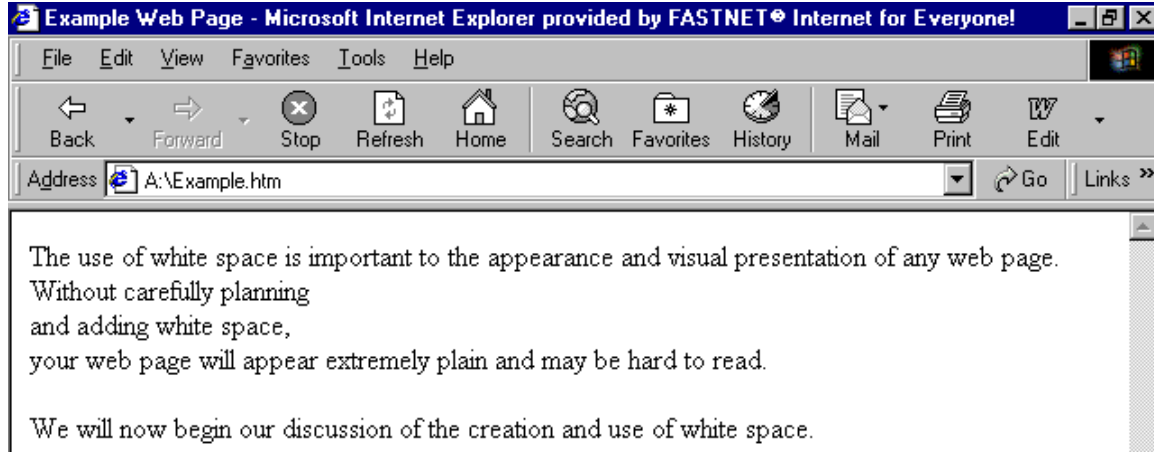
```
<HTML>

<HEAD><TITLE>Example Web Page</TITLE><HEAD>

<BODY>
  <P>The use of white space is important to the
appearance and visual presentation of any web page.
Without carefully planning <BR> and adding white
space, <BR> your web page will appear extremely plain
and may be hard to read.</P>
<P>We will now begin our discussion of the creation
and use of white space.</P>

</BODY>
```

Will appear like this:



4. **Note:** The **
** tag **does not need a </BR> end tag.**

Character Space

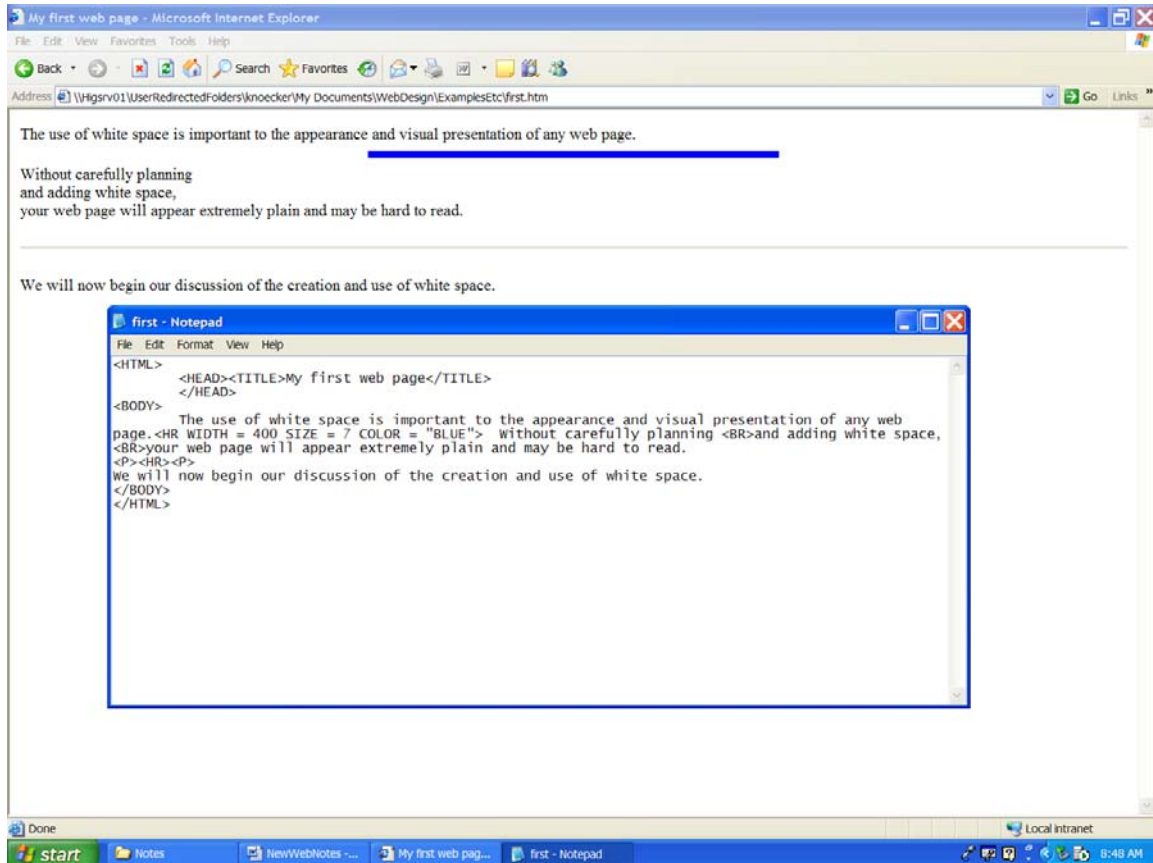
1. Used to skip a character
2. ** **
3. Example: To indent 3 characters - ** **; ** **; ** **;

Web Design

17

Horizontal Rules

1. **<HR>**
2. **Ends the current line** with the text to the left of the tag, **inserts a blank line, inserts a ruled line.** Then it **inserts another blank line**, and **starts a new line of text** below the ruled line.
3. Example:



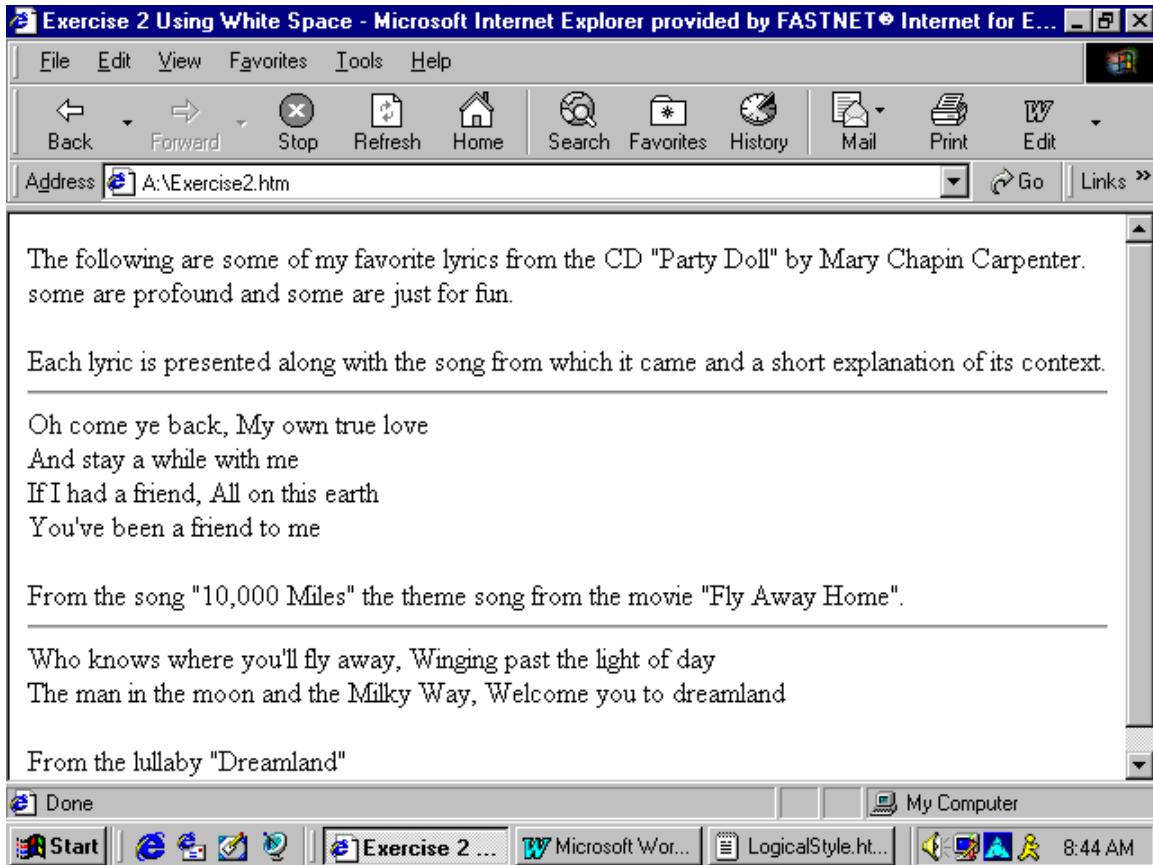
- Adding the **WIDTH** parameter sets the number of pixels horizontally that the horizontal rule will have
 - Adding the **SIZE** parameter sets the thickness in pixels
 - Adding the **COLOR** parameter sets the color of the line
4. **Note:** There is **no** `</HR>` needed.

Web Design

18

Exercise 2 Using White Space

1. Create a web page that appears as the example below.
2. Be sure to use the following tags:
 - `<P> ... </P>`
 - `
`
 - `<HR>`



3. Save this file as "Exercise2.htm"
4. Your instructor will give you the location in which to store it.

Physical Styles

Physical style tags allow you to change the physical appearance of any given section of text on your page. The reader will see these changes in the page after it is opened by the browser.

Bold Text

1. ** ... **
2. The text between the tags will appear as bold.

Italics Text

1. *<I> ... </I>*
2. The text between the tags will appear in italics.

Underlined Text

1. <U> ... </U>
2. The text between the tags will appear underlined.

Subscript Text

1. _{_{...}}
2. The text between the tags will appear as subscript
3. Example : The "2" in H₂O

Superscript Text

1. ^{^{...}}
2. The text between the tags will appear as super script.
3. Example: The "2" in πr^2

Typewriter Font

1. `<TT> ... </TT>`
2. The text between the tags will appear as fixed pitch typewriter font.

Big Font

1. **<BIG> ... </BIG>**
2. The text between the tags will appear in bigger font than the current font size.

Web Design

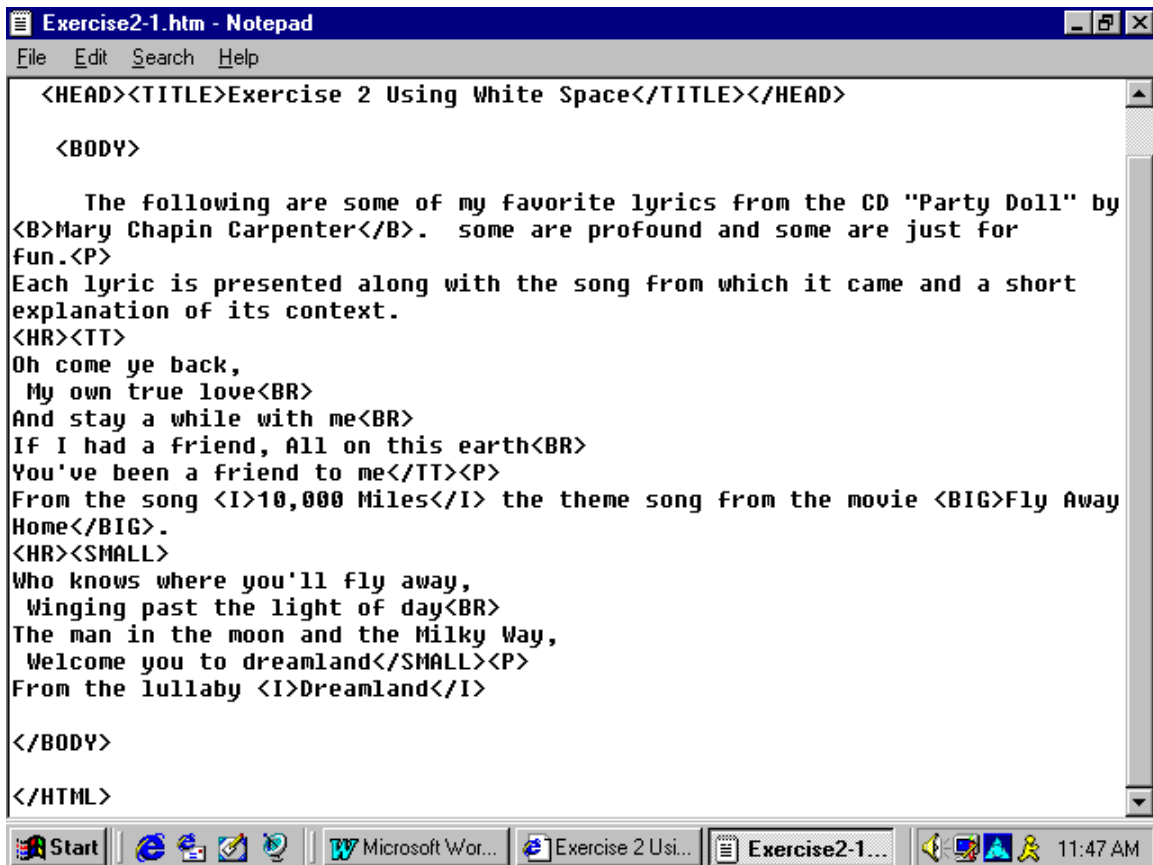
20

Small Font

1. `<SMALL> ... </SMALL>`
2. The text between the tags will appear in smaller font than the current font size.

To illustrate the physical styles, let's revisit our "Exercise2.htm" file that we have already created. We will make the following changes:

- All song titles will be in italics
- The name "Mary Chapin Carpenter" will appear in bold text
- The lyrics for "Dreamland" will appear smaller than the current font size
- The lyrics for "10,000 Miles" will appear in fixed typewriter font
- The name of the movie "Fly Away Home" will appear bigger than the current font size
- The code appears below:

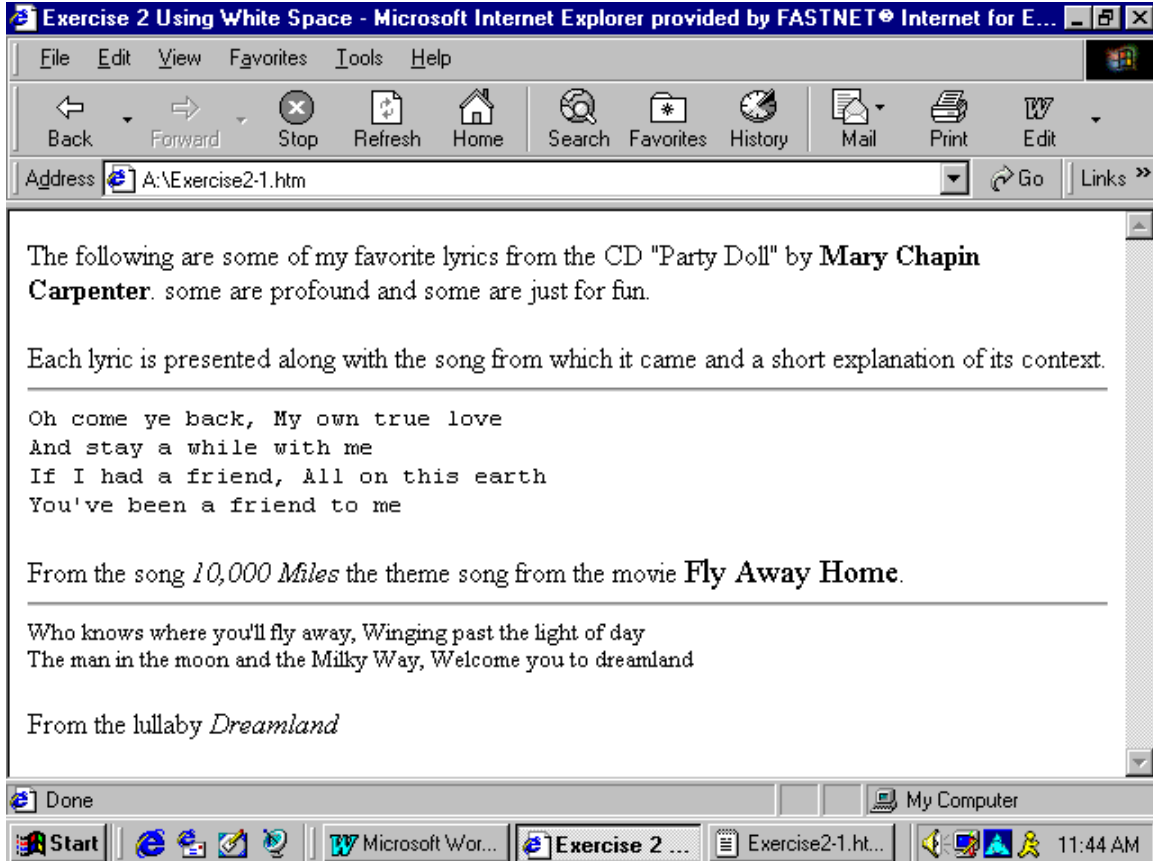


```
Exercise2-1.htm - Notepad
File Edit Search Help
<HEAD><TITLE>Exercise 2 Using White Space</TITLE></HEAD>
<BODY>
    The following are some of my favorite lyrics from the CD "Party Doll" by
<B>Mary Chapin Carpenter</B>. some are profound and some are just for
fun.<P>
Each lyric is presented along with the song from which it came and a short
explanation of its context.
<HR><TT>
Oh come ye back,
My own true love<BR>
And stay a while with me<BR>
If I had a friend, All on this earth<BR>
You've been a friend to me</TT><P>
From the song <I>10,000 Miles</I> the theme song from the movie <BIG>Fly Away
Home</BIG>.
<HR><SMALL>
Who knows where you'll fly away,
Winging past the light of day<BR>
The man in the moon and the Milky Way,
Welcome you to dreamland</SMALL><P>
From the lullaby <I>Dreamland</I>
</BODY>
</HTML>
```

Web Design

21

3. The web page will appear as below:



4. **Note:** The Bold and Big text fonts appear identical. In some cases they may be, depending upon the font size settings in these sections of the code.

Exercise 3-1 Using Physical Styles

1. Make the changes in the "Exercise2.htm" file to create the changes as displayed above.
2. After your changes are made, save the file and use the **Restore** button on the browser.
3. **Note:** Physical style tags can be used in combination. To do this, simply insert their start tags consecutively followed by the text, followed by the end tags in reverse order. For example to display the name **Mary Chapin Carpenter** in bold font with underlines, use the following code: `<U>Mary Chapin Carpenter<U>`
4. If you finish this exercise before others in the class, experiment with those physical style tags not used in the exercise:
 - `<SUB>`
 - `<SUP>`
5. Remember that in order to see the changes, you must save the "Exercise2.htm" file and then use the **Restore** button on the browser.
6. **Note:** Use restraint with physical styles as they can quickly make your page more difficult to read.

Logical Styles

While physical styles change the way parts of your text **look**, logical styles change the way parts of your text are **used**.

Emphasized

1. ` ... `
2. Used to place emphasis on the text within the tags.
3. It is usually in *Italics*

Block Quotes

1. `<BLOCKQUOTE> ... </BLOCKQUOTE>`
2. Used to display large quotations such as verses of poems or excerpts from books.
3. Includes **left and right indents**.

Cited Works

1. `<CITE> ... </CITE>`
2. Used for titles of a book or any other citation to give credit to another source.
3. Usually appears in *Italics*

Text Keyed In by the User

1. `<CODE> ... </CODE>` or `<KBD> ... </KBD>`
2. Used when the viewer of the page needs to input some information.
3. Usually appears in typewriter font.

To illustrate these logical styles, let's create a web page that does the following:

- Cite a published work
- Display an excerpt from the work
- Tell the user to click on the "NEXT" button
- Emphasize the work's importance

The code appears on the next page.

