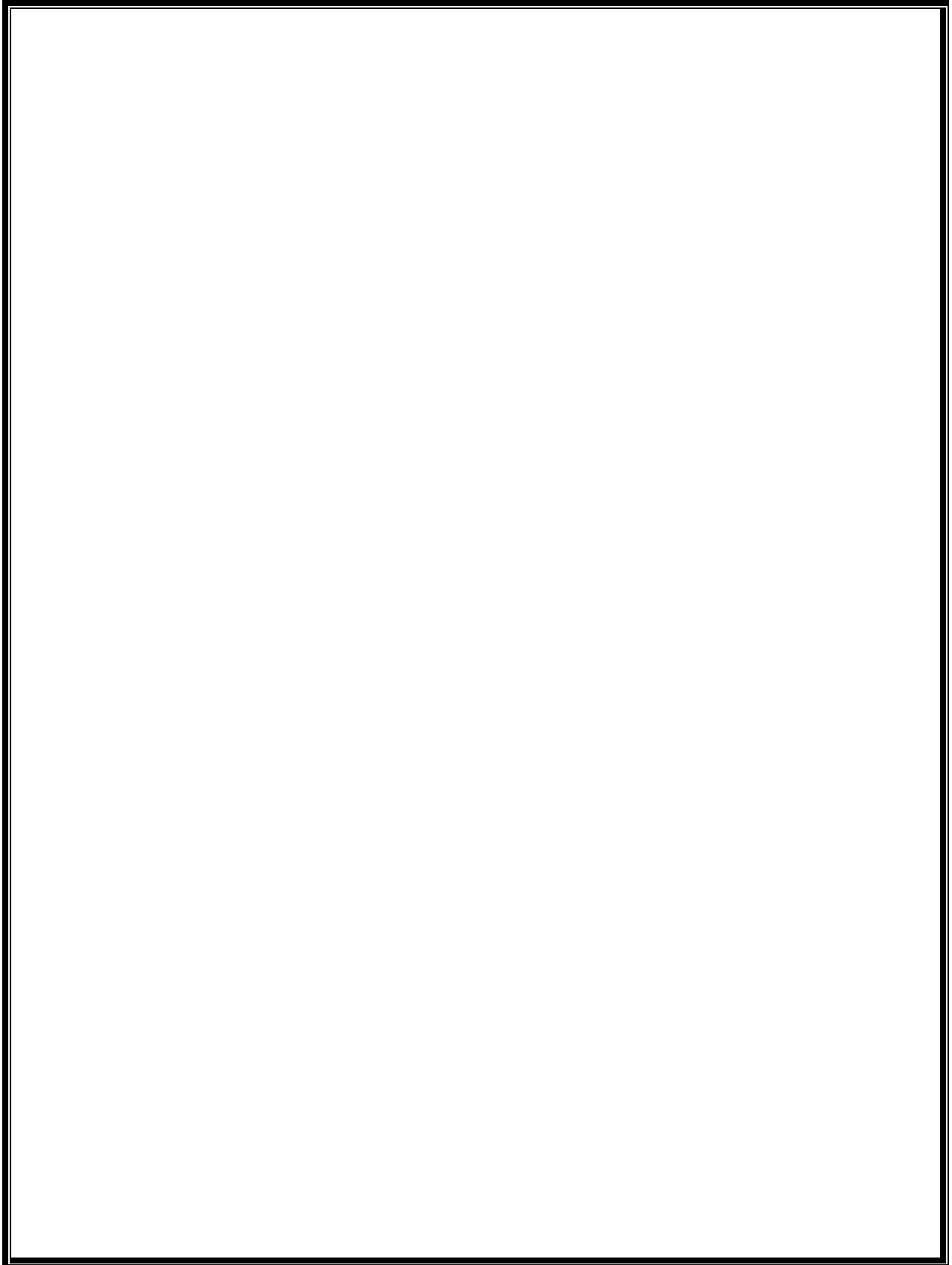




**DEPARTMENT OF PURE AND APPLIED SCIENCE**

**SCHOOL OF INFORMATION SCIENCE**

**BIT1202: Web Design**



## Course Purpose

At the end of the course, the student should be able to explain what is web design and describe its qualities and place in information science

## Learning Outcomes:

At the end of this unit the student should be able to:

- Describe and explain web design
- Demonstrate the ability to design a web
- Appreciate the value of web designing in information flow

## Content

History of the World Wide Web (WWW); Internet and the WWW; importance of web design; Web development using HTML, Dreamweaver etc; web services, feeds and blogs; web design tools; network and web security; Future trends

## Learning and Teaching Methodologies

Lectures, Discussions, Practical in computer laboratory, Demonstrations, Assignments

## Assessment

Type	Weighting (%)
Examination	70%
Continuous Assessment	30%
Total	100%

## Core Texts

1. Moseley R. (2006): Developing Web applications. John Wiley & Sons Ltd, England.
2. Deitel, H. M., Deitel P.J. & Goldberg A. B. (2004): Internet & World Wide Web: how to program. Upper Saddle River, N.J: Prentice Hall.
3. Niederst J. (2001): Learning Web Design; Beginners Guide to HTML, Graphics and Beyond. Prentice Hall, New Delhi. 8173661677.

## References

1. Faster, Smarter(2003) Web Page Creations; take Charge of Web Design and Production, Faster, Smarter, Better UML. Millhollon, Mary.. Prentice Hall, New Delhi .8120322452
2. Alan Denis(2002).Systems Analysis and Design an Object Oriented Approach with UML. Denis Alan..Libraries Unlimited. 0471413879

## Table of Contents

CHAPTER 1: INTRODUCTION TO THE INTERNET .....	- 1 -
1.1 What Is The Internet? .....	- 1 -
1.2 Connections. ....	- 2 -
1.3 History Of The Internet. ....	- 2 -
1.4 Email.....	- 4 -
1.4.1 E-Mailing Tips .....	- 6 -
1.4.2 Sending Attachments .....	- 7 -
1.5 Upload/Download Files.....	- 7 -
1.6 Access the World Wide Web .....	- 7 -
1.7 Internet Protocols .....	- 7 -
How Does Tcp/Ip Work For The Internet? .....	- 7 -
1.8 Routers.....	- 8 -
1.9 Internet Service Providers (ISP) .....	- 8 -
1.10 Internet Ethics.....	- 9 -
1.11 Internet Names And Addresses .....	- 9 -
1.12 Domain Name System.....	- 9 -
CHAPTER 2: THE WEB.....	- 11 -
2.1 Chapter Overview .....	- 11 -
2.2 Introduction To The Web .....	- 11 -
2.2.1 Why is it called the World Wide Web? .....	- 11 -
2.2.2 What can I find on the Web? .....	- 11 -
2.3 The History Of The Internet And The Web .....	- 13 -
2.3.1 Where does the Web fit into the story? .....	- 13 -
2.3.2 What practical use is the Web to me? .....	- 14 -
2.3.3 What's the relationship between the WWW and the Internet? .....	- 15 -
2.3.4 Features of WWW.....	- 15 -
2.4 Internet/Web Terminologies .....	- 15 -
2.5 Web Site planning guiding principles.....	- 16 -
2.5.1 Factors to be considered while planning for a website .....	- 16 -
2.6 Types Of Web Documents .....	- 18 -
2.6.1 Static document .....	- 18 -

2.6.2 Dynamic documents .....	19 -
2.6.3 Active documents .....	19 -
2.7 Web design tools.....	19 -
CHAPTER 3: CREATING YOUR FIRST WEB PAGE .....	20 -
3.1 Chapter Overview .....	20 -
3.2 What Is Html? .....	20 -
3.3 How HTML Works .....	20 -
3.4 HTML: Then and Now.....	20 -
3.5 HTML and the World Wide Web Consortium .....	21 -
3.6 The Limitations of HTML .....	21 -
3.7 The Need for Style Sheets .....	21 -
3.8 How Web Browsers Affect Your Work.....	22 -
3.8.1 Browser Compatibility Issues.....	22 -
3.8.2 Creating Cross-Browser Compatible Pages.....	22 -
3.8.3 Cutting-Edge Coding.....	22 -
3.8.4 Coding for Multiple Screen Resolutions.....	23 -
3.9 Bandwidth Concerns .....	23 -
3.10 Working with the Cache.....	23 -
3.11 Should You Use an HTML Editor? .....	23 -
3.12 Creating a New HTML Document .....	24 -
3.12.1 Saving a HTML Document .....	24 -
3.12.2 Closing and Opening a HTML Document .....	24 -
3.12.3 Loading a HTML Document in a Local Browser.....	25 -
CHAPTER 4: HTML ELEMENTS, TAGS & ATTRIBUTES .....	26 -
4.1 Chapter overview .....	26 -
4.2 HTML - Elements .....	26 -
4.2.1 <head> Element .....	27 -
4.2.2 <title> Element.....	27 -
4.2.3 Body Element <body> .....	28 -
4.2.3.1 Body Margins .....	29 -
4.2.3.2 Base Text .....	29 -
4.2.3.3 Base Links .....	29 -

4.2.3 Elements Reviewed .....	30 -
4.3 HTML Tags .....	30 -
4.3.1 Logical vs. Physical Tags .....	30 -
4.3.2 Nested Tags .....	32 -
4.3.3 Elements Without Closing Tags .....	32 -
4.4 HTML Attributes .....	33 -
4.4.1 Title Attribute .....	33 -
4.4.2 Align Attribute .....	34 -
Tips .....	35 -
4.5 Font .....	35 -
4.5.1 Font Size .....	35 -
4.5.2 Font Color .....	36 -
4.5.3 Font Face .....	36 -
4.6 Body Attributes .....	37 -
4.6.1 Bgcolor .....	37 -
4.6.2 Background .....	37 -
Tips .....	38 -
CHAPTER 5: TEXT FORMATTING .....	39 -
5.1 Chapter Overview .....	39 -
5.2 Paragraph Text .....	39 -
5.3 HTML Headings .....	40 -
5.4 Formatting Text Elements with Tags .....	41 -
5.4.1 Line Breaks .....	42 -
Tips .....	43 -
5.4.2 <p>Paragraph .....	43 -
5.4.2.1 Paragraph Alignment .....	43 -
5.4.3 <pre> Preformatting .....	44 -
5.4.4 Horizontal Rule .....	45 -
5.4.5 Comments in HTML .....	45 -
5.4.6 Abbreviating .....	46 -
5.4.7 HTML Character Entities .....	46 -
CHAPTER 6: HYPERLINKS .....	47 -

6.1 Chapter Overview .....	- 47 -
6.2 Hypertext Reference (href) .....	- 47 -
6.3 Link Targets .....	- 48 -
6.4 Email Links.....	- 49 -
Email Links:.....	- 49 -
Complete Email: .....	- 49 -
6.5 Default Links; Base .....	- 49 -
6.6 HTML Comments.....	- 50 -
6.6.1 Comment Tags: .....	- 50 -
Tips .....	- 51 -
CHAPTER 7: HTML LISTS.....	- 52 -
7.1 Chapter Overview .....	- 52 -
7.2 Introduction .....	- 52 -
7.3 Unordered Lists.....	- 52 -
7.4 Ordered Lists .....	- 53 -
7.4.1 Ordered List Types: .....	- 53 -
7.4.2 Numbered List Start Attribute .....	- 54 -
7.4.2.1 Numbered List Start - 4:.....	- 54 -
7.5 Definition Lists.....	- 54 -
7.5.1 Definition List Code:.....	- 54 -
Definition List Display: .....	- 54 -
Tips .....	- 54 -
CHAPTER 8: HTML IMAGES .....	- 55 -
8.1 Chapter Overview .....	- 55 -
8.2 Introduction to Images .....	- 55 -
8.2.1 Background Images.....	- 55 -
8.2.2 Inline Images .....	- 57 -
8.2.2.1 Source URLs .....	- 57 -
8.2.2.2 Image Height and Width Attributes.....	- 58 -
8.2.2.3 Alternative Attribute .....	- 60 -
8.2.2.4 Horizontally Align Images .....	- 60 -
For example: .....	- 60 -

8.2.3 Common Image Types.....	- 60 -
Tips .....	- 61 -
8.2.4 Image Links.....	- 61 -
CHAPTER 9: HTML TABLES .....	- 63 -
9.1 Chapter Overview .....	- 63 -
9.2 Introduction .....	- 63 -
9.3 Table Rows & Table Columns.....	- 65 -
9.3.1 Spanning Multiple Rows and Cells .....	- 65 -
9.3.2 Cell Padding and Spacing .....	- 66 -
9.4 Using Tables to implement Layout .....	- 68 -
9.5 The bgcolor Attribute.....	- 68 -
9.5.1 Web Colors.....	- 69 -
9.5.2 Coloring Fonts, Table Rows, & Table Columns.....	- 70 -
Scoreboard:.....	- 71 -
9.5.3 Color Codes .....	- 71 -
Tips .....	- 71 -
CHAPTER 10: HTML FORMS .....	- 72 -
10.1 Chapter Overview .....	- 72 -
10.2 Introduction .....	- 72 -
Tips .....	- 73 -
Attribute.....	- 74 -
What it does... ..	- 74 -
10.3 Input Element(s).....	- 74 -
10.3.1 Web Forms: Value Attribute .....	- 75 -
10.3.2 Name and ID Attributes .....	- 75 -
10.4 Text Fields .....	- 76 -
10.4.1 Text Fields: Size Attribute .....	- 76 -
10.4.2 Text Fields: Maxlength Attribute .....	- 77 -
10.5 Password Fields.....	- 77 -
10.5.1 Password Fields: Attributes .....	- 78 -
10.6 Reset Buttons.....	- 78 -
10.7 Submit Buttons .....	- 79 -



10.7.1 Form Submission - Action .....	79 -
Tips .....	80 -
10.8 Checkbox Forms .....	80 -
10.8.1 Checkboxes Selected.....	81 -
10.9 Radio Buttons.....	82 -
10.9.1 Radio Buttons: The Checked Attribute .....	83 -
10.10 Select Fields.....	84 -
10.10.1 Disabling Selection Fields.....	84 -
10.11 Hidden Field .....	85 -
10.12 Upload Field .....	85 -
10.12.1 Max File Size Field .....	86 -
10.13 Text Areas .....	86 -
10.13.1 Text area Wrap.....	87 -
10.13.2 Text Areas: Readonly .....	88 -
10.13.3 Text Areas: Disabled.....	89 -
CHAPTER 11: MULTIMEDIA ELEMENTS.....	90 -
11.1 Introduction .....	90 -
11.2 Music Files.....	90 -
11.2.1 Embed Attributes - Related to Display.....	90 -
11.2.2 Embed Attributes - Related to Functionality .....	91 -
Tips .....	91 -
11.3 Video Codes .....	91 -
11.3.1 Video Media Types.....	91 -
11.3.2 YouTube Videos .....	92 -
11.3.3 Embed Attributes .....	92 -
CHAPTER 12: META TAGS AND META DATA.....	94 -
12.1 Chapter Overview .....	94 -
12.2 Meta Tag Description.....	94 -
12.3 Keyword Meta Tags .....	94 -
12.4 Refresh and Redirect Meta .....	95 -
12.5 Revised Meta .....	95 -
Tips .....	96 -

CHAPTER 13: HTML STYLE ATTRIBUTES .....	- 97 -
13.1 Chapter Overview .....	- 97 -
13.2 Introduction .....	- 97 -
13.3 Styling.....	- 97 -
13.3.1 Div Element(s).....	- 98 -
13.3.2 Div inside of Div.....	- 99 -
13.4 Page Layouts and Templates.....	- 100 -
13.4.1 Standard Layout .....	- 100 -
Tips .....	- 102 -
CHAPTER 14: HTML FRAMES.....	- 103 -
14.1 Chapter Overview .....	- 103 -
14.2 Introduction .....	- 103 -
14.3 Adding a Banner or Title Frame .....	- 103 -
14.4 FrameBorder and FrameSpacing .....	- 104 -
14.5 Frame Name and Frame Target .....	- 104 -
14.5 Noresize and Scrolling.....	- 105 -
Tips .....	- 106 -
CHAPTER 15: SCRIPTING LANGUAGES .....	- 107 -
15.1 Introduction .....	- 107 -
15.2 Javascript Code .....	- 107 -
15.3 VBScript.....	- 108 -

## CHAPTER 1: INTRODUCTION TO THE INTERNET

### 1.1 What Is The Internet?

The Internet, sometimes called simply "the Net," is a worldwide system of computer networks - a network of networks in which users at any one computer can, if they have permission, get information from any other computer (and sometimes talk directly to users at other computers)

In essence the Internet is a term used to describe connection of thousands of computers, spanning all over the world. Some people may liken this to a single entity, but this is not true. The Internet is transitory, ever changing, reshaping and remolding itself. Ordinarily a collection of thousands of computers world wide might not attract so much attention. However people are using this new medium in ways that simply was not possible a mere five years ago.

### Just How Are All These Things Made Possible?

The Internet happens to be the single largest telecommunications system ever conceived by humankind. There are four basic building blocks to the Internet, **Hosts**, **Routers** and **Clients** and **Connections**. In most cases your computer falls under the "Client" category. Data is sent from your computer in the form of a "packet". You can liken a packet to be similar to an envelope; it surrounds your data and contains both a return and destination address. Your computer handles the packets for you, it's all done in the background, without your knowledge.

A **Router** is a special device. Basically routers sit at key points on the Internet and act like traffic cops at an intersection of hundreds of streets. The Router basically reads the destination address on the packets being sent by your computer and then forwards the packet to the appropriate destination. In some cases your data will travel through several routers before reaching its ultimate destination.

Diagram below shows connections to various routers

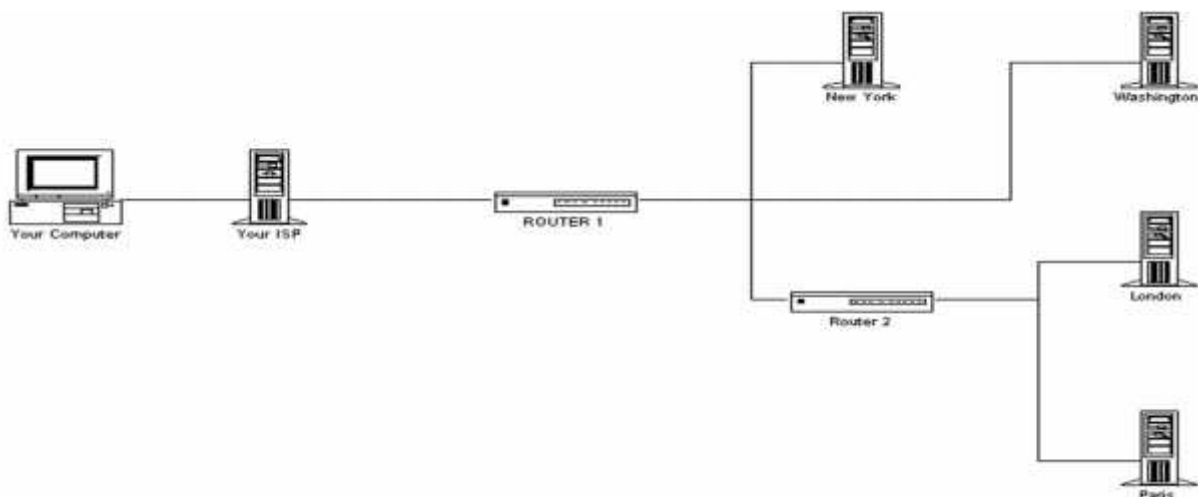


Fig 1.1 Sample Internet Connections

## 1.2 Connections.

This is a catch all term describing how you can connect from one point to another point. As an end user, your only concern is that the connection is good, but for a network engineer, this can mean several different types of technologies, including;

- Dial Up Phone Lines
- Fiber Optics
- ISDN
- Frame Relay
- Satellite Links

## 1.3 History Of The Internet.

In response to a need for secure computer-to-computer communications, DARPA, the Defense Advanced Research Projects Administration, commissioned a study in computer-to-computer technologies back in the early 1970's. From this beginning the Internet was born. During the next 20 years the Internet was used solely as a combination of military and academic network, linking computers first nationwide, then ultimately world wide. The idea behind the Internet is really very simple and can be conceptualized thusly;



Figure 1.2 Simple Connections

Two computers are connected via a single wire. In order for one computer to talk to the other, it sends a signal requesting permission to speak. If the other computer is busy, it replies with the equivalent of a "Please wait, I am busy" otherwise it replies, granting permission. Since both computers know what the other one is talking about, by virtue of the fact they are running similar software, the data can be passed from computer to computer. This is a very straightforward and trivial example. Now however, instead of a single wire, we replace the connection with the Internet, which can be millions of computers between the two computers wishing to talk.

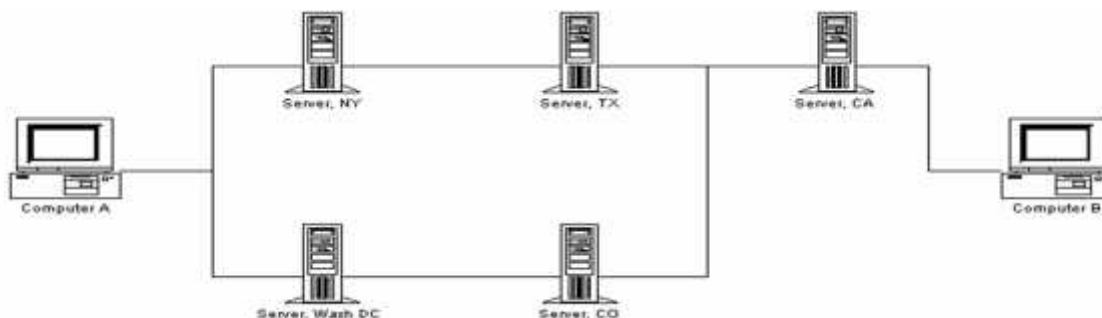


Figure 1.3 Sample Internet Connections 2

Computer A and Computer B wish to talk to each other, but there is nearly 3000 miles between them. Using the Internet, the number of places through which the data has to travel is really transparent to the user.

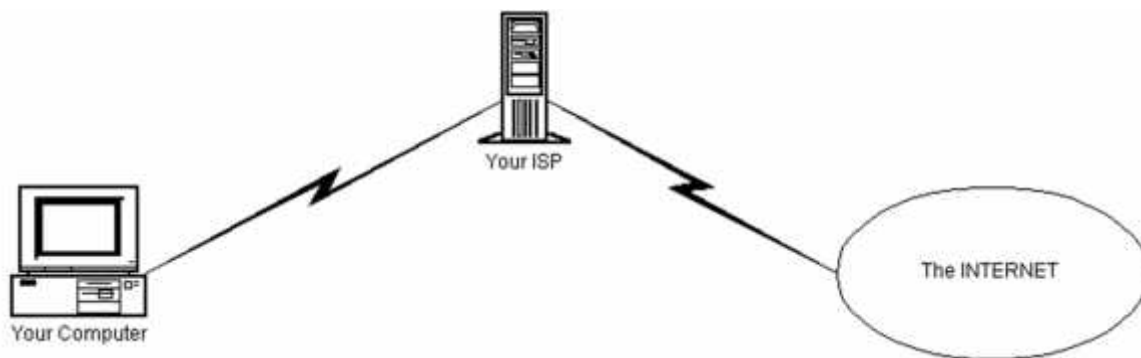
In effect, the link between Computer A and Computer B can take many paths. It can travel hundreds or even thousands of miles out of the way in order to reach the other computer. All you need to know is that it will get there.

