Mt Kenya University

P.O. Box 342-01000 Thika

Email: info@mku.ac.ke

Web: www.mku.ac.ke

DEPARTMENT OF INFORMATION TECHNOLOGY

COURSE CODE: BIT 3105

COURSE TITLE: ITERNET PROGRAMMING

Instructional manual for BBIT – Distance Learning

TABLE OF CONTENT

COURSE OUTLINE

BIT 3105: INTERNET PROGRAMMING

Pre-requisite: BIT 1202 - FUNDAMENTALS OF INTERNET

Purpose: To understand the fundamentals of programming for the internet in the business world.

Objectives: By the end of the course unit the learner will gain knowledge and skills in: -

- Designing and implementing static web content
- Dynamic web content in web servers -
- Defining the web control using various types of business information

Course Assessments: Continuous Assessment Tests 30%

End of semester examination 70%

Reference Books:

Evangelos P. Magturing (2002) Visual basic 6 BPB Duplications

Mark S et al (1996), Special Edition using internet HTML Que Publishing Co ltd

Alex H et all(2000),, Professional active server Wrox publishers programmer to programmer series

BIT 3105: INTERNET PROGRAMMING - TOPICS – Details

Week 1: Introduction

- Services provided by the internet
- Important components of the web
- Common internet protocols
- Internet architecture

Week 2: Web servers

- Introduction to web Servers
- Common features of a web server
- Web Server Types
- Web Server Error Messages
- Web server Overload
- Web Server software

Week 3: Static Content: Advanced HTML

- Introduction to static webpage
- HTML Forms
- HTML Doctypes

- HTML head Elements
- HTML Entities
- HTML Uniform Resource Locators

Week 4: Static Content: Introduction to XML

- Introduction to XML
- XML Tree
- XML Syntax
- XML Elements
- XML Attributes
- XML Validation
- XML Validator
- XML CSS

Week 5: Dynamic Content: Introduction to DHTML

- Introduction to Dynamic web page
- Introduction to DHTML
- DHTML JavaScript
- DHTML HTML Document Object Model
- DHTML Events
- DHTML CSS

Week 6-8: Dynamic Content: Introduction to PHP 8th Aug

- Introduction to PHP
- PHP Syntax
- PHP Variables
- PHP Operators
- Conditional Statements
- PHP Loops
- PHP Functions
- PHP Forms and User Input
- PHP $_GET Function
- PHP $_POST Function
- The PHP $_REQUEST Function

Week 9-10: PHP Database

- MySQL Introduction
- MySQL Connect
- MySQL Create
- MySQL Insert
- MySQL Select
- MySQL Where
- MySQL Order By

- MySQL Update
- MySQL Delete

Week 11: Advanced PHP

- PHP Include
- PHP File
- PHP File Upload
- PHP Cookies
- PHP Sessions
- PHP E-mail

Week 12: Multimedia websites

- Multimedia formats - Sound Formats, Video Formats, and Windows Formats
- Playing Sounds and Playing Videos
- Multimedia object element

Module Compiler: David Kibaara

CHAPTER ONE: INTRODUCTION TO INTERNET PROGRAMMING BASIC CONCEPTS

Learning Objectives:

By the end of this chapter the learner shall be able to;

i. Explain the services provided by the internet

ii. Explain the different internet protocols

iii. Explain the advantages and disadvantages of client/server architecture.

1.1 Services Provided by the Internet

Electronic Mail - E-mail, also known as electronic mail, is one of the most popular Internet services. E-mail allows you to send messages to one person, or to send a message simultaneously to a group of people. One of the greatest advantages of e-mail over other forms of communication is the convenience to the recipient. Messages wait in your mailbox until you open it. Another advantage of an Internet e-mail account is that you can check your e-mail from any location with an internet connection.

FTP (File Transfer Protocol)-This facility is a method of gaining limited access to another machine in the Internet, and obtaining files from it. You need full Internet connectivity, to do ftp interactively. FTP has many advantages, for example, it allows you to get new free software, or updated versions of old programs, as well as useful data for your research. The most common way of using FTP is via **anonymous FTP**. When you start an ftp connection, you will be asked for a user name and a password.

Telnet: logging in to Remote Network Computers - Telnet is the Internet facility that allows you to execute commands on a remote host (another computer, most likely one to which you do not have physical access) as if you were logged in locally. You need to know the name of the machine to which you want to connect, and to have a valid user name in it. There is no such thing as "anonymous" telnet.

Usenet Newsgroups Usenet newsgroups, also called bulletin boards, are a similar e-mail conferencing system, but are less intrusive to the subscriber than list serves since messages are posted to Usenet sites around the world instead of appearing in each subscriber's mailbox. Usenet refers to the huge collection of messages which are posted to tens of thousands of newsgroups worldwide. Millions of people around the world regularly read newsgroup messages, following their favourite topics of interest. New newsgroups are added and old ones deleted every day. Usenet can provide a unique information resource not readily accessible from any other source. If you are looking for personal anecdotes about products, especially computer-related hardware and software products, how-to information, practical advice, or the latest news stories, newsgroup archives may be a valuable resource.

Internet Chat

Communication on the Internet goes even further than personal e-mail, newsgroups and mailing lists, to encompass real-time conversations (synchronous communication) among two or more people. Chat is available on the Internet through Internet Relay Chat or IRC. It consists of thousands of chat channels, each covering a different topic and with participants from all over the world.

Web Conferencing Many institutions are discovering new ways to integrate Internet communications into their organizations. One of the most popular ways is through the use of web or online conferencing.

Web conferencing is currently being used by businesses for employee training, meetings and general communication. Educational institutions are using web conferencing as a way to enhance on-site classes or distance education classes. Web conferencing is a tool which provides a way for "students" to share information, ask questions, get answers, discuss problems and work collaboratively. Conferencing provides opportunities to solve issues by providing a dynamic exchange of text, graphics, HTML links to information, audio, and video in a structured conversation organized by topic. Web conferences may take place in "real-time" where all participants are communicating at the same pre-arranged time.

1.2 Requirements for connecting to the internet

Internet service provider – an internet service provider provides you with a connection to the internet and the software you will need to navigate. An Internet service provider (ISP) is a

company that provides access to the Internet. Access ISPs directly connect customers to the Internet using copper wires, wireless or fiber-optic connections.

Telecommunication line (Mobile or Wired) – a telephone line is required to connect you to the internet service provider.

Modem – a modem converts a digital signal received from a computer into an analogue signal that can be sent along ordinary telephone lines, and back to digital at the other end.

Web browser – a web browser is software used to view and download Web pages and various types of files such as text, graphics and video. Examples are Microsoft Internet Explorer or Netscape Navigator.

1.3 Important Components of the Web

Web Server

Web server can refer to either the hardware (the computer) or the software (the computer application) that helps to deliver content that can be accessed through the Internet. The most common use of Web servers is to host Web sites but there are other uses like data storage or for running enterprise applications. A web server is a computer programs that delivers (serves) content, such as web pages, using the Hypertext Transfer Protocol (HTTP), over the World Wide Web.

A **web search engine** is designed to search for information on the World Wide Web and FTP servers. The search results are generally presented in a list of results and are often called *hits*. The information may consist of web pages, images, information and other types of files. Some search engines also mine data available in databases or open directories. Unlike Web directories, which are maintained by human editors, search engines operate algorithmically or are a mixture of algorithmic and human input. Examples of search engines are yahoo, google, msn etc

Web crawler is an automated Web browser which follows every link it sees. The contents of each page are then analyzed to determine how it should be indexed (for example, words are extracted from the titles, headings, or special fields called meta tags). Data about web pages are stored in an index database for use in later queries. Some search engines, store all or part of the source page (referred to as a cache) as well as information about the web pages, whereas others, store every word of every page they find. When a user enters a query into a

search engine the engine examines its **index** and provides a listing of best-matching web pages according to its criteria, usually with a short summary containing the document's title and sometimes parts of the text. Exclusions from web crawler can be made by the use of robots.txt.

Domain Name System (**DNS**) is a hierarchical naming system for computers, services, or any resource connected to the Internet or a private network. It associates information with domain names assigned to each of the participants. Most importantly, it translates domain names meaningful to humans into the numerical (binary) identifiers associated with networking equipment for the purpose of locating and addressing these devices worldwide.
The Domain Name System makes it possible to assign domain names to groups of Internet users in a meaningful way, independent of each user's physical location. Internet domain names are easier to remember than IP addresses such as 208.77.188.166 (IPv4) or 2001:db8::1f70:6e8 (IPv6). The Domain Name System distributes the responsibility of assigning domain names and mapping those names to IP addresses by designating authoritative name servers for each domain. Authoritative name servers are assigned to be responsible for their particular domains, and in turn can assign other authoritative name servers for their sub-domains. The DNS database of Domain names and the corresponding IP addresses can not be held on one machine. As a truly distributed resource, it is maintained by many organisations, each manages a little bit of it. DNS defines a tree structure, and each node on the tree is owned by one of the naming authorities. The owner of a node can create any number of child nodes, but each must have a unique name.