Web Design

23

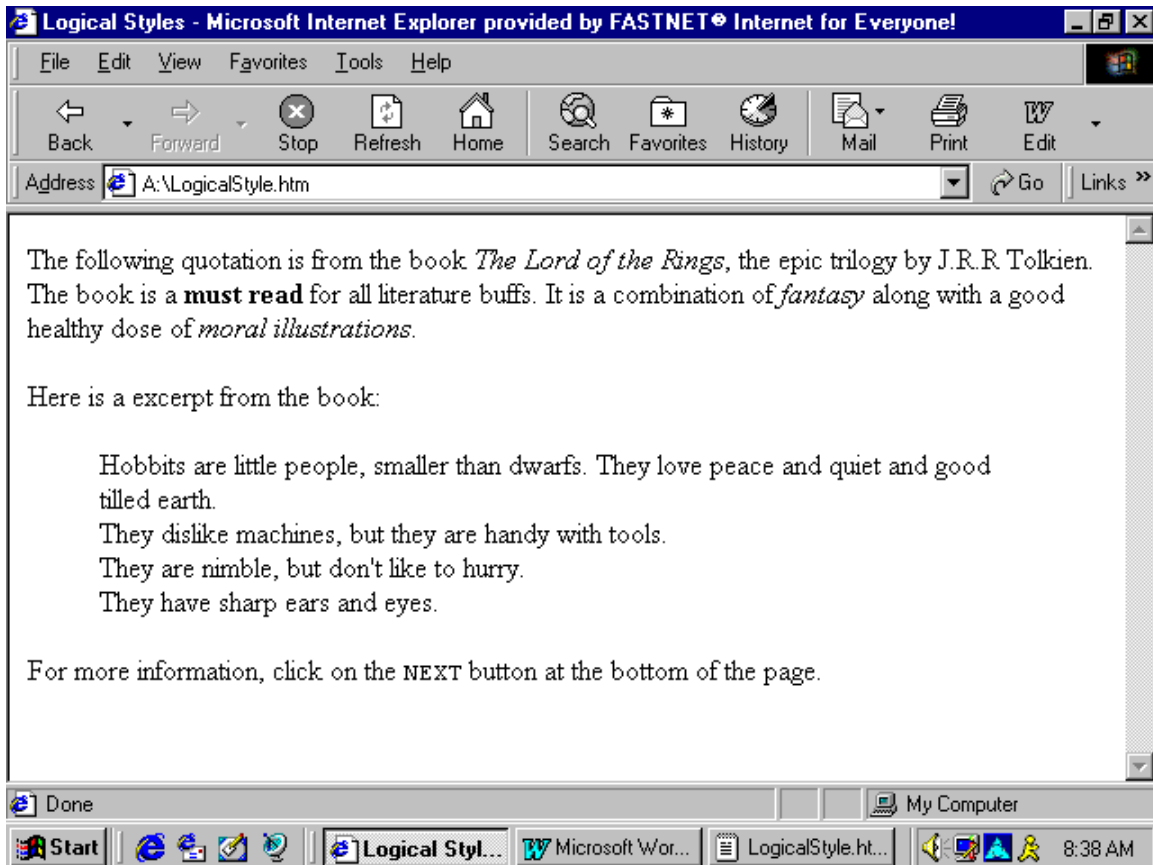
```
<HTML>

  <HEAD><TITLE>Logical Styles</TITLE></HEAD>

<BODY>
<P>
The following quotation is from the book <CITE>The Lord of the Rings</CITE>,
the epic trilogy by J.R.R Tolkien.<BR>
The book is a <STRONG>must read</STRONG> for all literature buffs. It is a
combination of <EM>fantasy</EM> along with a good healthy dose of <EM>moral
illustrations</EM>.</P>
<P>
Here is a excerpt from the book:
<BLOCKQUOTE>
Hobbits are little people, smaller than dwarfs. They love peace and quiet
and good tilled earth.<BR>
They dislike machines, but they are handy with tools.<BR>
They are nimble, but don't like to hurry.<BR>
They have sharp ears and eyes.</BLOCKQUOTE>
</P>
For more information, click on the <KBD>NEXT</KBD> button at the bottom of
the page.

</BODY>
</HTML>
```

The resulting page appears in the browser as below:

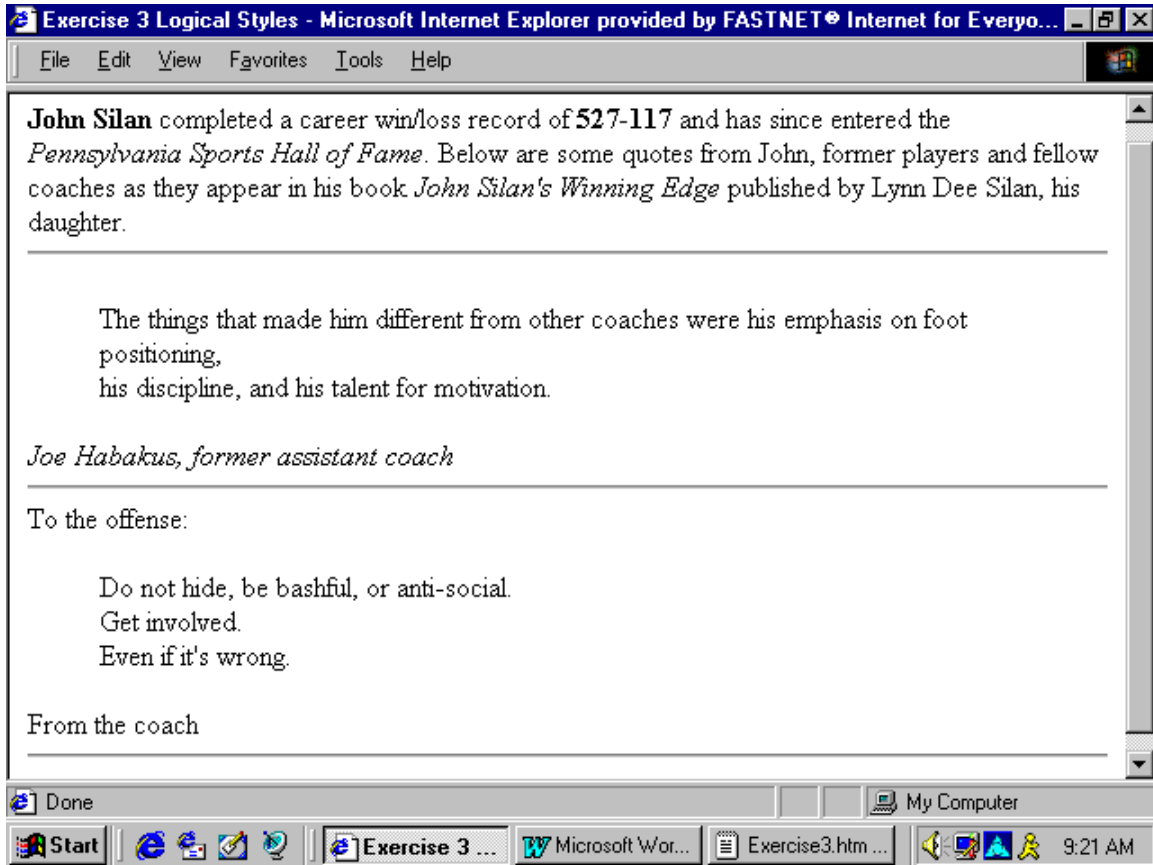


Web Design

24

Exercise 3-2

Create a web page that appears as below:



Headings

There may be times where you would like to emphasize some text more than others or perhaps indicate the priorities of some text over others. By assigning header tags to this text, you can create these effects.

Heading 1

1. `<H1> ... </H1>`
2. Will assign the largest heading font to the text inside the tags.
3. **Example:** `<H1>This text is in heading 1 font.</H1>`

Heading 2

1. `<H2> ... </H2>`
2. Will assign the text between the tags to have a font somewhat **less** emphatic than heading 1 font.

Headings 3 Through 7

1. There are seven levels of headings available with HTML.
2. They all are started and ended as illustrated above in the *Heading 1* and *Heading 2* sections.

Alignment

There are not many options available as far as alignment is concerned. About all you can do is center any text you would like to.

CENTER

1. `<CENTER> ... </CENTER>`
2. Used to place the text located between the tags in a horizontally centered position.
3. **Example:** `<CENTER>This text is centered</CENTER>`

Using Alignment With Headings

Text can be right aligned by assigning it to a heading and including the **style** with the “H” tag.

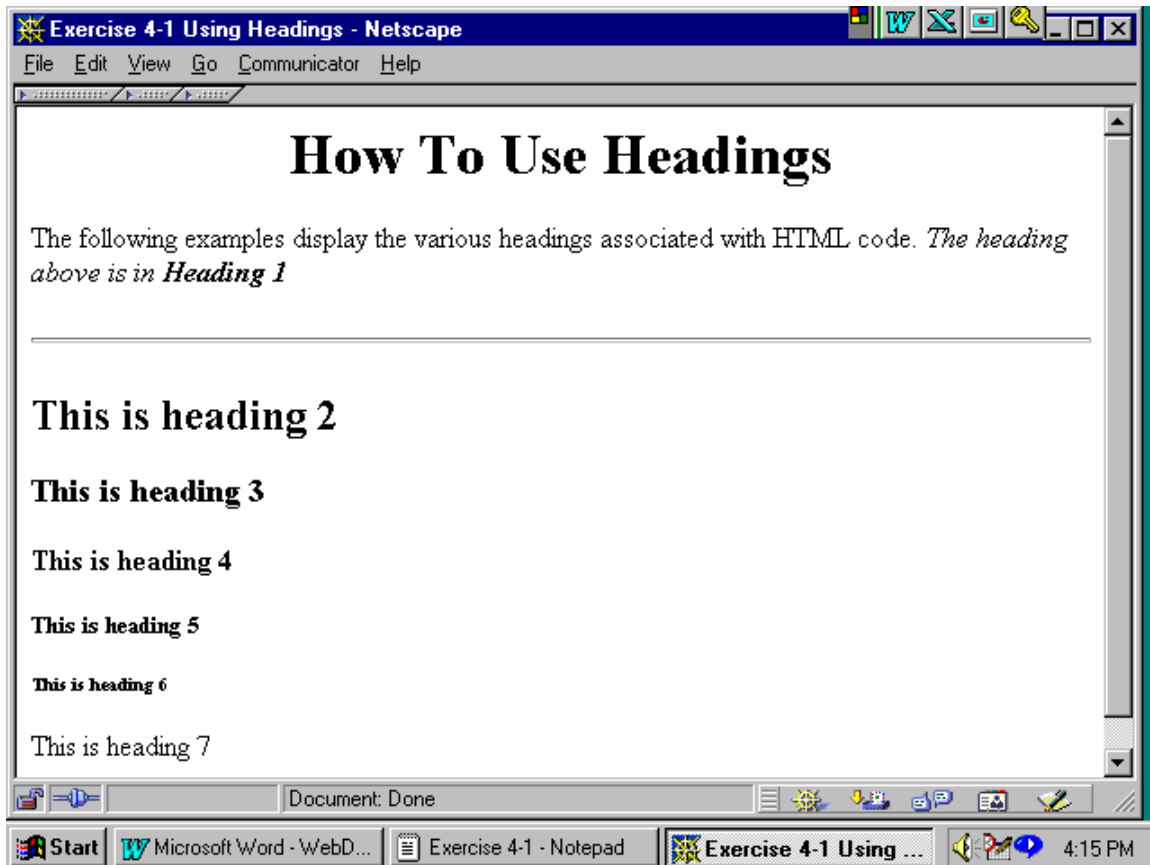
1. `<H4 STYLE = “TEXT-ALIGN: RIGHT”> ... </H4>`
2. Here, we have assigned a style to the “H4” heading that includes an alignment to the right.
3. It should be noted that the text will be right aligned **only in that particular instance of the H1 heading**. In any subsequent uses of H1, the text will revert back to left aligned.
4. Other options available with the **STYLE=** option:
 - TEXT-ALIGN = CENTER
 - TEXT-ALIGN = LEFT
 - TEXT-ALIGN = JUSTIFY (Justifies the text to the left and right margins.)

Web Design

26

Exercise 4-1 Using Headings and Text Alignment

Create a web page that appears like the one below.

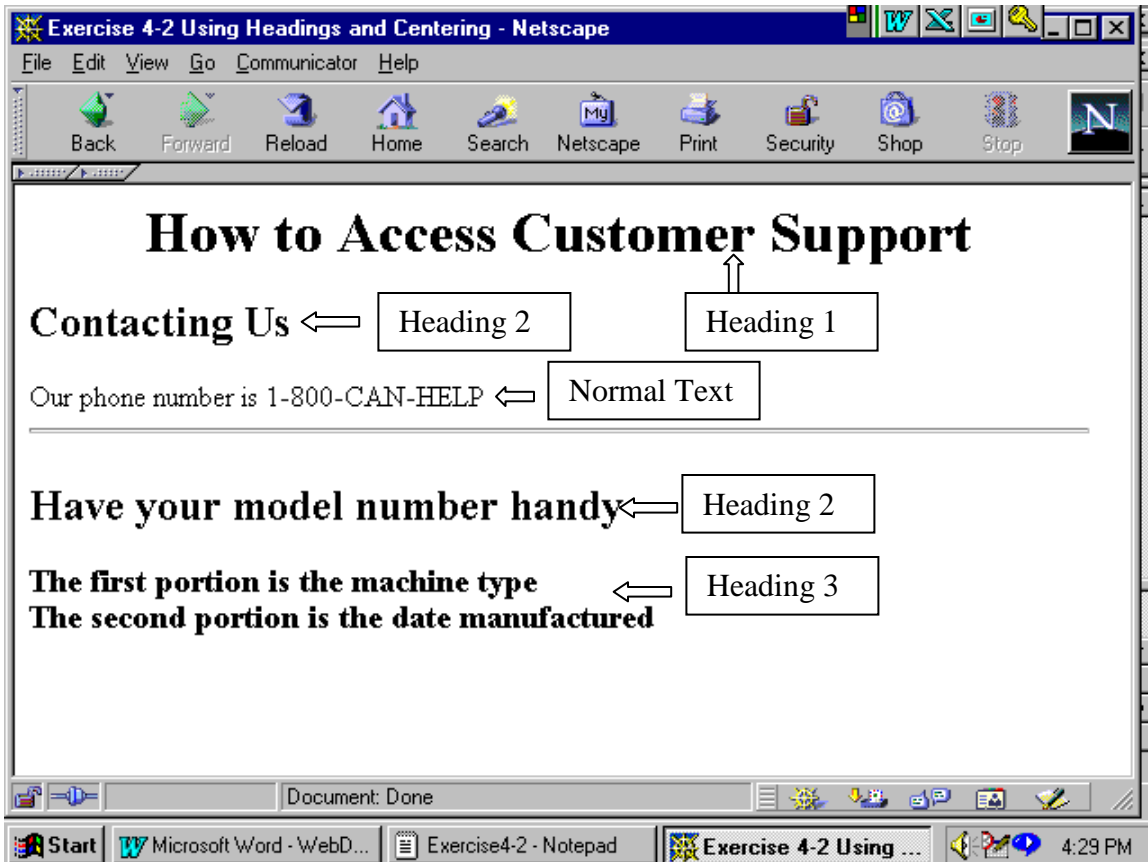


Notice that the main heading is centered and that the second sentence of the introductory paragraph is in italics with the “Heading 1” portion set to bold as well.

Exercise 4-2

Create a web page as it appears on the next page. Please note the following items:

1. The picture on the next page includes the headings requested for each line. These are **not** part of the HTML document.
2. When closing a heading tag, **the browser will automatically perform a carriage return.**



Lists

There are two types of lists that are supported by HTML. These are the **Bulleted List** and the **Numbered List**. They are used to set lists of things off from the rest of the document.

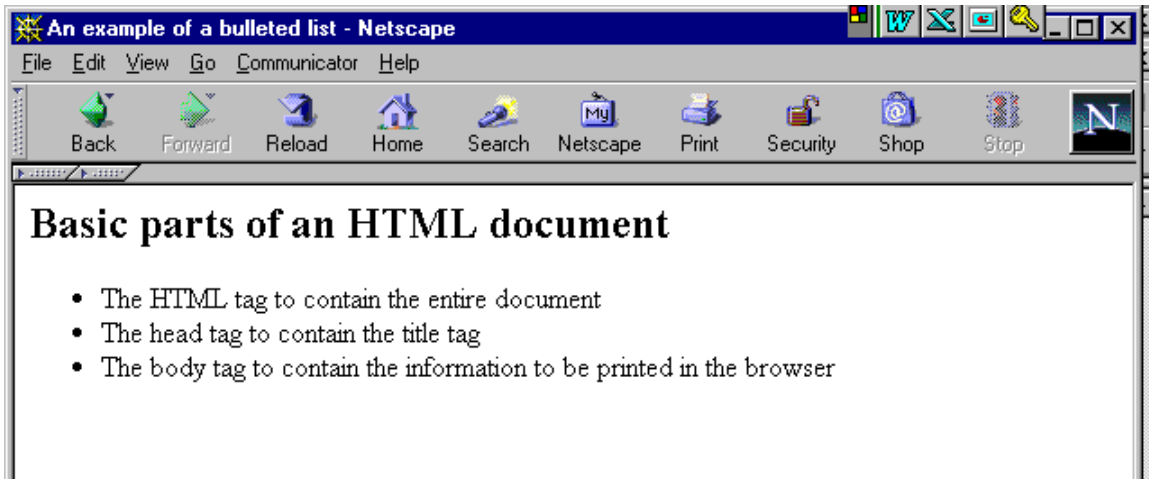
Bulleted Lists

1. ` `
2. The list starts out with the `` tag to designate the beginning of an unordered list.
3. Within the `` tag a `` tag precedes each member of the list. The `` tag will perform a carriage return and place a bullet before the text in the list.
4. There is **no end tag** for the `` tag.
5. After all of the listed items are entered, the `` tag ends the unordered list with a carriage return.

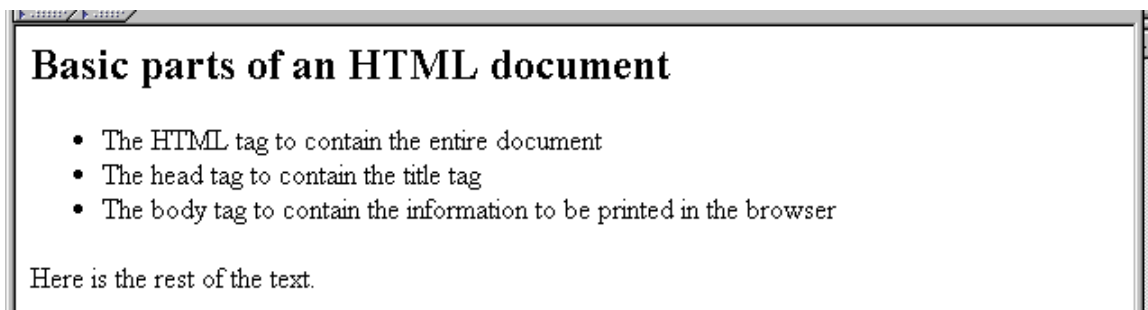
An example of a bulleted list appears on the next page.

```
UExample - Notepad
File Edit Search Help
<HTML>
<HEAD><TITLE>An example of a bulleted
list</TITLE></HEAD>
<BODY>
<H2>Basic parts of an HTML document</H2>
<UL>
<LI>The HTML tag to contain the entire document
<LI>The head tag to contain the title tag
<LI>The body tag to contain the information to be
printed in the browser
</UL>
</BODY>
</HTML>
```

Will result in the following web page in the browser:



A blank line and a carriage return will be included at the end of the list **without adding a <P> tag:**

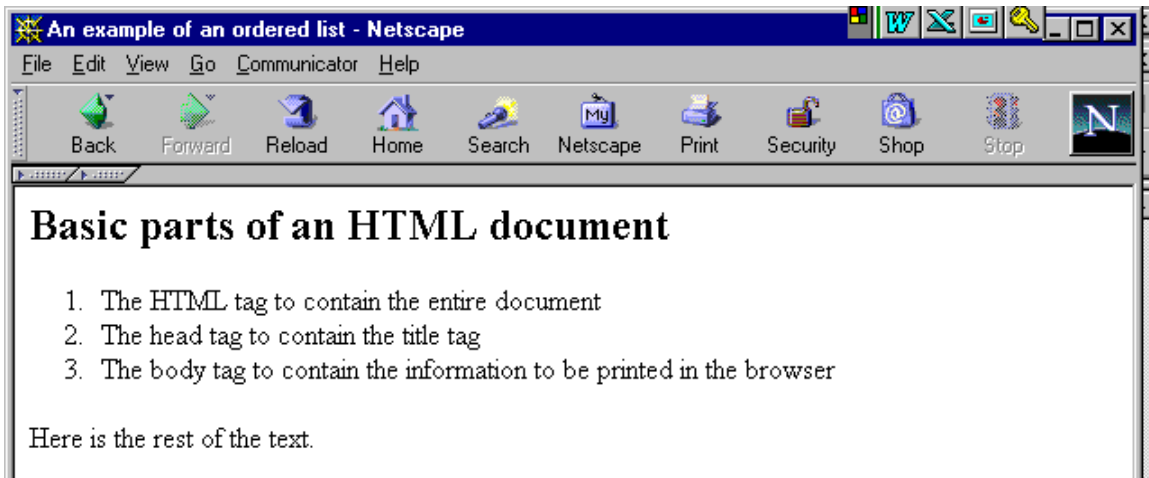


Numbered Lists

These are also referred to as **Ordered Lists**. Each member of the list is preceded by a number.

1. ` `
2. The procedure is the same as with an unordered list except for the start and end tags.