As the 1980's progressed the face of computing changed significantly, and with it, the Internet. More and more commercial and personal computers were going on-line, until, they exceeded the number of the original users.

The 1990's signaled the start of the "connected" era, with the end of the cold war, and improvements in military communications, the original Military users of the Internet left for other communications systems. The Internet was left much as it is today, a collection of internationally based users and computers.

With improvements on the desktop, there arose a need for better graphics on the Internet. The Internet up to this point, had been largely a text only system.

The graphics capabilities implemented were called HTML, and a means was invented to allow users to view these HTML files in their graphic format. With the Internet largely in place, all that was needed was to invent the transmission mechanism. That mechanism was dubbed the World Wide Web, or Web for short.



**Fig. 1.4 Sample Connections 2**

Figure 1.4 shows a typical connection scenario, with your personal or work computer connecting to the Internet via your ISP's server. Your ISP may go through several connections with other servers before reaching the "main backbone" of the Internet, the exact route is not information needed by the typical user.

There are two classes of computers on the Internet, HOSTS and CLIENTS. Unless you have a permanent link to the Internet and your machine is always connected and on-line, then you are

probably a client and not a host. As a client to the Internet, you should have the following abilities; (If you don't, talk to your Internet Service Provider)

- ✓ Send Email
- ✓ Upload/Download Files
- ✓ Access the World Wide Web



## 1.4 Email

It is a computer-based storage and forwarding of text based message. The standard e-mail is based on a TCP/IP service known as SMTP, simple mail transfer protocol. SMTP protocol specifies addressing conventions, the "@" sign, usernames, and locations. E-mail is so widespread that staff is sometimes bombarded with messages, including unsolicited e-mails known as SPAM mails.

Email is really fast - it is sent and retrieved in seconds, minutes at the most. Contrary to popular belief, it is not instantaneous. Postal mail is often called snail mail in comparison. Sending email is easy, too. All you need is access to the Internet, an email program, and the email address of the person with who you wish to communicate. You can send e-mail messages to one person at a time or to thousands of people all at the same time.

Like any other tool, electronic mail has its strengths and weaknesses. On the surface, it appears to be just a faster way of delivering letters, or their equivalent. Electronic mail differs from the other applications we are looking at because it is not an "end-to-end" service: the sending and receiving machine need not be able to communicate directly with each other to make it work. It is known as a "store and forward" service. Mail is passed from one machine to another until it finally arrives at its destination. This is completely analogous to the way the postal service delivers mail.

The postal service operates a store and forward network. You address a message and put it into a post box. The message is picked up by a truck and set to another place and stored there. It is stored and forwarded to another place. This step is repeated until it arrives at the recipient's mailbox. If the recipient's mailbox happens to be in place where the postal office cannot deliver directly (e.g., another country), you can still send the message; the postal service will pass the message to the postal service of that country for delivery.

In order for an email message to be sent, the person's address should be known. An email address, like a postal mail address, contains all the necessary information needed to deliver a message to someone. If there will be something wrong with the email address, it will be sent back to the sender having the MAILER DAEMON in the mail box.

Internet email addresses consist of a local part and a host part. The username refers to the mailbox, login name, or the userID of the recipients on that computer. The host part of the address should be recognizable to the user - a series of words separated by dots. The local part and the host part of an email address are separated by an @ sign.

E-mail Address Structure:

*username@domain\_name.top\_level\_domain.country\_top\_level\_domain*

Example of an E-mail address

allanmuthomi@aol.com

Username / account name / login name - a unique name given to Internet users to identify the user of the organization name. It may consist of any combination of letters and numbers as well as symbols.

Host name - computer connected to the Internet.

Domain - a named group of network hosts.

More and more people are using e-mails because:

- They are usually delivered in a matter of minutes, or even seconds. You can send and receive messages anytime no matter where you are.
- Because communication is faster, results come in fast also.
- Companies use e-mail to send electronic newsletters, price changes, product information and updates, and the like to whoever may need them. This eliminates the cost of postage, presentation and printing.
- You can send and receive electronic files that contain text, graphics, sounds, and moving videos.

Some drawbacks of using e-mails

- Not secure or confidential. Systems administrators can read your mails at any point as your message is sent from one place to another. Although the employees could be really practicing their business ethics, nothing can stop them from snooping in your mails.
- Special formatting may be lost at recipient's end.
- Occasionally unreliable because some problems might occur in the computer unknown to the sender or recipient.
- Presence of junk mails.
- Computer virus can be sent through e-mails.

But like dropping a letter in a mailbox, once you've clicked the send button, nothing can stop its delivery to recipient.

#### **1.4.1 E-Mailing Tips**

- 👍 Check your mailbox daily. It is rude to let your e-mail sit in your mailbox.
- 👍 Delete unwanted messages; they take up disk space.
- 👍 Download files you want to keep and save them in a file.
- 👍 Scan for viruses.
- 👍 When you reply to a message sent to a group, reply to a sender only, not to the group.
- 👍 Don't publicly post an e-mail sent to you by another person unless you have his or her permission.
- 👍 Don't type your message in ALL CAPS. It is the NET equivalent of shouting.
- 👍 Don't waste NET resources. For example, don't copy a file from a computer in the USA when the same file can be found closer to home. Log off from your ISP when you are simply typing e-mail. Log on when you are ready to send or receive messages.
- 👍 Be sensitive to possible national and ethnic differences between you and your recipients.
- 👍 Don't believe everything you read. People posting articles on the Internet are not always experts as they seem to be.



### **1.4.2 Sending Attachments**

Not all e-mail systems handle files in the same way. If you are not sure, send a test file to the recipient to check for compatibility.

Let the reader know what file format you are sending. Even with the same software vendor, files aren't always compatible. An earlier version of a program might not read a file created with a newer version of the same program.

### **1.5 Upload/Download Files.**

Upload/Download are two different faces on the same coin. Basically it refers to moving a file, either from a host computer to your client computer(Download) or from your client computer to some host computer(Upload).

### **1.6 Access the World Wide Web**

The Web blends the best and not-so-best of the textual information with the graphical capabilities of today's desktop systems. On the Web you will find information relating to almost any conceivable topic. More about the web in Chapter 2.

### **1.7 Internet Protocols**

A standardized set of rules and agreement on how to communicate that specify the format, timing, sequencing, and/or error checking for data transmissions. It is the languages through which computers on the Internet communicate thus called Computer Protocols.

Computer networking protocols are made up of sequences of binary values collected into groups called PACKETS, small chunks of data in the Internet that are then transmitted over a media like coax cable, fiber optic cable or phone lines as electrical signals or pulses of light.

- i. **TCP/IP:** A standard method of transmitting data in small packets across incompatible networks.

**TCP** - Transmission Control Protocol is the set of protocols that enables all the networks of the Internet to communicate and provides for reliable delivery of data between communicating applications.

**IP** - Internet Protocol allows a packet to traverse multiple networks on the way to its final destination. It takes care of addressing, or making sure that the routers know what to do with your data when it carries.

### **How Does Tcp/Ip Work For The Internet?**

Even though the individual can't communicate with one another, the various networks to which they're connecting can by using TCP/IP. All networks attempting to connect to the Internet must either support TCP/IP or first pass through what is known as a Gateway. The Gateway serves to translate the data into TCP/IP. When using the TCP/IP applications, various types are being transferred from one computer to another. TCP/IP breaks this information into

chunks called PACKETS. Each packet contains a piece of information or documents, plus some ID tags, such as the addresses of the ending and receiving computers. That's why the Internet is known as a packet-switched network. The switches are computers called ROUTERS, which are programmed to figure out the best packet routes, just as a travel agent might help you find the best flight with the fewest layovers. Routers are the airport hubs of the Internet; they connect the networks and shuttle packets back and forth.

- ii. **FILE TRANSFER PROTOCOL (FTP):** The oldest commonly used method of transferring files on the Internet. It works in two different modes. The first is that it can be used to transfer files between any two accounts on the Internet (however, this requires knowing the passwords of both accounts). The other mode is known as anonymous FTP. An anonymous FTP site enables anyone to connect into the system and download files.

Publicly accessible FTP servers are called anonymous servers because of the procedure used to connect to them. Traditionally, the account logged on to a server was anonymous; currently it is common to use the FTP account. Similarly, the traditional password was guest, but most current systems request the use of your e-mail address as the password.

FTP by itself has no facility to let the user know in advance where a particular file is available. FTP as a facility of sharing documents is somewhat on the decline because of the newer tools, but it is still the primary way to acquire computer software over the Internet.

- iii. **TELNET (REMOTE ACCESS):** The Telnet protocol makes it possible for you to log onto a remote system - after you've logged onto the Internet itself. Telnet is a way of "moving" from place to place on the Net. The key concept is this: Just as the FTP command gives you access to and lets you transfer files from only a portion of a remote computer, the Telnet command gives you access to a remote system and lets you run a specific program on that system. Also the Telnet allows you to create e-mail.

## **1.8 Routers**

These are devices that connect network together. It's a set of computers that connect the different pieces of the Internet. Sometimes Ethernets, sometimes token rings, and sometimes telephone lines. They act as postal substation in the Internet and make decisions about how to route data (packets).

## **1.9 Internet Service Providers (ISP)**

Internet service providers are participating in a competitive market. For any given kind of service, there are usually several providers available - and several different price structures. And

examples of providers are as follows: Mozcom, PhilWorld, SkyInet, Virtual Asia, Cybernet, Easynet, WorldTouch, and Infocom to name a few. They sell Internet services such as access, email, web page hosting, etc. They supply information and entertainment services – as well as Internet access, keyword searches, on-line shopping, games, weather information, company profiles, etc., of course, these information may be provided with a fee. Access to ISP's is either via dial-up or dedicated.

**DIAL-UP** – connect through modems and the standard telephone line, connections are metered or may vary.

**DEDICATED** – connect through a high-speed line from the user LAN to the ISP.

### **1.10 Internet Ethics**

The sense of community on the Internet is very strong. If you join this community, and become a netizen, you will become part of it more quickly and easily by following a few of its unwritten laws. The members of the Internet community not only tolerate, but also encourage, individuality. In return, they expect individual responsibility and to respect the right of others.

### **1.11 Internet Names And Addresses**

Each computer, or hosts, has a name and a numerical address (both unique). An Internet computer name is usually several words separated by periods, such as interface.edu.ph. An Internet address – technically an ID address - consist of four numbers, each less than 256, also separated by period, for example, 165.220.27.34. When saying these names and addresses out loud, you should substitute "dot" for "period" to sound as though you belong. This is known as dotspeak, and there is a whole lot of it in the Internet.

The idea if for people to use the computers' names when accessing resources, and to let the computers and routers works with the IP addresses.

### **1.12 Domain Name System**

The Domain Name System (DNS) is the worldwide system of distributed database of names and addresses. These databases provide the "translation" from names to numbers and vice-versa. DNS names are constructed in a hierarchical naming fashion, which you can think of as a worldwide organization chart.

Since IP addresses, like 192.172.10.254, are not user friendly and could cause typing errors; the domain name system (DNS) was created so people would not have to remember several confusing numbers. Domain names enable short, alphabetical names to be assigned to IP addresses DNS provides a set of procedures that converts or translates domain names into IP address and vice versa. DNS provides distributed look up.

#### **1.12.1 How to obtain domain name**

Organization chooses a desired name that must be unique, registers it with a central authority and is placed under one top-level domain. This name is subject to international law for trademarks and copyrights. A domain name is divided into four levels:

#### **(i) First level**

It is an extension and is assigned according to what kind of domain it represents. Examples of top-level domains are:

<b>Domain name</b>	<b>Type of domain</b>
.Edu	Educational institution
.Gov	Government organization
.Mil	Military organization
.Net	Network service provider. Eg. ISPs
.Com	Commercial organization
.Org	Organizations, normally a not for profit.
.Au	Australian domain
.Uk	United Kingdom domain
.Ke	Kenyan domain
.Za	South African domain
.ph	Philippines domain
.co.ke	Domain for a commercial organization in Kenya
.or.ke	Domain for a not for profit organization in Kenya
.ac.ke	Domain for a an academic institutionin Kenya

#### **(ii) Second level**

The name one chooses to identify themselves on the Internet. E.g. [www.shawks.com](http://www.shawks.com)

#### **iii) Third level**

This is a division within a company. E.g. If the company shawks Ltd had a division of sales, then they could be assigned a unique domain name like [www.sales.shawks.com](http://www.sales.shawks.com)

#### **iv) Fourth level**

This would either be for a company's subdivision or an individual computer. e.g. Assume the sales department in Shawks Ltd had two sub-divisions, products and services. Then the domain names for the two sub-divisions would be: [www.services.sales.shawks.com](http://www.services.sales.shawks.com) and [www.products.sales.shawks.com](http://www.products.sales.shawks.com) .

## CHAPTER 2: THE WEB

### 2.1 Chapter Overview

#### 2.2 Introduction To The Web

The World Wide Web (also called the WWW, W3, or simply "the Web") is a huge global database of information, which is stored and distributed by the millions of linked computers that make up the Internet. Using the Web, you can access information from all over the world, and can display it in the form of documents called "Web pages" on your own desktop computer.

Many different types of information are available on the Web. In addition to plain text, Web pages can also contain pictures, video clips and sounds, and they can even be programmed to "interact" with the person viewing them. This ability to provide multi-media information content means that the Web is now an extremely powerful educational tool. The ability to use the Web is rapidly becoming a fundamental academic skill.

##### 2.2.1 Why is it called the World Wide Web?

The really distinctive feature of the Web is the way in which documents are "linked" to one another. Most Web pages contain links to other, related documents located at other sites around the world. This global, web-like structure of interlinked documents is the reason for the name "World Wide Web". Using a mouse (or a keyboard) you can "follow links" from one document to the next, exploring and assessing the information as you go. This is a potentially endless activity, sometimes dismissively referred to as "surfing".

Who is in charge of the Web?

Nobody "owns" the Web, and no central authority is responsible for organizing or regulating the availability of information on it. This is both a major strength (because it allows uncensored freedom of information) and a disadvantage (since it can be hard to locate the information that you require). Despite this de-regulated structure, you should remember that the actual content of any information provided on the Web is, in most cases, the intellectual property of whoever originally created it, and is therefore subject to copyright law in the same way as any other form of publication.

##### 2.2.2 What can I find on the Web?

The short answer to this is "almost anything". Tens of millions of documents are currently available on the Web, covering a huge range of topics. Some of the most useful for academic users include:

- ✓ **Teaching/Learning resources:** lecture notes, interactive tutorials, discussion articles
- ✓ **Research:** project descriptions, databases, conference proceedings, grants
- ✓ **Computing:** software archives, tutorials, manuals, discussions
- ✓ **Administration:** staff lists, course details, minutes of meetings

- ✓ **Government:** legislation, political parties, United Nations, European Union
- ✓ **Environment:** conservation, climate change, wildlife, pollution
- ✓ **Career:** vacancy listings, career agencies, on-line newspapers
- ✓ **Travel:** tourist guides, accommodation, timetables, weather
- ✓ **Culture:** art galleries, music, cinema, literature, religion
- ✓ **Recreation/Entertainment:** Games, music, video tutorials, movies, sport, recipes, TV listings, magazines, personal home-pages
- ✓ **Social Media:** Collectively describes a set of tools that foster interaction, discussion and community, allowing people to build relationships and share information. They offer services that allow you to connect with other people of similar interests and background. Usually they consist of a profile, various ways to interact with other users, ability to setup groups, etc. Popular social media Tools include:
  - **Blogs and Forums:** Online forums allow members to hold conversations by posting messages. Blog comments are similar except they are attached to blogs and usually the discussion centers around the topic of the blog post.
  - **Microblogging:** Services that focus on short updates that are pushed out to anyone subscribed to receive the updates. Twitter is the most popular.
  - **Social Networks :** Services that allow you to connect with other people of similar interests and background. Usually they consist of a profile, various ways to interact with other users, ability to setup groups, etc Facebook, LinkedIn, Google Plus, Orkut for networking
  - **Bookmarking Sites:** Services that allow you to save, organize and manage links to various websites and resources around the internet. Most allow you to “tag” your links to make them easy to search and share. The most popular are Delicious and StumbleUpon.
  - **Social News:** Services that allow people to post various news items or links to outside articles and then allows it’s users to “vote” on the items. The voting is the core social aspect as the items that get the most votes are displayed the most prominently. The community decides which news items get seen by more people. The most popular are Digg and Reddit.
  - **Media Sharing:** Services that allow you to upload and share various media such as pictures and video. Most services have additional social features such as profiles, commenting, etc. The most popular are YouTube & Flickr



Keep in mind that, while these are the 6 different types of social media, there can be overlap among the various services. For instance, Facebook has microblogging features with their “status update”. Also, Flickr and YouTube have comment systems similar to that of blogs.

It may help to imagine the Web as a huge de-centralised library. If you need information about, for example, cinema in France, forest destruction in South America, or job vacancies in Ireland, then all of that information is probably available somewhere on the Web.

## 2.3 The History Of The Internet And The Web

It is useful to know about the historical development of the Internet and the Web, since this can help you to understand how they work, and why they exist in their current forms. It also helps to highlight the difference between the Internet and the World Wide Web. People often mistakenly refer to the Web as "the Internet", and fail to realise that the Web is only one recently-invented way of *using* the Internet.

How was the Internet invented?

The Internet is actually more of a historical accident than an invention. The original concept was developed during the Cold War, as a way of defending America's communication channels against nuclear attack. The RAND Corporation, a U.S. think-tank, was given the task of devising a communications system that could still function, even if a substantial portion of it were to be destroyed.

Such a structure obviously had to be de-centralised, since a large central control site would be too obvious a target. For this reason, it was decided to use a **network** of interlinked computers, each of which had equal authority to send, pass and receive messages. The messages themselves would be split up into "packets" (small units of data) each of which would be "addressed" to the recipient computer. But the exact route that each packet took through the network would be unimportant. If part of the network were destroyed, the packets would simply take another route to their destination, using the remaining network links.

An early network of four computers, operating on these principles, was established in 1969, and was called ARPANET. By 1972, this had grown to 37 computers, as scientists and researchers began to appreciate the advantages of rapid communication. Through the '70s, additional networks were developed, linked to one another by computers called "**gateways**". Newsgroups and mailing lists were set up to facilitate communications between users. The introduction of the Joint Academic Network (**JANET**) in the UK in 1984, and the National Science Foundation Network (NSFNET) in 1986 laid the foundations of what we now know as the Internet - a loose coalition of networks, all linked together by means of gateways, supporting millions of users world-wide.

### 2.3.1 Where does the Web fit into the story?

The World Wide Web is a more recent development. The concept was devised in 1989 by Tim Berners-Lee, a communications specialist at the European Particle Physics Laboratory, 'CERN' in Geneva. He wanted to design an information system by which researchers could share their results as rapidly as possible, over the Internet. In order to navigate through this information, he chose to use a system of "**hypertext**", which allows documents to be linked, using key words, to other similar documents elsewhere on the Internet. The language in which Berners-Lee's Web documents were authored, and in which the hypertext links were defined, became known as Hypertext Markup Language (**HTML**).

At first, the Web was used only as an experimental tool for exchanging text-based information. However, the development of more advanced, graphical Web-viewing programs (also called "**browsers**") such as **Mosaic** and **Netscape**, in 1993-94 enabled people to publish and view Web documents which contained pictures. More recently, these capabilities have been extended to include sound and video content. These recent developments have led to a dramatic increase in the popularity of the Web, which is today the most high-profile application of Internet technology.

### **2.3.2 What practical use is the Web to me?**

Learning how to use the Web, and how to publish your own information on it, is not prohibitively difficult. However, it does require an investment of both time and patience. Before making this investment, it is sensible to consider what uses might be made of the Web in a University context, and what motivations an educator might have for wishing to learn about this technology.

There is no doubt that Web is already changing the ways in which people present information to one another. Increasingly, educators are realising the potential of this new technology for developing learning resources that are both more **interactive** and contain more **multimedia** content than traditional lecture and text-book approaches. By combining video and sound clips with static pictures and text, it is possible to make information both clearer and more interesting to use. By incorporating form-based "multiple choice"-type quizzes into Web pages, it is possible to test whether the user has understood the material fully.

In addition, the use of **hypertext** on the Web means that documents can be logically linked to one another. Information can therefore be presented in a non-linear format, and material can be linked to other related resources elsewhere on the Internet. The accessibility of the Web increases the potential for "distance-learning" and "asynchronous learning", i.e. freeing the students to choose their own times and places for study, and to progress through the material at their own rate.



### **2.3.3 What's the relationship between the WWW and the Internet?**

The World Wide Web is just one of the many services that the Internet provides. Some other services provided by the Internet are email, FTP, gopher, telnet and usenet.

Almost every protocol type available on the Internet is accessible on the web including the following components: Email, FTP, Telnet, User news, HTTP

### **2.3.4 Features of WWW**

- ✓ It has its own protocol i.e. HTTP
- ✓ It creates a convenient and user friendly environment
- ✓ It is the fastest components of Internet since it gathers together all the protocols into a single system.
- ✓ It relies on the hypertext as means of Information retrieval.
- ✓ It has the ability to work with multimedia and advanced programming languages i.e. text, graphics, video and audio.
- ✓ It is a delivery medium, content provider and subject matter.
- ✓ It connects users to almost any part of the Internet.
- ✓ It is used to explore intellectual, verbal knowledge and effective learning.
- ✓ It contains complex virtual web of connections and consist of files.
- ✓ It provides real-time collaboration, interactive pages and automatic push of information to client computers.

## **2.4 Internet/Web Terminologies**

### **(1) FTP**

The Internet allows you to copy files between your computer and other computers on the Internet by using file transfer protocol (ftp). You connect your computer to an ftp server, an Internet host computer that stores files for transfer. You may be required to log in to retrieve a file, which varies from software, and text files to graphic files.

### **(2) TCP/IP ( Transmission Control Protocol/Internet Protocol)**

A special set of protocols that is used to send data. It is a reliable connection oriented protocol that controls and manages application level services between computers. Some applications use it for transport.  
in a more reliable way.

### **(3) E-mail**

This is online communication between computer users. It is quick, convenient, efficient and cheap way to communicate with both individuals and groups.

### **(4) TELNET**

It's a service that enables remote log in. Users are permitted to log in onto a host and perform tasks as if they are working on the remote computer itself.

## **(5) USENET/newsgroups, mailing lists**

A huge network of discussion groups

## **(6) IRC**

This is an Internet service that allows a number of users to connect to the same network node and communicate in real time.

## **2.5 Web Site planning guiding principles.**

### **2.5.1 Factors to be considered while planning for a website**

#### **a. Audience**

What do you want to communicate and to whom. A site influenced by “accurate” information about its intended and actual audience has a higher probability of successfully communicating its intended

message and information. When planning for this, define the target audience and critical information about them.