URLs Addresses for web sites are called URLs (Uniform Resource Locators). Most of them begin with http (HyperText Transfer Protocol), followed by a colon and two slashes. For example, the URL for the Florida Centre for Instructional Technology is http://fcit.usf.edu/ . 20
Some of the URL addresses include a directory path and a file name. Consequently, the addresses can become quite long. For example, the URL of a web page may be: http://fcit.usf.edu/holocaust/default.htm. In this example, "default.htm" is the name of the file which is in a directory named "holocaust" on the FCIT server at the University of South Florida.

Top-level domain Each part of a domain name contains certain information. The first field is the host name, identifying a single computer or organization. The last field is the top-level domain, describing the type of organization and occasionally country of origin associated with the address.

Top-level domain names include:

.com	Commercial
.edu	Educational
.gov	US Government
.int	Organization
.mil	US Military
.net	Networking Providers
.org	Non-profit Organization

Domain name country codes include, but are not limited to:

.au	Australia
.de	Germany
.fr	France
.nl	Netherlands
.uk	United Kingdom
.us	United States
.ke	Kenya

Routers

All of these networks rely on Network Access Point (NAPs), backbones and **routers** to talk to each other. What is incredible about this process is that a message can leave one computer and travel halfway across the world through several different networks and arrive at another computer in a fraction of a second! The routers determine where to send information from one computer to another. Routers are specialized computers that send your messages and those of every other Internet user speeding to their destinations along thousands of pathways. A router has two separate, but related, jobs:

- It ensures that information doesn't go where it's not needed. This is crucial for keeping large volumes of data from clogging the connections of "innocent bystanders."

- It makes sure that information does make it to the intended destination.

In performing these two jobs, a router is extremely useful in dealing with two separate computer networks. It joins the two networks, passing information from one to the other. It also protects the networks from one another, preventing the traffic on one from unnecessarily spilling over to the other. Regardless of how many networks are attached, the basic operation and function of the router remains the same. Since the Internet is one huge network made up of tens of thousands of smaller networks, its use of routers is an absolute necessity

1.4 Common Internet Protocols

The internet protocols are derived from the **TCP/IP** (Transmission Control Protocol/ Internet Protocol) suite of protocols which is the set of protocols used to communicate across the internet. It is also widely used on many organizational networks due to its flexibility and wide array of functionality provided. Microsoft who had originally developed their own set of protocols now is more widely using TCP/IP, at first for transport and now to support other services. For the purpose of this unit we will discuss the most common and important protocols.

ARP - Address Resolution Protocol enables the packaging of IP data into ethernet packages. It is the system and messaging protocol that is used to find the ethernet (hardware) address from a specific IP number. Without this protocol, the ethernet package could not be generated from the IP package, because the ethernet address could not be determined.

RARP - Reverse address resolution protocol is used to allow a computer without a local permanent data storage media to determine its IP address from its ethernet address.

IP- Internet Protocol, Except for ARP and RARP all protocols' data packets will be packaged into an IP data packet. IP provides the mechanism to use software to address and manage data packets being sent to computers.

TCP - A reliable connection oriented protocol used to control the management of application level services between computers. It is used for transport by some applications.

UDP - An unreliable connection less protocol used to control the management of application level services between computers. It is used for transport by some applications which must provide their own reliability.

FTP - File Transfer Protocol allows file transfer between two computers with login required.

HTTP - Hypertext Transfer Protocol is used to transport HTML pages from web servers to web browsers. The protocol used to communicate between web servers and web browser software clients.

BOOTP - the Bootstrap Protocol, or BOOTP, is a network protocol used by a network client to obtain an IP address from a configuration server. BOOTP is usually used during the bootstrap process when a computer is starting up. A BOOTP configuration server assigns an IP address to each client from a pool of addresses.

DHCP - Dynamic host configuration protocol is a method of assigning and controlling the IP addresses of computers on a given network. It is a server based service that automatically assigns IP numbers when a computer boots. This way the IP address of a computer does not need to be assigned manually. This makes changing networks easier to manage. DHCP can perform all the functions of BOOTP.

RIP - Routing Information Protocol is used to dynamically update router tables on WANs or the internet. A distance-vector algorithm is used to calculate the best route for a packet. RFC 1058, 1388 (RIP2).

OSPF - Open Shortest Path First dynamic routing protocol. A link state protocol rather than a distance vector protocol. It tests the status of its link to each of its neighbours and sends the acquired information to them.

POP3 - Post Office Protocol version 3 is used by clients to access an internet mail server to get mail. It is not a transport layer protocol.

IMAP4 - Internet Mail Access Protocol version 4 is the replacement for POP3.

Telnet is used to remotely open a session on another computer. It relies on TCP for transport and is defined by RFC854.

1.5 Internet Architecture

Internet is by definition a meta-network, a constantly changing collection of thousands of individual networks intercommunicating with a common protocol.

The Internet's architecture is described in its name, a short form of the compound word "inter-networking". This architecture is based in the very specification of the standard TCP/IP protocol, designed to connect any two networks which may be very different in internal hardware, software, and technical design. Once two networks are interconnected,

communication with TCP/IP is enabled end-to-end, so that any node on the Internet has the near magical ability to communicate with any other no matter where they are. This openness of design has enabled the Internet to grow to a global scale.

The companies running the Internet backbone operate very high bandwidth networks relied on by governments, corporations, large organizations, and other Internet service providers. Their technical infrastructure often includes global connections through underwater cables and satellite links to enable communication between countries and continents.

Each communication packet goes up the hierarchy of Internet networks as far as necessary to get to its destination network where local Routing takes over to deliver it to the addressee. In the same way, each level in the hierarchy pays the next level for the bandwidth they use, and then the large backbone companies settle up with each other. Bandwidth is priced by large Internet service providers by several methods, such as at a fixed rate for constant availability of a certain number of megabits per second, or by a variety of use methods that amount to a cost per gigabyte. Due to economies of scale and efficiencies in management, bandwidth cost drops dramatically at the higher levels of the architecture.

The internet is based on client/server architecture where sever and the client resides in different machines. The **client–server model** (see Fig 1) is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients. Often clients and servers communicate over a computer network on separate hardware, but both client and server may reside in the same system. A server machine is a host that is running one or more server programs which share their resources with clients. A client does not share any of its resources, but requests a server's content or service function. Clients therefore initiate communication sessions with servers which await incoming requests.

Advantages client/server architecture

- In most cases, client/server architecture enables the roles and responsibilities of a computing system to be distributed among several independent computers that are known to each other only through a network. This creates an additional advantage to this architecture: greater ease of maintenance. For example, it is possible to replace, repair, upgrade, or even relocate a server while its clients remain both unaware and unaffected by that change.

- All data is stored on the servers, which generally have far greater security controls than most clients. Servers can better control access and resources, to guarantee that only those clients with the appropriate permissions may access and change data.
- Since data storage is centralized, updates to that data are far easier to administer in comparison to a P2P paradigm. In the latter, data updates may need to be distributed and applied to each peer in the network, which is time-consuming as there can be thousands or even millions of peers.
- Many mature client/server technologies are already available which were designed to ensure security, friendliness of the user interface, and ease of use.
- It functions with multiple different clients of different capabilities.

Disadvantages client/server architecture

- As the number of simultaneous client requests to a given server increases, the server can become overloaded. Contrast that to a P2P network, where its aggregated bandwidth actually increases as nodes are added, since the P2P network's overall bandwidth can be roughly computed as the sum of the bandwidths of every node in that network.
- The client/server paradigm lacks the robustness of a good P2P network. Under client/server, should a critical server fail, clients' requests cannot be fulfilled. In P2P networks, resources are usually distributed among many nodes. Even if one or more nodes depart and abandon a downloading file, for example, the remaining nodes should still have the data needed to complete the download.