By changing the code from the `` tags to the `` tags, we get the following result:

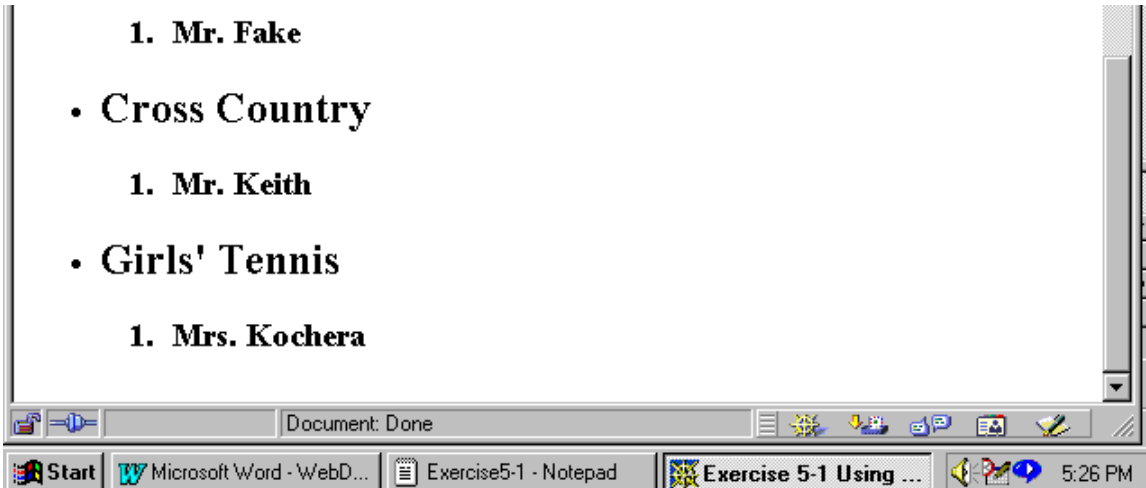


Exercise 5-1 Using Lists

Create the following web page with the title in `<H1>`, the bulleted list in `<H2>`, and the numbered list in `<H3>`:

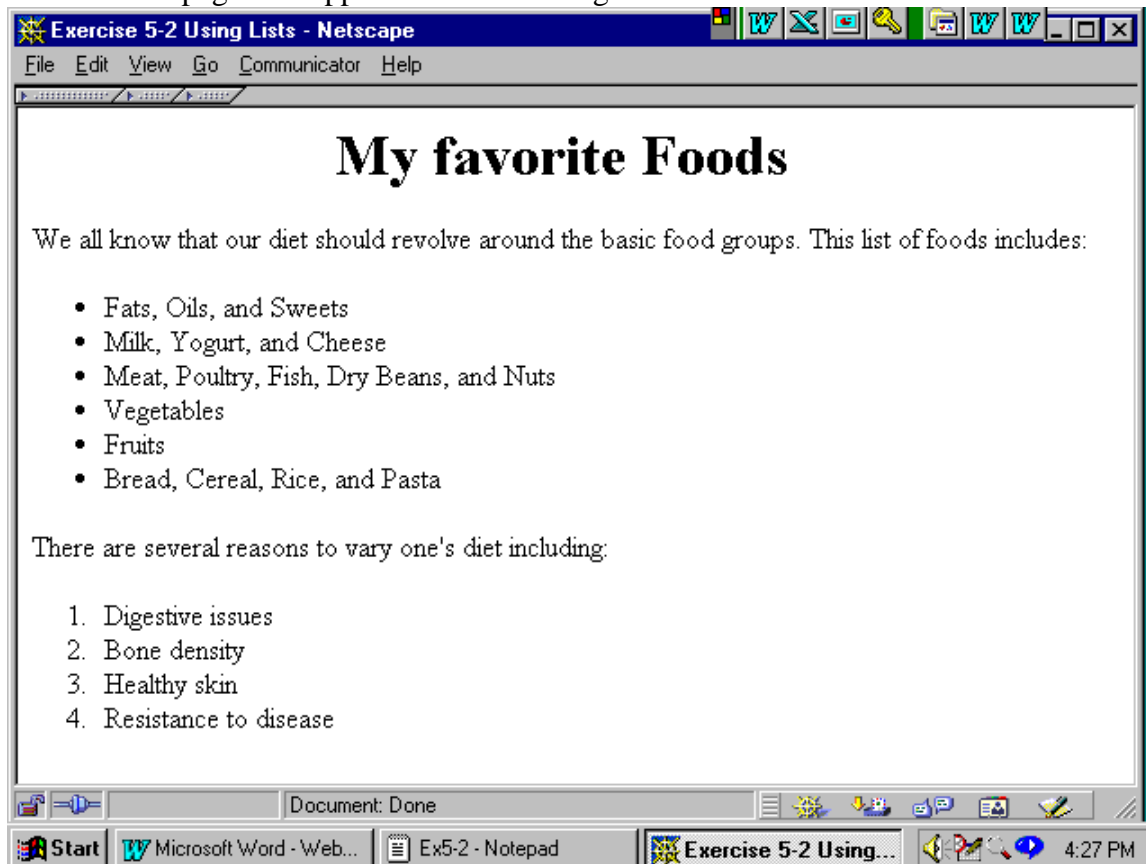


Note: The rest of the page appears on the next page.



Exercise 5-2 Using Lists

Create a web page as it appears below utilizing both ordered and unordered lists.

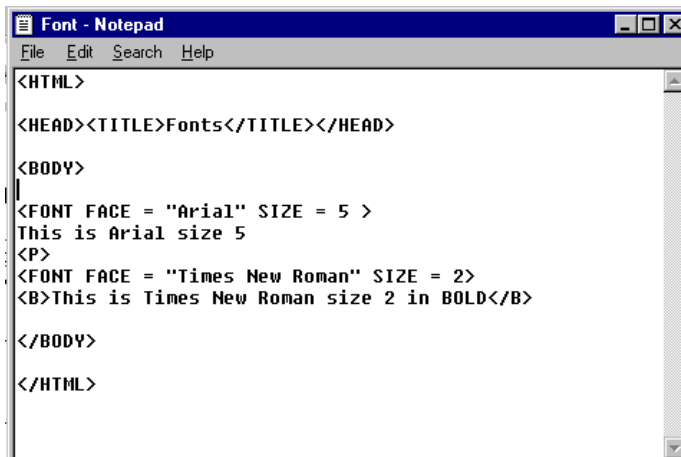


Setting Fonts

There are several fonts that are used fairly universally in most browsers. These include **Arial** and **Times New Roman**. You can experiment with some other fonts based upon the lists in your word processor. Some may work and some may not. Whatever the font you decide upon, **do not use too many different fonts on the same page** as this is distracting to the reader.

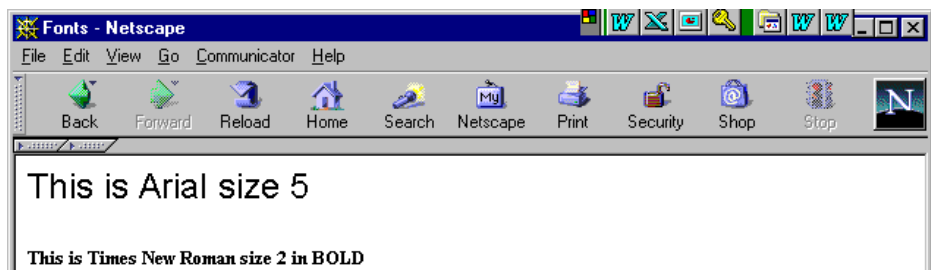
1.
 - Sets the **FONT FACE** for all text following the tag
 - The font face is the name of the font i.e. Arial would be
 - The name of the font face **must be in quotation marks**
 - This setting remains in force until another tag changes it
2.
 - Sets the **FONT SIZE** for all text following the tag
 - The allowable sizes range from **1 through 7** with 1 being the smallest i.e. to set the font size to 5
3. or
 - Sets the **FONT COLOR** for all text following the tag
 - Can be set by utilizing either the **color name** or by using the **RGB code** as is explained on page 33.
4. As an example, the following code:

Note: The bold tags are inserted after the font is set.



```
Font - Notepad
File Edit Search Help
<HTML>
<HEAD><TITLE>Fonts</TITLE></HEAD>
<BODY>
<FONT FACE = "Arial" SIZE = 5 >
This is Arial size 5
<P>
<FONT FACE = "Times New Roman" SIZE = 2>
<B>This is Times New Roman size 2 in BOLD</B>
</BODY>
</HTML>
```

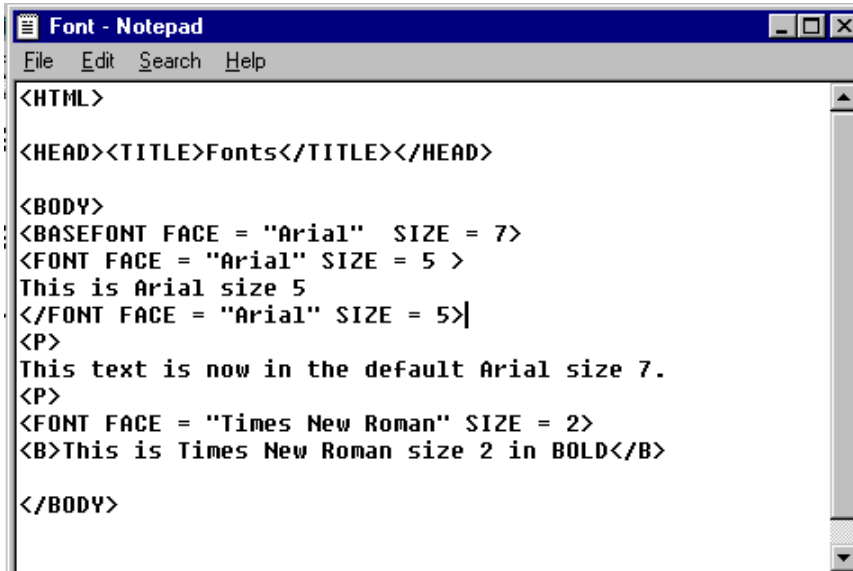
Results in the following page:



Base Fonts

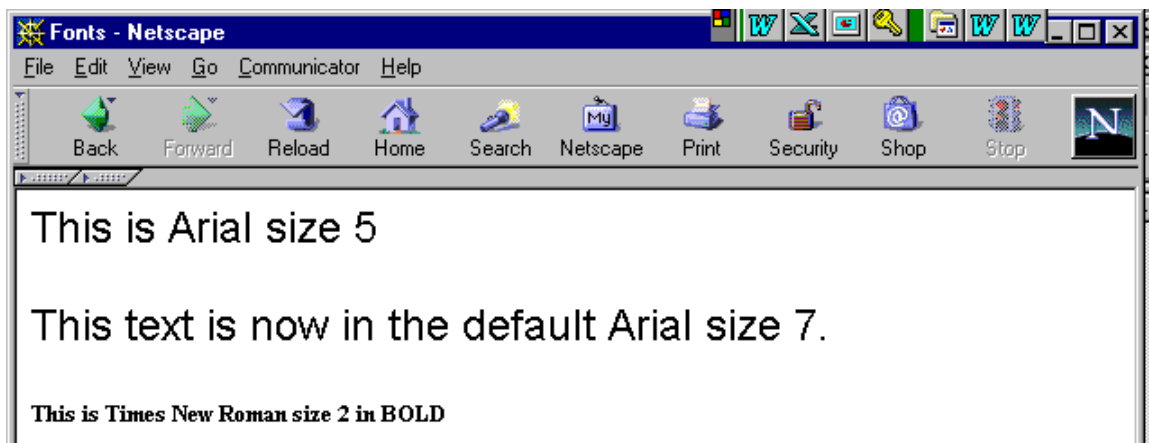
The default font for the entire document can be set by including the **BASEFONT** attribute immediately after the **<BODY>** tag. The size and face attributes can be used within the **BASEFONT** tag.

1. `<BASEFONT FACE = "Times New Roman" SIZE = 5>`
 - This example sets the default font for the entire document to Times New Roman size 5.
2. By changing the code from above to the following:



```
Font - Notepad
File Edit Search Help
<HTML>
<HEAD><TITLE>Fonts</TITLE></HEAD>
<BODY>
<BASEFONT FACE = "Arial" SIZE = 7>
<FONT FACE = "Arial" SIZE = 5 >
This is Arial size 5
</FONT FACE = "Arial" SIZE = 5|
<P>
This text is now in the default Arial size 7.
<P>
<FONT FACE = "Times New Roman" SIZE = 2>
<B>This is Times New Roman size 2 in BOLD</B>
</BODY>
```

The resulting page looks like the following:



Note: After we ended the Arial Size 5 tag, the font **defaulted** to the Arial size 7 **BASEFONT** and then was overwritten by the Times New Roman size 2 font tag.

Color

Color can be added to both the background as well as the text and any links within a web page. By default, the text color is black and the background is light gray or white. If you decide to add color, care must be taken to ensure that the color combinations render the text readable. For instance, a light gold background with white text would be very difficult to read. If one of the colors is light, the other should be dark and vice versa.

RGB Values

HTML colors are controlled by issuing a six character code that determines a color's **red, green, and blue** values. These values are hexadecimal in nature. That is, they can have up to sixteen (16) values. Since there are only ten digits available, the other six possible values are covered by the letters A through F with **F** being the **largest** and **0** being the **smallest**. The **first two** digits set the **red** portion of the color. The **second two** digits set the **green** portion of the color. The **last two** digits set the **blue** portion of the color.

Full Red = #FF0000

Full Green = #00FF00

Full Blue = #0000FF

Black = #000000

White = #FFFFFF

Background Color

1. **BGCOLOR** = "#....."
2. Placed inside the <BODY> tag
 - i.e. <BODY BGCOLOR = "#FF99FF">
 - Sets the background color to **pink**
3. It will take some trial and error to arrive at the desired color.

Text Color

1. **TEXT** = "#....."
2. Placed inside the <BODY> tag
 - i.e. <BODY BGCOLOR = "#FF99FF" TEXT = "#FFFFFF">
 - Sets the background color to **pink** and the text color to **white**

Link Color

1. **LINK** = "#....."
2. Placed inside the <BODY> tag
 - i.e.<BODY BGCOLOR = "#FF99FF" TEXT = "#FFFFFF" LINK = "#0000FF">
 - Sets the background color to **pink**, the text color to **white** and the link color to **blue**

Web Design

34

Named Colors

Some basic colors can be designated by using their name. These colors include **black, white, maroon, green, lime, olive, navy, teal, aqua, silver, gray, purple, fuchsia, and yellow**. Some browsers support more colors but try to stick to these basic colors. They are designated using the same syntax as the RGB colors.

1. `<BODY BGCOLOR = "NAVY" TEXT = "WHITE">`

- Sets the background color to a generic **navy** and the text color to a **white**.

Note: For best control and to be sure the browser used by your reader will support the color name, **try to use the RGB color scheme**. In this way, you can be sure that your reader will get the desired effect.

Exercise 6-1 Setting Colors

Create a web page with the following text:



Change the background and text colors to the following combinations:

Background	Text
Pink	Blue
Dark Green	Light Blue
Red	Black
Purple	Yellow
Tan	Black
Lavender	White

Note: You will probably have to experiment. It may be a good idea to write down the RGB code you used to get the desired shades. If you finish this exercise before the rest of the class, experiment with more colors.

Inserting Images

Images can be used to add color and variety to your pages. They can be scanned, taken with a digital camera, or gotten from the Internet. There are several formats from which you can choose.

Image Formats

1. GIF Files

- Graphical Interchange Format files
- Maximum of 256 colors
- Are compressed to reduce their size
- Can include a transparent color which allows the browser to eliminate excessive background color
- Can be **interlaced**, which means that the browser can gradually display a better and better image as it receives the packets
- Supports a simple animation technique where several images can be stored and displayed one after the other to simulate animation
- Have an extension of **.GIF**

2. JPEG Files

- Joint Photographic Expert Group
- Can have either 16.7 million or 2 billion colors
- Use a compression technique that slightly diminishes the picture quality
- Supports imaging similar to the GIF interlacing
- Does not support transparent background
- Does not support animation
- Have an extension of **.JPG**

3. BMP

- Windows bitmap

4. PCX

- Another bitmap format

5. TIF

- Tagged Image File

6. PIC

- Macintosh picture file

Guidelines for Images

1. **Don't over-do the use of images** as they can quickly make your page distracting to the reader and **extremely large** rendering them **slow to download**.

Web Design

36

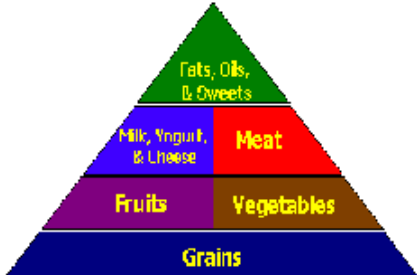
2. ``
 - Loads the image from the file *School.gif* which is **stored in the same location as the HTML code file.**
3. If you use the **ALT** attribute inside the IMG tag, you can display a short description of the picture being loaded before it is visible.
 - ``
 - The phrase *Picture of a school* is displayed until the graphic is loaded.
4. Use **HEIGHT** and **WIDTH** attributes to set the dimensions of your graphic.
 - ``
 - Inserts the image with a **height of 100 pixels** and a **width of 50 pixels**
5. Use the **BORDER** attribute to eliminate or set the border size around the graphic
 - ``
 - The **BORDER = 0** attribute sets the border size to **0 pixels**
 - Text can be placed right next to the image
 - If the border were set **higher than zero**, text or other images would be **spaced that number of pixels**
6. It is suggested that you use **GIF** format files for images created by **drawing or paint programs** and **JPEG** format for **photographic images**
7. Transparent GIF files should be used to insert images seamlessly into your page with no background
8. Pictures and images found on the web or in books or magazines are copyrighted and cannot be used in your web page without permission from the author.

Exercise 7-1 Using Bitmap Graphics

Create a web page as it appears on the next page. **Your instructor will tell you where to get the images you need.** Please be sure to include the following:

- The Text is red
- The background is silver
- The base font is Arial size 5
- The top heading is Heading 1
- Copy the images to your diskette because **the images used in a web page should be stored in the same location as the source code for the page.**
- The text "Nutrition Pyramid" below the picture should be **bold** and *italics*
- The text "Base Area" below the picture should be underlined.
- Set the **HEIGHT** of the image to 300 pixels and the **WIDTH** to 400 pixels

The Nutrition Pyramid



Regardless of age, it is imperative that everyone maintains a balanced diet. There is no better way than to follow the **Nutrition Pyramid**

Notice How the more nutritious foods are located in the larger Base Area of the pyramid. That is because these are the foods we should eat more of.

Start | Microsoft Word - Web... | ex7-1 - Notepad | Nutrition Pyramid -... | 6:38 PM

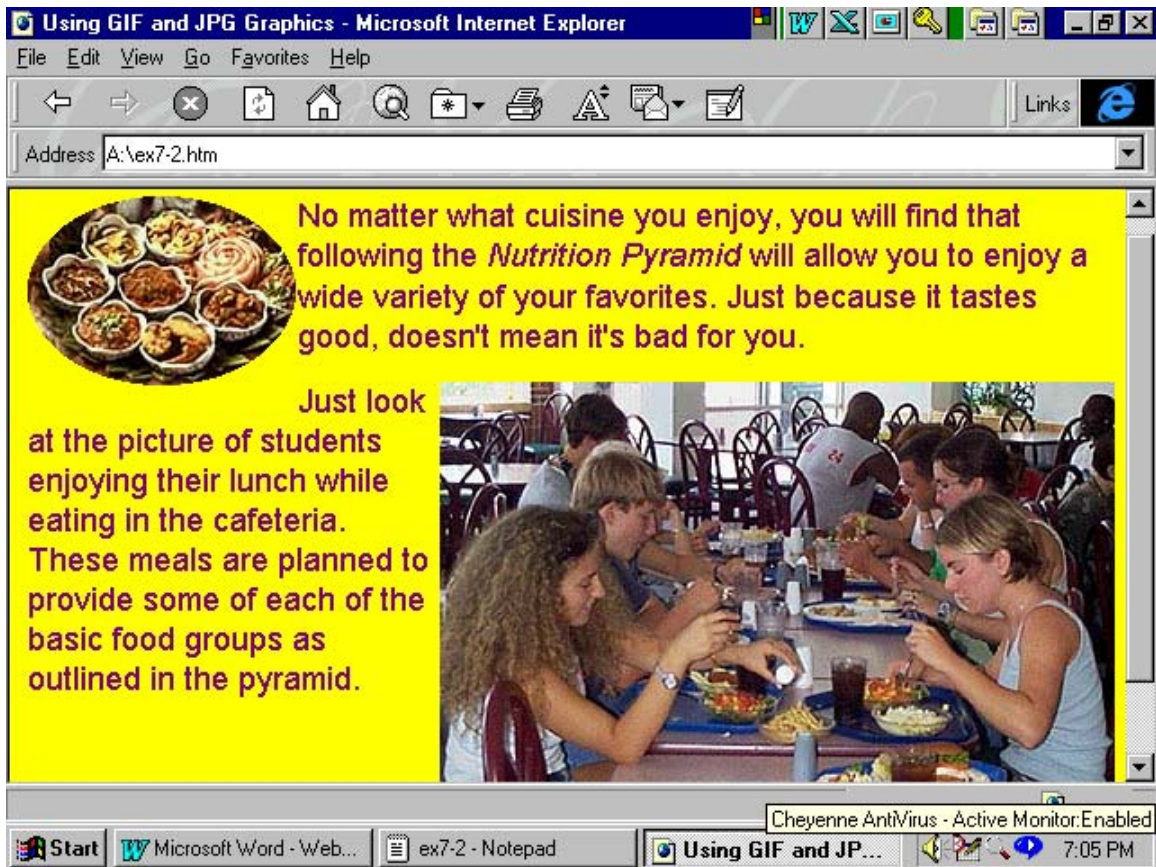
Exercise 7-2 Using GIF and JPG Graphics

Create a web page as it appears on the following page.

- Your instructor will tell you where you can obtain the pictures
- The background color is yellow
- The text is purple
- Use the **ALIGN = LEFT** and **ALIGN = RIGHT** in the IMG tag to locate the pictures and have the text wrap to the other side of each of the pictures

Web Design

38



Note: See how the GIF file at the top does not have the square look of either the JPG file at the bottom or the BMP file in the previous exercise. That is because of the GIF format's **transparent color** property.

Adding Space Around Graphics

Sometimes, you may want the text to not run up against the image. Looking at the page you created in exercise 7-2, you can see how close the text has come to the pictures. To fix this, you can include **vertical** and **horizontal space** within the IMG tag.