#### **b. Purpose Statement**

This should be clear in regard to the following elements;

***Subject area:*** What do we want to convey?

***Audience:*** Whom do we want to convey to?

***Level of detail:*** How much information do we gather and maintain for dissemination on the web?

***Users expected benefit or response:*** What will the users of the website gain from it?

#### **c. Domain Information**

Define what information is necessary for the developers to know and what information will be provided to users. E.g Are there specialized databases to which developers or users must gain access? Is there an existing store of online material that will serve as a basis for user information Plan for the acquisition of domain information. e.g. how will the information be obtained?

Plan for updating and maintaining the information

How will it be updated?

Who will update?

What will be the costs? etc.

#### **d. Web Specification**

This document acts as a guidebook for the designers and implementers. The main objective is to note that the developers have tools, training and time necessary to develop the site according to the specifications

**Note:**

The web plan should flow out of and be integrated with the strategic plan of an organization.

**e. Web Hosting**

A web host stores web site and transmits it to the internet for public view. To put a web site on the internet you need three things: a domain name, a web site and a web host account. How it works is that your domain name points to your web host's server which is where your site is physically housed. Pages are uploaded to web server using FTP. Hosting your web site on your own server is always an option. Here are some

problems to consider:

**i. Hardware Expenses**

To run a "real" web site, you will have to buy some powerful server hardware. Don't expect that a low cost PC will do the job. You will also need a permanent (24 hours a day ) high speed connection to your office, and such connections are expensive.

**ii. Software Expenses**

Don't forget to count the extra cost for software licenses. Remember that server licenses often are much higher than client licenses. Also note that some server software licenses might have limits on number of concurrent users.

**iv. Labor Expenses**

Don't expect low labor expenses. Remember that you have to install your own hardware and software. You also have to deal with bugs and viruses, and keep your server constantly running in an environment where "everything could happen". Renting a server from an Internet Service Provider (ISP) is a common option.

Here are some advantages:

**I) Connection Speed**

Most providers have very fast connections to the Internet.

**II) Powerful Hardware**

Service providers often have many powerful web servers that can be shared by several companies. You can also expect them to have an effective load balancing, and necessary backup servers.

**III) Security and Stability**

Internet Service Providers are specialists on web hosting. Expect their servers to have more than 99% up time, the latest software patches, and the best virus protection.

Things to consider:

### **24-hour support**

Make sure your Internet service provider offers 24-hours support. Don't put yourself in a situation where you cannot fix critical problems without having to wait until the next working day. Toll-free phone could be vital if you don't want to pay for long distance calls.

### **Daily Backup**

Make sure your service provider runs a secure daily backup routine; otherwise you may lose some valuable data.

### **Traffic Volume**

Study the provider's traffic volume restrictions. Make sure that you don't have to pay a fortune for unexpected high traffic if your web site becomes popular.

### **Bandwidth or Content Restrictions**

Study the provider's bandwidth and content restrictions. If you plan to publish pictures or broadcast video or sound, make sure that you can.

### **Email Capabilities**

Make sure your provider fully supports the email capabilities you need.

### **Database Access**

Make sure your provider fully supports the database access you need if you plan to use databases from your site.

## **2.6 Types Of Web Documents**

### **2.6.1 Static document**

Documents are stored as a file on a server. The same content is delivered every time that URL is accessed.

#### **Advantages:**

They are simple, reliable, fast and the documents can be cached locally at a client.

#### **Disadvantages:**

Inflexible as content can only be changed by updating the file and Information can become boring easily.

### **2.6.2 Dynamic documents**

The documents are created by a program like CGI -script.

#### **Advantages**

Information is timely and always reflects the latest information.

#### **Disadvantages**

They are not reliable.

Require high cost of executing and maintenance.

Slow to access

### **2.6.3 Active documents**

These are documents that contain executable elements that are executed by the client on arrival.

Executable elements are in script language such as JavaScript, Active X, Java applets e.t.c

#### **Advantages**

Documents reflect the latest information.

#### **Disadvantages**

High cost of execution and maintenance. It is complex and poses great security risks from servers and codes.

## **2.7 Web design tools**

HTML documents are plain-text (also known as ASCII) files that can be created using any text editor. E.g. of text editors include notepad, web-edit, word processors like MS Word, DOS edit, Netscape composer

Some WYSIWYG editors can also be used e.g. Front page, Outlook Express, Claris Home page, Adobe PageMill. One can also use graphic tools like Photoshop, Paint, Animated GIF construction set, PageMaker etc.

### **Self Evaluation**

1. Define:
  - a. Web hosting
  - b. Web server
2. Discuss guiding principles during web development
3. What are the factors to consider in leasing a web server space

## CHAPTER 3: CREATING YOUR FIRST WEB PAGE

### 3.1 Chapter Overview

In this chapter you will explore the variable factors that affect Web design. You will learn how the Hypertext Markup Language (HTML), the language used to create documents on the World Wide Web, is constantly evolving, and preview the new markup language that will change how you design for the Web. You will see how Web browsers affect the way users view your content, and how variations in the user's browser choice, screen resolution, and connection speed pose specific challenges to creating Web pages that display properly in different computing platforms. Finally, you will consider what type of software tool you should use to create your HTML code.

When you complete this chapter, you will be able to:

- 👉 Describe the current state of HTML
- 👉 Describe how Web browsers display your work
- 👉 Code for multiple screen resolutions
- 👉 Understand bandwidth concerns
- 👉 Create a HTML document

### 3.2 What Is Html?

HTML stands for HyperText Markup Language and HyperLink is the link that links one file to another. You markup text files with HTML tags, which are pieces of code enclosed by the less than sign ( < ) and a greater than sign ( > ). Web browser reads these tags when formatting HTML files onscreen.

### 3.3 How HTML Works

Browser sends request for HTML files to remote servers on the Internet by using addresses called URLs (Uniform Resource Locator). When data returns, the browser interprets the HTML tags and displays the data as WEB pages.

The Server you are connected to routes request from your system to other server on the Internet, and transmits the HTML files back to you.

The Internet is a worldwide network of servers. Your request bounces from server to server until the URL address of the HTML file you want is found. The data is then returned over the Internet to your system.

The WEB server holds the HTML file that you are looking for.

### 3.4 HTML: Then and Now

When Tim Berners-Lee first proposed HTML at the European Laboratory for Particle Physics (CERN) in 1989, he was looking for a way to manage and share large amounts of information among colleagues. As the idea developed, Berners-Lee named the mesh the World Wide Web.

He created an application of the **Standard Generalized Markup Language (SGML)**, a standard system for specifying document structure, and called it the Hypertext Markup Language.

When Berners-Lee created HTML, he adopted only the elements of SGML necessary for representing basic office documents such as memos and reports. The need for new markup languages and standards to address these demands is handled by the **World Wide Web Consortium (W3C)**.

### **3.5 HTML and the World Wide Web Consortium**

After the initial surge of interest in HTML and the Web, a need arose for a standards organization to set recommended practices that would guarantee the open nature of the Web. The W3C was founded in 1994 at the Massachusetts Institute of Technology to meet this need. The W3C, led by Tim Berners-Lee, sets standards for HTML and provides an open, non-proprietary forum for industry and academic representatives to add to the evolution of this new medium.

### **3.6 The Limitations of HTML**

HTML is a **markup language**, a structured language that lets you identify common sections of a document such as headings, paragraphs, and lists. An HTML file includes text and HTML markup elements that identify these sections. The HTML markup elements indicate how the document sections appear in a browser. For example, the <h1> element tags in the following code indicate that the text is first-level heading:

```
<h1>Welcome to My Web Page</h1>
```

HTML adopts many features of SGML, including the cross-platform compatibility that allows different computers to download and read the same file from the Web. HTML is not a What You See Is What You Get (WYSIWYG) layout tool. It was intended only to express logical document structure, not formatting characteristics. Because HTML was not designed as a layout language, many editing programs create less-than-standard code to accomplish a certain effect. You cannot rely on the HTML editor's WYSIWYG view to test your Web pages. Despite its limitations, HTML is ideal for the Web because it is an open, non-proprietary, cross-platform compatible language.

### **3.7 The Need for Style Sheets**

Browser developers to help HTML authors bypass the design limitations of HTML introduced style elements such as <font>. Designers and writers who are accustomed to working with today's full-featured word processing programs want the same ability to manipulate and position objects precisely on a Web page as they can on the printed page. Mixing style information within the structure, as is the case in most of the Web today, limits the cross-platform compatibility of the content. This separation of style and structure was accomplished in 1996 by the W3C's specification for a Web style language. The style language, named **Cascading Style**

**Sheets (CSS)**, allows authors to create style rules for elements and express them externally in a document known as a **style sheet**.

### **3.8 How Web Browsers Affect Your Work**

One of the greatest challenges facing HTML authors is designing pages that multiple browsers can display properly. Every browser contains a program called a **parser** that interprets the markup tags in an HTML file and displays the results in the **canvas area** of the browser interface.

#### **3.8.1 Browser Compatibility Issues**

As different browsers competed for market share, a set of proprietary HTML elements evolved for the use of each particular browser. Some examples of these elements are `<font>` and `<center>`, which were developed specifically for the Netscape browser. **Deprecated elements** are those that the W3C has identified as obsolete and will not be included in future releases of HTML. However, it is likely that browsers will support such elements for some time. Avoid using proprietary elements unless you are sure that your audience is using only the browser for which they were designed. The newer browsers offer much better support for the standards released by the W3C. Most HTML authors do not have the luxury of knowing the age, type, or operating system of the browser that will be used to view their Web pages.

#### **3.8.2 Creating Cross-Browser Compatible Pages**

How can you handle the demands of different browsers while designing attractive Web pages? Some HTML authors suggest that you use an older version of HTML to ensure portability. Others say that you should push the medium forward by coding to the latest standard and using the most recent enhancements.

##### ***Lowest Common Denominator Coding***

Although it can be difficult to create pages that are always displayed properly, it is not impossible. One way to create portable pages is to use a lowest-common-denominator approach. This approach provides the greatest acceptance across browsers, because the authors choose to code their HTML using the next-to-last release of HTML.

#### **3.8.3 Cutting-Edge Coding**

Another strategy to adopt when designing your Web site is to stay at the cutting edge. By requiring the latest software, some designers insist that their users keep up with them. This design strategy can result in visually exciting and interactive sites that keep pace with the latest technology. **Plug-ins** are helper applications that assist a browser in rendering a special effect. Without the plug-in, your user will not see the results of your work.

##### ***Browser-Specific Coding***

Some Web sites are coded for one particular browser or brand of browsers only. The author may have wanted to use a unique enhancement for the site, or may have found that the site did not render properly in other browsers. On the Web, you never can be sure of the type of browser



your user has. However, this method of browser-specific coding may be viable on a company **intranet**, where you know or you can specify that all users have the same brand and version of browser.

### ***Solving the Browser Dilemma***

You must test your work in as many browsers as possible during the entire development process to make sure that your pages will render properly. Knowing your audience is a major step towards correctly implementing your site.

### **3.8.4 Coding for Multiple Screen Resolutions**

A computer monitor's **screen resolution** is the width and height of the computer screen in pixels. Most monitors can be set to at least two resolutions, whereas larger monitors have a broader range from which to choose. Screen resolution is a function of the monitor's capabilities and the computer's video card. The three most common screen resolutions (traditionally expressed as width X height in pixels) are 640 X 480, 800 X 600, and 1024 X 768.

### **3.9 Bandwidth Concerns**

Connection speed is another factor that should influence your Web page design. Most users simply will not wait longer than 10-20 seconds for a page to load. If your pages download slowly, your users probably click to go to another site before they see even a portion of your content. Because the single biggest factor influencing the speed at which your pages render is the size and number of graphics on your Web pages, you should keep your page designs simple with few graphics.

### **3.10 Working with the Cache**

All Web pages are stored on computers called Web servers. When you type a Uniform Resource Locator (URL) address in your browser, it connects to the appropriate Web server and requests the file you specified. The browser checks to see if it has any of the specified images stored locally on the computer's hard drive in the **cache**. The cache is the browser's temporary storage area for Web pages and images. You can make use of the browser's caching capabilities by reusing graphics as much as possible throughout your site.

### **3.11 Should You Use an HTML Editor?**

There are a variety of HTML editing programs, such as Adobe PageMill, Microsoft FrontPage, and Macromedia Dreamweaver, to name a few. Some code-based HTML editors, such as Macromedia HomeSite, forgo a WYSIWYG approach. They have become popular because they include many powerful enhancements that Notepad lacks, such as multiple search-and-replace features and syntax checking, while still allowing you to manipulate code at the tag level.

Many of the latest office applications now convert documents to HTML. As with the browsers, authoring packages interpret tags based on their own built-in logic. Therefore, a page that you

create in an editing package may look quite different in the editing interface than it does in a browser.

### 3.12 Creating a New HTML Document

After being exposed to some HTML facts, it is time to create a new WEB document. Creating a new WEB document is not as hard as some of you might think. All you need is a text editor like notepad or WordPad and you can begin to type your code in it. After keying all your codes, you need to save your HTML document like any other document in other application software. Proceed with the following procedure to save the document:

#### 3.12.1 Saving a HTML Document

After keying all your codes, you need to save your HTML document like any other document in other application software. Proceed with the following procedure to save the document:

1. Click on the File Menu
2. Select Save
3. Type a file name as index.html
4. Click on the Save in drop list
5. Select Desktop
6. Click on Save button



#### 3.12.2 Closing and Opening a HTML Document

After saving your document, to close the document you need to follow the steps below:

1. Click on the File Menu
2. Select exit

After you close your document, you realised that there are certain things that you need to modify. You need to re-open your document. Follow the steps below:

1. Click on the Start button
2. Move your mouse over to program
3. Move over to accessories
4. Select notepad
5. Click on File menu
6. Select open
7. Click on the Save in drop list
8. Select Desktop
9. Click on the name of your document
10. Click on Open button

### **3.12.3 Loading a HTML Document in a Local Browser**

After typing several codes, you have to load your document to a browser to view how it looks like. Proceed with the steps below:

1. Click on File menu
2. Select save
3. Minimise the notepad
4. Launch internet browser such as Netscape Navigator or Internet Explorer
5. Click on File menu of the browser
6. Select open
7. Click on Browse button
8. Click on the Save in drop list
9. Select Desktop
10. Select your document name
11. Click on open button
12. Click on OK button
13. Once you load your document, the next time you change or modify your codes, all you need to do is to save the changes and click on the Refresh button on the browser.

## **ASSIGNMENT**

### **Discussion Questions**

1. Discuss some browser compatibility issues.
2. Discuss when it would be appropriate to use a HTML code generating tool(e.g. Dreamweaver) as opposed to a text editor like Notepad.

### **Projects to Assign**

Examine the HTML source code for some professionally designed web pages, and compare that with what you created in the chapter exercises. Chart some differences or similarities.

## CHAPTER 4: HTML ELEMENTS, TAGS & ATTRIBUTES

### 4.1 Chapter overview

Upon completion of this lesson, students should be able to;

- 👉 Plan and create a HTML document
- 👉 Preview and edit a Web page
- 👉 Describe the use of the head and body elements in an HTML document.
- 👉 Describe these types of HTML tags: opening, closing, and empty
- 👉 Describe the use of attributes within HTML tags.
- 👉 Describe the use of HTML comments and whitespace

### 4.2 HTML - Elements

When you land on a website, all the items you see in front of you -- the paragraph texts, the page banners, and the navigation links are all *elements* of the web page. The term element is a just a name given to any piece of a web page. Many HTML elements are actually invisible to visitors and work quietly behind the scenes to provide web crawlers and search engines useful information about the site.

An element consists of three essential pieces: an opening tag, the content, and a closing tag.

1. `<p>` - opening paragraph tag
2. **Element Content** - "Once upon a time..."
3. `</p>` - closing tag

#### A Complete HTML Element:

```
<p>Once upon a time...</p>
```

A single page can contain hundreds or thousands of elements, but when all is said and done, every HTML page should have a bare minimum of four critical elements: the *HTML*, *head*, *title*, and *body* elements.

#### `<html>` Element...`</html>`

`<html>` is the element that begins and ends each and every web page. Its sole purpose is to hold each web element nicely in position and serves the role of book cover; all other HTML elements are encapsulated within the `<html>` element. The tag also lets web browsers know, "Hey, I'm an HTML web page!", so that the browser knows how to render it. Remember to close your HTML documents with the corresponding `</html>` tag to signify the end of the HTML document.

If you haven't already, it is time to open up Notepad or Notepad++ and have a new, blank document ready to go. Copy the following HTML code into your text editor.

#### HTML Element Code:

```
<html>  
</html>
```

Now save your file by selecting - **Menu** and then **Save**. Click on the **Save as Type** drop down box and select the option **All Files**. When you're asked to name your file, name it - *index.html*. Double-check that you did everything correctly and then press - **Save**. Now open your file in a new web browser so that you have the ability to *refresh* the page and see any new changes.

If you opened up your *index.html* document, you should be staring at your very first blank (white) web page!

#### 4.2.1 <head> Element

The <head> is usually the first element contained inside the <html> element. While it is also an element container that encapsulates other HTML elements, these elements are not directly displayed by a web browser. Instead they function behind the scenes, providing more descriptive information about the HTML document, like its page title and other meta data. Other elements used for scripting (JavaScript) and formatting (CSS) are also contained within the <head> element, and we will eventually introduce how to utilize those languages. For now, the head element will continue to lay empty except for the *title* element, which will be introduced next.

Here's a sample of the initial setup.

##### HTML Head Element Code:

```
<html>
  <head>
</head>
</html>
```

If you've made the code changes and refreshed the browser page, you will notice that we still have nothing happening on the page. So far, all we've done is add a couple of necessary elements that describe the web page document to the web browser. Content -- the stuff you can see -- will come next!

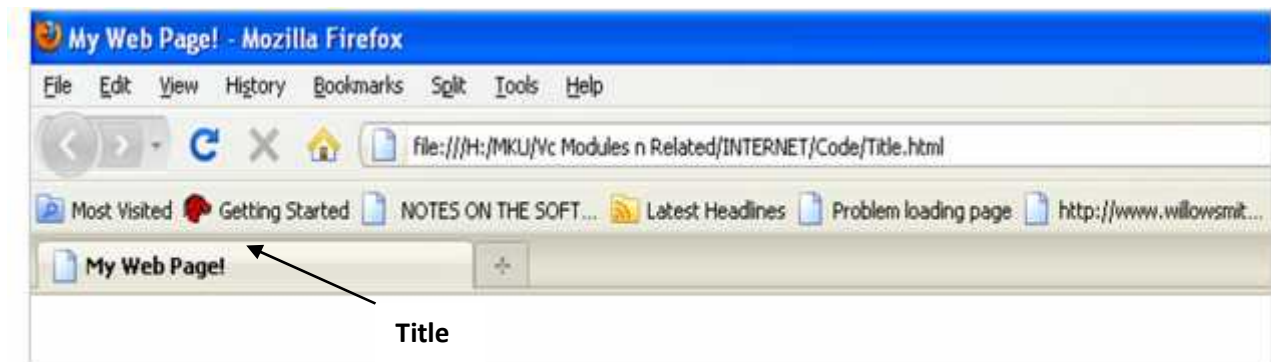
#### 4.2.2 <title> Element

The <title> element adds a title to a web page. Web page titles are displayed at the top of any browser window or at the top of browser tabs. They are probably the first thing seen by web surfers as pages are loaded, and web pages you bookmark are saved using the web pages' titles. A proper title makes a good first impression, and any page caught without a title is considered unprofessional, at best.

##### HTML Title Element Code:

```
<html>
  <head>
    <title>My Web Page!</title>
  </head>
</html>
```

Save the file and refresh the browser. You should see "My Web Page!" in the upper-left bar of your browser window.

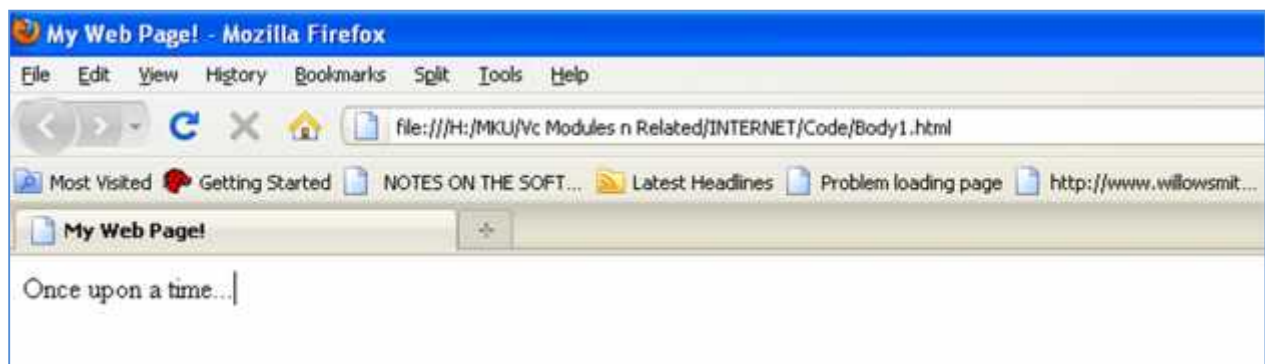


#### 4.2.3 Body Element <body>

The element that encapsulates all the visual elements (paragraphs, pictures, tables, etc.) of a web page is the <body> element. The <body> tag serves as the element containing all the content for the website. Tables, lists, forms, paragraphs, and everything else must be placed within the body element to ensure each element is displayed on your site.

##### HTML Body Element Code:

```
<html>
  <head>
    <title>My Web Page!</title>
  </head>
  <body>
    <p>Once upon a time...</p>
  </body>
</html>
```



### 4.2.3.1 Body Margins

#### Unique Attributes

- **leftmargin** - Sets a left hand margin for your body element.
- **topmargin** - Sets a margin along the top of your body element.

A unique set of margin attributes are available to the body tag. These attributes work much like those of a word processing program, allowing you set pixel value margins for the left, right, top, or bottom of your website. Setting these attributes means that all the content you place within your body tags will honor the preset margin.

#### HTML Code:

```
<body topmargin="50">  
<body leftmargin="50">
```

### 4.2.3.2 Base Text

The text attribute sets the text color of all text contained within the body tags. Think of it as a means to set the color of your text, unless otherwise noted. Basically, you use these tags to set a base color scheme, which you can later modify through additional tags inside of the body.

#### HTML Code:

```
<body text="red" >
```

**or**

```
<body text="rgb(255,0,0)" >
```

### 4.2.3.3 Base Links

Along the same lines, we may also specify base colors for visited or unvisited links. This method has deprecated, and we recommend that you use Cascading Style Sheets (CSS) instead.

#### HTML Code:

```
<body link="white" vlink="black" >
```

**or**

```
<body link="rgb(255,255,255)" vlink="rgb(0,0,0)" >
```

Setting a baselink is a great way to ensure your viewers will not receive the annoying error message that occurs with broken links.

### 4.2.3 Elements Reviewed

To recap quickly: we've laid out a set of four essential elements that are used to create a strong foundation and structure for your web page. The <html> element encapsulates **all** page content and elements, including two special elements: the <head> and <body> elements. The <head> element is a smaller container for elements that work behind the scenes of web pages, while the <body> element houses content elements such as web forms, text, images, and web video.