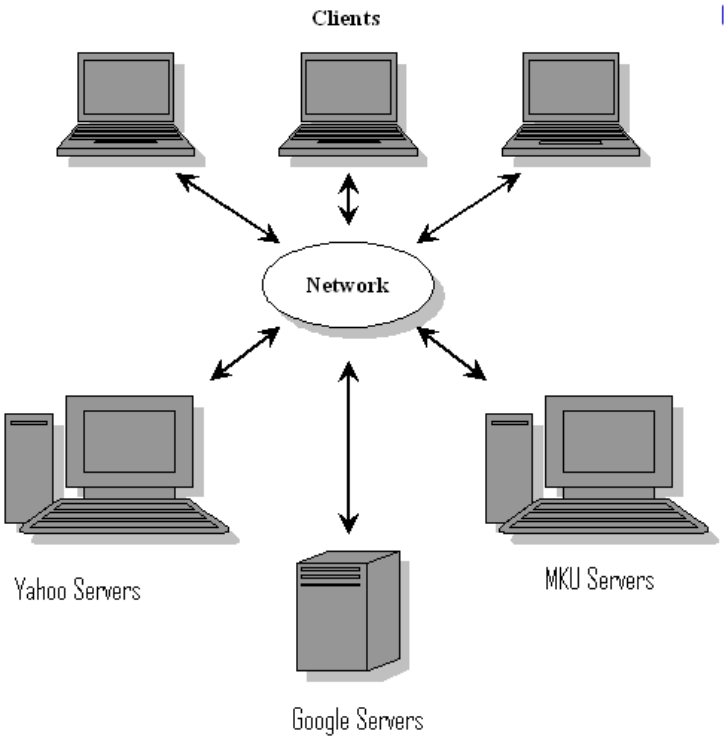


Fig 1.1: Client/Server Architecture

Chapter Review Questions

1. Explain in brief
    a. Web Conferencing
    b. Web crawler
    c. Top-level domain
    d. URLs
    e. Routers
    f. HTTP
    g. FTP
2. Explain in brief the concept of e-mail.
3. What are the basic objectives of FTP?
4. Outline the advantages of Client/Server architecture
5. Outline the disadvantages of Client/Server architecture

Suggested Further Reading

1. Mark S et al (1996), Special Edition using internet HTML Que Publishing Co ltd

2. Alex H et all(2000),, Professional active server Wrox publishers programmer to programmer series
3. http://www.w3schools.com

CHAPTER TWO: WEB SERVERS

Learning Objectives:

By the end of this chapter the learner shall be able to;

i. Explain web serves
ii. Explain the types of web servers
iii. solve the web server overloading problem and server errors
iv. Explain the web server products in the market

2.1 Introduction to web Servers

Web server is an application (software) or computer (hardware) that responds to HTTP requests by returning 'web' resources (e.g., HTML files, images, etc) over the Internet. It is a piece of software that accepts connections from clients and sends them documents. The most common use of Web servers is to host Web sites but there are other uses like data storage or for running enterprise applications.

The primary function of a web server is to deliver web pages on the request to clients (See fig 2.1). This means delivery of HTML documents and any additional content that may be included by a document, such as images, style sheets and JavaScripts. A client, commonly a web browser or web crawler, initiates communication by making a request for a specific resource using HTTP and the server responds with the content of that resource or an error message if unable to do so. The resource is typically a real file on the server's secondary memory. While the primary function is to serve content, a full implementation of HTTP also includes ways of receiving content from clients. This feature is used for submitting web forms, including uploading of files.

Fig 2.1. Communication between the web server and the client (browser)

Many generic web servers also support server-side scripting. This means that the behaviour of the web server can be scripted in separate files, while the actual server software remains unchanged. Usually, this function is used to create HTML documents "on-the-fly" as opposed to returning fixed documents. This is referred to as dynamic and static content respectively. The former is primarily used for retrieving and/or modifying information from databases. The latter is, however, typically much faster and more easily cached.

2.2 Common features of a web server

The following are the common features of a web server: -

- *Virtual hosting* to serve many Web sites using one IP address. Virtual hosting is a method for hosting multiple domain names on a computer using a single IP address. This allows one machine to share its resources, such as memory and processor cycles, to use its resources more efficiently.
- *Large file support* to be able to serve files whose size is greater than 2 GB on 32 bit OS.
- *Bandwidth throttling* to limit the speed of responses in order to avoid saturating the network and to be able to serve more clients. This is done by controlling the number of requests a server responds to within a specified period of time
- *Server-side scripting* to generate dynamic Web pages, still keeping Web server and Web site implementations separate from each other. Server-side scripting is a web server technology in which a user's request is verified by running a script directly on the web server to generate dynamic web pages. It is usually used to provide interactive web sites that interface to databases or other data stores.

2.3 Web server implementations

The following are the main two ways web server software can be implemented: -

1. *In the OS kernel* - An in-kernel Web server (Microsoft IIS on Windows) will usually work faster, because, as part of the system, it can directly use all the hardware resources it needs, such as non-paged memory, CPU time-slices, network adapters, or buffers.
2. User space (like other application) implementation - Web servers that run in user-mode have to ask the system the permission to use more memory or more CPU resources. Not only do these requests to the kernel take time, but they are not always satisfied because the system reserves resources for its own usage and has the responsibility to share hardware resources with all the other running applications. Also, applications cannot access the system's internal

buffers, which cause useless buffer copies that create another handicap for user-mode web servers.

2.4 Server Types

The following is the list of web server types: -

- Proxy Servers - Proxy servers sit between a client program (typically a Web browser) and an external server (typically another server on the Web) to filter requests, improve performance, and share connections.
- Mail Servers - mail servers move and store mail over corporate networks (via LANs and WANs) and across the Internet.
- Application Servers - application servers occupy a large chunk of computing territory between database servers and the end user, and they often connect the two. An application server is a software framework that provides an environment where applications can run, no matter what the applications are or what they do. It is dedicated to the efficient execution of procedures (programs, routines, scripts) for supporting the construction of applications.
- Real-Time Communication Servers - Real-time communication servers, formerly known as chat servers or IRC (Internet Relay Chat) Servers, and still sometimes referred to as instant messaging (IM) servers, enable large numbers users to exchange information instantaneously.
- FTP Servers - File Transfer Protocol makes it possible to move one or more files securely between computers while providing file security and organization as well as transfer control.
- Collaboration Servers – this is the server that hosts collaboration software. Collaboration software designed to enable users to collaborate, regardless of location, via the Internet or a corporate intranet and to work together in a virtual atmosphere.
- List Servers - List servers offer a way to better manage mailing lists, whether they are interactive discussions open to the public or one-way lists that deliver announcements, newsletters or advertising.
- Telnet Servers - A Telnet server enables users to log on to a host computer and perform tasks as if they're working on the remote computer itself.
- Virtual Servers, Virtual Web servers - are a very popular way of providing low-cost web hosting services. Instead of requiring a separate computer for each server, dozens of virtual servers can co-reside on the same computer. In most cases, performance is not affected and

each web site behaves as if it is being served by a dedicated server. However, if too many virtual servers reside on the same computer, or if one virtual server starts hogging resources, Web pages will be delivered more slowly. In 2009, the number of virtual servers deployed exceeded the number of physical servers. Today, server virtualization has become near ubiquitous in the data centre. Virtual hosting is a method for hosting multiple domain names on a computer using a single IP address. This allows one machine to share its resources, such as memory and processor cycles, to use its resources more efficiently.

- Secure server - A Web server that supports any of the major security protocols, like SSL(Secure Sockets Layer), that encrypt and decrypt messages to protect them against third party tampering. Making purchases from a secure Web server ensures that a user's payment or personal information can be translated into a secret code that's difficult to crack. Major security protocols include SSL, SHTTP(Secure Hypertext Transfer Protocol), and IPSec(Internet Protocol Security).
- PWS - Short for personal Web server, a generic term for a Web server that is used to host Web page files for an individual. Web server program for individuals hosting Web page files from a personal computer. Personal Web Server is a smaller-scale version of Microsoft's IIS technology and is therefore limited in its capabilities. It is designed to support Web sites that obtain limited traffic and/or to be used as a staging server for building pages that will be transferred to a server that can handle large amounts of traffic.
- Commerce server - Web software that runs some of the main functions of an online storefront such as product display, online ordering, and inventory management. The software works in conjunction with online payment systems to process payments.

2.5 Some of the Web Server Error Messages

List of HTTP Response Codes:

- 500 Internal Server Error - A generic error message, given when no more specific message is suitable.
- 501 Not Implemented - The server either does not recognise the request method, or it lacks the ability to fulfill the request.
- 502 Bad Gateway - The server was acting as a gateway or proxy and received an invalid response from the upstream server.