1. ``
 - Here, we have designated a 20 pixel space to the left and right of the picture and a 10 pixel space above and below the picture.

Exercise 7-3 Adding Space Around a Picture

Add a 20 pixel space to the left and right of both pictures in exercise 7-2 and a 30 pixel space above and below it. Notice the difference.

Web Design

39

Project #1

Create a web page on **one** of the following subjects:

- Your family
- Your favorite vacation spot
- A question that you answer on the web page

The project must contain the following:

1. At least two (2) pictures
 - Can consume no more than $\frac{1}{4}$ of the page
 - Must have text to one side of them
 - Include either borders around them or horizontal / vertical space
2. A horizontal rule
3. At least two (2) different font faces
4. At least two (2) different font sizes
5. At least two (2) different font colors
6. A background color
7. At least one list
8. A title
9. At least two (2) distinct paragraphs
10. Use of at least two (2) physical font styles
11. A header at the top of the page containing the title of the page in one heading size with your name on another line in a different heading size

Links

As you create your web site, you will probably find it necessary to include more than one page. For instance, you may have so much information that, to put it all in one page, would make it way too long. So, you would like to group the information in certain areas and devote a separate page for each. In order to access these pages, you will need to include a link to each page.

1. You need to know the URL of the page to access or, if it is another page within your site, its name.
2. Start with the **<A>** tag
 - Signifies an **address**
3. Add the **HREF** attribute inside the **<A>** tag which indicates the address of the page linked to
 - ** Go to my page **
 - The above example creates text on the page that says "Go to my page"
 - This text is a link to the **http://www.mypage.com** web site
 - When the user left clicks on the text, it will automatically open the linked site
 - **Go to page 2 **
 - The above example creates text on the page that says "Go to page 2"
 - This text is a link to the "Page2.htm" file stored in the same location as the current page
 - When the user left clicks on the text, it will automatically open that page
4. Links can be created to another place on the same page
 - Use the **NAME** attribute inside the **<A>** tag to **name a portion of the page**
 - ****
 - This tag appears at the point you would like a link to point
 - Use the **** tag to reference this portion of the page
 - **Go to that place**
 - This tag creates the text "Go to that place" as a link to the portion of the page named "PLACE"
5. Pictures can also be used as links
 - Instead of providing text after the **** tag, use the **** tag
 - ****
 - The above example uses the image named "Bookpage.gif" as a link to the "Page2.htm" page, **both** of which are stored in the same location as the current page.

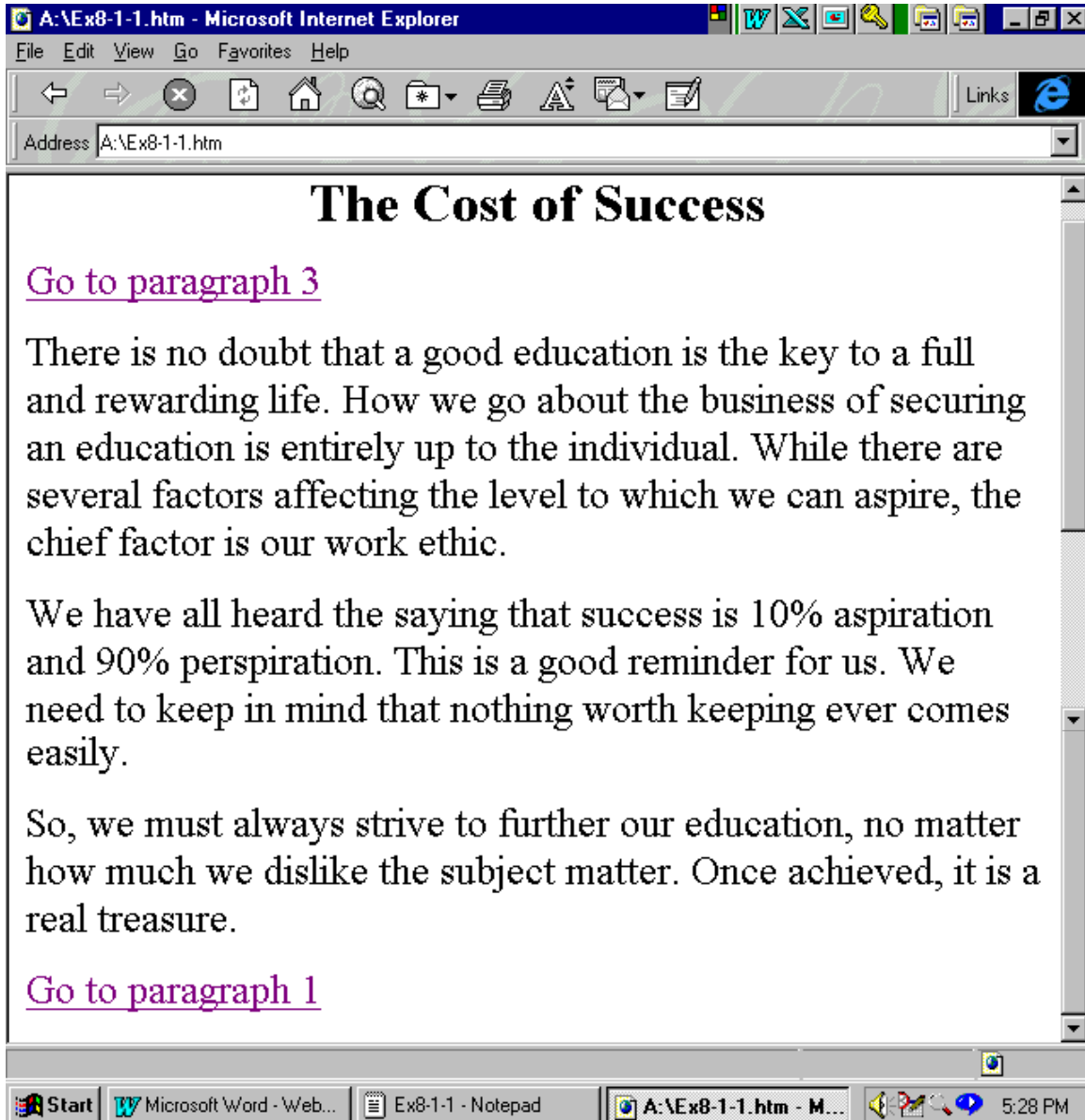
Assigning Color to Links

Link text is colored blue by default. You can change the color of links by including the **LINK** attribute within the **<BODY>** tag

1. **<BODY BGCOLOR = "#000000" TEXT = "#FFFFFF" LINK = "#FFFF00">**
 - In the above example, the background color for the page is set to black, the text color to white and the link color to yellow.
2. **<BODY LINK = "#FFFF00" VLINK = "#FF0000">**
 - The **VLINK** attribute sets the color for a visited link (the above example is red).

Exercise 8-1 Using Links Within a Page

Create the following web page, creating links to the third paragraph in that page. Then, at the bottom of the page, provide a link to the first paragraph.




When you click on either the “Go to paragraph 3” or “Go to paragraph 1” links, you should be able to scroll back and forth within the page without using the scroll bars.

Exercise 8-2 Using Links From Page to Page

Create two web pages as they appear below (you may use different pictures if you like). Create links on both pages to allow the user to move between the pages. Include the following items:

1. Set the **link color**
2. Set the **visited link color**
3. Set the text color for the **heading and the text**
4. Choose a **background color**
5. Set the **height and width** attributes for both pictures
6. Set some **space between the pictures and the text**


The African Lion



There is probably no animal on the African plain more dreaded than the African lion. With its piercing roar, fiery eyes, and formidable teeth, it strikes fear in every animal in the area.

It's hard to believe that the adult lion grew from such a lovable, cuddly baby. To see a picture of the [lion cub](#) click on this link.

The Lion Cub



See how cute an cuddly this lion cub is? Some people raise lions from cubs and become very attached to them. Part of the reason is because they are so cute as cubs.

Unfortunately, some of these people end up injured or even killed by the animals they love.

They forgot that, even though they once were cute little fur balls, now they are dangerous carnivores whose instincts cannot always be altered by their up-bringing. To see an adult [lion](#) click on this link.

Note: The underlined words on both pages are the links to the other page.

Graphic Links

As you create web pages, you may want to link your pages via a picture. In this case, you may want the user to click on a picture in order to move to another web page. The picture is actually the link.

1. Start with the ADDRESS TAG `<A >` and include the HREF attribute to point to the desired page.
 - ``
 - In the above example, the user would click on the *picture1.gif* image on the page in order to move to *page1.htm*.
2. The **attributes** discussed in the previous sections can be included within the IMG tag as if it were a normal picture.
 - ``
 - In the above example, not only is the *picture1.gif* image a link to *page1.htm*, but the image is aligned **RIGHT** with a border of 15 pixels.
 - We could have also included height and width attributes

Exercise 9-1

Change the code in exercise 8-2 to make the lion and lion cub pictures the links for each of the two pages.

Image Maps

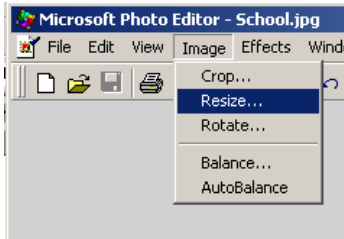
You can designate different portions of a picture as links to other sites, pages, or areas of the same page. To do this, you need to use the `<MAP>` tag along with the `<AREA>` and `` tags.

1. Find or create an image to serve as the image map
2. Decide upon the rectangular boundaries within the image that will serve as the links to various areas
 - Use the **NAME** attribute inside the `<MAP>` tag to designate the name of the image map
 - `<MAP NAME = "IMGMAP1">`
3. Use the `<AREA>` tag to designate the portion of the image to use as a link
 - Use the **SHAPE** attribute to designate the shape to use and use the **COORDS** attribute to designate the points (in pixels) that serve as boundaries for the link
 - `<MAP NAME = "IMGMAP1">`
`<AREA SHAPE = RECT COORDS = 0,40,39,79>`
 - In the example above, the area designated as a link is **rectangular** with the coordinates 0, 40, 39, and 79 representing the **left, top, right,** and **bottom** boundaries of the area
 - These numeric boundaries are expressed in pixels
 - Some graphics software packages provide these coordinates

When using an image for an image map, you need to do the following things:

Determining Image Coordinates

1. Insert the image into the page to see if the size is correct
 - If not, open the image in an image editing application such as “Photo Editor” (one of the tools available in Microsoft Office).

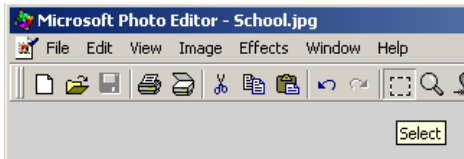


- When opened, use the *Resize* option from the *Image* menu option
- Then, use the *percentage option* to change both the height and width of the image simultaneously, eliminating the possibility of distorting the appearance of the image.

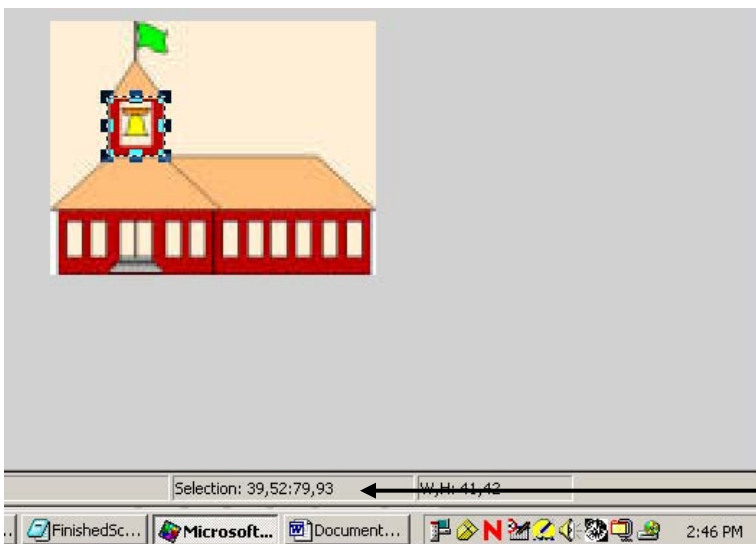


- To the left is an example of how you would **double the size** of an image by increasing both its width and height to 200%.
- When you have resized the image, overwrite the old image file by saving it back to its original location as its same name.

2. Determine the location of the desired mapping area by using the *Select* tool in the toolbar.



- You can now drag a square around the desired area and this area's coordinates will appear at the bottom of the window.



3. These coordinates can then be used to construct the image map.

Web Design

45

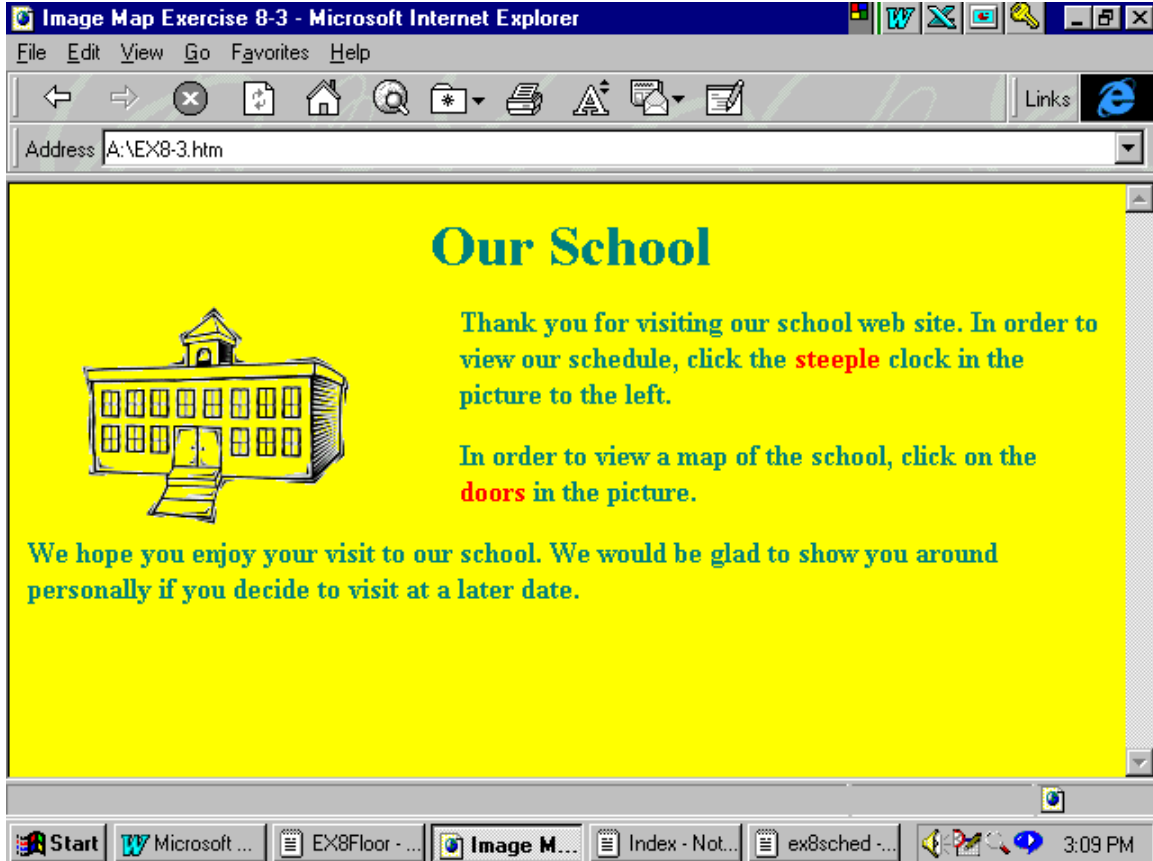
4. Use the **HREF** attribute to designate the page that will open when the link is clicked
 - `<MAP NAME = "IMGMAP1">`
`<AREA SHAPE = RECT COORDS = 0,40,39,79`
`HREF = "PAGE1.HTM">`
5. Additional areas can be designated within the same map
 - `<MAP NAME = "IMGMAP1">`
`<AREA SHAPE = RECT COORDS = 0,40,39,79`
`HREF = "PAGE1.HTM">`
`<AREA SHAPE = RECT COORDS = 40,0,109,79`
`HREF = "PAGE2.HTM">`
6. Close the MAP tag with `</MAP>`
 - `<MAP NAME = "IMGMAP1">`
`<AREA SHAPE = RECT COORDS = 0,40,39,79`
`HREF = "PAGE1.HTM">`
`<AREA SHAPE = RECT COORDS = 40,0,109`
`HREF = "PAGE2.HTM">`
`</MAP>`
7. Insert the image using the `` tag with the SRC attribute and designate the image map by using the **USEMAP** attribute
 - `<MAP NAME = "IMGMAP1">`
`<AREA SHAPE = RECT COORDS = 0,40,39,79`
`HREF = "PAGE1.HTM">`
`<AREA SHAPE = RECT COORDS = 40,0,109,79`
`HREF = "PAGE2.HTM">`
`</MAP>`
``
 - In the above example, we chose the *Picture1.jpg* file as our image and we chose the *IMGMAP1* image map as our map

Example of an Image Map

1. First, we will get a graphic from the Internet. For our example, we will use a GIF image of a school.
2. We create the home page and call it "INDEX.HTM" because, by default, web servers will search for the *INDEX* file as the home page.
3. After the image is inserted into the INDEX page, we need to estimate where the mapped areas will be. In our example, we will have the user click on the **clock steeple** on the picture to see the daily **bell schedule**. He/she will click on the **doors** to see a **map of the building**.
4. The INDEX page appears on the next page along with the code to create it.

Web Design

46



```
File Edit Search Help
<HTML>
<HEAD><TITLE>Image Map Exercise 8-3</TITLE></HEAD>
<BODY BASEFONT SIZE = 5 FACE = "ARIAL" bgcolor = "yellow"
<H1 ALIGN = CENTER><B>Our School</B></H1>
<MAP NAME = "IMGMAP1">
  <AREA SHAPE = RECT COORDS = 55,5,85,40
    HREF = "EX8SCHED.HTM">
  <AREA SHAPE = RECT COORDS = 50,70,80,90
    HREF = "EX8FLOOR.HTM">
</MAP>
<IMG SRC = "School.gif" USEMAP = "#IMGMAP1" ALIGN = LEF
```

Map tag where we
1) Designate the name of the map
2) Designate the shapes as rectangles
3) Designate the borders of each rectangular

We place the image into the page and designate it as an image map (USEMAP attribute) and name the map image used

Web Design

47

When we click on the steeple area, we open the schedule page.

Schedule - Microsoft Internet Explorer

File Edit View Go Favorites Help

Address A:\ex8sched.htm

The Daily Schedule

Below is a schedule for the beginning and end of each period. You will notice that there are 3 minutes allowed for changing classes.

1. 8:00 - 8:42
2. 8:45 - 9:27
3. 9:30 - 10:12
4. 10:15 - 10:57
5. 11:00 - 11:42
6. 11:45 - 12:27 or 12:18 - 1:00
7. 1:03 - 1:45
8. 1:50 - 2:32

This is an ordered (numbered) list.

[BACK TO HOME PAGE](#) ← Link back to the home page.

Shortcut to A:\EX8SCHED.HTM (local)

Start Microsoft ... EX8Floor ... Schedul... Index - Not... ex8sched ... 3:19 PM

If we click on the doors, we will open the floor plan page.

Address A:\EX8Floor.htm

Our Floorplan

Hopefully you will be able to find your way from room to room with the help of this map. If you have any questions, please call the office and we will be happy to help you.

[BACK TO THE HOME PAGE](#) ← Link to home

Restrooms
Telephones
Information
Classroom
Elevator
Stairs
Ramp/Access

Entrance at 58th Street

3:24 PM

Web Design

48

Exercise 10 -1 Creating an Image Map

Using the picture below named “*ANIMALS.JPG*” create a web site where the user can click on at least **two (2)** of the animals appearing in the picture and go to pages about those particular animals that **you created**. **One (1)** link should be to a page on the **Internet**. Your instructor will tell you where to access the *ANIMALS.JPG* file to complete this exercise. You should include the following:

1. Each page that you create should include at least one picture of the animal **and** at least a few sentences about it.
2. Each page should contain a link back to the home page
3. The home page should be named “*INDEX.HTM*”
4. Don't be afraid to include as many of the tags we've done before as you can. The practice will be good for you.
5. If you finish before the rest of the class, add links to pages about other animals.



Creating Tables

Tables are used for two different things. First, they can be used to represent data in sets of columns and rows. Second, they can be used to provide a layout for a web page. We will discuss both uses separately.

Using Tables to Display Data

Perhaps you would like to display data that you have gathered from an Excel spreadsheet. Let us say that the data looks like the data below.