From here on out, the examples we provide will assume that you have a firm understanding of these key elements and know to place the majority of your HTML code inside of the <body> element.

## 4.3 HTML Tags

A web browser reads an HTML document from top to bottom, left to right. Each time the browser finds a tag, the tag is rendered accordingly. Paragraph tags render paragraph text, image tags render images, etc. By adding tags to an HTML document, you are not only coding HTML, but also signaling to the browser, "Hey look, this is paragraph text; now treat it as such!"

Do you recall that HTML elements are comprised of three major parts: the opening tag, the content, and the closing tag? As you will learn, there are infinite combinations of HTML tags/elements, including those used for forms, images, and lists. Everything displayed on an web page requires the use of a tag or two.

### HTML Tag Code:

```
<tag>Content</tag>
```

```
<p>This text will be rendered like a paragraph.</p>
```

Tags should always be written in lowercase letters if you plan on publishing the page online, as this is the web standard for XHTML and Dynamic HTML.

### HTML More Tag Code:

```
<body>Body Tag
<p>Paragraph Tag</p>
<h2>Heading Tag</h2>
<b>Bold Tag</b>
<i>Italic Tag</i>
</body>
```

#### 4.3.1 Logical vs. Physical Tags

In HTML there are both logical tags and physical tags.

**Logical (*Idiomatic Elements*) tags** are designed to describe (to the browser) the enclosed text's meaning. A logical character tag describes how the text is being used, not necessarily how it is



formatted. An example of a logical tag is the `<strong> </strong>` tag. By placing text in between these tags you are telling the browser that the text has some greater importance. By default all browsers make the text appear bold when in between the `<strong>` and `</strong>` tags.

Tag	Description	Renders as
<code>&lt;cite&gt;</code>	citation	emphasizes the text in italics.
<code>&lt;code&gt;</code>	code sample	Displays some characters as code usually in Courier font (i.e., fixed-width font)
<code>&lt;del&gt;</code>	deleted text	displays text with a line through it; renders differently in Netscape and Internet Explorer
<code>&lt;dfn&gt;</code>	definition	italics; renders differently in Netscape and Internet Explorer
<code>&lt;em&gt;</code>	emphasis	emphasizes the text in some way usually as italic.
<code>&lt;ins&gt;</code>	inserted text	underlined; renders differently in Netscape and Internet Explorer
<code>&lt;kbd&gt;</code>	code sample	fixed-width font
<code>&lt;samp&gt;</code>	code sample	fixed-width font
<code>&lt;strong&gt;</code>	strong	Text is emphasized more strongly than usually as bold.
<code>&lt;var&gt;</code>	program variable	italics

### *What's so logical about logical tags?*

The original intent of HTML was to mark up text to indicate the *purpose* of each part of the document. Text within the `<H1 ...>` is a header, text within `<CODE>` is code from a program.

Although logical tags indicate different types of information, most of them are usually rendered in one of just a few ways: italics, bold, or monospace (all characters the same width):

<i>This is emphasized text</i> <i>This is cited text</i> <b>This is strong text</b>	This is sample text This is code text This is keyboard text
---	---

So if logical tags just look like bold or italics, why use them at all? Why not just use `<B>` when you want bold? Logical tags have lost favor to "formatting" tags, which indicate the physical appearance of the document (for example `<B>` for BOLD). This loss of popularity is unfortunate, because logical tags still serve some important purposes:

- Logical tags allow the browser to render that information in the manner most appropriate for that browser. Text that should be emphasized (`<EM>`) may be best emphasized in Windows with italics, and bold in Unix.
- Logical tags help you, the author, keep track of what you are saying, without the distraction of presentation. If you need to indicate someone's address, use `<ADDRESS>`, knowing it will be presented in an appropriate manner.

**Physical tags** on the other hand provide specific instructions on how to display the text they enclose. They controls how the characters are formatted. Examples of physical tags include:

Tag	Description	Renders as
<b>	bold	displays text as bold
<big>	big	displays text in a big font
<i>	italics	displays text as italic
<s> or <strike> *	strike	displays text with a line through it
<small>	small	displays text in a small font
<sub>	subscript	displays the text as subscript — text that displays below the baseline of the text
<sup>	superscript	displays the text as superscript — text that has baseline above the baseline of the rest of the text
<tt>	teletype	displays the text with fixed-width font
<u>	underline	underlines the text

\* Both <s> and <strike> tags are deprecated in HTML 4.0 so instead use the <del> tag.

Physical tags were invented to add style to HTML pages because style sheets were not around, though the original intention of HTML was to not have physical tags. Rather than use physical tags to style your HTML pages, you should use style sheets.

#### 4.3.2 Nested Tags

When you enclose an element in with multiple tags, the last tag opened should be the first tag closed.

For example:

```
<p><b><em>This is NOT the proper way to close nested tags.</p></em></b>
```

```
<p><b><em>This is the proper way to close nested tags. </em></b></p>
```

**Note:** It doesn't matter which tag is first, but they must be closed in the proper order.

#### 4.3.3 Elements Without Closing Tags

There are a few elements that do not require formal closing tags. In a way, they still have the 3 parts (opening/closing and content), but they are consolidated into one tag. The reason for this is that these tags do not really require any content to be placed within them, but sometimes may require attributes such as source URLs for images.

One prime, easy example of an HTML tag that does not require a closing tag is the line break tag.

**HTML Line Break Code:**

```
<br />
```

To tell the browser we want to place a line break (carriage return) onto the site, it is not necessary to type `<br>linebreak</br>`. This would require a tremendous amount of effort, and if every line break tag needed all three components as other tags do, life would become redundant real quick. The better solution is to combine the opening and closing tags into a single format and shorten the number of characters needed to render a line break. Other tags, such as image tags and input tags, have also been modified in such a manner.

#### HTML Code:

```
 -- Image Tag  
<br /> -- Line Break Tag  
<input type="text" size="12" /> -- Input Field
```

As you can see from the above, the web browser is capable of interpreting the image tag as long as we inform the browser where the image file is located, using the *src* attribute. Some HTML elements have combined closing tags.

### 4.4 HTML Attributes

Tags can have attributes. Attributes can provide additional information about the HTML elements on your page. The `<tag>` tells the browser to do something, while the attribute tells the browser how to do it. For instance, if we add the *bgcolor* attribute, we can tell the browser that the background color of your page should be blue, like this:

```
<body bgcolor="blue">
```

This tag defines an HTML table: `<table>`. With an added border attribute, you can tell the browser that the table should have no borders:

```
<table border="0">
```

Attributes always come in name/value pairs like this: `name="value"`. Attributes are always added to the start tag of an HTML element and the value is surrounded by quotes. In short, attributes are like blue print schematics informing the browser how to render an HTML element. As an HTML tag is processed, the web browser looks to these attributes as guides for the construction of web elements. Without any attribute values specified, the browser will render the element using the default setting(s). Common examples of attributes;

#### 4.4.1 Title Attribute

The *title* attribute titles an HTML element and adds a tiny text pop-up to any HTML element, offering your web viewers a tool-tip mechanism where information can be found or where a better description of an HTML element can be seen.

Much like the tool tips found in word processing programs, this attribute can add spice to any page while offering the user priceless interactivity. As the mouse hovers over the element, a tool tip is displayed, giving the viewer just one extra piece of information.

**HTML Title Attribute:**

```
<h2 title="Hello There!">Titled Heading Tag</h2>
```

Hover your mouse over the display heading and watch the *title* attribute in action!



The *title* attribute is one that has not deprecated and should be used often. Many search engines are capable of identifying this attribute inside your HTML elements, granting increased search rankings based on the relevance of the *title* attribute text.

**4.4.2 Align Attribute**

If you wish to change the horizontal alignment of your elements, you may do so using the *align* attribute. It allows you to align things left, right, or center. By default, most elements are automatically aligned left, unless otherwise specified.

**HTML Align Attribute:**

```
<h2 align="center">Centered Heading</h2>
```

**HTML Align Attribute Code:**

```
<h2 align="left">Left-aligned heading</h2>
```

```
<h2 align="center">Centered Heading</h2>
```

```
<h2 align="right">Right-aligned heading</h2>
```



HTML attributes, including *align*, used to be the primary source for the customization of web elements, but they have now lost their market share to Cascading Style Sheets (CSS). Since most HTML attributes are now deprecated, they should ultimately be avoided in professional-level web design. Nonetheless, having an understanding of HTML attributes will prove to be a tremendous aid for anybody looking to move into professional web development using Cascading Style Sheets.

### Tips

- Use several different attributes to enhance a tag.
- Completely customize your site through various tag/attribute combos.
- Use the title attribute to give your user a more interactive experience and to help your SEO rank!

## 4.5 Font

The `<font>` tag provides no real functionality by itself, but with the help of a few attributes, this tag is used to change the style, size, and color of HTML text elements. The *size*, *color*, and *face* attributes can be used all at once or individually, providing users with the ability to create dynamic font styles for any HTML element.

**Note:** The `<font>` and `<basefont>` tags are **deprecated** and should not be used for published work. Instead, use CSS styles to manipulate your font.

### 4.5.1 Font Size

Set the size of your font with *size*. The range of accepted values goes from 1 -- the smallest, to 7 -- the largest. The default size of a font is 3.

#### HTML Font Size Code:

```
<p>  
<font size="5">Here is a size 5 font</font>  
</p>
```

#### HTML Font Size Attribute:

Here is a size 5 font.

## 4.5.2 Font Color

Set the color of your font with *color*.

### HTML Font Color Code:

```
<font color="#990000"> This text is hex color #990000</font>  
<br />
```

```
<font color="red">This text is red</font>
```

### HTML Font Color Attribute:

This text is hex color #990000

This text is red

## 4.5.3 Font Face

Choose a different font face by specifying any font you have installed. Font face is synonymous with font type. As a web designer, be aware that if you specify a custom font type and users viewing the page don't have the exact same font installed, they will not be able to see it. Instead the chosen font will default to Times New Roman. To reduce the risk of running into this situation, you may provide a list of several fonts with the *face* attribute, such as outlined below.

### HTML Font Face Code:

```
<p>  
<font face="Georgia, Arial, Garamond"> This paragraph has had its font...</font>  
</p>
```

### HTML Font Face Attribute:

This paragraph has had its font formatted by the font tag!

In the example above, we have changed the font face (type) of a paragraph element and specified a list of similar fonts to apply to this element in the case that some of our viewers do not have these fonts installed.

## HTML Font - Attribute Review

### HTML Font Attributes:

Attribute=	"Value"	Description
size=	"Num. Value 1-7"	Size of your text, with 7 being biggest
color=	"rgb,name,or hexadecimal"	Font color
face=	"name of font"	Font type



#### HTML Code:

```
<p><font size="7" face="Georgia, Arial" color="maroon">Customize  
your font to achieve a desired look.</font></p>
```

## 4.6 Body Attributes

The **<body>** tag has two attributes where you can specify backgrounds. The background can be a color or an image.

### 4.6.1 Bgcolor

The bgcolor attribute specifies a background-color for an HTML page. The value of this attribute can be a hexadecimal number, an RGB value, or a color name:

```
<body bgcolor="#000000">  
<body bgcolor="rgb(0,0,0)">  
<body bgcolor="black">
```

The lines above all set the background-color to black.

### 4.6.2 Background

The background attribute can also specify a background-image for an HTML page. The value of this attribute is the URL of the image you want to use. If the image is smaller than the browser window, the image will repeat itself until it fills the entire browser window.

```
<body background="clouds.gif">  
<body background="http://profdevtrain.austincc.edu/html/graphics/clouds.gif">
```

The URL can be relative (as in the first line above) or absolute (as in the second line above).

If you want to use a background image, you should keep the following in mind:

- ✓ Will the background image increase the loading time too much?
- ✓ Will the background image look good with other images on the page?
- ✓ Will the background image look good with the text colors on the page?

- ✓ Will the background image look good when it is repeated on the page?
- ✓ Will the background image take away the focus from the text?

**Note:** The bgcolor, background, and the text attributes in the <body> tag are deprecated in the latest versions of HTML (HTML 4 and XHTML). The World Wide Web Consortium (W3C) has removed these attributes from its recommendations. Style sheets (CSS) should be used instead (to define the layout and display properties of HTML elements).

#### **4.6.2.1 Web Safe Colors**

A few years ago, when most computers supported only 256 different colors, a list of 216 Web Safe Colors was suggested as a Web standard. The reason for this was that the Microsoft and Mac operating system used 40 different "reserved" fixed system colors (about 20 each). This 216 cross platform web safe color palette was originally created to ensure that all computers would display all colors correctly when running a 256 color palette.

#### **16 Million Different Colors**

The combination of Red, Green and Blue values from 0 to 255 gives a total of more than 16 million different colors to play with (256 x 256 x 256). Most modern monitors are capable of displaying at least 16,384 different colors. To assist you in using color schemes, check out <http://colorshemadesigner.com/>. This site lets you test different color schemes for page backgrounds, text and links.

#### **Tips**

- **Don't use font tags if at all possible! Use CSS Styles instead!**
- Fonts add character and originality to sites, so long as they're not overused or unreadable.
- Use consistent fonts throughout your site.
- Nobody enjoys websites that are hard to read, so keep your fonts legible!
- Use formatting tags rather than the font tag for bold or italic texts.

#### **Self Assessment Questions**

1. What are HTML tags?
2. Where is the text of the title tag displayed?
3. What steps are involved in creating a simple HTML document?
4. How do you create a comment tag?
5. How can you display your HTML document in a web browser?

#### **Assignment**

Think of a topic for your own web page. Now create your own HTML text file that includes a <title> tag and a few introductory sentences. Save the HTML file and reload it in your web browser. Keep this file handy as you will add to it in later lessons.



## CHAPTER 5: TEXT FORMATTING

### 5.1 Chapter Overview

Text is the backbone of any web page. It plays a double role; it provides content for web surfers to enjoy as they skim through articles and articles of information, but it also gives Search Engines and Spiders keywords and meta data. It is because of text on web pages that we have a network of seemingly endless information available at our fingers.

At the end of this chapter, you should be able to

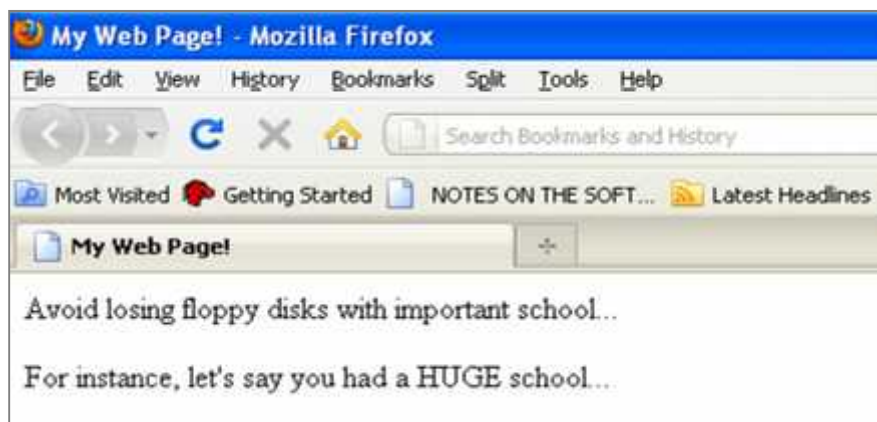
- 👉 Format characters
- 👉 Customize fonts
- 👉 Create HTML Headings
- 👉 Align text

### 5.2 Paragraph Text

Any text containing more than a few lines (or sometimes even more) should exist inside of a paragraph tag `<p>`. This tag is reserved specifically for blocks of text, such as those you would expect to find inside your favorite novel.

#### HTML `<p>` Tag Code:

```
<html>
<head>
  <title>My Web Page!</title>
</head>
<body>
  <p>Avoid losing floppy disks with important school...</p>
  <p>For instance, let's say you had a HUGE school...</p>
</body>
</html>
```



Well written HTML documents can gain popularity through Search Engine Optimization and careful coding of your HTML elements.

Precision is important when writing your code. Web spiders are a little forgiving when it comes to malformed HTML elements. For best results, do your best to ensure your code is complete and accurately constructed.

## 5.3 HTML Headings

HTML has heading tags which can be used as headers or subheaders. By placing text inside of an `<h1>` heading tag, for example, the text displays bold and the size of the text increases to a 24pt font size. Heading tags are numbered 1 through 6, and they change size depending on which tag you choose, with 1 being the largest font size at 24pt, and 6 being the smallest font size at 8pt.

### HTML Heading Element:

```
<body>
  <h1>Headings</h1>
  <h2>are</h2>
  <h3>great</h3>
  <h4>for</h4>
  <h5>titles</h5>
  <h6>and subtitles</h6>
</body>
```

Place these lines into your HTML file and you should see results similar to what you see below.



Notice that each heading has a line break (a blank line) rendered before and after it. This is a built-in attribute associated with the heading tag. Each time you place a heading tag, your web browser automatically places a line break in front of your beginning tag and after your ending tag, just like it does with <p> tags. This is expected behavior, and as a designer you need to take these line breaks into consideration when designing a layout.

## 5.4 Formatting Text Elements with Tags

As you begin to place more and more text elements onto your website, you may find yourself wanting to incorporate **bold** or *italic* properties in your text elements. HTML offers a handful of special tags that can be utilized to do just that. Formatting elements such as <b> should be used sparingly, and what we mean by that is that you should only use them to bold or italicize one or two words at a time. If you wish to bold an entire paragraph, the better solution would involve Cascading Style Sheets (CSS) and adjust the element's font-weight property. Try out the code given below and take note of the effect it creates on your web document. The expected outcome is provided to guide you;;

### HTML Text Formatting Tags:

```
<p>An example of <b>Bold Text</b></p>
<p>An example of <em>Emphasized Text</em></p>
<p>An example of <strong>Strong Text</strong></p>
<p>An example of <i>Italic Text</i></p>
<p>An example of <sup>superscripted Text</sup></p>
<p>An example of <sub>subscripted Text</sub></p>
<p>An example of <del>struckthrough Text</del></p>
<p>An example of <code>Computer Code Text</code></p>
```

### HTML Formatting Text (Expected Outcome):

An example of **Bold Text**

An example of *Emphasized Text*

An example of **Strong Text**

An example of *Italic Text*

An example of <sup>superscripted Text</sup>

An example of <sub>subscripted Text</sub>

An example of ~~struckthrough Text~~

An example of `Computer Code Text`

All of these tags add a pinch of flavor to HTML text elements, from paragraphs to lists and text links

### 5.4.1 Line Breaks

A line break is used in HTML text elements, and it is the equivalent of pressing **Enter** or **Return** on your keyboard. In short, a line break ends the line you are currently on and resumes on the next line. Earlier, we mentioned that each paragraph element begins and ends with a line break, which creates an empty space between the start of a paragraph element and the end of a paragraph element.

As we've mentioned, line break elements are a little different from most of the tags we have seen so far because they do not require a closing tag. Instead, their opening and closing tags are combined into a single line break element. Placing `<br />` within the code is the same as pressing the return key in a word processor. Use the `<br />` tag within other elements such as paragraphs (`<p>`).

#### HTML Format Text:

```
<p>
Allan Muthomi<br/>
Box 9688<br />
G.P.O, Nairobi<br />
</p>
```

#### Address:

Allan Muthomi

Box 9688

G.P.O, Nairobi

We have created an address for a letterhead and used a line break `<br />` tag inside of a paragraph element to add some line breaks to make this text appear more like an address. Here's another look as we add a signature element to the same letter.

#### HTML Text Format:

```
<p>Sincerely,<br />
<br />
<br />
Trizah Nimos<br />
Vice President</p>
```

#### Closing Letter:

Sincerely,

Trizah Nimos

*Vice President*

### **Tips**

- Remember that there is a special tag for line breaks and horizontal rule tags.
- Use this page as a reference if you ever need to format your website's text in the future.

## **5.4.2 <p>Paragraph**

### **5.4.2.1 Paragraph Alignment**

#### **Paragraph Justification**

Paragraphs can be formatted in HTML much the same as you would expect to find in a word processing program. Here the align attribute is used to "justify" our paragraph.

#### **HTML Code:**

```
<p align="justify">For instance, let's say you had a HUGE school or work...</p>
```

#### **Justified Text Alignment:**

For instance, let's say you had a HUGE school or work project to complete. Off hand, the easiest way to transfer the documents from your house could be a 3.5" floppy disk. However, there is an alternative. Place your documents, photos, essays, or videos onto your web server and access them from anywhere in the world.

#### **Paragraph Centering**

#### **HTML Code:**

```
<p align="center">For instance, let's say you had a HUGE school or work...</p>
```

#### **Centered Text Alignment:**

For instance, let's say you had a HUGE school or work project to complete. Off hand, the easiest way to transfer the documents from your house could be a 3.5" floppy disk. However, there is an alternative. Place your documents, photos, essays, or videos onto your web server and access them from anywhere in the world.

Each line of the paragraph has now been centered inside the display window.

#### **Paragraph Align Right**

**HTML Code:**

```
<p align="right">For instance, let's say you had a HUGE school or work...</p>
```

**Right Text Alignment:**

For instance, let's say you had a HUGE school or work project to complete. Off hand, the easiest way to transfer the documents from your house could be a 3.5" floppy disk. However, there is an alternative. Place your documents, photos, essays, or videos onto your web server and access them from anywhere in the world.

Every line of the paragraph above is now aligned to the right hand side of the display box.

**5.4.3 <pre> Preformatting**

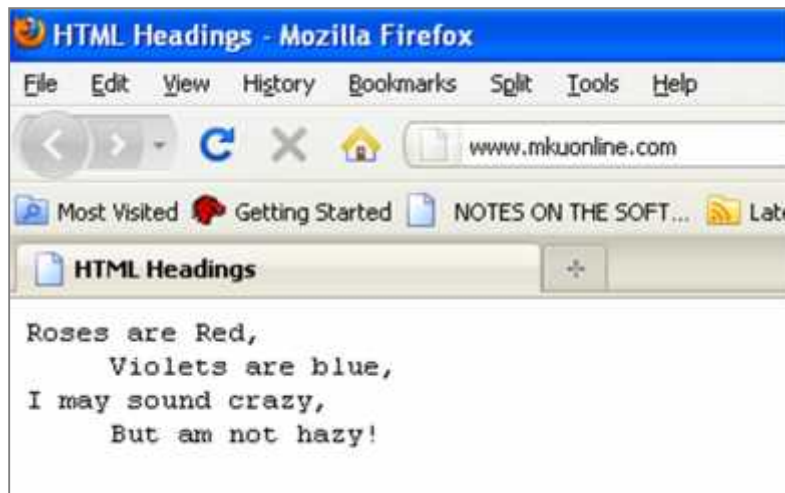
A web browser interprets your HTML document as being one long line. Sure, you may have tabs and line breaks in Notepad which align your content so it's easier for you to read, but your browser ignores those tabs and line breaks.

We showed you that you can get around this problem by using the <br/> tag, but tabs and spacing are quite different from one another and can be absolutely annoying at times. One simpler solution might be to use the <pre> tag, which stands for previously formatted text.