- 503 Service Unavailable - The server is currently unavailable (because it is overloaded or down for maintenance).Generally, this is a temporary state.
- 504 Gateway Timeout - The server was acting as a gateway or proxy and did not receive a timely response from the upstream server.
- 505 HTTP Version Not Supported - The server does not support the HTTP protocol version used in the request.
- 506 Variant Also Negotiates (RFC 2295) - Transparent content negotiation for the request results in a circular reference.
- 507 Insufficient Storage
- 509 Bandwidth Limit Exceeded - This status code, while used by many servers, is not specified in any RFCs.
- 510 Not Extended - Further extensions to the request are required for the server to fulfill it.
- 400 Bad File Request - Usually means the syntax used in the URL is incorrect (e.g., uppercase letter should be lowercase letter; wrong punctuation marks).
- 401 Unauthorized - Server is looking for some encryption key from the client and is not getting it. Also, wrong password may have been entered. Try it again, paying close attention to case sensitivity.
- 403 Forbidden/Access Denied - Similar to 401; special permission needed to access the site, a password and/or username if it is a registration issue. Other times you may not have the proper permissions set up on the server or the site's administrator just doesn't want you to be able to access the site.
- 404 File Not Found - Server cannot find the file you requested. File has either been moved or deleted, or you entered the wrong URL or document name. Look at the URL. If a word looks misspelled, then correct it and try it again.
- 408 Request Timeout - Client stopped the request before the server finished retrieving it. A user will either, hit the stop button, close the browser, or click on a link before the page loads. Usually occurs when servers are slow or file sizes are large.
- Connection Refused by Host - Either you do not have permission to access the site or your password is incorrect.

- File Contains No Data - Page is there but is not showing anything. Error occurs in the document. Attributed to bad table formatting, or stripped header information.
- Bad File Request - Browser may not support the form or other coding you are trying to access.
- Failed DNS Lookup - The Domain Name Server can't translate your domain request into a valid Internet address. Server may be busy or down, or incorrect URL was entered.
- Host Unavailable - Host server down. Hit reload or go to the site later.
- Unable to Locate Host - Host server is down, Internet connection is lost, or URL typed incorrectly.
- Refused by the Server - The Web server is busy.

2.6 Web server Overload

Web Server Overload causes

- Too much legitimate web traffic. Thousands or even millions of clients connecting to the web site in a short interval
- Distributed Denial of Service attacks - is an attempt to make a computer resource unavailable to its intended users. It generally consists of the concerted efforts of a person or people to prevent an Internet site or service from functioning efficiently or at all, temporarily or indefinitely. Perpetrators of DoS attacks typically target sites or services hosted on high-profile web servers such as banks, credit card payment gateways, and even root name servers.
- Computer worms that sometimes cause abnormal traffic because of millions of infected computers (not coordinated among them)
- viruses can cause high traffic because of millions of infected browsers and/or Web servers
- Internet bots. Traffic not filtered/limited on large web sites with very few resources. Internet bots are software applications that run automated tasks over the Internet. Typically, bots perform tasks that are both simple and structurally repetitive, at a much higher rate than would be possible for a human alone. The largest use of bots is in web spidering, in which an automated script fetches, analyzes and files information from web servers at many times the speed of a human.

- Internet (network) slowdowns, so that client requests are served more slowly and the number of connections increases so much that server limits are reached
- Web servers (computers) partial unavailability. This can happen because of required or urgent maintenance or upgrade, hardware or software failures, back-end (e.g., database) failures, etc.; in these cases the remaining web servers get too much traffic and become overloaded.

Web server Overload symptoms

- Requests are served with (possibly long) delays (from 1 second to a few hundred seconds)
- 500, 502, 503, 504 HTTP errors are returned to clients (sometimes also unrelated 404 error or even 408 error may be returned)
- TCP connections are refused or reset (interrupted) before any content is sent to clients
- In very rare cases, only partial contents are sent (but this behaviour may well be considered a bug, even if it usually depends on unavailable system resources).

Web server Anti-overload techniques

To partially overcome above load limits and to prevent overload, most popular Web sites use common techniques like:

- managing network traffic, by using:
    - Firewalls to block unwanted traffic coming from bad IP sources or having bad patterns
    - HTTP traffic managers to drop, redirect or rewrite requests having bad HTTP patterns
    - Bandwidth management and traffic shaping, in order to smooth down peaks in network usage;
- Deploying Web cache techniques. A web cache is a mechanism for the temporary storage (caching) of web documents, such as HTML pages and images, to reduce bandwidth usage, server load, and perceived lag. A web cache stores copies of documents passing through it; subsequent requests may be satisfied from the cache if certain conditions are met.
- using different domain names to serve different (static and dynamic) content by separate Web servers, i.e.:
    - http://images.example.com

- http://www.example.com

- using different domain names and/or computers to separate big files from small and medium sized files; the idea is to be able to fully cache small and medium sized files and to efficiently serve big or huge (over 10 - 1000 MB) files by using different settings
- using many Web servers (programs) per computer, each one bound to its own network card and IP address
- using many Web servers (computers) that are grouped together so that they act or are seen as one big Web server
- adding more hardware resources (i.e. RAM, disks) to each computer
- Using more efficient computer programs for Web servers, etc.

2.7 Basic Server software features

The following are the features that you will find in most of the web server software's. Some web server software's will have additional features.

- Handling of static files, index files, and auto-indexing
- Handling dynamic Content
- Load balancing
- Fault tolerance
- Security support e.g. SSL or HTTPs
- Name- and IP-based virtual servers
- FLV streaming and MP4 streaming
- Web page access authentication
- Compression of files e.g. gzip
- Ability to handle many simultaneous connections
- Administration console

2.8 Web Server software on the internet

Table 2.1: The market share for the top 5 web server software according to Netcraft survey in March 2011

Vendor	Product	Web Sites Hosted	Percent
Apache	Apache	179,720,332	60.31%
Microsoft	IIS	57,644,692	19.34%
Igor Sysoev	nginx	22,806,060	7.65%
Google	GWS	15,161,530	5.09%
lighttpd	lighttpd	1,796,471	0.60%

The **Apache HTTP Server**, is web server software notable for playing a key role in the initial growth of the World Wide Web. In 2009 it became the first web server software to surpass the 100 million website milestone. Apache was the first viable alternative to the Netscape Communications Corporation web server (currently known as Oracle iPlanet Web Server), and has since evolved to rival other web servers in terms of functionality and performance. Typically Apache is run on a Unix-like operating system.

Apache is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation. The application is available for a wide variety of operating systems, including Unix, Mac OS X and Microsoft Windows.

Internet Information Services (IIS) – formerly called Internet Information Server – is a web server application and set of feature extension modules created by Microsoft for use with Microsoft Windows. It is the most used web server after Apache HTTP Server. IIS 7.5 supports HTTP, HTTPS, FTP, FTPS and SMTP.

nginx is a lightweight, high-performance Web server/reverse proxy and e-mail (IMAP/POP3) proxy, licensed under a BSD (Berkeley Software Distribution)-like license. It runs on Unix, Linux, BSD variants, Mac OS X, Solaris, and Microsoft Windows. Nginx quickly delivers static content with efficient use of system resources. It can deploy dynamic HTTP content on a network using FastCGI handlers for scripts, and can serve as a very capable software load balancer.

2.9 Definition of other web server related terms

- **Common Gateway Interface (CGI)** is a standard that defines how web server software can delegate the generation of web pages to a stand-alone application or an executable file. Such applications, known as CGI scripts, can be written in any programming language, although scripting languages are often used.

- **Virtual Web site** - A Web site hosted on a server that shares resources with other Web sites, as opposed to a single machine dedicated to processing HTTP requests for a single Web site. Web sites on the same server will share common resources. Also called shared Web hosting.

- **Hosting server** - A server dedicated to hosting a service or services for users. Hosting servers are most often used for hosting Web sites but can also be used for hosting files, images, games and similar content. Hosting servers can be shared among many clients (shared hosting servers) or dedicated to a single client (dedicated servers), the latter of which is particularly common for larger Web sites where the hosting needs of the Web site owner necessitate more control and/or bandwidth.

- **Web host** - A Web host is in the business of providing server space, Web services and file maintenance for Web sites controlled by individuals or companies that do not have their own Web servers. Many ISPs, such as America Online, will allow subscribers a small amount of server space to host a personal Web page. Other commercial ISPs will charge the user a fee depending on the complexity of the site being hosted.

- **Web stack** - The term used to refer to software stacks in Web development environments. The stack of software, mainly comprised of open source software, will contain an operating system, Web server, database server, and programming language. e.g. WAMP server and LAMP software bundle.

- **Traffic shaping** (also known as "packet shaping" or ITMPs: Internet Traffic Management Practices) is the control of computer network traffic in order to optimize or guarantee performance, improve latency, and/or increase usable bandwidth for some kinds of packets by delaying other kinds of packets that meet certain criteria.

- The **Slashdot effect**, also known as slashdotting, occurs when a popular website links to a smaller site, causing a massive increase in traffic. This overloads the smaller site, causing it to slow down or even temporarily close. The name stems from the huge influx of web traffic those results from the technology news site Slashdot linking to websites. The effect has been associated with other websites or metablogs such as Drudge Report, Reddit, Twitter and

Digg, leading to terms such as being Drudged and the Reddit effect. Typically, less robust sites are unable to cope with the huge increase in traffic and become unavailable – common causes are lack of sufficient data bandwidth, servers that fail to cope with the high number of requests, and traffic quotas. Sites that are maintained on shared hosting services often fail when confronted with the Slashdot effect.