NAME	CLASS	HOMEROOM
Carrie A. Tune	Sophomore	Rm. 101
Vic Tory	Senior	Rm. 105

To represent this data in a web page, we will have to do the following:

1. Provide **TABLE** tags
 - `<TABLE`
2. Add the **BORDER** attribute to establish a border around the table. As with graphics, this border will be in pixels.
 - `<TABLE BORDER = 10>`
3. Create the first table row by using a set of `<TR>...</TR>` tags
 - `<TABLE BORDER = 10>`
`<TR>`
`</TR>`
`</TABLE>`
4. Use a set of `<TH> ... </TH>` tags to create column **headers** in the table
 - `<TABLE BORDER = 10>`
`<TR>`
`<TH>NAME</TH>`
`<TH>CLASS</TH>`
`<TH>HOMEROOM</TH>`
`</TR>`
`</TABLE>`

Web Design

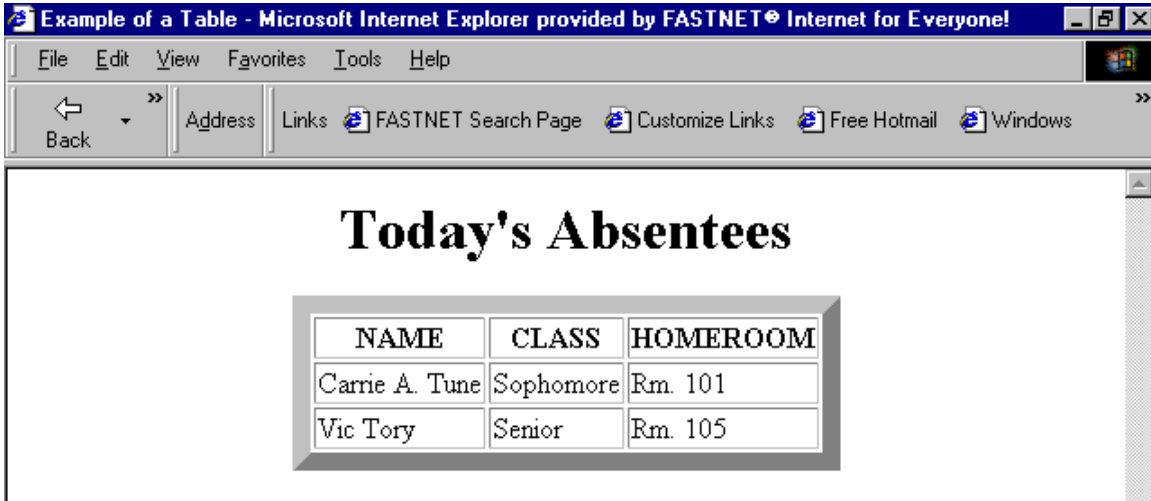
50

5. Additional rows for the table can be added by adding additional `<TR>` tags
- If the contents of the row are to appear as **data** rather than headings, use the `<TD> ... </TD>` tags.
 - This information will be left aligned in that particular row and column and will not be bold.
 - ```
<TABLE BORDER = 10>
 <TR>
 <TH>NAME</TH>
 <TH>CLASS</TH>
 <TH>HOMEROOM</TH>
 </TR>
 <TR>
 <TD>Carrie A. Tune</TD>
 <TD>Sophomore</TD>
 <TD>Rm. 101</TD>
 </TR>
 <TR>
 <TD>Vic Tory</TD>
 <TD>Senior</TD>
 <TD>Rm. 105</TD>
 </TR>
</TABLE>
```

6. The following code will yield the web page as it appears on the next page.

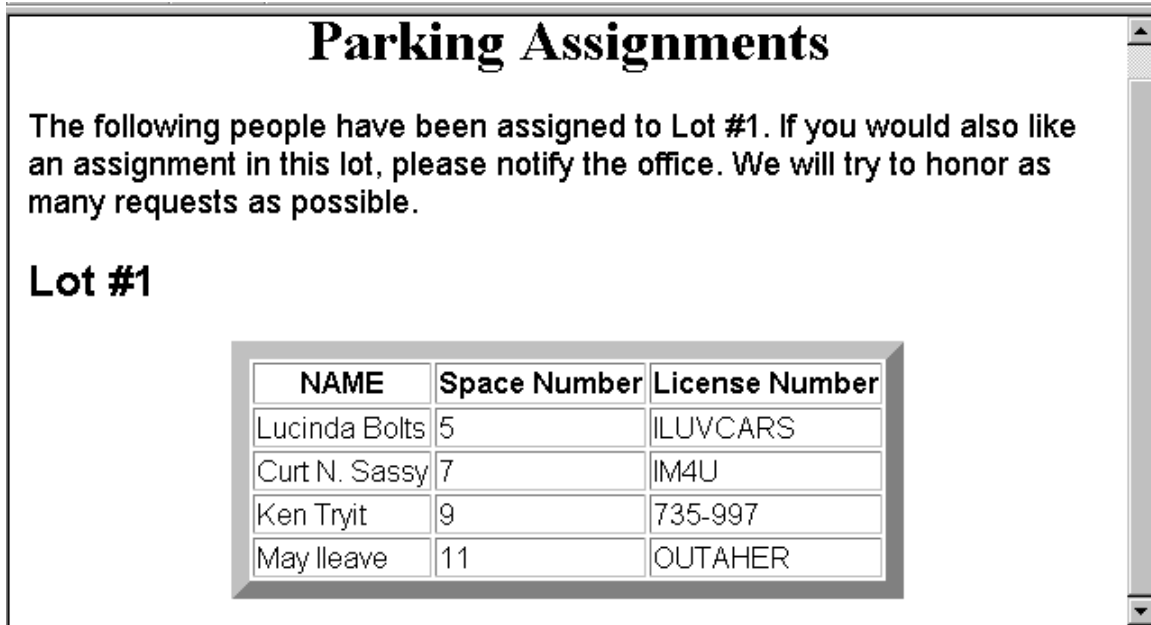
```
<HTML>
<HEAD><TITLE>Example of a Table</TITLE></HEAD>
<BODY>
<H1 ALIGN = CENTER>Today's Absentees</H1>
 <TABLE BORDER = 10 ALIGN = CENTER>
 <TR>
 <TH>NAME</TH>
 <TH>CLASS</TH>
 <TH>HOMEROOM</TH>
 </TR>
 <TR>
 <TD>Carrie A. Tune</TD>
 <TD>Sophomore</TD>
 <TD>Rm. 101</TD>
 </TR>
 <TR>
 <TD>Vic Tory</TD>
 <TD>Senior</TD>
 <TD>Rm. 105</TD>
 </TR>
```

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



## Exercise 11-1 Creating a Table for Data

Create a web page to display the information as it appears below.

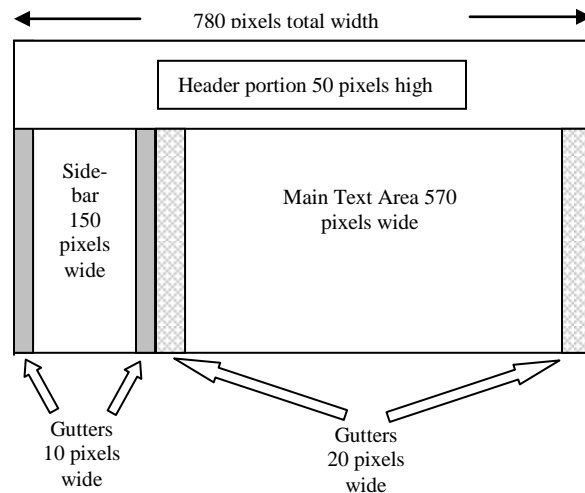


## Using Tables for Page Layouts

We have seen that a table can be used to display data in an orderly manner. For this reason, tables are effective in creating a page layout. Each cell of a table can be viewed as a small page within a page. Attributes of tags within a given cell will pertain only to that particular cell or portion of the page. So, it is possible to center a picture inside the left-most 25% of the page while centering text between the border of this column and the middle of the page.

The following steps are necessary to create a table layout for a web page.

1. Determine what your layout will look like. In the example we will build, our layout will look like the diagram below.



2. Start with a set of `<TABLE>` tags to begin the layout of the page including attributes for
  - **BORDER = 0** (This will eliminate any border areas around the outside of the page )
  - **CELLSPACING = 0** (This attribute will set spaces between columns.)
  - **CELLPADDING = 0** (This attribute will widen the lines between columns.)
  - By setting these attributes to zero, we can be assured that our pixel measurements will not be rendered inaccurate because we have placed artificial space between columns or around the outside of the page.
  - **WIDTH = 600** (This attribute establishes the width of the table as 780 pixels.)
  - **<TABLE BORDER = 0 CELLSPACING = 0 CELLPADDING = 0 WIDTH = 600>**

## Web Design

53

### 3. Create the **Header Row**

- `<TABLE BORDER = 0 CELLSPACING = 0 CELLPADDING = 0 WIDTH = 780>`

```
<TR>
```

```
<TD BGCOLOR = "SILVER" HEIGHT = 50 COLSPAN = 6>
```

- We have designated the header row as having a silver background, a height of 50 pixels, and a span of six columns (Sidebar + Main Text + 4 Gutters).

### 4. Now we can place a heading inside of the first row ( our header area ).

- `<TABLE BORDER = 0 CELLSPACING = 0 CELLPADDING = 0 WIDTH = 600>`

```
<TR>
```

```
<TD BGCOLOR = "SILVER" HEIGHT = 50 COLSPAN = 6>
```

```
<H1 FONT = "RED" ALIGN = CENTER>This is the Heading
```

```
Area</H1>
```

```
</TD>
```

### 5. Now that the header area is set up, we need to set up the second row that will contain the six columns.

- `<TABLE BORDER = 0 CELLSPACING = 0 CELLPADDING = 0 WIDTH = 600>`

```
<TR>
```

```
<TD BGCOLOR = "SILVER" HEIGHT = 50 COLSPAN = 6>
```

```
<H1 FONT = "RED" ALIGN = CENTER>This is the Heading
```

```
Area</H1>
```

```
</TD>
```

```
<TR>
```

```
<TD BGCOLOR = "YELLOW" WIDTH = 10 HEIGHT = 600></TD>
```

```
<TD BGCOLOR = "WHITE" WIDTH = 150 VALIGN = TOP>
```

```
<P>This text is left aligned inside the second column<P>
```

```
<P><CENTER>This text is centered<P></TD>
```

```
<TD BGCOLOR = "YELLOW" WIDTH = 10 ></TD>
```

```
<TD BGCOLOR = "GREEN" WIDTH = 20 ></TD>
```

```
<TD BGCOLOR = "WHITE" WIDTH = 390 VALIGN = TOP>
```

```
<H2 ALIGN = CENTER>This is the Main Text Area</H2>
```

```
<P><P>
```

```

```

I have inserted my lion picture to illustrate how, even though I have `<B>`aligned it left`</B>` it is left aligned inside this particular table column. Any attributes set within this `<B><I>TD</I></B>` tag have no effect on any of the other columns.

```
<P>
```

Notice that we have designated the side-bar area as a white background with its surrounding gutters colored YELLOW. Meanwhile, the main text area also has a background of white with its gutters colored `<FONT COLOR = "GREEN">GREEN</FONT COLOR = "BLACK">`. `<P>`

```
Very extremely cool!!!</TD>
```

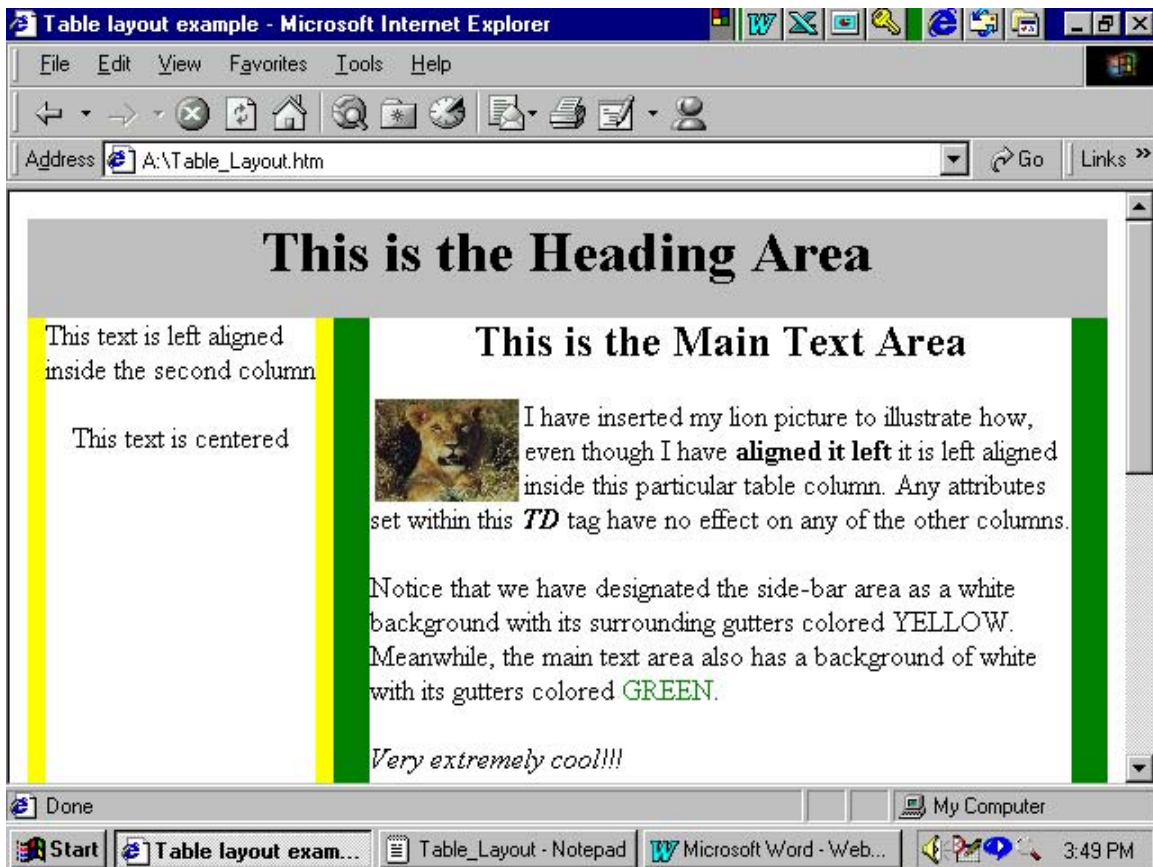
```
<TD BGCOLOR = "GREEN" WIDTH = 20 HEIGHT = 600></TD>
```

# Web Design

54

## 6. Items of importance from the preceding code

- Use the **VALIGN = TOP** attribute in any column in which text will be added. This will ensure that any text added to the column will start at the top of the column instead of the middle. This is because technically, we are dealing with a cell and HTML will always want to center data within a cell.
- When setting up a series of columns next to each other (as with the six columns underneath the header row), use the **HEIGHT** attribute in the **first column only** in order to set the height of the entire row. (**HEIGHT = 600**)
- Notice that the **total** of all six column **widths** equal the **overall width** (600 pixels).
- A picture of the resulting page appears on the next page. Notice how the text butts against the gutter columns. If we changed the **gutter background color to white**, we would then create a 30 pixel barrier between text in the side-bar and text in the main text area.

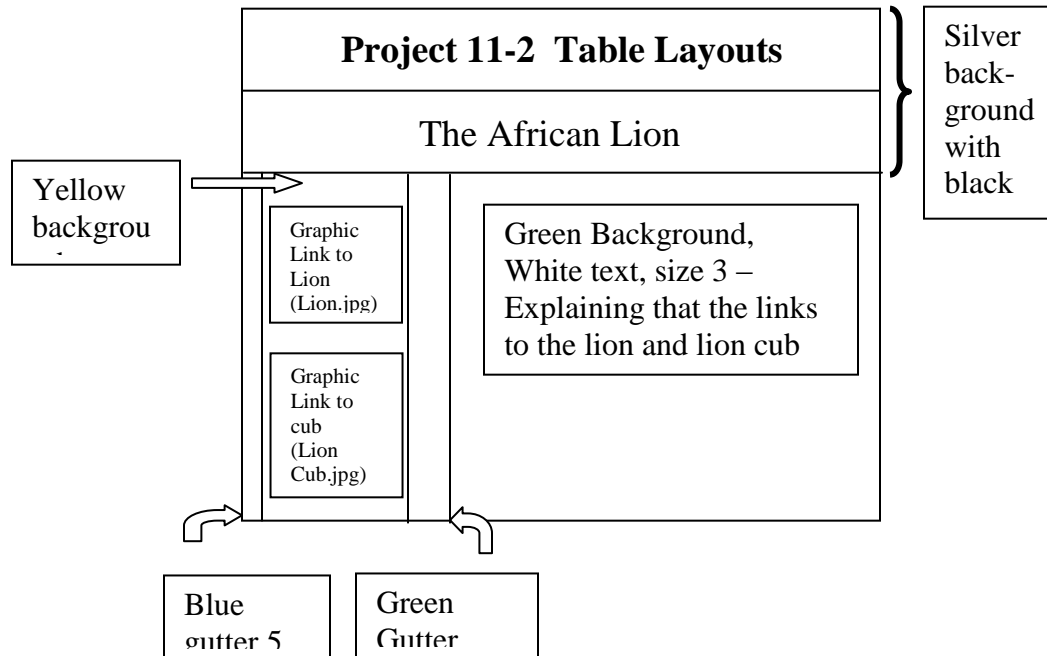


# Web Design

55

## Exercise 11-2 Creating a Page Layout Using a Table

Using the *Lion.jpg* and the *Lion Cub.jpg* files on the network, create a web page as it appears in the diagram below. Utilize a table to set up the various areas of the page.



### **Using a Graphic as a Background**

Instead of defining a specific color for the background of your web page, you may want to specify a graphic to appear as the background. There are several issues to keep in mind.

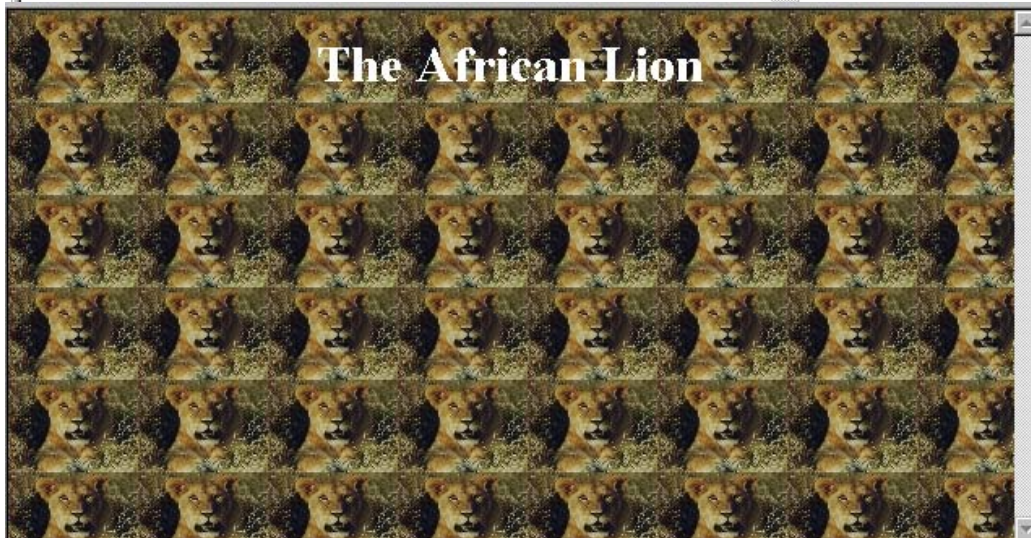
1. The browser will **tile** the graphic in order to cover the entire page area. Therefore, you should choose a graphic that lends itself well to being seamed together with copies of itself.
  - If the graphic does not cover the entire page, the browser will butt copies of the same graphic together until it does cover the page.
2. Text and link colors, etc. must be chosen carefully in order to be readily visible when the page is viewed.
3. Use the **BACKGROUND** attribute inside the <BODY> tag and designate the graphic.
  - <BODY BACKGROUND = "Lion.Jpg">

# Web Design

56

4. The following code will yield the page below it

```
Background - Notepad
File Edit Search Help
<HTML>
<HEAD><TITLE>Background Graphic</TITLE></HEAD>
<BODY BACKGROUND = "LION.JPG">
<H1 ALIGN = CENTER>The African
Lion</H1>
</BODY>
</HTML>
```



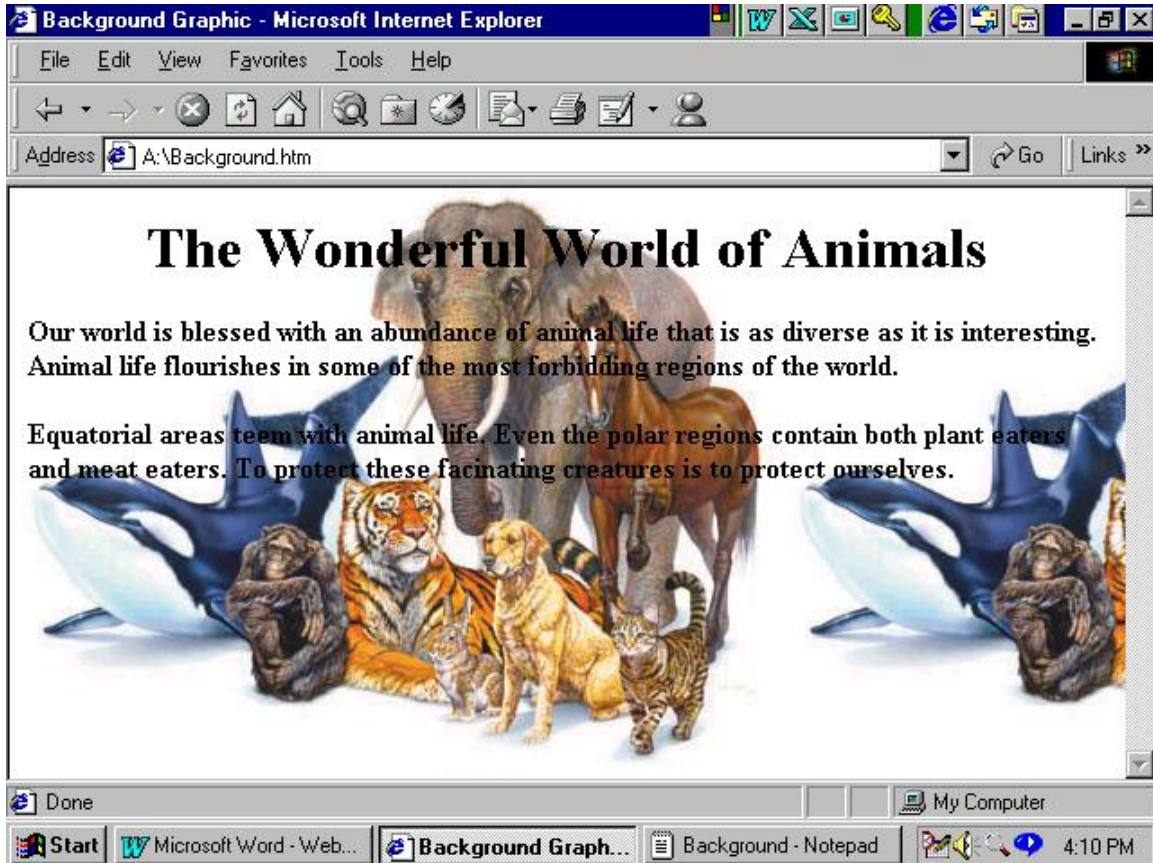
Note the **tiling** of the graphic.