Use the <pre> tag for any special circumstances where you wish to have the text appear exactly as it is typed. Spaces, tabs, and line breaks that exist in your actual code will be preserved with the <pre> tag.

**HTML Pre Code:**

```
<pre>
Roses are Red,
  Violets are blue,
I may sound crazy,
  But am not hazy!
</pre>
```




#### 5.4.4 Horizontal Rule

The `<hr>` element is used for horizontal rules that act as dividers between sections, like this:



The horizontal rule does not have a closing tag. It takes attributes such as align and width. For instance:

This Code Would Display	Would Display
<code>&lt;hr width="50%" align="center"&gt;</code>	

#### 5.4.5 Comments in HTML

The comment tag is used to insert a comment in the HTML source code. A comment can be placed anywhere in the document and the browser will ignore everything inside the brackets. You can use comments to write notes to yourself, or write a helpful message to someone looking at your source code.

This Code Would Display	Would Display
<code>&lt;p&gt; This html comment would &lt;!-- This is a comment --&gt; be displayed like this. &lt;/p&gt;</code>	This HTML comment would be displayed like this.

Notice you don't see the text between the tags `<!--` and `-->`. If you look at the source code, you would see the comment. To view the source code for this page, in your browser window, select **View** and then select **Source**.

**Note:** You need an exclamation point after the opening bracket `<!--` but not before the closing bracket `-->`.

HTML automatically adds an extra blank line before and after some elements, like before and after a paragraph, and before and after a heading. If you want to insert blank lines into your document, use the `<br>` tag.

### 5.4.6 Abbreviating

The code `<abbr title="World Wide Web">WWW</abbr>` would display **WWW** but when you hold your mouse pointer over the WWW, the text *world wide web* (in the title attribute) will appear in.

### 5.4.7 HTML Character Entities

Some characters have a special meaning in HTML, like the less than sign (<) that defines the start of an HTML tag. If we want the browser to actually display these characters we must insert character entities in place of the actual characters themselves. A character entity has three parts: an ampersand (&), an entity name or an entity number, and finally a semicolon (;). The & means we are beginning a special character, the ; means ending a special character and the letters in between are sort of an abbreviation for what it's for. To display a less than sign in an HTML document we must write: **&lt;** or **&#60;**; The advantage of using a name instead of a number is that a name is easier to remember. The disadvantage is that not all browsers support the newest entity names, while the support for entity numbers is very good in almost all browsers.

Entity Name	Looks Like	Description
&nbsp;		no-break space
&excl;	!	inverted exclamation mark
&pound;	£	pound sterling sign
&brvbar;	¦	broken (vertical) bar
&copy;	©	copyright sign
&reg;	®	registered sign
&frac14;	¼	fraction one-quarter
&frac12;	½	fraction one-half
&frac34;	¾	fraction three-quarters
&iquest;	¿	inverted question mark
&amp;	&	ampersand



## CHAPTER 6: HYPERLINKS

### 6.1 Chapter Overview

The World Wide Web got its spidery name from the plentiful connections (links) that link websites together with the click of a button. What most people don't know is that HTML links are actually HTML anchors constructed using anchor tags (<a>).

#### HTML Text Link:

```
<a>I am a text link!</a>
```

#### HTML Text Link:

I am a text link!

While the example above appears and feels like a text link when viewed through a web browser, the element is incomplete as it is missing a vital attribute that references another web page called a Hypertext Reference (*href*).

### 6.2 Hypertext Reference (href)

To create a hyperlink, you use the tag in conjunction with the *href* attribute (*href* stands for Hypertext Reference). The value of the href attribute is the *URL*, or, location of where the link is pointing to.

A Hypertext Reference (*href*) is an HTML attribute of an anchor (link) tag that requires a valid URL in order to properly direct a user to a different location. In other words, this Hypertext Reference is where users will navigate to if they do click on this link. Use the demonstration below as a reference.

#### HTML Text Link Code:

```
<a href="http://www.shawksInc.com/" target="_blank">Shawks Inc Home</a>
<br />
<a href="http://www.mku.ac.ke/" target="_blank">Mt Kenya University Home</a>
<br />
```

The address of a website is called a Uniform Resource Locator (a URL), and it acts like a street address for a website as a user is directed from one site to another. There are different types of URLs, and each has a slightly different look depending on the hyperlink reference. Here's a look at the different types of URLs.

Global - href="http://www.cnn.com/" Links to other domains outside your website domain.

Local - href="../internal/mypage2.html" Links to other pages within your website domain.

Internal(Named Anchors) - href="#anchorname" Links to anchors embedded in the current web page.

The name attribute is used to create a ***named anchor***. When using named anchors we can create links that can jump directly to a specific section on a page, instead of letting the user scroll around to find what he/she is looking for. Unlike an anchor that uses href, a named anchor doesn't change the appearance of the text (unless you set styles for that anchor) or indicate in any way that there is anything special about the text. Below is the syntax of a named anchor:

`<a name="top">Text to be displayed</a>`

To link directly to the top section, add a # sign and the name of the anchor to the end of a URL, like this:

This Code	Would Display
<pre>href="http://virtualcampus.mtkenyaUniversity.com/html/10links.html#top"&gt;Back to top of page &lt;/a&gt;</pre> <p>A hyperlink to the top of the page from within the file 10links.html will look like this:</p> <pre>&lt;a href="#top"&gt;Back to top of page &lt;/a&gt;</pre>	<p>Back to top of page</p>     <p>Back to top of page</p>

**Note:** Always add a trailing slash to subfolder references.

Named anchors are often used to create "table of contents" at the beginning of a large document. Each chapter within the document is given a named anchor, and links to each of these anchors are put at the top of the document. If a browser cannot find a named anchor that has been specified, it goes to the top of the document. No error occurs.

### 6.3 Link Targets

The *target* attribute defines how each link will open when clicked. Will each one open in a new window, or will each one open in the current browser window? As the web designer, you call the shots as to how a user navigates from page to page, so long as you know how to handle the *target* attribute.

### Link Targets:

Target	Description
_blank	Opens new page in a new browser window
_self	Loads the new page in the current window
_parent	Loads new page into a parent frame
_top	Loads new page into the current browser window, cancelling all frames

The two most important values are the top two, (target="\_blank" and target="\_self"). The *\_self* value is generally the default. It loads each new page in the current browser window, while *\_blank* opens up a new web browser window with each click of the text link.

The code below shows how you would link to facebook.com, a popular sports website. The *target* attribute is added to allow the browser to open facebook in a new window, so that the viewer can have a window that remains opened to our website. Try the example below and note the outcome;

```
<a href="http://www.facebook.com" target="_blank">FB</a>
```

Links are a big part of the user experience for any website. Always try to keep that in mind when working on a site's navigation. A web page that opens a new web browser window each time a user clicks a link is not the greatest way to entice users to stick around.

## 6.4 Email Links

Creating an email link is simple. If you want people to mail you about your site, a good way to do it is place an email link with a subject line already filled out for them.

### HTML Email Link Code:

```
<a href="mailto:email@Facebook.com?subject=Feedback" >Email@Facebook.com</a>
```

### Email Links:

[Email@Facebook.com](mailto:email@Facebook.com)

In some circumstances, it may be necessary to fill in the body of the email for the user as well.

### HTML Email Link Code:

```
<a href="mailto:email@Facebook.com?subject=Feedback&body=Sweet site!">  
Email@Facebook.com</a>
```

### Complete Email:

[Email@Facebook.com](mailto:email@Facebook.com)

## 6.5 Default Links; Base

Use the <base> tag in the *head* element to set a default URL for all links on a page to go to. It's always a good idea to set a base tag just in case your links become bugged somewhere down the line. Usually, you should set your base to your home page.

### HTML Base Link Code:

```
<head>  
  <base href="http://www.Facebook.com/" />  
</head>
```

## 6.6 HTML Comments

Comments are a great asset to new developers and a great way to place little notes for reminding yourself what various pieces of code are doing. Comments are also great ways to troubleshoot bugs and code errors, as they give you the ability to comment out lines of code one at a time to search for the exact line causing problems.

As a sprouting young web developer, HTML code comments are your friends! A comment is a way to control which lines of code are to be ignored by the web browser and which lines of code to incorporate into your web page. There are three main reasons why you may want your code to be commented out or ignored.

- Comment out elements temporarily rather than removing them, especially if they've been left unfinished.
- Write notes or reminders to yourself inside your actual HTML documents.
- Create notes for other scripting languages like JavaScript which requires them.

### 6.6.1 Comment Tags:

`<!-- Opening Comment Tag`

`-- > Closing Comment Tag`

As you can see, comments are also comprised of an opening and closing tag, (`<!-- -->`). Like other HTML elements, these tags can span across multiple lines of code, allowing you to comment out large blocks of HTML code. Any HTML elements that are encapsulated inside of the comment tags will be ignored by the web browser. This makes comment tags quite useful for debugging, as they allow the developer to temporarily comment out pieces of an HTML document without having to immediately delete the code from the HTML document.

### HTML Comment Code:

```
<!--Note to self: This is my banner image! Don't forget -->
```

```

```

Placing notes and reminders to yourself is a great way to remember your thoughts and to keep track of elements embedded inside the web page. Also, your code may exist for many, many years, and these notes to yourself are a great way to remember what was going on, since you may not remember five or more years down the road.

All combinations of text placed within the comment tags will be ignored by the web browser. This includes any HTML tags, scripting language(s), etc.

### HTML - `<!-- Commenting Existing Code -->`

As a web designer, you may sometimes have different websites in progress at once, and it might be easy to leave some HTML elements unfinished. In this case, you may comment out an

element until it is 100% ready for the site. Below, we have commented out an input form element, since we are not quite ready to receive input from our users.

**HTML Code:**

```
<!-- <input type="text" size="12" /> -- Input Field -->
```

Now when we are ready to show that element, we can simply remove the comment tags, and our browser will readily display the element! Comment out elements and bits of code that you may want to recall and use at a later date. Nothing is more frustrating than deleting bits of code only to turn around moments later and realize that you now need to recode them.

**Tips**

- Frequent use of comments allows you to recall your train of thought the next time you take a look at the HTML code.
- Commenting out elements or code chunks helps debug without having to delete and then retype code.

## CHAPTER 7: HTML LISTS

### 7.1 Chapter Overview

One of the best ways to communicate a great deal of information in a short amount of time is by using bulleted lists to convey the message. That philosophy was not lost on the early creators and designers of Web pages, and various tags allow for easy formatting of a number of styles of lists, including both bulleted and non-bulleted incarnations. This chapter will teach you how to create HTML lists and use them when creating a website.

By the end of this chapter you should be able to;

- ✓ Describe HTML lists
- ✓ Create HTML Lists
- ✓ Use HTML Lists appropriately in Web design

### 7.2 Introduction

HTML provides a simple way to show listed items. There are actually three different types of HTML lists, including unordered lists (bullets), ordered lists (numbers), and definition lists (think: dictionaries). Each list type utilizes its own unique list tag where the actual list tags themselves e.g. `<ul>` are nothing but container elements for list items (`<li>`). They work behind the scenes to identify the beginning and ending of an HTML list element.

### 7.3 Unordered Lists

An unordered list (`<ul>`) signifies to a web browser that all list items contained inside the `<ul>` tag should be rendered with a bullet preceding the text. The default bullet type for most web browsers is a full disc (black circle), but this can be adjusted using an HTML attribute called *type*.

#### HTML Default Bullet List Code:

```
<h4 align="center">Shopping List</h4>
<ul>
  <li>Milk</li>
  <li>Toilet Paper</li>
  <li>Cereal</li>
  <li>Bread</li>
</ul>
```

#### HTML Default Disc Bullets:

##### Shopping List

- Milk
- Toilet Paper
- Cereal
- Bread

To render a list with a different bullet type, add a *type* attribute to the unordered list element. Using the same code in the example above, replace the `<ul>` line from the previous example with any of the lines listed below to change the bullet from disc shape to another shape.

### HTML Unordered List Type Code:

```
<ul type="square">
```

```
<ul type="disc">
```

```
<ul type="circle">
```

### HTML Unordered List Types:

**type="square"**

- Milk
- Toilet Paper
- Cereal
- Bread

**type="disc"**

- Milk
- Toilet Paper
- Cereal
- Bread

**type="circle"**

- Milk
- Toilet Paper
- Cereal
- Bread

## 7.4 Ordered Lists

An ordered list is defined using the `<ol>` tag, and list items placed inside of an ordered list are preceded with numbers instead of bullets.

### HTML Numbered/Ordered List:

```
<h4 align="center">Goals</h4>
```

```
<ol>
```

```
<li>Find a Job</li>
```

```
<li>Get Money</li>
```

```
<li>Move Out</li>
```

```
</ol>
```

### HTML Numbered List:

#### Goals

1. Find a Job
2. Get Money
3. Move Out

The numbering of an HTML list can be changed to letters or Roman Numerals by once again adjusting the type attribute.

### HTML Code:

```
<ol type="a">
```

```
<ol type="A">
```

```
<ol type="i">
```

```
<ol type="I">
```

#### 7.4.1 Ordered List Types:

Lower-Case Letters	Upper-Case Letters	Lower-Case Numerals	Upper-Case Numerals
--------------------	--------------------	---------------------	---------------------

- |               |               |               |               |
|---------------|---------------|---------------|---------------|
| a. Find a Job | A. Find a Job | i. Find a Job | I. Find a Job |
| b. Get Money  | B. Get Money  | ii. Get Money | II. Get Money |
| c. Move Out   | C. Move Out   | iii. Move Out | III. Move Out |

The *start* attribute allows you to further customize an HTML ordered list by setting a new starting digit for the ordered list element.

#### 7.4.2 Numbered List Start Attribute

```
<h4 align="center">Goals</h4>
<ol start="4" >
  <li>Buy Food</li>
  <li>Enroll in College</li>
  <li>Get a Degree</li>
</ol>
```

##### 7.4.2.1 Numbered List Start - 4:

###### Goals

4. Buy Food
5. Enroll in College
6. Get a Degree

#### 7.5 Definition Lists

HTML definition lists (<dl>) are list elements that have a unique array of tags and elements; the resulting listings are similar to those you'd see in a dictionary.

- <dl> - opening clause that defines the start of the list
- <dt> - list item that defines the definition term
- <dd> - definition of the list item

These lists displace the word term (<dt>) in such a way that it rests just above the definition element (<dd>) to offer a very unique look. For best results we suggest bolding the definition terms with the bold tag <b>.

##### 7.5.1 Definition List Code:

```
<dl>
  <dt><b>Hero</b></dt>
  <dd>An ordinary man who has made remarkable achievement </dd>
</dl>
```

###### Definition List Display:

###### Hero

An ordinary man who has made remarkable achievement.

###### Tips

- Use the *start* and *type* attributes to customize list elements.
- It is possible to make lists of lists, which is helpful for creating some items, such as outlines.



## CHAPTER 8: HTML IMAGES

### 8.1 Chapter Overview

Images are a staple of any web designer, so it is very important that you understand how to use them properly. In order to place an image onto a website, one needs to know where the image file is located within the file tree of the web server -- the URL (Unified Resource Locator).

By the end of the chapter, you should be able to;

- 👉 Describe various web image formats
- 👉 Create background images
- 👉 Create inline images
- 👉 Create hyperlinked images

### 8.2 Introduction to Images

Use the `<img/>` tag to place an image on your webpage. Like the `<br/>` tag, `<img/>` tag does not require a formal ending tag. Instead, all we need to do to close this tag out with a slash (/) placed just inside the ending bracket (`</>`). For purposes of illustrations, we'll use the image `mtkenya.jpg` below. ``



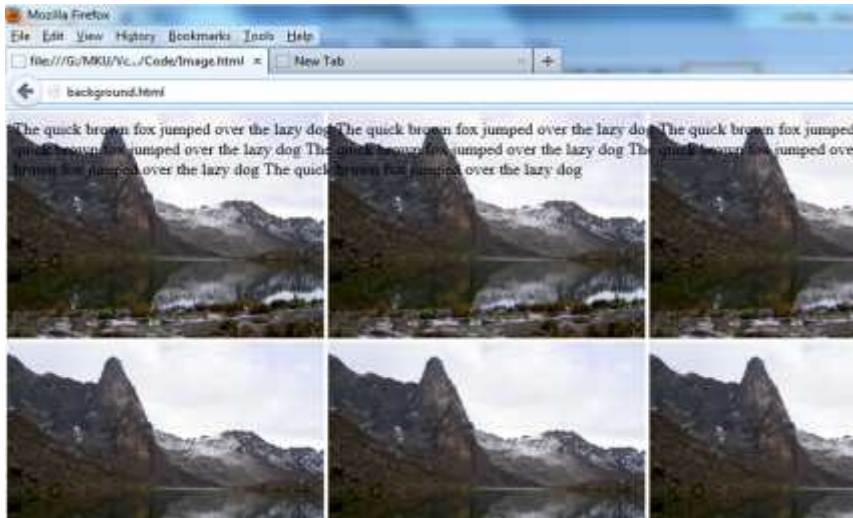
Images can either be used as *background* images (behind text) or *inline* images (together with text).

#### 8.2.1 Background Images

Created behind text usually as water marks. Care should be taken such that the image does not interfere with the legibility of text in the foreground. These are created by defining them as attributes in the body element.

```
<body background="mtkenya.jpg">
```

Take note that for this to work, your source code, webpage (HTML document you have created) and the picture should be in the same folder. You have to note and use the picture's file extension(.jpg, .gif, .png, .bmp e.t.c) in your code.



There are two design issues you need to take note of here.

- i. Whilst the background image may be used for aesthetic purposes, it should not interfere with elements on the foreground. For example, it should not interfere with legibility of text like it has done in our case. To correct this, open the original image in a graphics editor and edit it to be a watermark. This can also be achieved by increasing the brightness.
- ii. Unless if necessary, tiled backgrounds like in our case are visually disruptive and should be avoided. Using a graphics editor to change the size of the original image appropriately then refreshing the webpage would remedy this.

The illustration below captures the implementation of point (i) and (ii) above. Using Microsoft's Picture Manager, the figure was increased to 500% of the original size and the brightness increased.



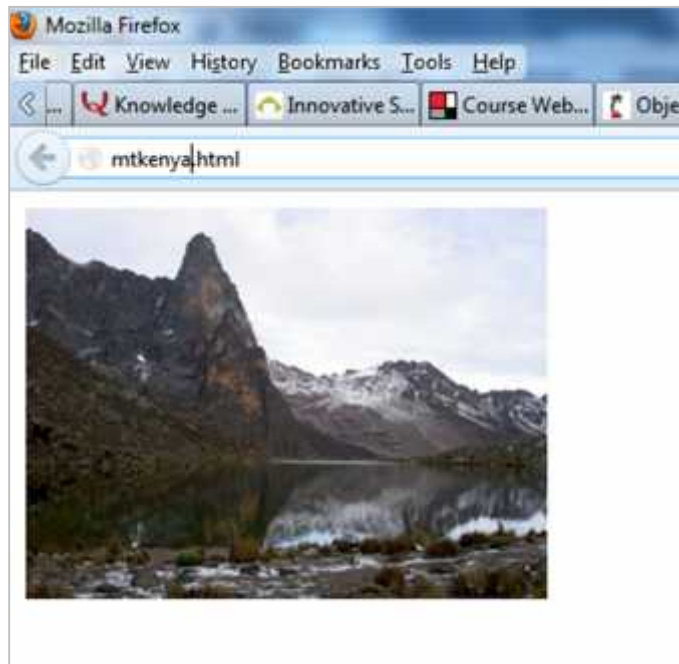
### 8.2.2 Inline Images

These are called to the webpage by the *src* attribute. Src stands for "source". The source attribute (*src*) is what makes an image tag display an image. An image source is a URL value and should point to the directory location of an image file. For example, the code

```

```

would give the following result;



An image source value is essentially the URL of a picture file and tells the web browser where the image is located so that it can then display the image correctly.

#### 8.2.2.1 Source URLs

Image source URLs can be either local or global, meaning that the picture files you wish to display on your website can be either hosted locally on your machine (local) or hosted elsewhere on some other web site domain (global).

- **Global:** <http://www.mkuonline.com/pics/graduation.gif>
- **Local:** `pics/mtkenya.jpg`

Local URLs are relative to the file path of the web page itself. For example, if the picture file is placed inside the same directory as the web page, then the local URL for the image would simply be the name of the image, since it is residing in the same directory as the HTML page.

### Local URLs Explained:

Local Src	Location Description
-----------	----------------------

<code>src="sunset.gif"</code>	picture file resides in same directory as .html file
-------------------------------	--

<code>src="pics/sunset.gif"</code>	picture file resides in the <i>pics</i> directory
------------------------------------	---

<code>src="../sunset.gif"</code>	picture resides one folder "up" from the .html file
----------------------------------	---

<code>src="../pics/sunset.gif"</code>	picture file resides in the <i>pics</i> directory, one folder "up" from the .html file.
---------------------------------------	---

Pictures must reside on the same web host as your .html file in order for you to use local URLs. A URL cannot contain drive letters such as C:\, since a *src* URL is a relational interpretation based on the location of the .html file and the location of the picture file. Something like `src="C:\www\web\pics\"` will *not* work.

Each URL format has its pros and cons. Using the URL of pictures on other sites poses a problem if the other site happens to change the physical location of the picture file. Copying the file directly to your web server solves this problem. However, as you continue to upload picture files to your system, you may eventually run short on hard drive space. Use your best judgment based upon your situation.

### 8.2.2.2 Image Height and Width Attributes

*Height* and *width* are HTML attributes that define an element's height and width properties. These values can either be percentage-based (%) or rely on pixel sizes.

#### HTML Height and Width Attributes:

```

```

#### HTML Height and Width (Pixels):



Above, we've used hard-coded pixel values for the height and width of the sunset image to ensure that this image will always render 50 pixels high by 100 pixels wide. By hard-coding these values, we are ensuring that the image will only display 50 pixels high by 100 pixels wide,

even if the picture file itself happens to be much larger. If the dimensions of the picture are much larger, then we risk some severe skewing as the browser tries to shrink our image into our small box.

Height and width values can also be a percentage. Percentage values are relative to the parent HTML element (usually the body), which means if you have a parent element like a <body> element that is the whole screen (1024x768), then an image with a height and width of 100% will take up that entire body element (1024x768). In our example below, we have placed the image in a table element that is about 400 pixels wide by 200 pixels tall.

#### HTML Height and Width Code:

```
<table height='200' width='400'>
  <tr>
    <td>
      
    </td>
  </tr>
</table>
```

#### HTML Height and Width (Percentage):



Here's a few things to remember when trying to place images on your web page:

1. **Maintain the same height to width ratio.** The ratio is critical, and must be maintained to avoid skewing.
2. **Always scale down.** -- Larger images will always scale down nicely and continue to look sharp.

If no height or width attribute is specified inside the <img> tag, the browser will use the actual dimensions of the image file to render the image. This can cause problems with the page layout if the picture file is too large, as other HTML elements will be moved further down the page in the event of an over-sized image.