- **Load balancing** is a computer networking methodology to distribute workload across multiple computers or a computer cluster, network links, central processing units, disk drives, or other resources, to achieve optimal resource utilization, maximize throughput, minimize response time, and avoid overload. Using multiple components with load balancing, instead of a single component, may increase reliability through redundancy. The load balancing service is usually provided by dedicated software or hardware, such as a multilayer switch or a Domain Name System server.

- **Bandwidth management** is the process of measuring and controlling the communications (traffic, packets) on a network link, to avoid filling the link to capacity or overfilling the link, which would result in network congestion and poor performance of the network.

Chapter Review Questions

1. What is a web server
2. outline the two implementations of a web server
3. list some of the HTTP server error response codes
4. what are the caused of web server overload
5. explain three examples of web server software

Suggested Further Reading

1. Mark S et al (1996), Special Edition using internet HTML Que Publishing Co ltd
2. Alex H et all(2000),, Professional active server Wrox publishers programmer to programmer series
3. http://www.w3schools.com

CHAPTER THREE: STATIC CONTENT - ADVANCED HTML

Learning Objectives:

By the end of this chapter the learner shall be able to;

1. explain what is a static website
2. explain the html doctypes
3. explain the HTML head elements
4. create the HTML forms

3.1 Static Websites

With static web sites, requests for pages are handled by a web server delivering the content of these HTML files, "as is". Their content don't change often. A static web page (sometimes called a flat page) is a web page that is delivered to the user exactly as stored. static web page displays the same information for all users, from all contexts, subject to modern capabilities of a web server to negotiate content-type or language of the document where such versions are available and the server is configured to do so. Static web pages are often HTML documents stored as files in the file system and made available by the web server over HTTP.

Advantages of a static website:

- Quick to develop
- Cheap to develop
- Cheap to host
- Inherently publicly cacheable (i.e. a cached copy can be shown to anyone).
- No particular hosting requirements are necessary.
- Can be viewed directly by a web browser without needing a web server or application server, for example directly from a CD-ROM or USB Drive.

Disadvantages of a static website

- Requires web development expertise to update site
- Site not as useful for the user since content can get stagnant
- Any personalization or interactivity has to run client-side (ie. in the browser), which is restricting.
- Maintaining large numbers of static pages as files can be impractical without automated tools.

3.2 HTML Doctypes

A doctype declaration refers to the rules for the markup language, so that the browsers render the content correctly.

Example, An HTML document with a doctype of HTML 4.01 Transitional:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Title of the document</title>
</head>
<body>
The content of the document.....
</body>
</html>
```

HTML Different Doctypes

The doctype declaration is not an HTML tag; it is an instruction to the web browser about what version of the markup language the page is written in. The doctype declaration refers to a Document Type Definition (DTD). The DTD specifies the rules for the markup language, so that the browsers render the content correctly. The doctype declaration should be the very first thing in an HTML document, before the <html> tag. Always add a doctype to your pages. This helps the browsers to render the page correctly.

The following are the types of doctypes:

1. *HTML 4.01 Strict*

This DTD contains all HTML elements and attributes, but does NOT INCLUDE presentational or deprecated elements (like font and center). Framesets are not allowed:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
```

2. *HTML 4.01 Transitional*

This DTD contains all HTML elements and attributes, INCLUDING presentational and deprecated elements (like font). Framesets are not allowed:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

3. *HTML 4.01 Frameset*

This DTD is equal to HTML 4.01 Transitional, but allows the use of frameset content:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
```

3.3 HTML head Elements

The head element is a container for all the head elements. Elements inside <head> can include scripts, instruct the browser where to find style sheets and provide meta information. The following tags can be added to the head section: <title>, <base>, <link>, <meta>, <script>, and <style>.

The HTML title Element

Define a document title. Required in every HTML and XHTML document. User agents may use the title in different ways. For example:

- Web browsers usually display it in a window's title bar when the window is open, and (where applicable) in the task bar when the window is minimized.
- It may become the default filename when saving the page.
- Search engines' web crawlers may pay particular attention to the words used in the title.
- provides a title for the page when it is added to favourites.

The title element must not contain other elements, only text. Only one title element is permitted in a document.

A simplified HTML document:

```
<html>
<head>
<title>Title of the document</title>
</head>
<body>
The content of the document.....
</body>
</html>
```

The HTML base Element

The <base> tag specifies a default address or a default target for all links on a page. Specifies a base URL for all relative href and other links in the document. Must appear before any element that refers to an external resource. HTML permits only one base element for each document. The base element has attributes, but no contents.

```
<head>
<base href="http://www.w3schools.com/images/" />
<base target="_blank" />
</head>
```

The HTML link Element

The <link> tag defines the relationship between a document and an external resource. Specifies links to other documents, such as previous and next links, or alternate versions. A common use is to link to external stylesheets, using the form:

<head>
<link rel="stylesheet" type="text/css" href="url" title="description_of_style">
</head>

A document's head element may contain any number of link elements. The link element has attributes, but no contents.

The HTML style Element

The <style> tag is used to define style information for an HTML document. Inside the style element you specify how HTML elements should render in a browser:

```
<head>
<style type="text/css">
body {background-color:yellow}
p {color:blue}
</style>
</head>
```

3.4 HTML Meta

The HTML meta Element

Metadata is information about data. The <meta> tag provides metadata about the HTML document. Metadata will not be displayed on the page, but will be machine parsable. Meta elements are typically used to specify page description, keywords, author of the document, last modified, and other metadata. The <meta> tag always goes inside the head element. The metadata can be used by browsers (how to display content or reload page), search engines (keywords), or other web services.

Keywords for Search Engines

Some search engines will use the name and content attributes of the meta element to index your pages. The following meta element defines a description of a page:

```
<meta name="description" content="Free Web tutorials on HTML, CSS, XML" />
```

The following meta element defines keywords for a page:

```
<meta name="keywords" content="HTML, CSS, XML" />
```

The intention of the name and content attributes is to describe the content of a page.

Note: A lot of webmasters have used <meta> tags for spamming, like repeating keywords (or using wrong keywords) for higher ranking. Therefore, most search engines have stopped using <meta> tags to index/rank pages.

3.5 HTML Scripts

The HTML script Element

The <script> tag is used to define a client-side script, such as a JavaScript. The script element either contains scripting statements or it points to an external script file through the src attribute. The required type attribute specifies the MIME type of the script. Common uses for JavaScript are image manipulation, form validation, and dynamic changes of content. The script below writes Hello World! to the HTML output:

```
<script type="text/javascript">
document.write("Hello World!")
</script>
```

The HTML noscript Element

The <noscript> tag is used to provide an alternate content for users that have disabled scripts in their browser or have a browser that doesn't support client-side scripting. The noscript element can contain all the elements that you can find inside the body element of a normal HTML page. The content inside the noscript element will only be displayed if scripts are not supported, or are disabled in the user's browser:

3.6 HTML Entities

Reserved characters in HTML must be replaced with character entities. Some characters are reserved in HTML. It is not possible to use the less than (<) or greater than (>) signs in your text, because the browser will mix them with tags. To actually display reserved characters, we must use character entities in the HTML source code. A character entity looks like this:

&entity_name;

OR

&#entity_number;

To display a less than sign we must write: **<** or **<**

Note: The advantage of using an entity name, instead of a number, is that the name is easier to remember. However, the disadvantage is that browsers may not support all entity names (the support for entity numbers is very good).

Non-breaking Space

A common character entity used in HTML is the non-breaking space ( ). Browsers will always truncate spaces in HTML pages. If you write 10 spaces in your text, the browser will remove 9 of them, before displaying the page. To add spaces to your text, you can use the   character entity.

HTML Useful Character Entities

Note: Entity names are case sensitive!

Result	Description	Entity Name	Entity Number
	non-breaking space		

<	less than	<	<
>	greater than	>	>
&	ampersand	&	&
¢	cent	¢	¢
£	pound	£	£
¥	yen	¥	¥
€	euro	€	€
§	section	§	§
©	copyright	©	©
®	registered trademark	®	®
™	trademark	™	™

3.7 HTML Uniform Resource Locators

A URL is another word for a web address. A URL can be composed of words, such as "w3schools.com", or an Internet Protocol (IP) address: 192.68.20.50. Most people enter the name of the website when surfing, because names are easier to remember than numbers. When you click on a link in an HTML page, an underlying <a> tag points to an address on the world wide web. A Uniform Resource Locator (URL) is used to address a document (or other data) on the world wide web.

A web address, like this: http://www.w3schools.com/html/default.asp follows these syntax rules:

scheme://host.domain:port/path/filename

Explanation:

- **scheme** - defines the **type** of Internet service. The most common type is **http**
- **host** - defines the **domain host** (the default host for http is **www**)
- **domain** - defines the Internet **domain name**, like w3schools.com
- **:port** - defines the **port number** at the host (the default port number for http is **80**)
- **path** - defines a **path** at the server (If omitted, the document must be stored at the root directory of the web site)
- **filename** - defines the name of a document/resource

Common URL Schemes

The table below lists some common schemes:

Scheme	**Short for....**	**Which pages will the scheme be used for...**
http	HyperText Transfer Protocol	Common web pages starts with http://. Not encrypted
https	Secure HyperText Transfer	Secure web pages. All information exchanged are

	Protocol	encrypted
ftp	File Transfer Protocol	For downloading or uploading files to a website. Useful for domain maintenance
file		A file on your computer

3.8 HTML URL Encoding

URL encoding converts characters into a format that can be transmitted over the Internet.

URL - Uniform Resource Locator

Web browsers request pages from web servers by using a URL. The URL is the address of a web page, like: **http://www.w3schools.com.**

URL Encoding

URLs can only be sent over the Internet using the ASCII character-set. Since URLs often contain characters outside the ASCII set, the URL has to be converted into a valid ASCII format. URL encoding replaces non ASCII characters with a "%" followed by two hexadecimal digits. URLs cannot contain spaces. URL encoding normally replaces a space with a + sign.

URL Encoding Examples

Character	**URL-encoding**
€	%80
£	%A3
©	%A9
®	%AE
À	%C0
Á	%C1
Â	%C2
Ã	%C3
Ä	%C4
Å	%C5

3.9 HTML Forms

HTML Forms are used to select different kinds of user input. HTML forms are used to pass data to a server. A form can contain input elements like text fields, checkboxes, radio-buttons, submit buttons and more. A form can also contain select lists, textarea, fieldset, legend, and label elements. The <form> tag is used to create an HTML form:

```
<form>
.
input elements
.
</form>
```

The Input Element

The input element is used to select user information. An input element can vary in many ways, depending on the type attribute. An input element can be of type text field, checkbox, password, radio button, submit button, and more. The most used input types are described below.

Text Fields

<input type="text" /> defines a one-line input field that a user can enter text into:

```
<form>

First name: <input type="text" name="firstname" /><br />

Last name: <input type="text" name="lastname" />

</form>
```

How the HTML code above looks in a browser:

First name:

Last name:

Note: The form itself is not visible. Also note that the default width of a text field is 20 characters.

Password Field

<input type="password" /> defines a password field:

```
<form>

Password: <input type="password" name="pwd" />

</form>
```

How the HTML code above looks in a browser:

Password: *****

Note: The characters in a password field are masked (shown as asterisks or circles).

Radio Buttons

<input type="radio" /> defines a radio button. Radio buttons let a user select ONLY ONE one of a limited number of choices:

```
<form>

<input type="radio" name="sex" value="male" /> Male<br />

<input type="radio" name="sex" value="female" /> Female

</form>
```

How the HTML code above looks in a browser:

Male

Female

Checkboxes

<input type="checkbox" /> defines a checkbox. Checkboxes let a user select ONE or MORE options of a limited number of choices.

```
<form>

<input type="checkbox" name="vehicle" value="Bike" /> I have a bike<br />

<input type="checkbox" name="vehicle" value="Car" /> I have a car

</form>
```

How the HTML code above looks in a browser:

I have a bike

I have a car

Submit Button

<input type="submit" /> defines a submit button.

A submit button is used to send form data to a server. The data is sent to the page specified in the form's action attribute. The file defined in the action attribute usually does something with the received input:

```
<form name="input" action="html_form_action.asp" method="get">

Username: <input type="text" name="user" />
```

<input type="submit" value="Submit" />

</form>

How the HTML code above looks in a browser:

Username:

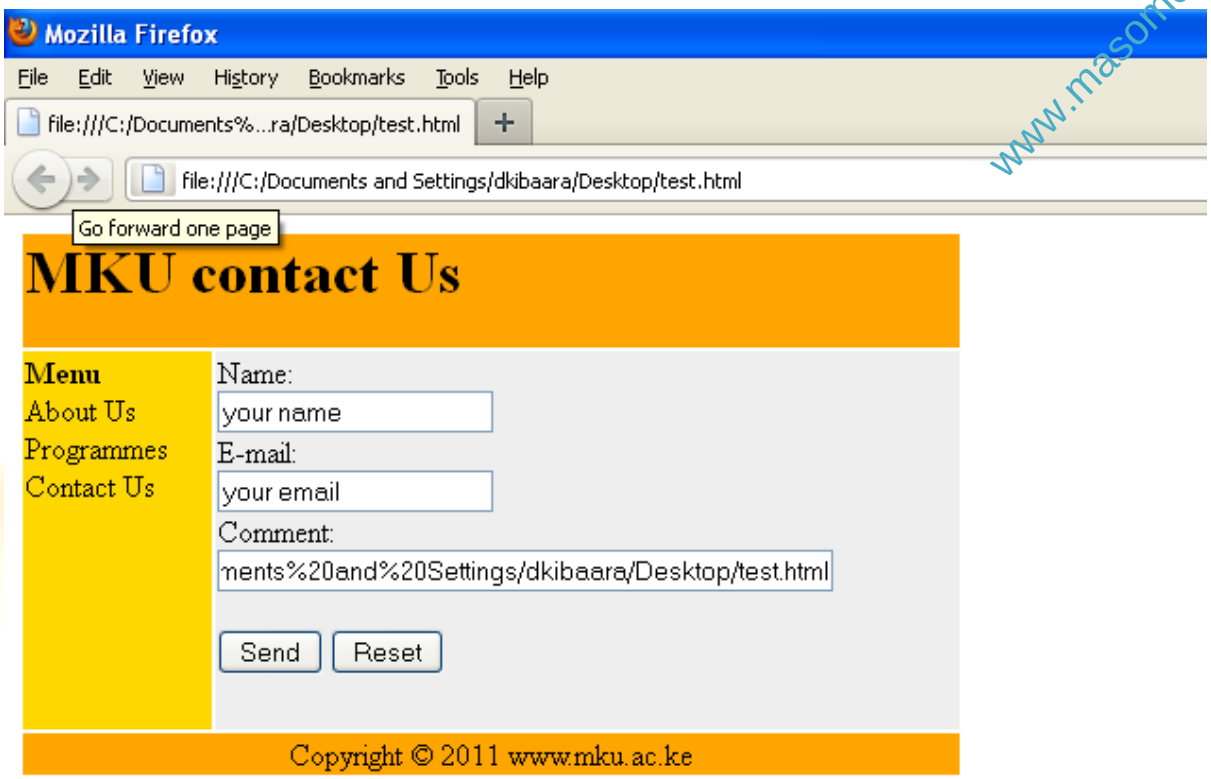
If you type some characters in the text field above, and click the "Submit" button, the browser will send your input to a page called "html_form_action.asp". The page will show you the received input.

HTML Form Tags

Tag	Description
<form>	Defines an HTML form for user input
<input />	Defines an input control
<textarea>	Defines a multi-line text input control
<label>	Defines a label for an input element
<fieldset>	Defines a border around elements in a form
<legend>	Defines a caption for a fieldset element
<select>	Defines a select list (drop-down list)
<optgroup>	Defines a group of related options in a select list
<option>	Defines an option in a select list
<button>	Defines a push button

Chapter Review Questions

1. Define a HTML form
2. Write the HTML code to display the following window



3. Define the following HTML doctypes
    a. HTML 4.01 Strict
    b. HTML 4.01 Transitional
    c. HTML 4.01 Frameset

Suggested Further Reading

1. Mark S et al (1996), Special Edition using internet HTML Que Publishing Co ltd
2. Alex H et all(2000),, Professional active server Wrox publishers programmer to programmer series
3. http://www.w3schools.com

CHAPTER FOUR: INTRODUCTION TO XML (EXTENSIBLE MARKUP LANGUAGE)

Learning Objectives:

By the end of this chapter the learner shall be able to;

i. explain XML

ii. Explain how XML is used

iii. Explain XML tree, XML Syntax Rules and XML Elements

4.1 Introduction XML

Extensible Markup Language (XML) is a set of rules for encoding documents in machine-readable form. It is defined in the XML 1.0 Specification produced by the W3C. XML is designed to transport and store data. The design goals of XML emphasize simplicity, generality, and usability over the Internet. It is a textual data format with strong support via Unicode for the languages of the world. Although the design of XML focuses on documents, it is widely used for the representation of arbitrary data structures, for example in web services.

Many application programming interfaces (APIs) have been developed that software developers use to process XML data, and several schema systems exist to aid in the definition of XML-based languages. Hundreds of XML-based languages have been developed, including RSS, Atom, SOAP, and XHTML. XML-based formats have become the default for most office-productivity tools, including Microsoft Office (Office Open XML), OpenOffice.org (OpenDocument), and Apple's iWork.

The Difference Between XML and HTML

XML is not a replacement for HTML. XML and HTML were designed with different goals:

- XML was designed to transport and store data, with focus on what data is
- HTML was designed to display data, with focus on how data looks

XML Does Not DO Anything

Maybe it is a little hard to understand, but XML does not DO anything. XML was created to structure, store, and transport information. The following example is a note to Tove, from Jani, stored as XML:

<note>

<to>Tove</to>

<from>Jani</from>

<heading>Reminder</heading>

<body>Don't forget me this weekend!</body>

</note>

The note above is quite self descriptive. It has sender and receiver information, it also has a heading and a message body. But still, this XML document does not DO anything. It is just information wrapped in tags. Someone must write a piece of software to send, receive or display it.

With XML You Invent Your Own Tags

The tags in the example above (like <to> and <from>) are not defined in any XML standard. These tags are "invented" by the author of the XML document. That is because the XML language has no predefined tags. The tags used in HTML are predefined. HTML documents can only use tags defined in the HTML standard (like <p>, <h1>, etc.). XML allows the author to define his/her own tags and his/her own document structure.

XML is Not a Replacement for HTML

XML is a complement to HTML. It is important to understand that XML is not a replacement for HTML. In most web applications, XML is used to transport data, while HTML is used to format and display the data. The best description of XML is this: *XML is a software- and hardware-independent tool for carrying information.*

How is XML Used

XML is used in many aspects of web development, often to simplify data storage and sharing.

1. XML Separates Data from HTML - If you need to display dynamic data in your HTML document, it will take a lot of work to edit the HTML each time the data changes. With XML, data can be stored in separate XML files. This way you can concentrate on using HTML for

layout and display, and be sure that changes in the underlying data will not require any changes to the HTML.

2. XML Simplifies Data Sharing - In the real world, computer systems and databases contain data in incompatible formats. XML data is stored in plain text format. This provides a software- and hardware-independent way of storing data. This makes it much easier to create data that can be shared by different applications.

3. XML Simplifies Data Transport - One of the most time-consuming challenges for developers is to exchange data between incompatible systems over the Internet. Exchanging data as XML greatly reduces this complexity, since the data can be read by different incompatible applications.

4. XML Simplifies Platform Changes - Upgrading to new systems (hardware or software platforms), is always time consuming. Large amounts of data must be converted and incompatible data is often lost. XML data is stored in text format. This makes it easier to expand or upgrade to new operating systems, new applications, or new browsers, without losing data.

5. XML Makes Your Data More Available - Different applications can access your data, not only in HTML pages, but also from XML data sources. With XML, your data can be available to all kinds of "reading machines" (Handheld computers, voice machines, news feeds, etc), and make it more available for blind people, or people with other disabilities.

6. XML is Used to Create New Internet Languages - A lot of new Internet languages are created with XML. Here are some examples:

    a. XHTML

    b. WSDL for describing available web services

    c. WAP and WML as markup languages for handheld devices

    d. RSS languages for news feeds

    e. RDF and OWL for describing resources and ontology

    f. SMIL for describing multimedia for the web

XML Key terminology

- (Unicode) Character - By definition, an XML document is a string of characters. Almost every legal Unicode character may appear in an XML document.

- Processor and Application - The processor analyzes the markup and passes structured information to an application. The specification places requirements on what an XML processor must do and not do, but the application is outside its scope. The processor (as the specification calls it) is often referred to colloquially as an XML parser.

- Markup and Content - The characters which make up an XML document are divided into markup and content. Markup and content may be distinguished by the application of simple syntactic rules. All strings which constitute markup either begin with the character "<" and end with a ">", or begin with the character "&" and end with a ";". Strings of characters which are not markup are content. E.g. <markup>content</markup>

- Tag - A markup construct that begins with "<" and ends with ">". There are three types of tags: start-tags, for example <section>, end-tags, for example </section>, and empty-element tags, for example <line-break />.

- Element - A logical document component that either begins with a start-tag and ends with a matching end-tag or consists only of an empty-element tag. The characters between the start- and end-tags, if any, are the element's content, and may contain markup, including other elements, which are called child elements. An example of an element is <Greeting>Hello, world.</Greeting>. Another is <line-break />.

- Attribute - A markup construct consisting of a name/value pair that exists within a start-tag or empty-element tag. In the example (below) the element img has two attributes, src and alt: . Another example would be <step number="3">Connect A to B.</step> where the name of the attribute is "number" and the value is "3".

- XML Declaration - XML documents may begin by declaring some information about themselves, which include the version of XML and character encoding used, as in the following example.

  <?xml version="1.0" encoding="UTF-8" ?>

4.2 XML Tree

XML documents form a tree structure that starts at "the root" and branches to "the leaves".

An Example XML Document

XML documents use a self-describing and simple syntax:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note>
 <to>Tove</to>
 <from>Jani</from>
 <heading>Reminder</heading>
 <body>Don't forget me this weekend!</body>
</note>
```

The first line is the **XML declaration**. It defines the XML version (1.0) and the encoding used (ISO-8859-1 = Latin-1/West European character set). The next line describes the **root element** of the document (like saying: "this document is a note"):

```
<note>
```

The next 4 lines describe 4 **child elements** of the root (to, from, heading, and body):

```
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
```

And finally the last line defines the **end of the root element**:

```
</note>
```

You can assume, from this example, that the XML document contains a note to Tove from Jani.

XML Documents Form a Tree Structure

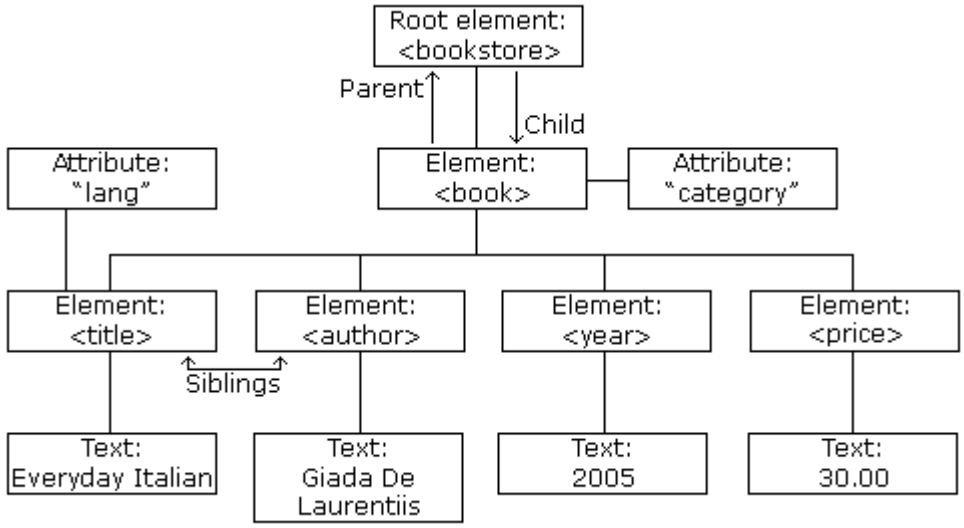
XML documents must contain a root element. This element is "the parent" of all other elements. The elements in an XML document form a document tree. The tree starts at the root and branches to the lowest level of the tree. All elements can have sub elements (child elements):

```
<root>
```

```
  <child>
    <subchild>.....</subchild>
  </child>
</root>
```

The terms parent, child, and sibling are used to describe the relationships between elements. Parent elements have children. Children on the same level are called siblings (brothers or sisters). All elements can have text content and attributes (just like in HTML).

Example:



The image above represents one book in the XML below:

```
<bookstore>
  <book category="COOKING">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
```

```
<book category="CHILDREN">
  <title lang="en">Harry Potter</title>
  <author>J K. Rowling</author>
  <year>2005</year>
  <price>29.99</price>
</book>
<book category="WEB">
  <title lang="en">Learning XML</title>
  <author>Erik T. Ray</author>
  <year>2003</year>
  <price>39.95</price>
</book>
</bookstore>
```

The root element in the example is <bookstore>. All <book> elements in the document are contained within <bookstore>. The <book> element has 4 children: <title>,< author>, <year>, <price>.

4.3 XML Syntax Rules

The syntax rules of XML are very simple and logical. The rules are easy to learn, and easy to use.