# Web Design

57

## Exercise 12-1 Using Background Graphics

Create a web page using the *Animals.Jpg* file copied to your diskette from the network and used in Exercise 10-1 to create the following web page:



## Adding Sound Files to the Web Page

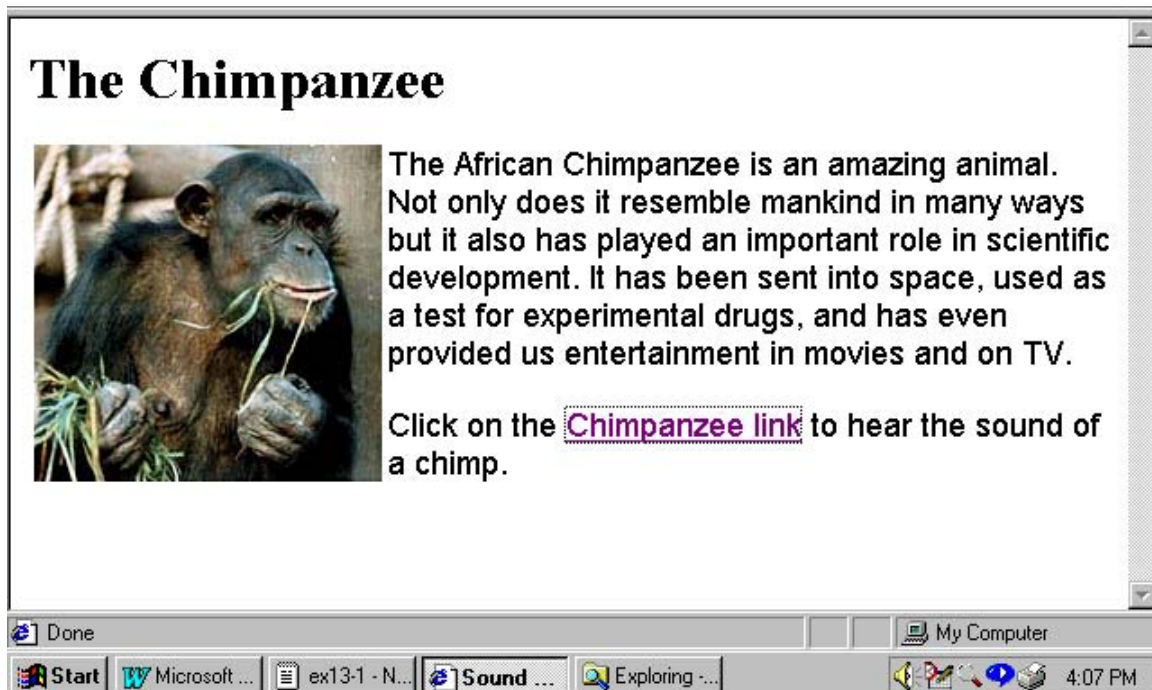
Not only can you insert graphic objects to a web page, but you can also add sound. There are two ways that this can be done.

### Inserting a Link to a Sound File

1. Save the desired sound file to the same folder as the other web site files.
  - Normally, this will be a **.WAV** file
2. In the code, do the following:
  - Add a **<A>** tag
  - Inside the **<A>** tag, add the **HREF** attribute pointing to the sound file, just like you would a web page.
    - **<A HREF = "SOUND.WAV">**
    - Add the **text link** to be printed on the page.
      - **<A HREF = "SOUND.WAV">Click here to hear the sound**
    - Close the **<A>** tag
      - **<A HREF = "SOUND.WAV">Click here to hear the sound</A>**

### Exercise 13-1 Creating a Link to a Sound File

Create the web page as it appears below. Your instructor will tell you where to get the graphic and sound files.



### Embedding a Sound File

When you **embed** a sound file into a page, the browser will automatically open the audio control software to play the sound when the page is opened. **Your sound playing software must be configured to work with your browser in order for this to work.**

Instructions to embed a sound file appear below:

1. Save the sound file to the same folder as the rest of your web site files.
2. In the code, include the following
  - **<EMBED SRC = "SOUND.WAV">**
3. The browser will display the sound player and controls necessary to play the sound.

### Exercise 13-2 Using EMBED to Play a Sound File

Use the web page created in Exercise 13-1 and embed the Chimpanzee sound file in this page.

**Note:** If you have a problem playing the sound file, adjustments may need to be made to your browser.

## What is JavaScript?

JavaScript is a programming language used to add additional functions to web pages. With it, the programmer can include many graphic and text effects to enhance his/her HTML coded web pages.

## How To Start a JavaScript

All JavaScript commands are enclosed within the `<SCRIPT>` tag and the `</SCRIPT>` tag. In this way, the browser “knows” to run the code between as JavaScript code **if it is able**. Not all browsers are capable of running JavaScript. This is especially true of the older browsers such as Internet Explorer 3.0 or earlier versions. If you have Internet Explorer version 5.0 or later, you can feel safe in running JavaScript code.

### *Example of a Simple JavaScript*

```
<SCRIPT LANGUAGE=JAVASCRIPT TYPE = “TEXT/JAVASCRIPT”>

 document.write(“This is a simple JavaScript example.”)

</SCRIPT>
```

### Parts of the Script Tag

1. **LANGUAGE** – Tells the browser that the language being used in the script is JavaScript.
2. **TYPE** – Tells the browser that the particular type of JavaScript being used is the TEXT version.

### *Parts of the JavaScript Language*

1. **Objects**- Things that are JavaScript can be used to manipulate or enhance. Some examples are pictures, buttons, forms, check boxes, and many more.
2. **Properties** – Aspects of objects that can be controlled or enhanced by JavaScript code. Some examples are form, name, type, value and many more. It should be noted that **different objects have different properties** depending upon their functions.
3. **Methods** – Different things that the object is expected to do. Some examples are `click()`, `selected()`, `write()`, and many more. Again, depending upon the object, it will have its own set of properties which, in turn, have their own set of methods associated with them.
4. **Event Handlers** – Actions performed by the user. For example, the user may be expected to click on a radio button or a check box. In this example, the event handler used would be the **onclick** event.

By referring to our example above, we can see that the **document** object’s **write** method was set to **“This is a simple JavaScript example.”**. There was no property involved in

# Web Design

61

this example. The write method was directly linked to the document object with “**dot notation**”.

## Value Types

Type	Description	Example
1. Number	Has a numeric value	3.72
2. String	A series of text characters	“My Web Site”
3. Boolean	A condition is either true or false	true
4. Null	Empty and meaningless	
5. Object	Any value associated with the object	
6. Function	Value returned by a function	

## Operators

Operator	Description	Example
1. + (Numeric)	Adds two values together	$X + Y$
2. + (String)	Adds two strings together	book + mark
3. -	Subtracts two numbers	$X - Y$
4. *	Multiplies two numbers	$X * Y$
5. /	Divides two numbers	$X / Y$
6. %	Modulus division (the result returned is the <b>remainder</b> of the division of the two numbers)	$X \% Y$
7. X++, ++X	Adds one to the value of the numeric variable ( $X = X + 1$ )	X++ or ++X
8. X--, --X	Subtracts one from the value of the variable ( $X = X - 1$ )	--X or X--
9. -X	Reverses the sign of the variable	-X

## Assignments

Assignment	Description	Example
1. =	Sets variable equal to a value	$X = Y$
2. +=	Sets variable equal to its current value plus another value ( $X = X + Y$ )	$X = +Y$
3. -=	Sets variable equal to its current value minus another value ( $X = X - Y$ )	$X -= Y$
4. *=	( $X = X * Y$ )	$X *= Y$
5. /=	( $X = X / Y$ )	$X /= Y$
6. %=	( $X = X \% Y$ )	$X \% = Y$

## Comparisons

Comparison	Description	Example
1. ==	Returns <b>true</b> if X and Y are equal	$X = Y$

## Web Design

62

- |       |                                                        |            |
|-------|--------------------------------------------------------|------------|
| 2. != | Returns <b>true</b> if X and Y are not equal           | X != Y     |
| 3. >  | Returns <b>true</b> if X is greater than Y             | X > Y      |
| 4. >= | Returns <b>true</b> if X is greater than or equal to Y | X >= Y     |
| 5. <  | Returns <b>true</b> if X is less than Y                | X < Y      |
| 6. <= | Returns <b>true</b> if X is less than or equal to Y    | X <= Y     |
| 7. && | Returns true if both conditions are true               | X>Y && A<B |
| 8.    | Returns <b>true</b> if either condition is true        | X>Y    A<B |
| 9. !  | Returns <b>true</b> if the condition is false          | !X >Y      |

### Where Do the Scripts Go?

JavaScript code can be placed either in the **header** (referred to as a **header script**) or the **body** (referred to as a **body script**).

### Hiding Scripts From Older Browsers

As was mentioned before, older browsers may have a problem with JavaScript scripts. For this reason, it is sometimes advisable to hide scripts from these older browsers. An example appears below:

```
<SCRIPT LANGUAGE=JAVASCRIPT TYPE="TEXT/JAVASCRIPT">

 <!-- - Hide script from old browsers

 document.write("This is a simple JavaScript example")

 // End hiding script from old browsers - ->

</SCRIPT>
```

1. <!-- - This signals the opening of an HTML comment
2. // This signals the beginning of a **single-line** JavaScript comment
3. - - > This signals the end of an HTML comment

By using this technique, an older browser will just assume the “document.write(...” section is merely an HTML comment and will not attempt to process the script. On the other hand, a browser that can handle JavaScript code will go ahead and process the script anyway.

## Multiple-Line JavaScript Comments

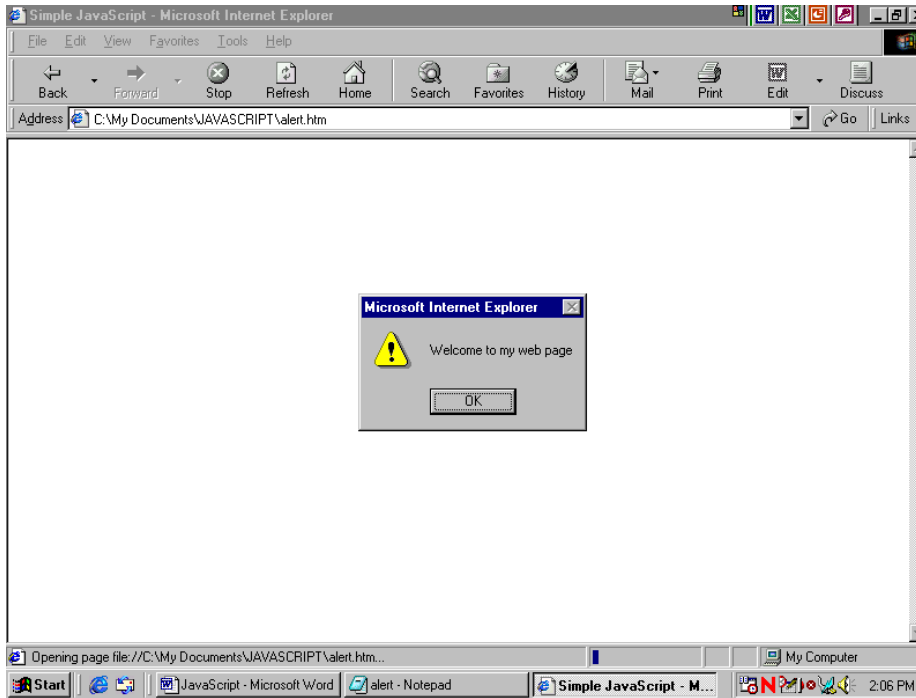
Any JavaScript comments that span more than one line of text are started with the `/*` designation and are ended with the `*/`.

Example:

**`/* This is an example of a JavaScript comment that spans more than one line of text. Notice how the comment is ended here. */`**

## User Alerts

Sometimes, you want the user to be shown an alert message. For instance, maybe the browser the user has is too old to process scripts. Or, perhaps you would just like the user to be alerted to something.



- The script that produced the page to the left appears below.
- Notice the **OK** button. The alert message is cleared when this button is clicked.

```
<HTML>
<HEAD><TITLE>Simple JavaScript</TITLE></HEAD>
<BODY>
 <SCRIPT LANGUAGE=JAVASCRIPT TYPE="TEXT/JAVASCRIPT">
 <!-- Hide script from old browsers
 alert("Welcome to my web page")
 // End hiding script from old browsers -->
 </SCRIPT>
</BODY> </HTML>
```

## USING <NOSCRIPT>

The <NOSCRIPT> tag relegates operation of its contents to only those browsers that cannot handle JavaScript code.

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

 <SCRIPT LANGUAGE=JAVASCRIPT TYPE="TEXT/JAVASCRIPT">
 <!-- Hide script from old browsers
 alert("Welcome to my web page")
 // End hiding script from old browsers -->
 </SCRIPT>
</HEAD>
<BODY>
 <NOSCRIPT>
 This browser cannot handle JavaScript.
 </NOSCRIPT>
</BODY>
</HTML>
```

In the example above, we have done two things.

1. The script has been moved to the Header
2. In the body, the NOSCRIPT tag has been used to print the message “This browser cannot handle JavaScript” **if the browser is too old or JavaScript has been turned off.**

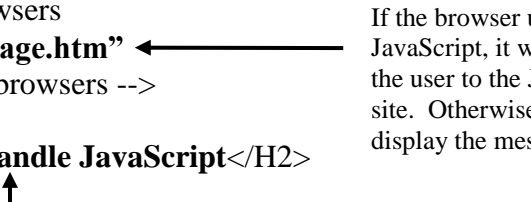
## Redirecting the User

There may be some instances where a set of circumstances may dictate sending the user to a different page or site. For instance, the user’s browser may not be able to handle JavaScript code and should be sent to a site consisting of HTML code only.

In the following code, the user opens a page containing some JavaScript code. If the user’s browser can handle the scripts, he/she is sent to a different site, otherwise, a message is printed in plain HTML code.

```
<HTML>
<HEAD><TITLE>Simple JavaScript</TITLE></HEAD>
<BODY>
 <SCRIPT LANGUAGE=JAVASCRIPT TYPE="TEXT/JAVASCRIPT">
 <!-- Hide script from old browsers
 window.location = "Javapage.htm" ←—————
 // End hiding script from old browsers -->
 </SCRIPT>
 <H2> Your browser cannot handle JavaScript</H2>
</BODY></HTML>
```

If the browser understands JavaScript, it will redirect the user to the JAVAPAGE site. Otherwise, it will display the message.



## **Redirecting Using a Link**

In the previous example, the user was automatically redirected to a page or site that is suited to his/her browser capabilities. Perhaps you would like the user to see the initial page and then, after clicking on a link, be directed to the appropriate site.

Let's change the code for the initial page:

```
<HTML>

<HEAD><TITLE> Redirect Using a Link</TITLE></HEAD>
<BODY BGCOLOR = "SILVER">

<H1 ALIGN = CENTER><A HREF = "NOJAVA.HTM"
 onClick = "window.location='JAVAPAGE.HTM'
 return false">Click here to be directed to the appropriate site</H1>

</BODY></HTML>
```

Here, the script is embedded within the anchor tag for the page designed for non-JavaScript browsers ( <A HREF = "NOJAVA.HTM" )

**onClick** = " This is the beginning of the JavaScript. Everything within the double quotation marks will be executed when the link is clicked.

**window.location=** Using dot notation, this command points to the site or page to be opened when the **click event** is executed.

**'JAVAPAGE.HTM'** This is the name of the page to be opened when the click event takes place. Notice its enclosure within **single quotation marks** or apostrophes. This is to fulfill its need to be quoted without prematurely ending the script's quotation sequence.

**return false"** This command **stops processing the user's click** to be sure the HTML coded site (NOJAVA.HTM) **does not execute**. The ending quotation mark ends the script.

## Detecting Browsers

In order to better customize web pages for particular browsers (given the fact that some browsers handle certain JavaScript codes better than others), you may want to be able to redirect the user to different sites based upon his/her browser.

Below is an example of code designed to test for “Netscape Navigator”:

```
<HTML><HEAD><TITLE>Browser Check</TITLE></HEAD>

<BODY BGCOLOR = “GREEN”>

<H2>
 <SCRIPT LANGUAGE=JAVASCRIPT TYPE=”TEXT/JAVASCRIPT”>
 <!-- -Hide Script from old browsers

 if(navigator.appname == “Netscape”){
 document.write(“You’re running Netscape Navigator.”)
 }
 else{
 document.write(“You are not running Netscape Navigator”)
 }

 // End hiding script from old browsers - ->
 </SCRIPT>

</H2></BODY></HTML>
```

**if(** The **if** conditional checks to see if the expression inside the parenthesis is true (the name of the browser is “Netscape”).

**navigator.appname == “Netscape”** Again, using dot notation, we use the **appname property** of the **navigator object** within the **if conditional** to see if the appname property is indeed “Netscape”

{ The first set of brackets signals the beginning of the actions to take place if the conditional is true. In this case, we will use the **write method** of the **document object** to display the message “You are running Netscape Navigator”.

} The first closing bracket ends the actions if the conditional is true.

**else {** Here, we begin the set of actions to take place if the conditional is not true. In this case, we use the write method of the document object to print “You are not running Netscape Navigator”

} The second closing bracket signals the end of the not-true conditional

## Detecting Plug-ins

Plug-ins are pieces of software that enhance a web page's appearance or functionality. Some plug-ins allow the site to show videos. Some add the ability to play sound files. It is nice to be able to check for specific plug-ins in the user's browser that are needed to operate certain aspects of your site.

In the example below, we will write the code to check for the QuickTime plug-in from Apple. If it is loaded, we will play a movie called "Campfire.mov".

```
<HTML><HEAD><TITLE>Check for QuickTime</TITLE></HEAD>

<BODY BGCOLOR="BLUE">

<H2 ALIGN = CENTER>The Campfire Video

 <SCRIPT LANGUAGE=JAVASCRIPT
 TYPE=TEXT/JAVASCRIPT">
 <!-- Hide script from old browsers

 if(navigator.plugins["QuickTime Plug-in 2.0"]) {
 document.write("<embed src='campfire.mov' width=320 height=250
 loop=false autoplay=true>")
 }
 else {
 document.write("<img src='firextngsh.gif'
 width=320 height=250>")
 }

 //End hiding script from old browsers - ->
 </SCRIPT></H2>
</BODY></HTML>
```

The example above only works with Netscape Navigator. However, the general logic can be seen. The only additional object and method combination is the **navigator** object and its **plugins** method (again, only applicable to Netscape Navigator). Indeed, the QuickTime movie will not be displayed at all with Internet Explorer.

Internet Explorer will be able to display movies if Microsoft's ActiveX plug-in has been installed. In this case, ActiveX will automatically install the plug-in that is needed.

## Loops

Loops are used to repeat instructions within scripts several times. For instance, to print the numbers 0 through 20 followed by a message, we could use the code as it appears below.

```
<HTML>
<HEAD><TITLE>Example of Loops</TITLE></HEAD>

<BODY>

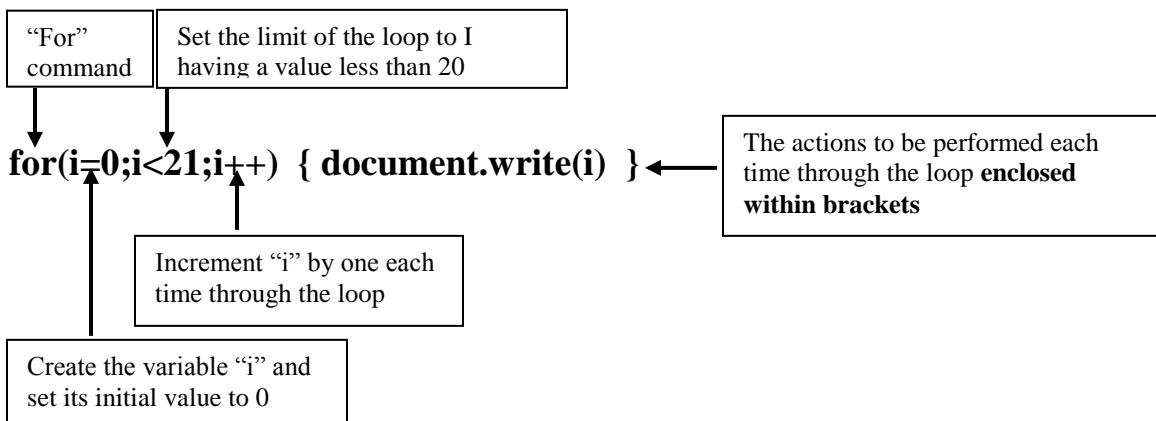
<SCRIPT LANGUAGE=JAVASCRIPT TYPE="TEXT/JAVASCRIPT">
 <!--Hide scripts from old browsers

 for(i=0;i<21;i++) {
 document.write(i) }

 //End hiding scripts from old browsers-->

</SCRIPT>
<H1><CENTER>Finished with the loop </CENTER></H1>
</BODY></HTML>
```

The highlighted area inside the script tags is the **“for loop”** and is broken down in this manner:



## If Logical Test

It will become important to test for conditions within the code. One way to do this is with the "If" test. Let's say we want to only print those numbers greater than 15.