Another concept to keep in mind is that as a browser begins rendering HTML components, it handles them one after another in sequence. Before it can move from one element on to the next, the browser needs to know the size and shape of an element. If this information is provided in the

tag, that's one less step required by the browser to render an image element and will result in the page loading faster for your users.

### 8.2.2.3 Alternative Attribute

The *alt* attribute specifies alternate text to be displayed if for some reason the browser cannot find the image, or if a user has image files disabled in their web browser settings. Text-only browsers greatly depend on the *alt* attribute since they are not capable of displaying pictures.

#### HTML Alternative Attribute (alt):

```

```

#### HTML Alternative Text Attribute:

The *alt* attribute is also an attribute that search engines may look for when displaying images. The text value contained within this attribute must reflect the substance of the image in order to receive "credit" from a search engine.

### 8.2.2.4 Horizontally Align Images

Images can be aligned horizontally to the right or to the left of other elements using the *align* attribute. Image elements are aligned to the left by default.

1. align
  - right
  - left

For example:

```
..... Align to the right  
..... Aligns in the middle (center)  
..... Aligns to the left
```

HTML Image Align: Right:

As you can see, the image's right edge has now been aligned with the right edge of the display box. Since the display box is the parent element, this is the desired behavior for the align attribute. If we take this example a step further, you can achieve some really great designs by embedding aligned images inside of paragraph <p> elements.



### 8.2.3 Common Image Types

**Gifs** are best used for banners, clip art, and buttons. The main reason for this is that .gif files can have transparent backgrounds -- a priceless attribute when it comes to web design. On the down side, .gif files are limited to only 256 colors and any .gif image containing more than a few



colors tends to have a larger file size than their .jpeg or .png counterparts. Large picture files are a plague of web design!

**Jpegs** have an unlimited color wheel and a high compression rate, which downsizes your load times and saves on hard drive space. Although .jpeg (or .jpg) files don't allow for transparent backgrounds, their size/quality ratio is outstanding. It's best to use .jpeg files for photo galleries or artwork. Avoid using .jpeg files for graphical designs, though; stick to using them for thumbnails, backgrounds, and photo galleries.

**PNG** image files are the best of both worlds. They have a large color wheel, low file size, and allow for transparencies like .gif images do. With a high-compression rate and transparent coloring, they might just be the best format for any web graphics.

When in doubt, try saving an image in multiple formats and decide which is better, keeping file size and quality in mind.

### Tips

- Find a good graphics editing program to edit your images with.
- Defining a height and width for your images will allow your page to load gracefully

### 8.2.4 Image Links

Image links are constructed as you might expect, by embedding an <img> tag inside of an anchor element <a>. Like HTML text links, image links require opening and closing anchor tags, but instead of placing text between these opening and closing tags, the developer needs to place an image tag -- with a valid source attribute value of course.

#### Image Link Code:

```
<a href="http://www.espn.com" target="_blank">  
    
</a>
```

#### Image Link:



By default, many browsers add a small border around image links. This default behavior is intended to give web viewers the ability to quickly decipher the difference between ordinary images and image links. Since this default is different from web browser to web browser, it may be best to squelch this ambiguity and set the *border* attribute of the image tag to 0 or 1.

#### Image Border Code:

```
<a href="http://www.espn.com" target="_blank">  
    
</a>
```

**HTML Image Link; No Border:**



#### **8.2.4.1 Image Thumbnails**

Thumbnails are by far the most common type of image link seen in today's world. To create a thumbnail, one must save a low-quality version of a picture with smaller dimensions. Then, one should link this low-quality picture to its higher-quality counterpart.

Thumbnails are intended to give your audience quick previews of images without them **having** to wait for the larger, higher-quality image to load. Photo galleries make heavy use of thumbnails, and they will allow you to display multiple pictures on one page with ease.

**Thumbnail Code:**

```
<a href="mtkenya.jpg">  
    
</a>
```

**HTML Thumbnails:**





## CHAPTER 9: HTML TABLES

### 9.1 Chapter Overview

An HTML table is an element comprised of table rows and columns, much like you'd see when working with an application such as Excel. Tables are container elements, and their sole purpose is to house other HTML elements and arrange them in a tabular fashion -- row by row, column by column. By the end of the chapter you should be able to;

- 👉 Create HTML Tables
- 👉 Apply different color designs to a HTML table
- 👉 Use tables as a layout tool

### 9.2 Introduction

Tables may seem difficult at first, but after working through this lesson, you'll see that they aren't so horrible. A table element consists of three different HTML tags including the `<table>` tag, `<tr>` (table rows), and the `<td>` (table columns) tags.

#### HTML Table Code:

```
<table border="1">
  <tr>
    <td>Row 1 Cell 1</td>
    <td>Row 1 Cell 2</td>
  </tr>
  <tr>
    <td>Row 2 Cell 1</td>
    <td>Row 2 Cell 2</td>
  </tr>
</table>
```

#### Basic HTML Table Layout:

Row 1 Cell 1	Row 1 Cell 2
Row 2 Cell 1	Row 2 Cell 2


We've adjusted the formatting of the code by adding additional spaces before some of the table elements, but this has no bearing on the rendering of the element. It simply helps keep track of each tag/element and helps us ensure we don't miss an opening or closing tag which would prevent our table element from rendering correctly. We've also added a *border* attribute to ensure the table cells/rows are more visible to our readers.

Content elements like HTML lists, images, and even other table elements can be placed inside each table cell. Doing so aligns the elements in a tabular fashion and provides structure.

**HTML Table Code:**

```
<table border="1">
  <tr>
    <td width="50%">
      <ul>
        <li>List Item 1</li>
        <li>List Item 2</li>
        <li>List Item 3</li>
      </ul>
    </td>
    <td>
      <ul>
        <li>List Item 4</li>
        <li>List Item 5</li>
        <li>List Item 6</li>
      </ul>
    </td>
  </tr>
  <tr>
    <td>
      <p>Avoid losing floppy disks with important school...</p>
    </td>
    <td>
      <a href="http://www.espn.com" target="_blank" rel="nofollow">
        
      </a>
    </td>
  </tr>
</table>
```

**HTML Table 2:**

<ul style="list-style-type: none"><li>• List Item 1</li><li>• List Item 2</li><li>• List Item 3</li></ul>	<ul style="list-style-type: none"><li>• List Item 4</li><li>• List Item 5</li><li>• List Item 6</li></ul>
Avoid losing floppy disks with important school...	

HTML tables allow the web designer to align page content in a tabular fashion while spanning elements horizontally across the web page, rather than stacking them up one on top of another.

The width attribute can be used to define the width of your table. It can be defined as a fixed width or a relative width. A fixed table width is one where the width of the table is specified in pixels. For example, this code, `<table width="550">`, will produce a table that is 550 pixels wide.

A relative table width is specified as a percentage of the width of the visitor's viewing window. Hence this code, `<table width="80%">`, will produce a table that occupies 80 percent of the screen.

There are arguments in favor of giving your tables a relative width because such table widths yield pages that work regardless of the visitor's screen resolution. For example, a table width of 100% will always span the entire width of the browser window whether the visitor has a 800x600 display or a 1024x768 display (etc). Your visitor never needs to scroll horizontally to read your page, something that is regarded by most people as being very annoying.

### 9.3 Table Rows & Table Columns

A table can contain an infinite number of table rows. Each table row is essentially a table element itself, with an opening and closing tag (`<tr>` `</tr>`). Table columns are also considered child elements of HTML tables, and like table rows, an HTML table may contain an infinite number of table data cells (`<td>` `</td>`).

Table rows and columns are container elements that house other HTML elements such as text links, images, and lists, as we've seen in previous examples. Below, we've applied a background color to the table example in order to help distinguish the different table elements.

#### HTML Table Code:

```
<table border="1">
  <tr title="You are looking at Row 1" bgcolor="silver">
    <td>Row 1 Cell 1</td>
    <td>Row 1 Cell 2</td>
  </tr>
  <tr title="You are looking at Row 2" bgcolor="aqua">
    <td>Row 2 Cell 1</td>
    <td>Row 2 Cell 2</td>
  </tr>
</table>
```

#### HTML Table Code Example:

Row 1 Cell 1	Row 1 Cell 2
Row 2 Cell 1	Row 2 Cell 2

#### 9.3.1 Spanning Multiple Rows and Cells

Use *rowspan* to span multiple rows merging together table rows and *colspan* to span across multiple columns.

#### HTML Table Rowspan Attribute:

```
<table border="1">
  <tr>
```

```

        <td><b>Column 1</b></td>
        <td><b>Column 2</b></td>
        <td><b>Column 3</b></td>
    </tr>
    <tr>
        <td rowspan="2">Row 1 Cell 1</td>
        <td>Row 1 Cell 2</td>
        <td>Row 1 Cell 3</td>
    </tr>
    <tr>
        <td>Row 2 Cell 2</td>
        <td>Row 2 Cell 3</td>
    </tr>
    <tr>
        <td colspan="3">Row 3 Cell 1</td>
    </tr>
</table>

```

#### HTML Colspan and Rowspan Attributes:

Column 1	Column 2	Column 3
Row 1 Cell 1	Row 1 Cell 2	Row 1 Cell 3
	Row 2 Cell 2	Row 2 Cell 3
Row 3 Cell 1		

### 9.3.2 Cell Padding and Spacing

With the *cellpadding* and *cellspacing* attributes, you will be able to adjust the spacing between table cells. Setting the *cellpadding* attribute determines how much space will exist between a table cell border and the elements contained within it, whereas *cellspacing* determines how much space will exist between each table cell. Color has been added to the table below to emphasize these attributes.

#### HTML Cellpadding/Cellspacing Code:

```

<table border="1" cellspacing="10" bgcolor="rgb(0,255,0)">
    <tr>
        <td><b>Column 1</b></td>
        <td><b>Column 2</b></td>
    </tr>
    <tr>
        <td>Row 1 Cell 1</td>
        <td>Row 1 Cell 2</td>
    </tr>

```

```

<tr>
  <td>Row 2 Cell 1</td>
  <td>Row 2 Cell 2</td>
</tr>
</table>

```

### HTML Cellspacing and Padding:

Column 1	Column 2
Row 1 Cell 1	Row 1 Cell 2
Row 2 Cell 1	Row 2 Cell 2

And now we will change the *cellpadding* of the table and remove the *cellspacing* from the previous example. This should clearly demonstrate the difference between *cellpadding* and *cellspacing*.

### HTML Code:

```

<table border="1" cellpadding="10" bgcolor="rgb(0,255,0)">
  <tr>
    <td><b>Column 1</b></td>
    <td><b>Column 2</b></td>
  </tr>
  <tr>
    <td>Row 1 Cell 1</td>
    <td>Row 1 Cell 2</td>
  </tr>
  <tr>
    <td>Row 2 Cell 1</td>
    <td>Row 2 Cell 2</td>
  </tr>
</table>

```

### HTML Cell Pads:

Column 1	Column 2
Row 1 Cell 1	Row 1 Cell 2
Row 2 Cell 1	Row 2 Cell 2

The value you specify for padding and spacing is interpreted by the browser as a pixel value. So a value of 10 is simply 10 pixels wide. Most HTML attributes that use numeric values for their measurements represent a pixel value.

## 9.4 Using Tables to implement Layout

One very common practice with HTML, is to use HTML tables to format the layout of an HTML page. A part of this page is formatted with two columns. As you can see on this page, there is a left column and a right column. This text is displayed in the left column. An HTML `<table>` is used to divide a part of this Web page into two columns. The trick is to use a table without borders, and maybe a little extra cell-padding. No matter how much text you add to this page, it will stay inside its column borders.



## 9.5 The bgcolor Attribute

Tables can be given background colors using the *bgcolor* attribute. The *bgcolor* attribute is used to set the background color of an HTML element. *Bgcolor* is one of those attributes that has become deprecated with the implementation of Cascading Style Sheets. The reason we've included it in this tutorial is because it will give us an opportunity to introduce web colors and also add some life to our HTML web page as we continue to progress through this tutorial. It will serve as a visual aid for you as you are learning the mechanics of building a table.

Without much effort, we can bring that boring white web page to life by adding some color with the *bgcolor* attribute.

### HTML Bgcolor Code:

```
<body bgcolor="silver">
  <p>This page now has a SILVER background!</p>
</body>
```

















### HTML Bgcolor:

This page now has a SILVER background!

### 9.5.1 Web Colors

Our example uses the text value, which is one of three different types of color values that can be used with the *bcolor* attribute. Below is a table of the 16 basic HTML color values that are available to HTML web designers.

#### HTML Basic Colors:

	Black		Gray		Silver		White
	Yellow		Lime		Aqua		Fuchsia
	Red		Green		Blue		Purple
	Maroon		Olive		Navy		Teal

While the table above illustrates only 16 colors, 16 is surely not the limit to our color wheel. As we mentioned, HTML supports three different types of color values including *text values* (which we've pretty much covered above), *numeric*, (RGB) and *hexadecimal values*. We'll go into more detail regarding these values so just sit tight. This next example offers a sneak peak at what these values may look like.

#### HTML Code:

```
<table bgcolor="#ff0000" border="1"><tr>
<td>A red colored table background using hexadecimal values "#FF0000".</td>
</tr></table>
```

```
<table bgcolor="rgb(0, 0, 255)" border="1"><tr>
<td>A blue colored table background using numeric, RGB values "rgb(0, 0, 255)".</td>
</tr></table>
```

```
<table bgcolor="lime" border="1"><tr>
<td>A lime colored table background using color names.</td>
</tr></table>
```

#### Table Bgcolor Values:

A lime colored table background using color names.

A red colored table background using hexadecimal values "#FF0000".

A blue colored table background using numeric, RGB values "rgb(0, 0, 255)".

*Hexadecimal* and *numeric* color values (RGB) allow HTML developers to expand the color wheel beyond 16 colors. Way beyond 16, in fact. Nonetheless, there's no need to memorize 256+ unique color combinations; instead, we can use numeric and hexadecimal values and calculate color shades. We'll show you how to use them in our [HTML Color Codes](#) page.

### 9.5.2 Coloring Fonts, Table Rows, & Table Columns

Here's a few common examples of *bgcolor* in action.

#### HTML Bgcolor Code:

```
<table>
  <tr bgcolor="#FFFF00"><td>This Row is Yellow!</td></tr>
  <tr bgcolor="#AAAAAA"><td>This Row is Gray!</td></tr>
  <tr bgcolor="#FFFF00"><td>This Row is Yellow!</td></tr>
  <tr bgcolor="#AAAAAA"><td>This Row is Gray!</td></tr>
  <tr bgcolor="#FFFF00"><td>This Row is Yellow!</td></tr>
  <tr bgcolor="#AAAAAA"><td>This Row is Gray!</td></tr>
</table>
```

#### Alternating Table Row Colors:

This Row is Yellow!
This Row is Gray!
This Row is Yellow!
This Row is Gray!
This Row is Yellow!
This Row is Gray!

Check out this "Scoreboard" we made with the use of *font color* and *bgcolor*!

#### HTML Code:

```
<table bgcolor="#000000">
  <tr><td bgcolor="#009900" align="right">
    <font color="#FFFF00">Green Bay</font></td>
    <td><font color="#FFFFFF">13</font></td></tr>
  <tr><td bgcolor="#0000FF" align="right">
    <font color="#DDDDDD">New England</font></td>
    <td><font color="#FFFFFF">27</font></td>
  </tr>
</table>
```



















### Scoreboard:

Green Bay	13
New England	27

### 9.5.3 Color Codes

Let's first review the 16 generic color values we mentioned previously before diving into the other, more complicated HTML coloring systems of *numeric* and *hexadecimal* values.

#### HTML String Color Codes:

	Black		Gray		Silver		White
	Yellow		Lime		Aqua		Fuchsia
	Red		Green		Blue		Purple
	Maroon		Olive		Navy		Teal

Any of the string values listed above such as "teal", "black", or "gray" can be passed as a color value to the HTML *bgcolor* attribute.

#### Tips

- If you are new to HTML, consider sticking with color names for setting your background color.
- There are 256 "true colors" with hexadecimal values. The colors are made out of hex-pairs for red, green, and blue. Examples: #99FFCC, #33AA44.
- Shades of gray occur when the 3 paired amounts of each color are equal. Example. "rgb(100,100,100)", "#333333," "#0A0A0A"
- Avoid bright, headache-causing color schemes! Keep your coloring distinct and purposeful.

## CHAPTER 10: HTML FORMS

### 10.1 Chapter Overview

A large part of developing a web based application involves handling interaction with user via their web browsers. One of the most common tasks in this area of web development involves presenting the user with forms to collect information, and then processing that information. Small wonder, then, that forms are the bedrock of user interaction on an Intranet. The ease with which HTML forms can be created is one of the technology's big benefits, for users and developers alike. HTML forms provide a rich set of data input features that emulate those found on common paper forms. For instance, a form can be used to capture manually-entered information to a database. A related application is the use of forms to input free text, such as a discussion forum message.

In this chapter, you learn:

- How to design and build forms using HTML
- What you can do with forms on an Intranet
- How online forms can replace paper flows
- Where Web-based forms don't fit

Upon completion of this chapter, you should be able to

- ✓ Design a form
- ✓ Create a text entry field
- ✓ Add radio buttons
- ✓ Add checkboxes
- ✓ Create a pull-down menu
- ✓ Add a push button

### 10.2 Introduction

HTML web forms are a composition of buttons, checkboxes, and text input fields embedded inside of HTML documents with one goal in mind: to capture user input. By doing things such as providing fields for user data such as names, phone number, and email addresses, web forms give users the opportunity to interact directly with a webpage.

HTML forms are placed on a web page using the `<form>` tag. This tag should encapsulate a series of other form elements, identifying them as a single cohesive web form.

#### HTML Form Element:

```
<form name="myWebForm" action="myServerSideScript.php" method="post">  
  <input type="checkbox" /> Checkbox 1<br />  
  <input type="text" /> Text Field 1<br />  
  <input type="submit" value="SUBMIT" />  
</form>
```

### HTML Web Form:



Checkbox 1

Text Field 1

HTML form elements rely on *action* and *method* attributes to identify where to send the form data for processing (action) and how to process the data (method). In the code above, we've inserted some make-believe values to represent what a typical HTML form might look like behind the scenes.

Unfortunately, HTML alone is unable to process form data. A scripting language such as PHP, PERL, and/or JavaScript must be used with HTML forms to process data captured by HTML form elements. For the purpose of following along, we can also adjust the *action* property slightly to have the form launch an email client instead of processing a make-believe server-side script. This will provide us with some form interactivity for us as we learn more about HTML forms.

### Email Form Element:

```
<form name="myWebForm" action="mailto:youremail@email.com" method="post">  
  <input type="checkbox" /> Checkbox 1<br />  
  <input type="text" /> Text Field 1<br />  
  <input type="submit" value="SUBMIT" />  
</form>
```

### HTML Email Form:



Checkbox 1

Text Field 1

Now when the **SUBMIT** button is clicked, the user should see their default email client launch.

HTML forms provide user interaction between visitors and the website while simultaneously collecting priceless user data from your users. They are a vital tool for any webmaster, and these days, it is common place to see form elements embedded in every web page.

### Tips

- Remember to set the name and value attributes for your forms so the document created will be neatly organized.
- Remember to place submit buttons with the form tags to submit the document correctly.

Attribute	What it does...
<b>ACTION (required)</b>	Specifies the URL to which FORM content is to be sent for processing. This is usually the address of a script or mailto target. Example: <FORM ACTION="mailto:g.benett@ieee.org">
<b>METHOD</b>	HTTP provides several methods for clients to request service: GET, POST, PUT, HEAD. By far the most prevalent is GET, which gets an object from the server, and POST, which posts data to an object on the server. As a rule of thumb, use POST when passing form data to a program, and GET when making a request or performing a search. Examples: <FORM ACTION="http://www.que.com/cgi-bin/readform" METHOD=POST> <FORM ACTION= <a href="http://www.que.com/cgi-bin/locate?G+Benett">http://www.que.com/cgi-bin/locate?G+Benett</a> METHOD=GET>

### 10.3 Input Element(s)

HTML input elements are form elements such as text fields, checkboxes, and buttons. The name comes from the <input> tag, which is the mark-up that identifies web form components. The <input> tag relies upon a few attributes to classify and name each form item, providing the web developer with a means to manipulate each element individually.

The *type* attribute determines what kind of input element to render to the screen. Options here include: *text*, *checkbox*, *radio*, *button*, *submit*, *reset*, *password*, and *hidden* form elements. Each has its own unique functionality and customizable presentation.

#### Input Element Code:

```
<form name="myWebForm" action="mailto:youremail@email.com" method="post">
  Check Me: <input type="checkbox" /><br />
  Name: <input type="text" /><br />
  Yes: <input type="radio" /> No: <input type="radio" /><br />
  <input type="submit" value="SUBMIT" />
  <input type="reset" value="RESET" />
</form>
```

#### Input Element sample output:

Check Me: ☐

Name:

Yes: ☐ No: ☐

### 10.3.1 Web Forms: Value Attribute

The *value* attribute plays a different role depending on the *type* of the input field. For example, when used with an HTML button, the *value* attribute defines the text inside of the button. When used with a text field, the *value* attribute populates the field with a default value.

#### HTML Input Element Code:

```
<form name="myWebForm" action="mailto:youremail@email.com" method="post">  
  Check Me: <input type="checkbox" /><br />  
  Name: <input type="text" value="David" /><br />  
  Yes: <input type="radio" /> No: <input type="radio" /><br />  
  <input type="submit" value="Send" />  
  <input type="reset" value="Clear" />  
</form>
```

#### HTML Input Elements:

Check Me: ☐

Name:

Yes: ☐ No: ☐

### 10.3.2 Name and ID Attributes

Setting the *name* and *id* attributes inside of form elements is a good habit. The element name and/or id will later serve as the link between your HTML form and any server-side script that you may deploy later on to process that data. Perhaps the best approach is to use both attributes in your code, since varying scripting languages demand one identifying attribute over the other.

#### Input Element Code:

```
<form name="myWebForm" action="mailto:youremail@email.com" method="post">  
  Check Me: <input name="" id="" type="checkbox" /><br />  
  Name: <input name="userName" id="userName" type="text" /><br />  
  Yes: <input name="radioItem" id="radioItem" type="radio" /> No: <input name="radioItem"  
id="radioItem" type="radio" /><br />  
  <input name="submitForm" id="submitForm" type="submit" value="SUBMIT" />  
  <input name="resetForm" id="resetForm" type="reset" value="RESET" />  
</form>
```

#### HTML Input Elements:

Check Me: ☐

Name:

Yes: ☐ No: ☐

## 10.4 Text Fields

Text fields offer a small rectangular box that's always ready to receive information from viewers. Users will notice that when they click these fields, the cursor will change from the typical arrow to a pipe character ( | ), allowing for text entries to be typed inside each input field.

A text field is placed on a web page using the `<input>` tag, with the *type* attribute set with a value of "text".

### Text Field Code:

```
<form name="myWebForm" action="mailto:youremail@email.com" method="post">  
First: <input title="Please Enter Your First Name" id="first" name="first" type="text" /> Last:  
<input title="Please Enter Your Last Name" id="last" name="last" type="text" />  
<input type="submit" value="SUBMIT" />  
</form>
```

### Text Field sample output:

First:  Last:

Text fields are designed to capture single words or phrases from the user. That information may then be processed through some kind of client/server side script (PHP, PERL, JavaScript). If you do plan on processing the data, be sure to include the *name* and *id* attributes. A descriptive *title* is also a great visual aid for providing a tool-tip display for your web elements.

### 10.4.1 Text Fields: Size Attribute

To modify the visual presentation of a text field, one needs to pass an integer value to the *size* attribute. The value represents how many characters a text field can display within the text field window.

As the web designer, it is your job to analyze and predict the average length of characters that will be entered into each field by your users. First and last names may generally vary from 8-24 characters in length, while a typical email address may range from 12-36 digits.

### Text Field Size code:

```
<form name="myWebForm" action="mailto:youremail@email.com" method="post">  
First: <input title="Please Enter Your First Name" id="first" name="first" type="text" size="12"  
><br />  
Last: <input title="Please Enter Your Last Name" id="last" name="last" type="text" size="18"  
><br />
```

```
<input type="submit" value="SUBMIT" />
</form>
```

#### Text Field Size sample output:

First:

Last:

If the user happens to enter more digits than the size attribute value, these characters will not be discarded; it just means that the user will not be able to see all of their input at once. Instead, they will be forced to scroll to the beginning and end of the input element, which tends to discourage user interaction.

### 10.4.2 Text Fields: Maxlength Attribute

*Maxlength* is an optional attribute that accepts an integer value. It allows the developer to restrict the number of characters a user can type in a specific text field.

#### HTML Text Field Maxlength:

```
<form name="myWebForm" action="mailto:youremail@email.com" method="post">
First: <input title="Please Enter Your First Name" id="first" name="first" type="text" size="12"
maxlength="3" value="David" /><br />
Last: <input title="Please Enter Your Last Name" id="last" name="last" type="text" size="18"
maxlength="3" value="Smith" /><br />
<input type="submit" value="SUBMIT" />
</form>
```

#### HTML Text Field Maxlength:

First:

Last:

We've also called upon the *value* attribute to place some text inside the text fields for you!

### 10.5 Password Fields

HTML password fields are designed to capture user input, but disguise each character with an asterisk (\*) instead of displaying the entered digits. They offer a user on-screen privacy while he or she is entering a password.

Password fields are placed on a website using the `<input>` tag and specify a value of "password" for the *type* attribute.

#### Password Field Code:

```
<form name="myWebForm" action="mailto:youremail@email.com" method="post">  
Password: <input type="password" title="Please Enter Your Password" size="8" /><br />  
<input type="submit" value="SUBMIT" />  
</form>
```

#### Password Fields Sample Output:

Password:

---

### 10.5.1 Password Fields: Attributes

Password form fields may be customized using the same attribute as outlined in the HTML Text Fields.

#### Password Input Field Code:

```
<form name="myWebForm" action="mailto:youremail@email.com" method="post">  
First: <input title="Please Enter Your First Name" id="first" name="first" type="text" size="12"  
maxlength="12" /> Last: <input title="Please Enter Your Last Name" id="last" name="last"  
type="text" size="18" maxlength="24" /><br />  
Password: <input type="password" title="Please Enter Your Password" size="8" maxlength="8"  
/><br />  
<input type="submit" value="SUBMIT" />  
</form>
```

#### Password Input Field sample output:

First:  Last:

Password:

Password fields offer a very thin layer of security by visually concealing passwords; they offer no security whatsoever against maintaining the integrity of the password data. From data is processed in plain text and can be readily sniffed by a hacker, unless **HTTPS** is used to encrypt the data.

### 10.6 Reset Buttons

A reset button allows users to basically clear their web form. It wipes values from all fields by "resetting" the form to its default appearance.

Set the *type* attribute of the `<input>` tag to "reset" to incorporate a reset button into a web form.



### HTML Reset Button Code:

```
<input type="reset" value="Reset" />  
<input type="reset" value="Start Over" />
```

### Two HTML Reset Buttons:

Placing a reset button inside of a form tag automatically associates the reset button with each form element and delivers a useful feature for your viewers.

### HTML Code:

```
<form action="myphp.php" method="post"> <input type="text" size="12" maxlength="12" />  
<input type="text" size="24" maxlength="24" />  
<input type="reset" value="Reset" /> </form>
```

### Reset Forms:



Fill out some information in the field boxes and press **reset** to experience a reset form!

## 10.7 Submit Buttons

Submit buttons send form data to a back-end process or application. The back-end process then verifies and processes the data, eventually passing the information into some database application.

Set the *type* attribute of the `<input>` tag to "submit" in order to place a submit button on a web page.

### HTML Submit Buttons:

```
<input type="submit" value="Submit" />  
<br /> <input type="submit" value="Send" />  
<br /> <input type="submit" value="Submit Form" /><br />
```

### Three HTML Submit Buttons:

Notice that in the above example, we also changed what was written on our button using the *value* attribute. This can be changed to any value you wish.

### 10.7.1 Form Submission - Action

Submission buttons send form data to whatever action has been designated by the *action* attribute of the encapsulating `<form>` element. We learned about the *action* attribute earlier in this chapter.

If you've been following along, we've also been using the deprecated *mailto* action to send form data to our default email client. This will allow us to get a sense of how form values are transferred to an action.

#### HTML Code:

```
<form method="post" action="mailto:youremail@youremail.com" >  
First:<input type="text" name="First" size="12" maxlength="12" />  
Last:<input type="text" name="Last" size="24" maxlength="24" />  
<input type="submit" value="Send Email" />  
</form>
```

#### Form Action:

First:

Last:

Fill out the above form, and as your mail program opens, you can change the email address to a personal address and then send the results using the form.

#### Tips

The HTML Button is probably the second most commonly used Graphical User Interface (GUI) component when developing HTML forms. There are three types of Button object available. The *type*= attribute is used to define which type of button is to be created. The three different types are:

- **type="BUTTON"** - The basic button which performs no action by default unless an event handler is assigned to it.
- **type="SUBMIT"** - The submit button. When pressed this button causes the data in the Form to be sent to the server using the settings defined in the enclosing `<Form>` tag. If the *onsubmit* attribute on the enclosing `<form>` tag has been specified this will be executed before the form data is submitted (useful for JavaScript form data validation).
- **type="RESET"** - The reset button. When pressed causes the fields in the Form to be either cleared, or reset to the default value (if one has been specified).

## 10.8 Checkbox Forms

Setting the *type* attribute of an `<input>` tag to *checkbox* places a checkbox element onto the web page.

Deploy checkbox elements in a situation when the user must check all boxes that apply (or none). A scripting language such as PHP will easily handle this form element, returning all elements the user has checked.

**HTML Checkbox Code:**

```
<form name="myWebForm" action="mailto:youremail@email.com" method="post">
```

```
<p>Please select every sport that you play.</p>
```

```
Soccer: <input type="checkbox" name="sports" value="soccer" /><br />
```

```
Football: <input type="checkbox" name="sports" value="football" /><br />
```

```
Baseball: <input type="checkbox" name="sports" value="baseball" /><br />
```

```
Basketball: <input type="checkbox" name="sports" value="basketball" />
```

```
</form>
```

**HTML Checkbox Form:**

Please select every sport that you play.

Soccer: ☐

Football: ☐

Baseball: ☐

Basketball: ☐

Checkboxes are used for instances where a user may wish to select multiple options, such as in the instance of a "check all that apply" question.

**10.8.1 Checkboxes Selected**

A checkbox element can be placed onto a web page in a pre-checked fashion by setting the *checked* attribute with a "yes" value. By doing so, this element will now default to a checked status each time the HTML page is loaded.

**HTML Checkbox Selected Code:**

```
<form name="myWebForm" action="mailto:youremail@email.com" method="post">
```

```
<p>Please select every sport that you play.</p> Soccer: <input type="checkbox" checked="yes"
```

```
name="sports" value="soccer" /> <br /> Football: <input type="checkbox" name="sports"
```

```
value="football" /> <br /> Baseball: <input type="checkbox" name="sports" value="baseball" />
```

```
<br /> Basketball: <input type="checkbox" checked="yes" name="sports" value="basketball" />
```

```
</form>
```

**HTML Pre-Selected Checkboxes:**

Please select every sport that you play.

Soccer: ☒

Football: ☐

Baseball: ☐

Basketball: ☒

## 10.9 Radio Buttons

Radio form elements allow the user to "bubble" in their choice and limit each question to only one selection per radio group.

Place a radio element on to your web page by setting the *type* attribute of the `<input>` tag to "radio".

### HTML Radio Input Code:

```
<form name="myWebForm" action="mailto:youremail@email.com" method="post">
<h4>Please select your favorite food category.</h4>
<input type="radio" name="food" /> : Italian<br />
<input type="radio" name="food" /> : Greek<br />
<input type="radio" name="food" /> : Chinese<br />
</form>
```

### HTML Radio Fields:

**Please select your favorite food category.**

☐ : Italian

☐ : Greek

☐ : Chinese

By naming each field similarly with a type of cuisine, we have created a relation, or a "grouping," of radio elements. This is how we link each element together and assure that the user is able to select only one answer.

Let's now take a look at how we can group together different sets of radio elements and simulate capturing two pieces of user data: gender and favorite food.

### HTML Multiple Radios:

```
<form name="myWebForm" action="mailto:youremail@email.com" method="post">
<h4>Please select your favorite food category.</h4>
<input type="radio" name="food" /> : Italian<br />
<input type="radio" name="food" /> : Greek<br />
<input type="radio" name="food" /> : Chinese<br />
```

```
<h4>Please select your gender.</h4>
<input type="radio" name="gender" /> : Male<br />
<input type="radio" name="gender" /> : Female<br />
</form>
```

### HTML Multiple Radio Fields:

**Please select your favorite food category.**

- ☐ : Italian
- ☐ : Greek
- ☐ : Chinese

**Please select your gender.**

- ☐ : Male
- ☐ : Female

Words/values applied to the *value* attribute is the value or 'answer' passed to any server-side script language we may have in place to record the results.

### 10.9.1 Radio Buttons: The Checked Attribute

By using the *checked* attribute, we adjust the form to load with a value already checked as the default setting.

#### HTML Code:

```
<form name="myWebForm" action="mailto:youremail@email.com" method="post">
<h4>Please select your favorite food category.</h4>
<input type="radio" name="food" checked="yes" /> : Italian<br />
<input type="radio" name="food" /> : Greek<br />
<input type="radio" name="food" /> : Chinese<br />
</form>
```

#### Default Italian:

- ☒ : Italian
- ☐ : Greek
- ☐ : Chinese

Using either/or logic, radios provide a very efficient way to capture very specific data from visitors. Remember to use *radio* elements only when you'd like the viewer to select only a single value, just as you might expect to see when taking a multiple-choice test in school.

## 10.10 Select Fields

HTML *select* fields provide essentially the same functionality as HTML Checkbox Fields. They allow the user to select one or more values from a pre-determined series of options.

Incorporating a select field into a web page is done using the `<select>` tag. List values are then added to the field using the `<option>` tag, similar to how list items `<li>` are added to ordered list elements (`<ol>`).

### HTML Drop Down List:

```
<select name="selectionField">
  <option value="CA" >California -- CA </option>
  <option value="CO" >Colorado -- CO</option>
  <option value="CN" >Connecticut -- CN</option>
</select>
```

### HTML Drop Down List:

By default, select fields, popularly called drop down lists, only allow the user to choose a single value. This behavior and appearance may be changed by adjusting the *multiple* and *size* attributes as demonstrated below.

### HTML Selection Field Code:

```
<select size="3" name="selectionField" multiple="yes" >
  <option value="CA" >California -- CA </option>
  <option value="CO" >Colorado -- CO</option>
  <option value="CN" >Connecticut -- CN</option>
</select>
```

### HTML Selection Element:

With the above settings, the user is now able to select multiple values by pressing and holding the **Control** (ctrl) key and clicking each value.

#### 10.10.1 Disabling Selection Fields

Disabling a selection field is achieved by setting the *disabled* attribute to "yes". But before doing that, you should set at least one of the values to be selected. Doing so renders a read-only selection field on the page that can inform your users of their selections without allowing them to alter the selection.

### HTML Read-Only Selection Field:

```
<select size="3" name="selectionField" multiple="yes" disabled="yes">
  <option value="CA" >California -- CA </option>
  <option selected value="CO" >Colorado -- CO</option>
  <option value="CN" >Connecticut -- CN</option>
</select>
```

## 10.11 Hidden Field

Hidden fields allow a coder to pass values to form elements in a subtle manner. An experienced web developer will utilize these fields to pass temporary, or session-based data, from one form to another or to store information that has already been entered in by the user.

Place a hidden input field into your web forms using the `<input>` tag and set the *type* attribute to "hidden". This field can be customized using any of the attributes discussed in the HTML Input and HTML Text Fields.

### HTML Hidden Input Field:

```
<form name="myWebForm" action="mailto:youremail@email.com" method="post">
First: <input title="Please Enter Your First Name" id="first" name="first" type="text" size="12"
maxlength="12" /> Last: <input title="Please Enter Your Last Name" id="last" name="last"
type="text" size="18" maxlength="24" /><br />
Password: <input type="password" title="Please Enter Your Password" size="8" maxlength="8"
/><br /><br />
<input type="hidden" name="orderNumber" id="orderNumber" value="0024" /><br />
<input type="submit" value="SUBMIT" />
<input type="reset" value="RESET" />
</form>
```

### HTML Hidden Fields:

First:  Last:

Password:

It is important to note that HTML hidden fields do not offer any data security. Like all HTML form elements, the data is processed in plain text and is readily accessible by any novice hacker.

## 10.12 Upload Field

Upload fields provide the interface that allows users to select a local file and upload it to the web server. An upload field renders as two parts -- an empty text field and a **Browse** button that opens up a local window explorer on the user's computer. This allows them to quickly browse to the local file and automatically fills in the file path inside of the text field.

Setting the *type* attribute of the `<input>` to "file" places the upload element on a web page.

### HTML Upload Field Code:

```
<form name="myWebForm" action="mailto:youremail@email.com" method="post">
<input type="file" name="uploadField" />
</form>
```

### HTML Upload Field:

#### 10.12.1 Max File Size Field

File transferring across the internet is a complicated process and should include many layers of security. HTML alone cannot ensure safe and secure file transferring, but it can offer a first line of defense. Using a MAX\_FILE\_SIZE hidden field can limit the size of files that are transferred.

Placing a restraint on an HTML upload field is accomplished by using a hidden input field with the *name* attribute set to **MAX\_FILE\_SIZE**.

### HTML MAX\_FILE\_SIZE Code:

```
<form name="myWebForm" action="mailto:youremail@email.com" method="post">
<input type="hidden" name="MAX_FILE_SIZE" value="500" />
<input type="file" name="uploadField" />
</form>
```

### HTML MAX\_FILE\_SIZE:

The *value* attribute specifies the maximum allowable kilobytes (KB) for any file selected by the user.

## 10.13 Text Areas

An HTML textarea is an oversized Text Field capable of capturing "blurb" type information from a user. If you've ever posted on a forum or left feedback on your favorite blog, you probably do so using an HTML textarea.

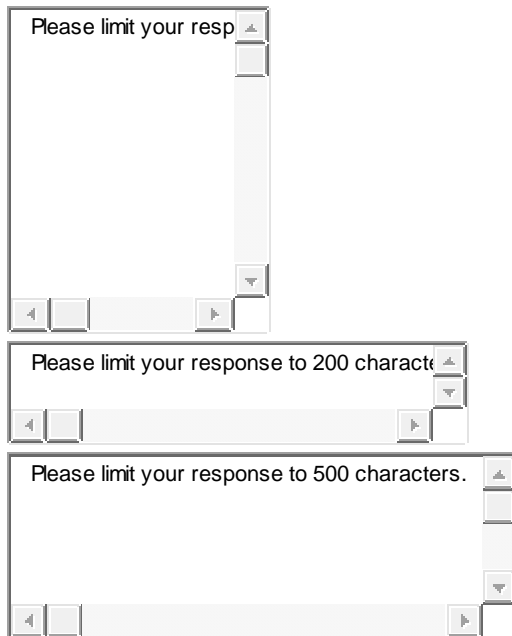
Embed textareas in HTML documents using the <textarea> tag. Any text placed between the opening and closing textarea tags will be rendered inside the textarea element as the "default" text. This makes for a great way to give users an example or description of how to go about filling out the text area field. Something like, "Please limit your response to 100 characters," would be an ideal description.

### HTML Textarea Code:

```
<textarea name="myTextArea" cols="20" rows="10">Please limit your response to 100
characters.</textarea><br />
<textarea name="myTextArea" cols="40" rows="2">Please limit your response to 200
characters.</textarea><br />
<textarea name="myTextArea" cols="45" rows="5">Please limit your response to 500
characters.</textarea><br />
```



### HTML Textarea Form Element:



As you may have noticed, the attributes *cols* (columns) and *rows* control the rendered size of the textarea. These constraints only impact how the textarea is rendered visually, and in no way do they limit the maximum number of characters a user can place inside the textarea. In fact, if you fill up the fields above with text, the fields will just continue to grow as you type and you will be able to scroll up and down as you please. Limits must be set with JavaScript and/or a server-side scripting language such as PHP.

#### 10.13.1 Text area Wrap

The *wrap* attribute refers to how the user input reacts when it reaches the end of each row in the text field. Wrapping can be defined using one of three values:

- soft
- hard
- off

"Soft" forces the words to wrap once inside the textarea but once the form is submitted, the words will no longer appear as such, and line breaks and spacing are not maintained.

"Hard" wraps the words inside the text box and places line breaks at the end of each line so that when the form is submitted the text will transfer as it appears in the field, including line breaks and spacing.

"Off" sets a textarea to ignore all wrapping and places the text into one ongoing line.

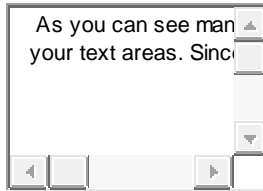
#### HTML Text Area Word Wrap Code:

```
<textarea cols="20" rows="5" wrap="hard">
```

As you can see many times word wrapping is often the desired look for your textareas. Since it makes everything nice and easy to read and preserves line breaks.

</textarea>

### HTML Text Area Word Wrap:



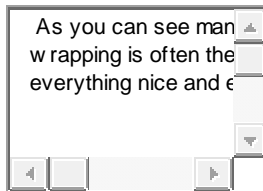
Here's a textarea with no word wrapping at all!

### HTML Text Area No Word Wrap:

```
<textarea cols="20" rows="5" wrap="off">
```

As you can see many times word wrapping is often the desired look for your textareas. Since it makes everything nice and easy to read. </textarea>

### HTML Text Area No Word Wrap:



## 10.13.2 Text Areas: Readonly

Setting a "yes" or "no" value for the *readonly* attribute determines whether or not a viewer has permission to manipulate the text inside the text field.

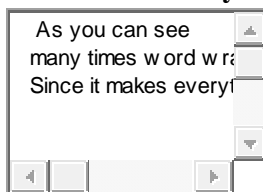
### HTML Readonly Attribute:

```
<textarea cols="20" rows="5" wrap="hard" readonly="yes">
```

As you can see many times word wrapping is often the desired look for your text areas. Since it makes everything nice and easy to read.

</textarea>

### HTML Read Only Text Areas:



This read-only behavior allows a web surfer to see and highlight the text inside the element, but he or she cannot alter it in any way. When highlighted, the user may also Copy (Ctrl + C on a PC, Ctrl-Click on a Mac) the text to local clipboard and paste it anywhere he/she pleases.

### 10.13.3 Text Areas: Disabled

Disabling the textarea altogether prevents the surfer from highlighting, copying, or modifying the field in any way. To accomplish this, set the *disabled* property to "yes".

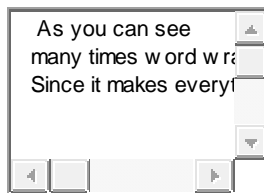
#### HTML Code:

```
<textarea cols="20" rows="5" wrap="hard" disabled="yes">
```

As you can see many times word wrapping is often the desired look for your text areas. Since it makes everything nice and easy to read.

```
</textarea>
```

#### Disabled Textareas:



Keep in mind that just because users are unable to copy from the screen directly doesn't prevent them from taking a screen capture or extracting the data from the source code. Disabling the textarea offers no security whatsoever.

## CHAPTER 11: MULTIMEDIA ELEMENTS

### 11.1 Introduction

Information scientists should be concerned with the provision of information in the formats most suited to the differing needs of various types of user, each of which must be clearly differentiated. Audiovisual materials can reach out to sections of the public for whom the traditional print-based materials have little impact, e.g. to those who are reluctant to use the printed word, and to those with visual and other handicaps. Knowledge of including multimedia in your website would thus provide you with an upper hand over using plain text. The term multimedia is used in our context to describe the integrated presentation of text, graphics, sound and images, including digital video via computer-based media. Multimedia includes the study of animation and games for mobile devices, web design, digital video, and audio production. Students who study multimedia are able to create and apply digital content for communication, education and entertainment.

### 11.2 Music Files

Inserting music onto a web page is relatively easy these days. In the past, multiple tags had to be used because browsers did not have a uniform standard for embedded media files. This is however not the case with the current HTML standards.

Music is inserted onto a web page with the use of the ***embed*** tag. There are other ways to link to music, but embed is now considered the standard for inserting media. Below is an minimalist example of the embed tag using the *src* attribute to define the media file's location.

#### HTML Embed Tag Code:

```
<embed src="beethoven.mid" />
```

```
<p>Above is an embedded media player. To stop the music press stop/pause.</p>
```

Depending on what kind of media software you or your visitor has installed, the above example will appear slightly different. To make your embedded player display properly, change the attributes associated with display.

#### 11.2.1 Embed Attributes - Related to Display

To customize the appearance of the embedded media player, be sure to set the following attributes.

- width - The width of the media player.
- height - The height of the media player.
- hidden - Determines if the media player is visible. If this value is set to "true", the media player will not be displayed. We recommend using this attribute only if you know that your visitors will not want the option to stop the music that is playing on your web page (The values are "true" or "false").

**HTML Code:**

```
<embed src="beethoven.mid" width="360" height="165" />
```

**11.2.2 Embed Attributes - Related to Functionality**

To customize the functionality of the embedded media player, be sure to set the following attributes.

- **autostart** - Allows media player to start automatically (values are "true" and "false")
- **loop** - Sets whether or not the media file will repeat (values are "true" and "false")
- **volume** - Sets the volume of the media file (values range from "0" to "100")

**HTML Code:**

```
<embed src="beethoven.mid" autostart="false" loop="false"  
volume="60" />
```

**Tips**

- Be careful when placing music on your website. If done poorly, users will be annoyed by the music and will leave.
- Only set the hidden attribute if you are certain your visitors will not want to stop the music.
- If you want your music to play over and over again, be sure to set the loop attribute to "true".

**11.3 Video Codes**

Video files, including YouTube videos, are embedded into an HTML document using the `<embed>` tag. The *src* attribute defines what video file to embed into the page. The `<embed>` tag does not require a closing tag. Here is a look at the `<embed>` tag with a global URL. Notice that its controls, including Play, Stop, Pause, and volume, are already included.