```
<HTML>
<HEAD><TITLE>Example of Loops</TITLE></HEAD>

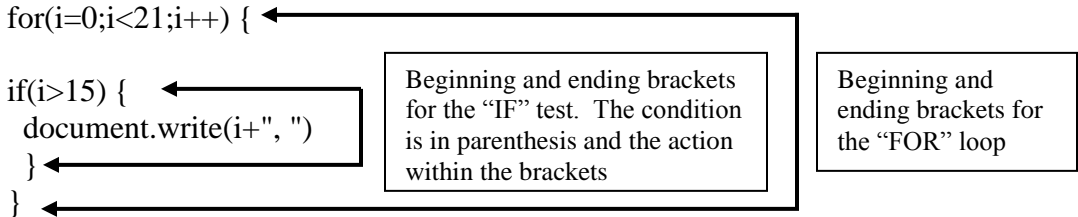
<BODY>

<SCRIPT LANGUAGE=JAVASCRIPT TYPE="TEXT/JAVASCRIPT">
 <!--Hide scripts from old browsers

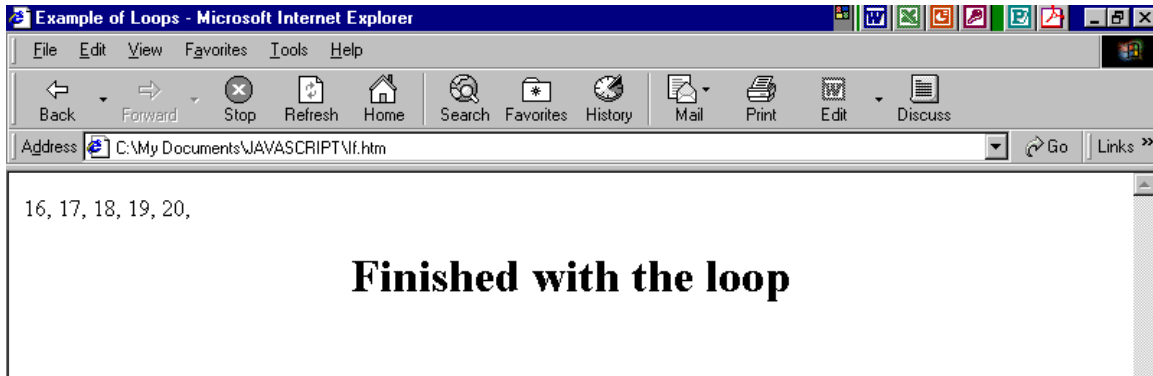
 for(i=0;i<21;i++) {
 if(i>15) {
 document.write(i+", ")
 }
 }

 //End hiding scripts from old browsers-->

</SCRIPT>
<H1><CENTER>Finished with the loop </CENTER></H1>
</BODY></HTML>
```



This code will result in the following browser response.



## Adding an ELSE Clause

Suppose we want something to happen if the test for the value of “i” returns false. In other words, its value is **not** greater than 15.

```
<HTML>
<HEAD><TITLE>Example of Loops</TITLE></HEAD>

<BODY>

<SCRIPT LANGUAGE=JAVASCRIPT TYPE="TEXT/JAVASCRIPT">
 <!--Hide scripts from old browsers

 total=0

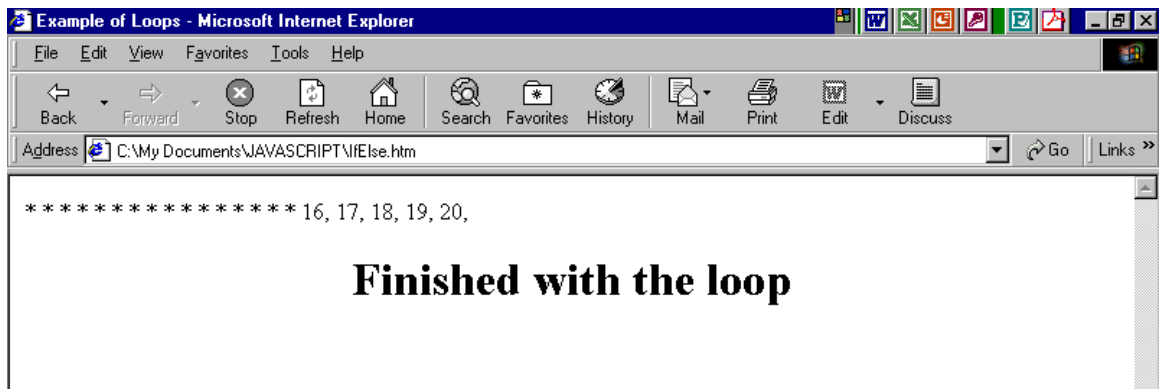
 for(i=0;i<21;i++) {

 if(i>15) {
 document.write(i+", ")
 }
 else {
 document.write("* ")
 }
 }

 //End hiding scripts from old browsers-->

</SCRIPT>
<H1><CENTER>Finished with the loop </CENTER></H1>
</BODY></HTML>
```

Here we have an example of the **ELSE** clause instructing the browser to insert a “\*” and a space if the **IF** test returns FALSE. The action to be taken is enclosed in its own set of brackets after the “else” statement.



## Functions, Forms, and Buttons

Functions are small “mini-scripts” that are called by a main script to perform a task. Data can be passed from the main script to a function as well. Below is an example of the use of a **function** in a **header script** and **command buttons** in a **body script**. Also note the introduction of the **FORM** object.

```
<HTML><HEAD><TITLE>Functions and Buttons</TITLE>
```

```
<SCRIPT LANGUAGE=JAVASCRIPT
TYPE="TEXT/JAVASCRIPT">
```

```
<!--Hide scripts from old browsers
```

```
function myMessage(message){
 alert(message)
}
```

The function is named “myMessage” and will initialize data from the calling statement as the variable “message”. It will then create an Alert Message including the data sent from the calling statement stored in “message”

```
//End hiding scripts from old browsers-->
```

```
</SCRIPT>
```

```
</HEAD>
```

```
<BODY BGCOLOR="Yellow">
```

```
<H2>Famous Presidential Quotes</H2>
```

```
<HR>
```

```
<FORM>
```

```
<INPUT TYPE=BUTTON VALUE="Lincoln" onClick="myMessage('Four score
and seven years ago...')">
```

```
<INPUT TYPE=BUTTON VALUE="Kennedy" onClick="myMessage('Ask not
what your country can do for you...')">
```

```
<INPUT TYPE=BUTTON VALUE="Nixon" onClick="myMessage('I am not a
crook!')">
```

```
</FORM>
```

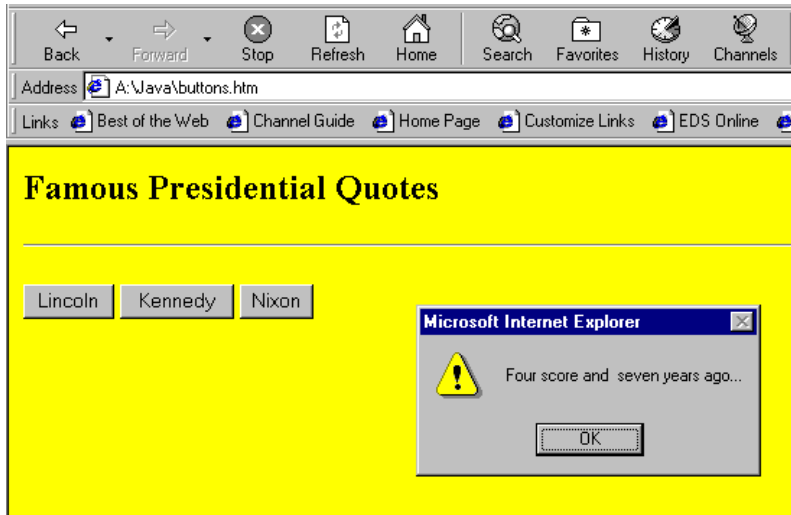
```
</BODY></HTML>
```

The command buttons are enclosed inside the form tags. A FORM is a set of objects designed to have the user enter information. In this case, it is to click a button. Within the form are a series of INPUT objects, each with its TYPE (button) and VALUE (label) properties set. The onClick EVENT HANDLER calls the “myMessage” function, passing a specific text to the function

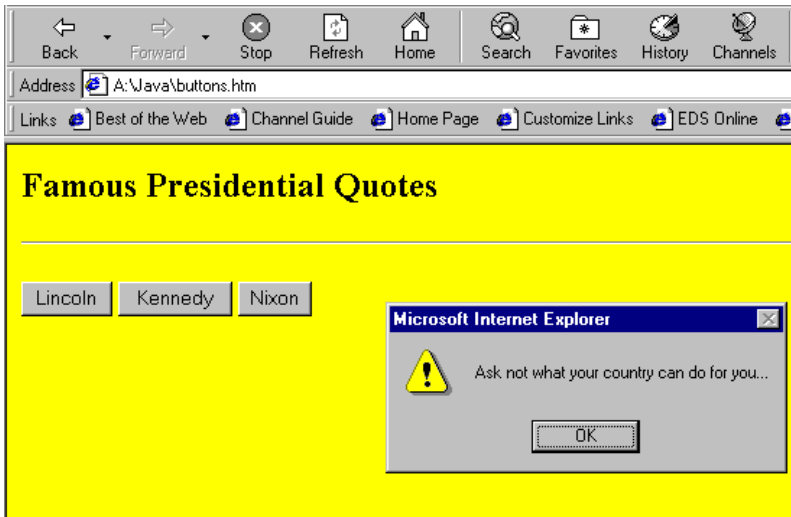
# Web Design

72

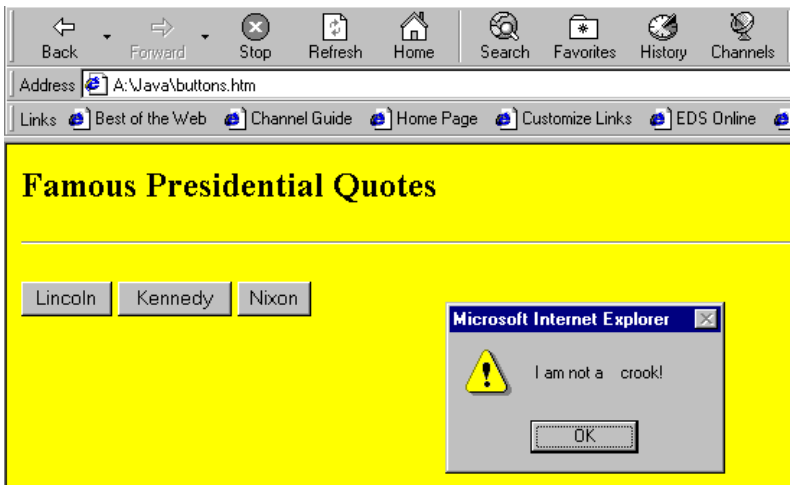
The results of the code on the previous page appears below:



This message appears when the “Lincoln” button is clicked.



This message appears when the “Kennedy” button is clicked



This message appears when the “Nixon” button is clicked.

## Graphics

### Mouse Rollovers

There may be instances where you would like to have a graphic change in some way when the mouse pointer rolls over it. In this example, we would like to create a document consisting of a face that faces to the right unless the mouse pointer is located on it. The process begins within the **header script**.

```

<HTML><HEAD><TITLE>Hard Drive Image Swap</TITLE>
<SCRIPT LANGUAGE=JAVASCRIPT TYPE="TEXT/JAVASCRIPT">

<!--Hide Script from Old Browsers
if(document.images){
 right = new Image
 left = new Image

 right.src = "right.gif"
 left.src = "left.gif"
}
else {
 left = ""
 right = ""
 document.face = ""
}
//End hiding script from old browsers -->

</SCRIPT>

</HEAD>

<BODY>
<A HREF = "rollover2.htm"
 onMouseover="document.face.src=left.src"
 onMouseout="document.face.src=right.src">
<IMG src = "right.gif"
 Width = 200 Height = 200 Border = 0
 NAME = "face">
</BODY></HTML>

```

**We check to see if the browser can handle Java Scripts inside the <HEAD> tag.**

**If the browser can handle scripts, we initialize "right" and "left" as new images.**

**Next, we designate the source graphic for both the "right" and "left" images.**

**In the event that the browser cannot handle scripts, we designate "right" and "left" as empty as well as the FACE of the document.**

**In the body of the document, we create a link using the <A HREF> tag. Within the tag, we designate the document linked to as "rollover2.htm".**

**Then, we designate the source file for the document face if the mouse is pointing to the link and if it is not pointing to the link.**

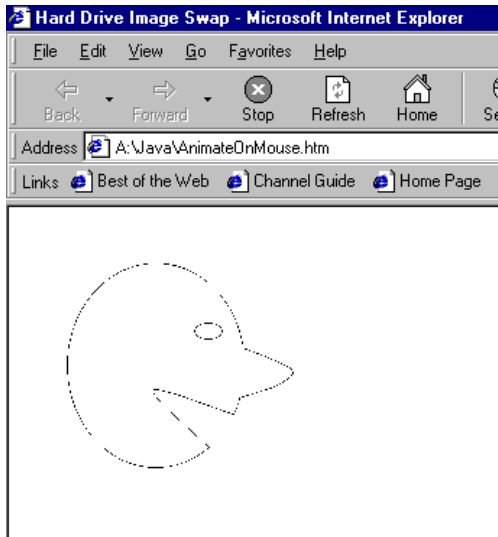
**Then, we designate the graphic to be displayed when the document opens.**

**Finally, we add the attributes we desire for the image.**

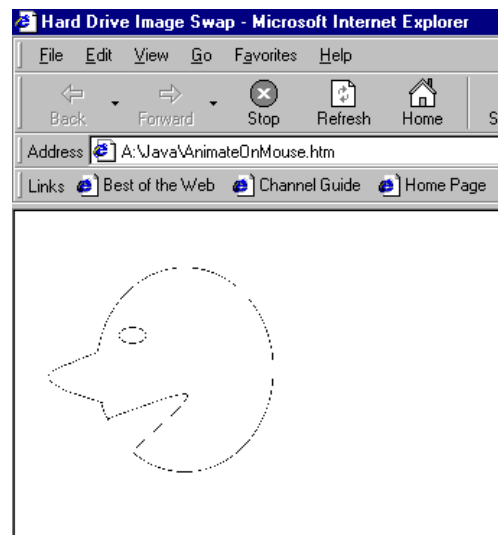
# Web Design

74

The results of the code on the previous page appears below:

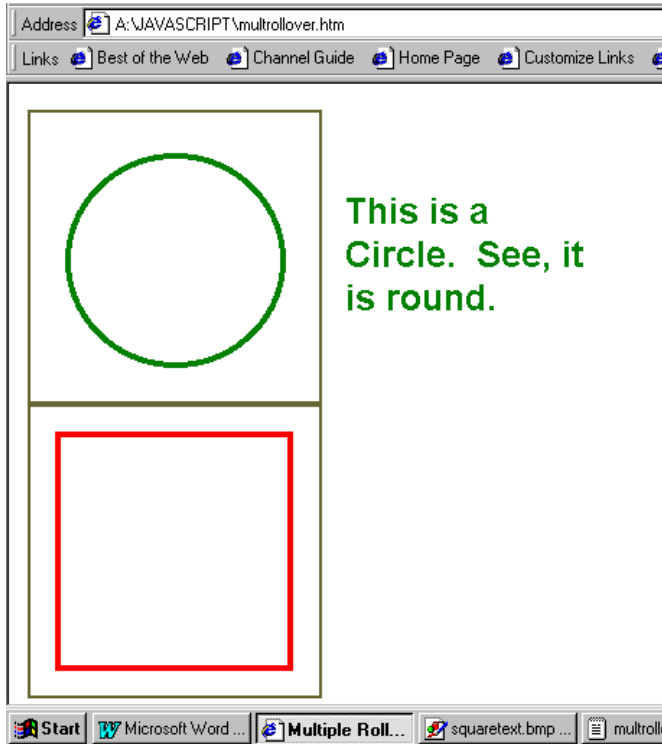


The document opens with the graphic link facing to the right. It will remain this way until the mouse pointer rests over top of it.

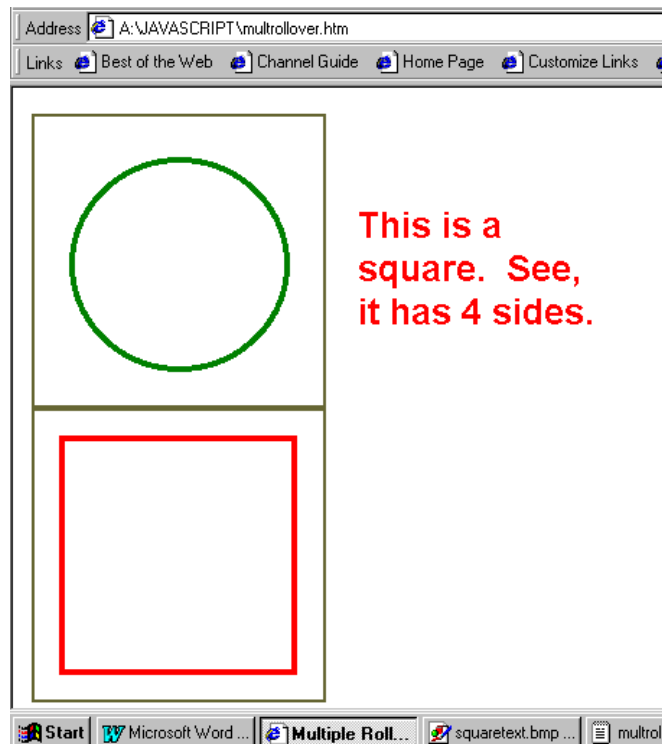


When the mouse pointer rests on the graphic link, it changes so that it is facing to the left. It will remain this way until the mouse pointer no longer rests over top of it, at which time, it will again face right. No click of the mouse is necessary.

## Multiple Image Rollover



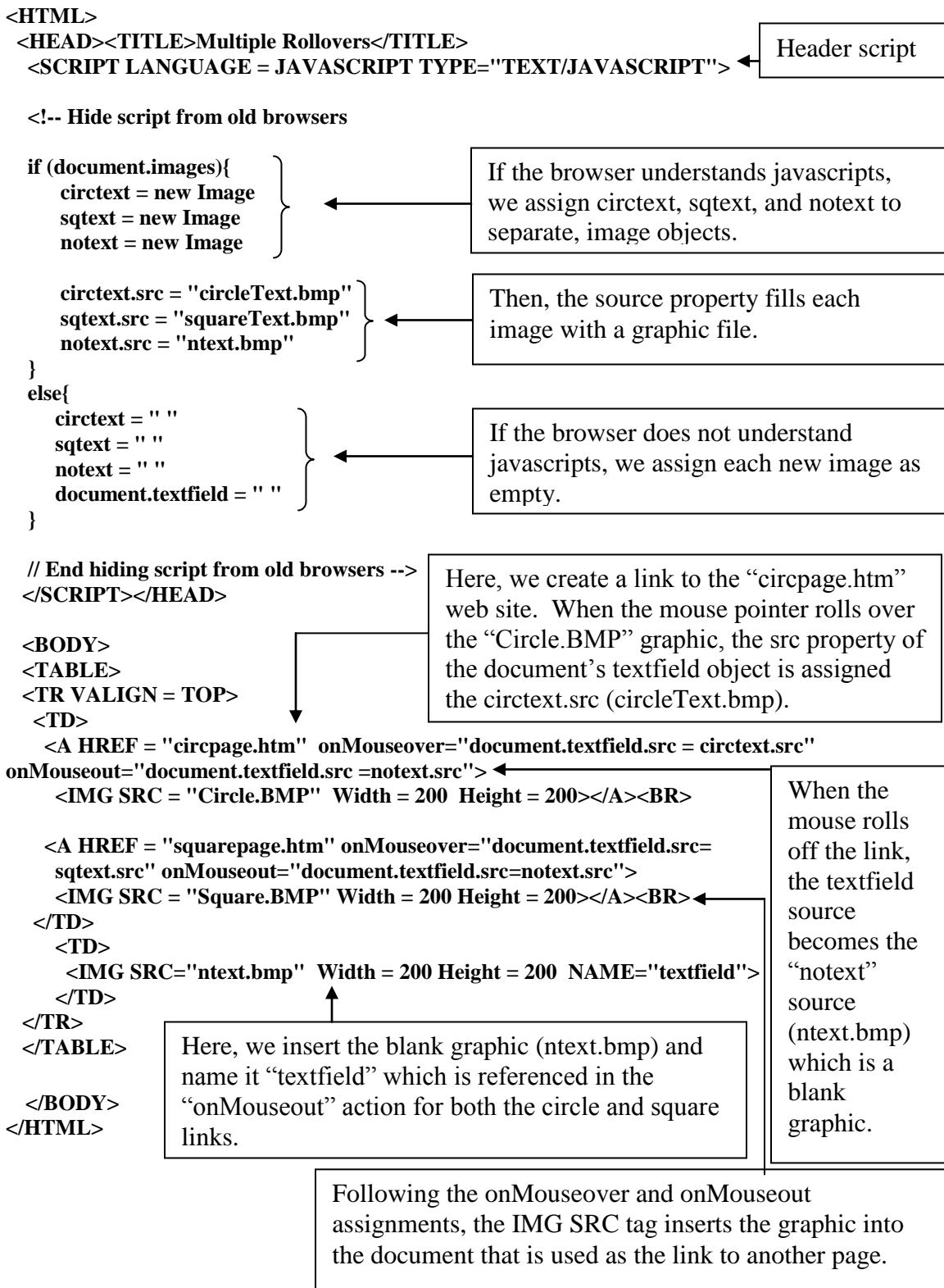
When the mouse pointer rolls over the circle picture, the text “This is a Circle. See, it is round.” appears. The circle picture is also a link to another page involving the circle.



When the mouse pointer rolls over the square picture, the text “This is a square. See, it has 4 sides.” appears. The square picture is also a link to another page involving the square.

# Web Design

76



## Frames

Frames are another method to divide web pages into separate working sections so that any one section can display contents independent of the other sections of the page. Unlike tables where changing the contents of one section of the table dictates changing the code for the page, frames allow one section to remain constant while another can display a totally different page.

In order to set up a frame, the user needs to provide the following:

1. A **frame set**, which divides the page into separate working sections.
2. **Child pages** that make up the contents of each of the sections created in the frame set.

### *Example of a Frame*

In this example, we will create a frame to provide a left column containing links to pages devoted to a month. The remainder of the frame will provide some information about that month. (In reality, this could be text describing activities that month or perhaps a table depicting the month in calendar form.)

#### The Frame Set

```
<HTML>
<HEAD><TITLE>Frame Set</TITLE>
<SCRIPT>
<!--Hide script from old browsers
/*Frame buster from http://www.usefulscripts.com*/
 if (self.parent.frames.length !=0){
 self.parent.location=document.location;}
// End hiding Script -->
</SCRIPT>
</HEAD>

<FRAMESET COLS= "30%,70%">
<FRAME SRC="leftstuff.htm" NAME="left" SCROLLING=AUTO>
<FRAME SRC="Jan.htm" NAME="content" SCROLLING=AUTO>
</FRAMESET>

</HTML>
```

Divide the page into columns comprising 30% and 70% of the page width.

Assign the pages to each of the newly created column sections. These are the HTML pages that will appear when the frame is loaded. It also names each of the columns. Further, scroll bars are provided for the frame if needed, otherwise, none will be present.

## Creating the Code For the Left Column

Here, we will create an array containing the various HTML pages we have created for each of the months. This will be done via a Java Script in the header. Then, we will access each page as an element of the array, placing each in the right section of the frame when its link is clicked in the left column.

```

<HTML>
<HEAD><TITLE>Stuff on Left</TITLE>

<SCRIPT LANGUAGE=JAVASCRIPT
 TYPE="TEXT/JAVASCRIPT">
 <!--Hide script from old browsers

pageArray = new Array("", "Jan.htm", "Feb.htm", "Mar.htm")

function setContent(thisPage){
 parent.content.document.location.href = pageArray[thisPage]
}

}

//End hiding script from old browsers -->
</SCRIPT>
</HEAD>

```

Create the array named "pageArray" and designate elements of the array as web pages. (Note: the "" signifies that there is no page assigned to element zero.)

Create a function called "setContent" that designates the referenced link to the parent page's "content" column document location as a member of the "pageArray" array.

```

<BODY BGCOLOR = "Blue" TEXT = "WHITE" LINK="RED"
VLINK="WHITE">

```

```

<H2> Navigation Bar</H2>

```

```

January

February

March


```

```

</BODY></HTML>

```

Set up the links to the month pages so that each link is referenced via the array as called by the "setContent" function.

# Web Design

79

## A Content Page

The code below is an example of one of the month pages. Again, it could just as easily contain a table representing the calendar as well as graphics. In essence, it is its own page within a page.

```
<HTML>
<HEAD><TITLE>January</TITLE>

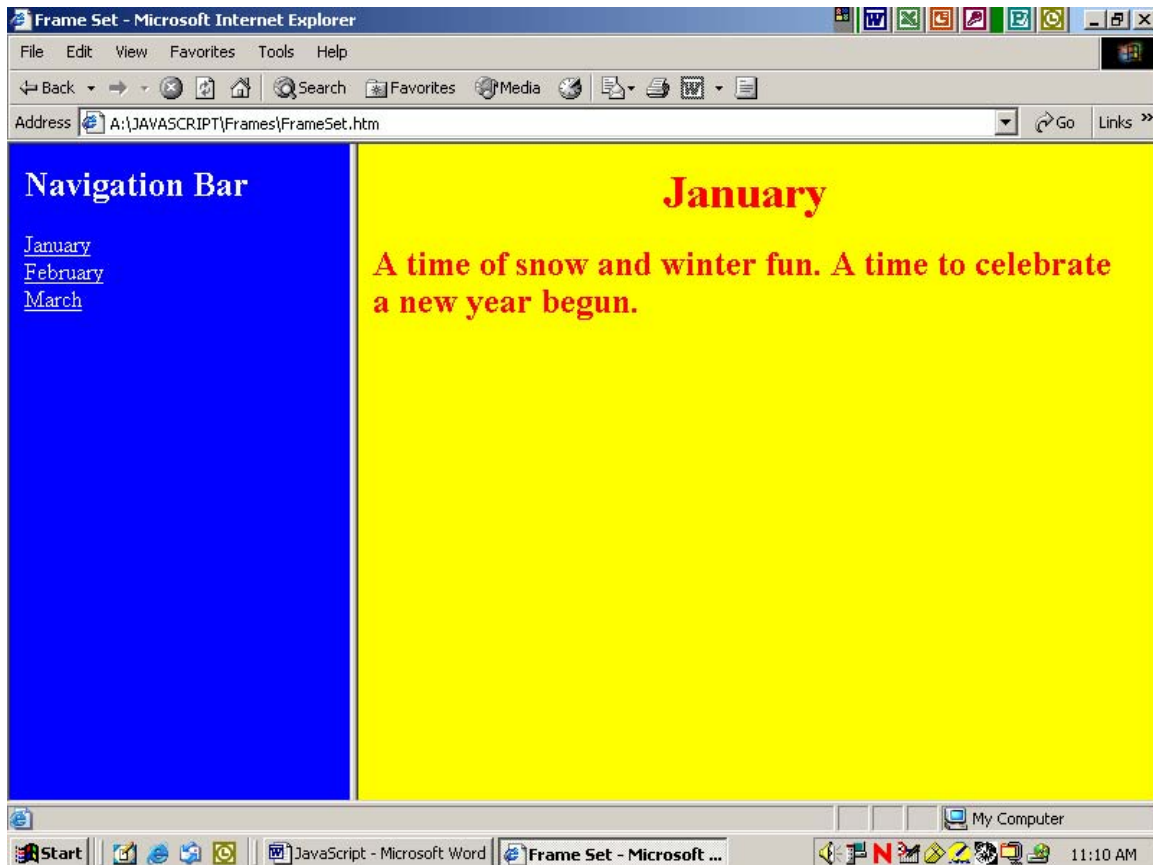
</HEAD>

<BODY BGCOLOR = "YELLOW" TEXT = "Red">

<CENTER><H1>January</H1></CENTER><P>
<H2>A time of snow and winter fun. A time to celebrate a new year begun.</H2>

</BODY></HTML>
```

## The Finished Product

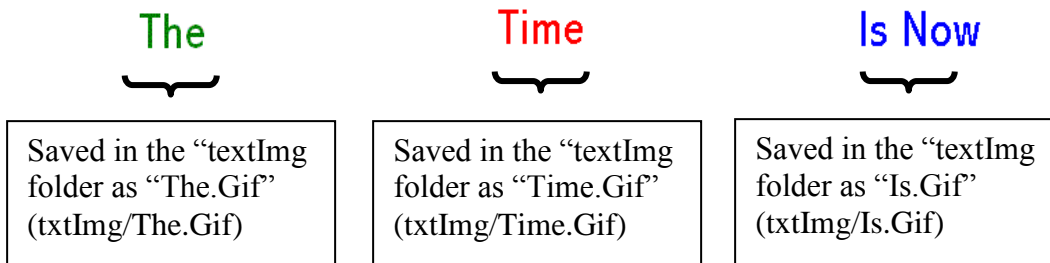


## Cycling Banners

You can create banners that cycle graphics in such a way as to convey a series of thoughts or to create animation.

### Add Images to an Array

In our case, we will create a series of images containing text using *Paint*:



```
<HTML>
<HEAD><TITLE>Cycling Banners</TITLE>
<SCRIPT LANGUAGE=JAVASCRIPT TYPE="TEXT/JAVASCRIPT">
<!-- Hide script from old browsers
textImages = new Array("textImg/The.Gif", "textImg/Time.Gif",
"textImg/Is.Gif")
thisImg = 2
imgCt = textImages.length
function rotate(){
 if(document.images){
 thisImg++
 if (thisImg==imgCt){
 thisImg = 0
 }
 document.cycleImages.src = textImages[thisImg]
 setTimeout("rotate()", 3 * 1000)
 }
}
//End hiding script from old browsers -->
</SCRIPT> </HEAD>
<BODY BGCOLOR = WHITE onLoad="rotate()">
<CENTER>

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

Set current image count to 2 so that, when opened, the function changes to image 0 (the first image).

The variable "imgCt" is set to the number of elements in the *textImages* array (in this case, 3).

## Create the Function to Change Images

```

<!-- Hide script from old browsers
textImages = new Array("textImg/The.Gif", "textImg/Time.Gif",
"textImg/Is.Gif")
thisImg = 2
imgCt = textImages.length
function rotate(){
 if(document.images){
 thisImg++
 if (thisImg==imgCt){
 thisImg = 0
 }
 document.cycleImages.src = textImages[thisImg]
 setTimeout("rotate()", 3 * 1000)
 }
}
//End hiding script from old browsers -->
</SCRIPT> </HEAD>

```

Name the function "rotate"

If the browser can handle images:

Increment the value of *thisImg* by 1.

If the value of *thisImg* = 3 then change it to 0.

The source of the cycling images is the *textImages* array.

The images will change every 3000 milliseconds (3 seconds).

## Finish the HTML Code to Use the Script

```

<BODY BGCOLOR = WHITE onLoad="rotate()">
<CENTER>


```

Start the banner by calling the *rotate* function in the BODY tag using the *onLoad* command.

Use the IMG SRC tag to designate the first image to appear (*The.Gif* located in the *textImg* folder). Assign the height and width of the image as usual.

Designate the name of the cycling image as referenced in the *rotate* function.

```

</CENTER> </BODY> </HTML>

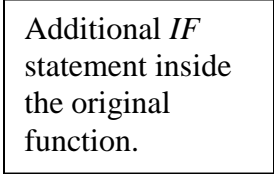
```

## ***Making the Banner Wait for Each Image to Fully Load Before Changing***

If your images are small, they will probably not take long to load. But if your images are large such as would be the case if they were pictures, they may not load before the time expires to start the next image. For instance, in our example above, we change the image every 3000 milliseconds or every 3 seconds. Large image files may not load in time.

To do this, add another IF statement inside the first IF statement in the *rotate* function.

```
function rotate(){
 if(document.images){
 if(document.cycleImages.complete){ ←
 thisImg++
 if (thisImg==imgCt){
 thisImg = 0
 }
 document.cycleImages.src = textImages[thisImg]
 } ←
 setTimeout("rotate()", 3 * 1000)
 }
}
```



Additional IF statement inside the original function.

## **Adding Links to Cycling Banners**

Perhaps you would like to make it possible for the user to link to another page or another site, depending upon which image is showing when he/she clicks on it. For instance, in our example, let's assume that we would like to link to the following pages based upon the image that is showing:

Image	URL
The.Gif	The.htm
Time.Gif	Time.htm
Is.Gif	Is.htm

```
<HTML>
<HEAD><TITLE>Cycling Banners</TITLE>
<SCRIPT LANGUAGE=JAVASCRIPT TYPE="TEXT/JAVASCRIPT">
<!-- Hide script from old browsers
textImages = new Array("textImg/The.Gif", "textImg/Time.Gif", "textImg/Is.Gif")
```

```
adURL = new Array("The.htm", "Time.htm", "Is.htm")
```

Create an array to hold the three pages to be linked to.

```
thisImg = 2
imgCt = textImages.length
function rotate(){
 if(document.images){
 if(document.cycleImages.complete){
 thisImg++
 if (thisImg==imgCt){
 thisImg = 0
 }
 document.cycleImages.src = textImages[thisImg]
 }
 setTimeout("rotate()", 3 * 1000)
 }
}
```

```
function newPage() {
 document.location.href = adURL[thisImg]
}
```

Create a function called *newPage* to change the current page as per the previously created array *adURL*.

```
//End hiding script from old browsers -->
</SCRIPT> </HEAD>
<BODY BGCOLOR = WHITE onLoad="rotate()">
<CENTER>
```

```

<IMG SRC="textImg/The.Gif" WIDTH=500 HEIGHT=300
 NAME="cycleImages">
</CENTER> </BODY> </HTML>
```

Make the IMG SRC tag included inside an HREF anchor that includes a reference to the *newPage* function.

### ***Exercise Using Banners***

Create a web page that includes a cycling banner somewhere in the page. Have the banner cycle between at least three (3) pictures. You can download these pictures to your local drive (preferably your diskette) from the Internet or you can create your own using a graphic application such as Paint.

Be sure to include the following items:

- Include code in your script to not allow the next image to load unless the previous one has finished loading.
- Practice changing the timing using the millisecond formula as per the notes in this section.
- Create a simple table to place your banner in a desired location on the page. To do this, you will need to place your JavaScript inside the appropriate cell in your table.





## ***Making the Slide Show a Wrap-Around***

Using the previous code, we are somewhat limited in our routine to change slides in that, if we are at the last slide and we click *Next>>*, we will not change slides because we are at the end of our show. Similarly, if we are at the first slide and click *<<Previous*, we will not change slides because we are at the first slide.

We would like to make our slide show move to the first slide if we click *Next>>* when at the last slide and we would like to show the last slide if we click *<<Previous* when at the first slide.

The code to accomplish this follows.

# Web Design

87

```
<HTML>
<HEAD><TITLE>Slide Show</TITLE>

<SCRIPT LANGUAGE = JAVASCRIPT TYPE = "TEXT/JAVASCRIPT">
<!-- Hide script from old browsers

myPics = new Array ("Pictures/Chimp.JPG", "Pictures/Lion.JPG",
 "Pictures/Lion Cub.Jpg")
curPic = 0
picCount = myPics.length - 1
function chngSlide(direction) {
 if (document.images) {
 curPic = curPic + direction
 if (curPic > picCount) {
 curPic = 0
 }
 if (curPic < 0) {
 curPic = picCount
 }
 document.thisPicture.src = myPics[curPic]
 }
}
// End hiding script from old browsers -->

</SCRIPT> </HEAD>

<BODY BGCOLOR = WHITE>

<TABLE BGCOLOR = RED BORDER = 3>
<TR>
<TH HEIGHT = 30 COLSPAN = 3>
 <H1><CENTER>Slide Show</H1></CENTER></TH>
<TR>
<TD HEIGHT = 300 WIDTH = 200></TD>
<TD WIDTH = 300>

 << Previous
 Next >>
</TD>
<TD WIDTH = 200></TD>
</TABLE>
```

Create the array containing the graphics to be used.

Set the value of the variable *curPic* to 0.

Set the value of the variable *picCount* to the number of elements in the array *myPics* minus 1.

Replace the functions *getPrevious* and *getNext* with one function called *chngSlide*. In this function we receive the value passed from the function call in the HTML code below (either 1 or -1) and use it to either add one to *curPic* or subtract one. We then check to see if the new value of *curPic* is **either less than zero or greater than the value of the number of pictures in the array (*picCount*)** and adjust its value to **either zero or *picCount* respectively**.

We then assign the source of the image object *thisPicture* to the array element calculated above.

In the link created for the **previous** slide, we call the function *chngSlide* and **pass a value of -1 to it** (the function's variable *direction* will **assume this value**).

Likewise for the link created to choose the **next** slide, we pass a value of 1 to the *chngSlide* function.

# Web Design

88

What is the World Wide Web? .....	1
How It Works.....	1
The Web Browser .....	2
Searching the Web .....	3
Search Engine .....	3
Web Index.....	6
The Speed of the Web.....	6
Contents of Your Web Site.....	6
Organizing the Data .....	6
Sequential Organization.....	6
Hierarchical Organization.....	7
Combination Sequential and Hierarchical Organization .....	7
Web Organization.....	7
What Should Each Page Include?.....	8
What Should Every Site Include?.....	8
Creating a Web Page.....	9
HTML Basics.....	9
Creating an HTML Document In Notepad .....	9
Basics of HTML .....	10
Exercise 1 Creating Your First Web Page .....	11
Editing an HTML Document .....	12
HTML Code.....	13
Adding White Space .....	13
Paragraph Tags.....	15
Line Breaks .....	16
Horizontal Rules .....	17
Exercise 2 Using White Space.....	18
Physical Styles .....	19
Bold Text .....	19
Italics Text .....	19
Underlined Text .....	19
Subscript Text .....	19
Superscript Text.....	19
Typewriter Font .....	19
Big Font .....	19
Small Font.....	20
Exercise 3-1 Using Physical Styles.....	21
Logical Styles.....	22
Emphasized.....	22
Block Quotes.....	22
Cited Works .....	22
Text Keyed In by the User .....	22
Exercise 3-2 .....	24
Headings .....	25
Heading 1 .....	25

# Web Design

89

Heading 2 .....	25
Headings 3 Through 7 .....	25
Alignment .....	25
CENTER .....	25
Using Alignment With Headings .....	25
Exercise 4-1 Using Headings and Text Alignment .....	26
Exercise 4-2 .....	26
Lists .....	27
Bulleted Lists .....	27
Numbered Lists .....	29
Exercise 5-1 Using Lists .....	29
Exercise 5-2 Using Lists .....	30
Setting Fonts .....	31
Base Fonts .....	32
Color .....	33
RGB Values .....	33
Background Color .....	33
Text Color .....	33
Link Color .....	33
Named Colors .....	34
Exercise 6-1 Setting Colors .....	34
Inserting Images .....	35
Image Formats .....	35
Guidelines for Images .....	35
Exercise 7-1 Using Bitmap Graphics .....	36
Exercise 7-2 Using GIF and JPG Graphics .....	37
Adding Space Around Graphics .....	38
Exercise 7-3 Adding Space Around a Picture .....	38
Project #1 .....	39
Links .....	40
Assigning Color to Links .....	40
Exercise 8-1 Using Links Within a Page .....	41
Exercise 8-2 Using Links From Page to Page .....	42
Graphic Links .....	43
Exercise 9-1 .....	43
Image Maps .....	43
Example of an Image Map .....	45
Exercise 10 -1 Creating an Image Map .....	48
Creating Tables .....	49
Using Tables to Display Data .....	49
Vic Tory .....	49
Exercise 11-1 Creating a Table for Data .....	51
Using Tables for Page Layouts .....	52
Exercise 11-2 Creating a Page Layout Using a Table .....	55
Using a Graphic as a Background .....	55
Exercise 12-1 Using Background Graphics .....	57

# Web Design

90

Adding Sound Files to the Web Page .....	58
Inserting a Link to a Sound File.....	58
Exercise 13-1 Creating a Link to a Sound File .....	58
Embedding a Sound File.....	59
Exercise 13-2 Using EMBED to Play a Sound File .....	59
What is JavaScript?.....	60
How To Start a JavaScript .....	60
Example of a Simple JavaScript .....	60
Parts of the Script Tag.....	60
Parts of the JavaScript Language .....	60
Value Types .....	61
Operators.....	61
Assignments.....	61
Comparisons .....	61
Where Do the Scripts Go? .....	62
Hiding Scripts From Older Browsers .....	62
Multiple-Line JavaScript Comments .....	63
User Alerts .....	63
USING <NOSCRIPT> .....	64
Redirecting the User .....	64
Redirecting Using a Link .....	65
Detecting Browsers.....	66
Detecting Plug-ins.....	67
Loops.....	68
If Logical Test.....	69
Adding an ELSE Clause .....	70
Functions, Forms, and Buttons .....	71
Graphics .....	73
Mouse Rollovers .....	73
Multiple Image Rollover.....	75
Frames.....	77
Example of a Frame .....	77
Cycling Banners.....	80
Add Images to an Array .....	80
Create the Function to Change Images .....	81
Finish the HTML Code to Use the Script .....	81
Making the Banner Wait for Each Image to Fully Load Before Changing .....	82
Adding Links to Cycling Banners.....	83
Image.....	83
Exercise Using Banners .....	84
Slide Shows.....	85
Making the Slide Show a Wrap-Around.....	86