**HTML Code:**

```
<embed src="http://www.Facebook.com/files/html/htmlexample.mpeg"  
autostart="false" />
```

You may start and stop your movie files by either pressing the buttons at the bottom of the object or by single-clicking on the object itself. The movie can be restarted by double-clicking your mouse.

**11.3.1 Video Media Types**

Flash (.swf) and MOV (.mov) file types are also supported by the `<embed>` tag.

- **.swf** - Macromedia's Flash file types - very high compression, great for the web!.

- **.wmv** - Microsoft's Window's Media Video file types - good quality, variable compression.
- **.mov** - Apple's Quick Time Movie format - good quality, variable compression.
- **.mpeg** - the accepted standard for web movie files created by the Moving Pictures Expert Group - good quality, variable compression.

The list above outlines some of the most common "internet-ready" video files. Macromedia's .swf and .mpeg formats may be the best options for use with the web because the high compression rate of these file types reduces file size and expedites the download/buffering periods for your page visitors.

You may also simply place the URL of your media files into the *href* attribute of an anchor tag, much like the concept of "thumbnailing" images.

#### **HTML Code:**

```
<a href="http://www.Facebook.com/pics/flash/motiontween1easy.swf">
motiontween1easy.swf</a>
```

#### **Flash Media:**

[motiontween1easy.swf](http://www.Facebook.com/pics/flash/motiontween1easy.swf)

### **11.3.2 YouTube Videos**

YouTube videos can be included in HTML documents, and Google offers the code to do so right on the same page as the video itself!

The code offered by YouTube includes a small handful of parameters that help customize the embedded video object, and if you dive deep enough into the code, you will be able to identify the `<embed>` element and see the *src* attribute pointing to the URL of the media file.

#### **YouTube Video Code:**

```
<object          width="425"          height="344"><param          name="movie"
value="http://www.youtube.com/v/opVb89Cmrtkamp;hl=en&fs=1">
  </param><param name="allowFullScreen" value="true">
  </param><param name="allowscriptaccess" value="always"></param>
  <embed  src="http://www.youtube.com/v/opVb89Cmrtk&hl=en&fs=1"  type="application/x-
shockwave-flash"    allowscriptaccess="always"    allowfullscreen="true"    width="425"
height="344">
  </embed>
</object>
```

### **11.3.3 Embed Attributes**

To customize the functionality of the embedded media player, be sure to set any of the following attributes.

- **autostart** - Controls the media's ability to start without prompting (values are "true" or "false")
- **hidden** - Controls whether or not the play/stop/pause embedded object is hidden or not (values are "true" or "false"; hide your embeded media if you just want background noise)
- **loop** - Controls the ability of the media to continuously play (values are "true" and "false")
- **playcount** - Sets a playcount which means the media will repeat itself  $x$  number of times, instead of continuously as with the loop attribute above (a playcount of "2" will repeat the video twice)
- **volumn** - Sets a numeric value for the loudness of your media (values are "0" through "100")

## CHAPTER 12: META TAGS AND META DATA

### 12.1 Chapter Overview

Meta tags are used to supply information for search engines that will not be seen by the web surfer. These invisible units provide a flag for search engines to investigate and will then present that data to any potential users that stumble across your site through a search engine.

In the past, meta tags were *the* primary means for your site to be recognized by web spiders, but webmasters abused meta tags to improve their rankings in search engines. As a result, search engines have since modified their approach to keep results accurate. They now rely less on meta tags. Nevertheless, you should still include meta for those search bots that still do recognize them.

### 12.2 Meta Tag Description

Search engines are the compasses of the web and help users navigate from site to site. Chances are, if you've used a search engine, you've probably seen the *description* meta tag in action.

Meta elements must be placed inside of the <head> element in order for them to be recognizable by web crawlers and bots. The <meta> tag generally requires the *name* and *content* attributes to be working together to present your web page in a good light.

#### HTML Code:

```
<head>
<meta name="description" content="Facebook contains webmaster tutorials." />
</head>
```

The *description* meta element allows the developer to summarize the content that can be found on the page and is often the first chance you'll have to attract visitors. These brief narratives and hooks are often the only opportunity you'll have to generate a lasting first impression.

### 12.3 Keyword Meta Tags

Keywords and/or phrases may be placed inside the *keyword* meta element. You should specify the most popular search terms you believe someone would use to reach your website. A few years back, you could spam this meta tag with any and every keyword possible to gain ranking on search engines. Now, however, repeated words, or words that do not pertain to the content of the site, will not benefit your search engine rankings.

#### HTML Code:

```
<head>
<meta name="keywords" content="keyword, key keywords, etc" />
</head>
```



Separate each phrase/word with a comma to create large lists. An example of the keywords meta tag for Facebook.com would go something like this:

**HTML Code:**

```
<head>
<meta name="keywords" content="HTML, XHTML, CSS, tutorials, Facebook" />
</head>
```

Keep in mind that driving traffic and having your site listed high in the search engine rankings is not as easy as placing keywords inside your meta element. The phrase "Search Engine Optimization (SEO)" was coined to describe the rigorous process involved in achieving rankings in search engines. While meta tags do play a small role in this process, they are by no means a one-stop shop for your SEO needs.

## 12.4 Refresh and Redirect Meta

Later down the road, you may need to redirect traffic to another domain. A common reason might be that you have just purchased a better domain name and would like to retain your old visitors, yet still use your new domain. With the *refresh* meta tag, you will be able to redirect visitors to the website of your choice or simply refresh your own page to update dynamic content automatically.

For the *refresh* meta tag, the *content* attribute accepts two arguments separated by a semicolon (;). The first argument specifies the number of seconds between refreshes or redirection and the 2nd argument is a URL of where the browser will relocate.

**HTML Redirect Meta Tag:**

```
<head>
<meta http-equiv="refresh" content="10; url=http://www.Facebook.com" />
</head>
```

The above code refreshes Facebook's home page every 10 seconds. A quick refresh may be necessary for news, stocks, or any other time-sensitive information. The most common use for this type of meta tag, however, is redirection. To redirect a viewer automatically, just change the URL to the new site, like shown below. This code will send your visitors to espn.com after landing at your site for five seconds.

**HTML Page Refresh Meta Tag:**

```
<head>
<meta http-equiv="refresh" content="5; url=http://www.espn.com" />
</head>
```

## 12.5 Revised Meta

The *revised* meta tag records when the last update was done to the site.

**HTML Code:**

```
<head>  
<meta name="revised" content="Happy New Year: 1/1/2003" />  
</head>
```

Don't forget to get a little meta with your pages!

**Tips**

- It's important not to repeat words in the description or keywords meta tags
- Meta is not the only way to have your site seen by search engines. Do not rely on meta tags alone to get your website listed on search engines.
- If your domain (.com) ever changes, remember to place a simple, "Our site has moved" message as the existing domain, and then use a redirect meta tag to make life easier for your viewers that already may have your site bookmarked.

## CHAPTER 13: HTML STYLE ATTRIBUTES

### 13.1 Chapter Overview

Understanding the HTML *style* attribute will provide you with a preview into the Cascading Style Sheet (CSS) world. In fact, the code we'll be using with *style* is indeed CSS code known as Internal CSS. CSS styling brings a whole new dimension to a website and offers endless customization of HTML elements and web page design.

### 13.2 Introduction

When the *style* attribute was introduced into the HTML language along with CSS, a number of HTML attributes and tags became obsolete. Manipulation of the fonts and color of HTML elements is now accomplished through CSS styling, instead of stacking bulky formatting tags one inside the other.

#### HTML Style: Inline CSS:

```
<p id="contentParagraph" style="color: #0900C4;">  
Here we've changed the font color of this paragraph to blue.  
</p>
```

#### HTML Styling:

Here we've changed the font color of this paragraph to blue.

In the HTML Font lesson, we achieved similar results, but the code used to do so was cumbersome and inefficient.

### 13.3 Styling

As we mentioned, the values passed to the *style* attribute are actually CSS code. This means that we can go ahead and pass a series of values at once, changing several properties in one go. Simply separate each CSS attribute with a semicolon (;).

#### HTML Font Styling:

```
<p id="contentParagraph" style="font-family: Georgia ; font-size: 12pt; color: #0900C4;">  
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut  
labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco  
laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in  
voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non  
proident, sunt in culpa qui officia deserunt mollit anim id est laborum.  
</p>
```

#### HTML Font Styling:

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco

laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Inline CSS with the HTML *style* attribute offers a great way to improve the visual display of web elements and pages. With this new understanding of HTML and CSS, you're well on your way to mastering web design.

### 13.3.1 Div Element(s)

The <div> tag is nothing more than a container unit that encapsulates other page elements and divides the HTML document into sections. Web developers use <div> elements to group together HTML elements and apply CSS styles to many elements at once. For instance, by wrapping a set of paragraph elements into a <div> element, the developer can take advantage of CSS styles and apply a font to all paragraphs at once by applying a font style to the <div> tag instead of coding the same style for each paragraph element. Group together text elements within a <div> tag to slice up HTML documents.

#### HTML Div Element Code:

```
<div id="myDiv" name="myDiv" title="Example Div Element">
  <h5>Subtitle</h5>
  <p>This paragraph would be your content paragraph...</p>
  <p>Here's another content article right here.</p>
</div>
```

With these text elements now grouped together under a <div> element, we can alter the appearance of each underlying element collectively by applying a *style* attribute to the <div> tag.

#### HTML Div Element Code:

```
<div id="myDiv" name="myDiv" title="Example Div Element" style="color: #0900C4; font:
Helvetica 12pt;border: 1px solid black;">
  <h5>Subtitle</h5>
  <p>This paragraph would be your content paragraph...</p>
  <p>Here's another content article right here.</p>
</div>
```

#### HTML Div Element in Action:

Subtitle

This paragraph would be your content paragraph...

Here's another content article right here.

Elements housed within a <div> tag acquire any styles or properties applied to the master div element. Therefore the paragraph and heading elements should now be rendered *blue* in a

*Helvetica* font. In addition, we've applied a border to the <div> element just to help visualize the grouping of elements together.

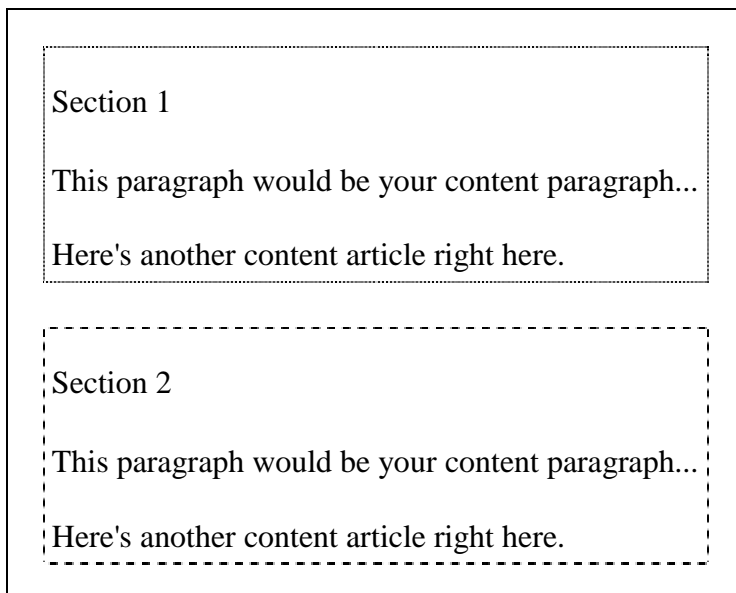
### 13.3.2 Div inside of Div

Placing <div> elements inside of other <div> elements allows these elements to be further subdivided.

#### HTML Div Subdivision:

```
<div id="myDiv" name="myDiv" title="Example Div Element" style="font-family: Helvetica;
font-size: 12pt; border: 1px solid black;">
  <div id="subDiv1" name="subDiv1" title="Subdivision Div Element" style="color: #FF0000;
border: 1px dotted black;">
    <h5>Section 1</h5>
    <p>This paragraph would be your content paragraph...</p>
    <p>Here's another content article right here.</p>
  </div>
  <br />
  <div id="subDiv2" name="subDiv2" title="Subdivision Div Element" style="color:
#FF00FF;border: 1px dashed black;">
    <h5>Section 2</h5>
    <p>This paragraph would be your content paragraph...</p>
    <p>Here's another content article right here.</p>
  </div>
</div>
```

#### Divs Inside of Divs:



This concept is the foundation of which most web pages are now built. HTML documents that are properly divided and subdivided are easy to maintain and modify.

## 13.4 Page Layouts and Templates

HTML layout is very basic. Not many options exist with the body tag alone. Tables, on the other hand, are the bread and butter of HTML layouts. Any element may be placed inside of a table, including tables themselves!

### HTML Code:

```
<table id="shell" bgcolor="black" border="1" height="200" width="300">
  <tr>
    <td>
      <table id="inner" bgcolor="white" height="100" width="100">
        <tr>
          <td>Tables inside tables!</td>
        </tr>
      </table>
    </td>
  </tr>
</table>
```

The white table (identified as inner) exists inside of the (shell) table, the black one. A light bulb should be going off inside of your head as you explore how this system will allow for the creation of limitless layouts.

### 13.4.1 Standard Layout

A fairly standard layout consists of a banner near the top, navigation, and your content or display box. These are the backbone to any great website.

### HTML Code:

```
<table cellspacing="1" cellpadding="0" border="0"
  bgcolor="black" id="shell" height="250" width="400">
  <tr height="50">
    <td colspan="2" bgcolor="white">
      <table title="Banner" id="banner" border="0">
        <tr><td>Place a banner here</td></tr>
      </table>
    </td>
  </tr>
  <tr height="200">
    <td bgcolor="white">
      <table id="navigation" title="Navigation" border="0">
        <tr><td>Links!</td></tr>
        <tr><td>Links!</td></tr>
        <tr><td>Links!</td></tr>
      </table>
    </td><td bgcolor="white">
```

```

        <table title="Content" id="content" border="0">
            <tr><td>Content goes here</td></tr>
        </table>
    </td>
</tr>
</table>

```

### Basic Layout:

Place a banner here	
Links!	Content goes here
Links!	
Links!	

This approach is basic, yet organized. The code becomes complex rather fast, so you will need to be sure to properly assign height and width values to your tables as well. The more specific you are about heights and widths, the less room there will be for error and debugging.

### HTML Code:

```

<table id="shell" title="Shell" height="250" width="400"
border="0" bgcolor="black" cellspacing="1" cellpadding="0">
    <tr height="50">
        <td bgcolor="white">
            <table title="banner" id="banner">
                <tr><td>Banner goes here</td></tr>
            </table>
        </td>
    </tr>
    <tr height="25">
        <td bgcolor="white">
            <table title="Navigation" id="navigation">
                <tr><td>Links!</td>
                <td>Links!</td>
                <td>Links!</td></tr>
            </table>
        </td>
    </tr>
    <tr>
        <td bgcolor="white">
            <table title="Content" id="content">

```

```
<tr>
  <td>Content goes here</td>
</tr>
</table>
</td>
</tr>
</table>
```

### Basic Layout 2:

Banner goes here
Links! Links! Links!
Content goes here

The code is quite a lot to look at, SO break it up and organize it in your own way to make things easier for you.

### Tips

- Your code can become quite complicated rather fast. Keep an organized spacing system so it becomes easy to spot where one table ends and the other begins.
- Use *cellspacing* to add usable borders to your content.
- Keep things neat. Be creative, yet organized.



## CHAPTER 14: HTML FRAMES

### 14.1 Chapter Overview

Frames allow for multiple .html documents to be displayed inside of one browser window at a time. This means that one page has no content on it, but rather tells the browser which web pages you would like to open. With the addition of CSS and PHP, frames have become outdated, but if you wish to use them, read on.

### 14.2 Introduction

#### A Generic Frame Page

Frames are most typically used to have a menu in one frame, and content in another frame. When someone clicks a link on the menu, that link is then opened in the content page. Here is a classic example of a basic "index" frameset with a menu on the left and content on the right.

#### HTML Code:

```
<html>
<body>
  <frameset cols="30%,*">
    <frame src="menu.html">
    <frame src="content.html">
  </frameset>
</body>
</html>
```

#### Frame Set:

Here's the example: [Frame Index](#)

- **frameset** - The parent tag that defines the characteristics of this frames page. Individual frames are defined inside it.
- **frameset cols="#%, \*"** - The width that each frame will have. In the above example, we chose the menu (the 1st column) to be 30% of the total page and used a "\*", which means the content (the 2nd column) will use the remaining width for itself (70%).
- **frame src=""** - The URL of the web page to load into the frame.

A good rule of thumb is to call the page which contains this frame information "index.html", as that is typically a site's main page.

### 14.3 Adding a Banner or Title Frame

Add a row to the top for a title and graphics with the code as follows:

#### HTML Code:

```
<html>
```

```

<body>
  <frameset rows="20%,*">
    <frame src="title.html">
    <frameset cols="30%,*">
      <frame src="menu.html">
      <frame src="content.html">
    </frameset>
  </frameset>
</body>
</html>

```

## 14.4 FrameBorder and FrameSpacing

You've probably noticed those ugly gray lines that appear between the frames. It is possible to remove these and manipulate the spacing between frames with *frameborder* and *framespacing*. These attributes appear within the *frameset* tag.

Note: *Framespacing* and *border* are the same attribute, but some browsers only recognize one or the other, so use both, with the same value, to be safe.

- **frameborder="#"** - Determines whether there will be a border.
- **border="#"** - Modifies the border width.
- **framespacing="#"** - Modifies the border width, used by Internet Explorer.

Here's an example of the same frameset without the borders.

### HTML Code:

```

<html>
<body>
  <frameset border="0" frameborder="0" framespacing="0" rows="20%,*">
    <frame src="title.html">
    <frameset border="0" frameborder="0" framespacing="0" cols="30%,*">
      <frame src="menu.html">
      <frame src="content.html">
    </frameset>
  </frameset>
</body>
</html>

```

### Frame Borders:

Here's a visual: [Visual](#)

## 14.5 Frame Name and Frame Target

How nice would it be to make each menu link load into the content page? We do this by naming each frame and setting the correct *base target* inside "menu.html".

**HTML Code:**

```
<html>
<body>
  <frameset rows="20%,*">
    <frame name="title" src="title.html">
    <frameset cols="30%,*">
      <frame name="menu" src="menu.html">
      <frame name="content" src="content.html">
    </frameset>
  </frameset>
</body>
</html>
```

**HTML Code:**

```
<html>
<head>
  <base target="content">
</head>
<body>
  <!-- Content Goes Here -->
</body>
</html>
```

**Frame Target:**

Here's the Visual: [Visual](#)

We first named the content frame "content" on our frame page, and then we set the base target inside "menu.html" to point to that frame. Our frame page is now a perfectly functional menu and content layout!

## 14.5 Noresize and Scrolling

It's possible to further customize the <frame> tag using the *noresize* and *scrolling* attributes.

**HTML Code:**

```
<html>
<body>
  <frameset border="2" frameborder="1" framespacing="2" rows="20%,*">
    <frame src="title.html" noresize scrolling="no">
    <frameset border="4" frameborder="1" framespacing="4" cols="30%,*">
      <frame src="menu.html" scrolling="auto" noresize>
      <frame src="content.html" scrolling="yes" noresize>
    </frameset>
  </frameset>
</body>
</html>
```

### **Noresize and Scrolling:**

Here's the Visual: [Visual](#)

- **noresize** - Determines whether the frames can be resized by the visitor or not. (values "true" and "false")
- **scrolling** - Determines whether scrolling is allowed in the frame or not (values "true" and "false")

We set the scrolling for our content frame to "yes" to ensure our visitors will be able to scroll if the content goes off the screen. We also set the scrolling for our title banner to no, because it does not make sense to have a scrollbar appear in the title frame.

### **Tips**

- Frames can be simple and well-organized. However, they are usually viewed as unacceptable by most web designers.
- Always set the scrolling and resize options to optimize load time.
- Using a simple menu and content frame design can reduce updates to massive sites. Instead of updating the menu on each page, you could simply update the menu.html file and be done with it!

## CHAPTER 15: SCRIPTING LANGUAGES

### 15.1 Introduction

A client-side *script* is a program that may accompany an HTML document or be embedded directly in it. The program executes on the client's machine when the document loads, or at some other time such as when a link is activated. HTML's support for scripts is independent of the scripting language.

Scripts offer authors a means to extend HTML documents in highly active and interactive ways. For example:

- Scripts may be evaluated as a document loads to modify the contents of the document dynamically.
- Scripts may accompany a form to process input as it is entered. Designers may dynamically fill out parts of a form based on the values of other fields. They may also ensure that input data conforms to predetermined ranges of values, that fields are mutually consistent, etc.
- Scripts may be triggered by events that affect the document, such as loading, unloading, element focus, mouse movement, etc.
- Scripts may be linked to form controls (e.g., buttons) to produce graphical user interface elements.

There are two types of scripts authors may attach to an HTML document:

- Those that are executed one time when the document is loaded by the user agent. Scripts that appear within a `SCRIPT` element are executed when the document is loaded. For user agents that cannot or will not handle scripts, authors may include alternate content via the `NOSCRIPT` element.
- Those that are executed every time a specific event occurs. These scripts may be assigned to a number of elements via the intrinsic event attributes.

The most popular scripting languages that are commonly used in HTML to make web pages come alive are JavaScript, PHP, Perl, ASP and VBScripts.

With HTML scripts, you can create dynamic web pages, make image rollovers for really cool menu effects, or even validate the data on your HTML forms before users submit their information. However, JavaScript and VBScript are very complicated compared to HTML. It may be simpler just to download someone else's scripting code and modify it for use on your web page (if they have given you permission to do so, of course!).

### 15.2 Javascript Code

If you want to insert JavaScript code into your HTML, you are going to use the `<script>` tag. Below is the correct code to insert embedded JavaScript code onto your site.

**HTML Code:**

```
<script type="text/javascript">
<!--script
***Some JavaScript code should go here***
-->
</script>
```

For JavaScript, you set the *type* attribute equal to "text/javascript", which is similar to the process of specifying CSS. We can also include a comment around the JavaScript code. This will prevent browsers that do not support JavaScript or have had JavaScript disabled from displaying the JavaScript code in the web browser.

### 15.3 VBScript

To insert VBScript code onto your website, you must once again make use of the <script> tag. Below is the correct code to insert VBScript code onto your site.

**HTML Code:**

```
<script type="text/vbscript">
<!--script
***The VBScript code should go in this spot***
-->
</script>
```

For VBScript, you set the *type* attribute equal to "text/vbscript", which is similar to specifying CSS. We also include a comment around the VBScript code. This will prevent browsers that do not support VBScript or have had VBScript disabled from displaying the VBScript code in the web browser.

**HTML - <!-- Commenting Scripts -->**

Scripting languages such as JavaScript and VBScript must be commented out as well. You will learn that it is only once they are placed within the <script> tags that the browser executes the scripts without causing errors.

**HTML Code:**

```
<script>
<!--
document.write("Hello World!")
//-->
</script>
```

With this example, we are jumping far ahead. Just be sure you understand when to use comments and where to look for them. They are a very useful tool for any large HTML project.