All XML Elements Must Have a Closing Tag

In HTML, some elements do not have to have a closing tag:

```
<p>This is a paragraph
<p>This is another paragraph
```

In XML, it is illegal to omit the closing tag. All elements must have a closing tag:

```
<p>This is a paragraph</p>
<p>This is another paragraph</p>
```

Note: You might have noticed from the previous example that the XML declaration did not have a closing tag. This is not an error. The declaration is not a part of the XML document itself, and it has no closing tag.

XML Tags are Case Sensitive

XML tags are case sensitive. The tag <Letter> is different from the tag <letter>.

Opening and closing tags must be written with the same case:

<Message>This is incorrect</message>

<message>This is correct</message>

XML Elements Must be Properly Nested

In HTML, you might see improperly nested elements:

This text is bold and italic

In XML, all elements must be properly nested within each other:

This text is bold and italic

In the example above, "Properly nested" simply means that since the element is opened inside the element, it must be closed inside the element.

XML Documents Must Have a Root Element

XML documents must contain one element that is the parent of all other elements. This element is called the root element.

```
<root>
  <child>
    <subchild>.....</subchild>
  </child>
</root>
```

XML Attribute Values Must be Quoted

XML elements can have attributes in name/value pairs just like in HTML. In XML, the attribute values must always be quoted. Study the two XML documents below. The first one is incorrect, the second is correct:

First document

```
<note date=12/11/2007>
  <to>Tove</to>
  <from>Jani</from>
</note>
```

Second document

```
<note date="12/11/2007">
  <to>Tove</to>
  <from>Jani</from>
</note>
```

The error in the first document is that the date attribute in the note element is not quoted.

Entity References

Some characters have a special meaning in XML. If you place a character like "<" inside an XML element, it will generate an error because the parser interprets it as the start of a new element.

This will generate an XML error:

```
<message>if salary < 1000 then</message>
```

To avoid this error, replace the "<" character with an entity reference:

```
<message>if salary &lt; 1000 then</message>
```

There are 5 predefined entity references in XML:

<	<	less than
>	>	greater than
&	&	ampersand

&apos;	'	apostrophe
&quot;	"	quotation mark

Note: Only the characters "<" and "&" are strictly illegal in XML. The greater than character is legal, but it is a good habit to replace it.

Comments in XML

The syntax for writing comments in XML is similar to that of HTML.

<!-- This is a comment -->

White-space is Preserved in XML

HTML truncates multiple white-space characters to one single white-space:

HTML:	Hello Tove
Output:	Hello Tove

With XML, the white-space in a document is not truncated.

XML Stores New Line as LF

In Windows applications, a new line is normally stored as a pair of characters: carriage return (CR) and line feed (LF). In Unix applications, a new line is normally stored as an LF character. Macintosh applications also use an LF to store a new line. XML stores a new line as LF.

4.4 XML Elements

An XML document contains XML Elements. An XML element is everything from (including) the element's start tag to (including) the element's end tag. An element can contain:

- other elements
- text
- attributes
- or a mix of all of the above...

Example

```
<bookstore>
  <book category="CHILDREN">
    <title>Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="WEB">
    <title>Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

In the example above, <bookstore> and <book> have element contents, because they contain other elements. <book> also has an attribute (category="CHILDREN"). <title>, <author>, <year>, and <price> have text content because they contain text.

XML Naming Rules

XML elements must follow these naming rules:

- Names can contain letters, numbers, and other characters
- Names cannot start with a number or punctuation character
- Names cannot start with the letters xml (or XML, or Xml, etc)
- Names cannot contain spaces

Any name can be used, no words are reserved.

Best Naming Practices

Make names descriptive. Names with an underscore separator are nice: <first_name>, <last_name>.

- Names should be short and simple, like this: <book_title> not like this: <the_title_of_the_book>.
- Avoid "-" characters. If you name something "first-name," some software may think you want to subtract name from first.
- Avoid "." characters. If you name something "first.name," some software may think that "name" is a property of the object "first."
- Avoid ":" characters. Colons are reserved to be used for something called namespaces (more later).
- XML documents often have a corresponding database. A good practice is to use the naming rules of your database for the elements in the XML documents.
- Non-English letters like éòá are perfectly legal in XML, but watch out for problems if your software vendor doesn't support them.

XML Elements are Extensible

XML elements can be extended to carry more information. Look at the following XML example:

<note>

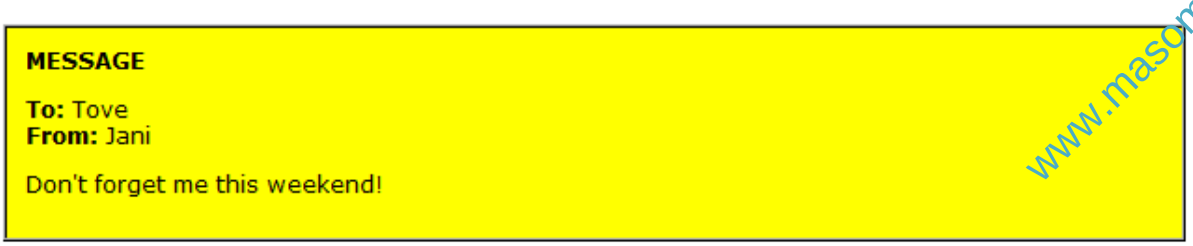
<to>Tove</to>

<from>Jani</from>

<body>Don't forget me this weekend!</body>

</note>

Let's imagine that we created an application that extracted the <to>, <from>, and <body> elements from the XML document to produce this output:



Imagine that the author of the XML document added some extra information to it:

<note>

<date>2008-01-10</date>

<to>Tove</to>

<from>Jani</from>

<heading>Reminder</heading>

<body>Don't forget me this weekend!</body>

</note>

Should the application break or crash? No. The application should still be able to find the <to>, <from>, and <body> elements in the XML document and produce the same output. One of the beauties of XML, is that it can be extended without breaking applications.

4.5 XML Attributes

XML elements can have attributes, just like HTML. Attributes provide additional information about an element.

In HTML, attributes provide additional information about elements:

<img src="computer.gif">

<a href="demo.asp">

Attributes often provide information that is not a part of the data. In the example below, the file type is irrelevant to the data, but can be important to the software that wants to manipulate the element:

<file type="gif">computer.gif</file>

XML Attributes Must be Quoted

Attribute values must always be quoted. Either single or double quotes can be used. For a person's sex, the person element can be written like this:

<person sex="female">

or like this:

<person sex='female'>

If the attribute value itself contains double quotes you can use single quotes, like in this example:

<gangster name='George "Shotgun" Ziegler'>

or you can use character entities:

<gangster name="George "Shotgun" Ziegler">

XML Elements vs. Attributes

Take a look at these examples:

First Example

<person sex="female">

<firstname>Anna</firstname>

<lastname>Smith</lastname>

</person>

Second example

<person>

<sex>female</sex>

<firstname>Anna</firstname>

<lastname>Smith</lastname>

</person>

In the first example sex is an attribute. In the last, sex is an element. Both examples provide the same information. There are no rules about when to use attributes or when to use elements. Attributes are handy in HTML. In XML avoid them. Use elements instead.

Best Practice

The following three XML documents contain exactly the same information:

A date attribute is used in the first example:

```
<note date="10/01/2008">
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

A date element is used in the second example:

```
<note>
  <date>10/01/2008</date>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

An expanded date element is used in the third: (BEST PRACTICE):

```
<note>
  <date>
    <day>10</day>
    <month>01</month>
    <year>2008</year>
  </date>
```

```
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
</note>
```

Why you need to avoid using XML Attributes

Some of the problems with using attributes are:

- attributes cannot contain multiple values (elements can)
- attributes cannot contain tree structures (elements can)
- attributes are not easily expandable (for future changes)

Attributes are difficult to read and maintain. Use elements for data. Use attributes for information that is not relevant to the data.

Don't end up like this:

```
<note day="10" month="01" year="2008"
to="Tove" from="Jani" heading="Reminder"
body="Don't forget me this weekend!">
</note>
```

XML Attributes for Metadata

Sometimes ID references are assigned to elements. These IDs can be used to identify XML elements in much the same way as the id attribute in HTML. This example demonstrates this:

```
<messages>
  <note id="501">
    <to>Tove</to>
    <from>Jani</from>
```

```
  <heading>Reminder</heading>
  <body>Don't forget me this weekend!</body>
 </note>
 <note id="502">
  <to>Jani</to>
  <from>Tove</from>
  <heading>Re: Reminder</heading>
  <body>I will not</body>
 </note>
</messages>
```

The id attributes above are for identifying the different notes. It is not a part of the note itself.

Metadata (data about data) should be stored as attributes, and the data itself should be stored as elements.

4.6 XML Validation

XML with correct syntax is "Well Formed" XML. XML validated against a DTD is "Valid" XML.

Well Formed XML Documents

A "Well Formed" XML document has correct XML syntax. See example below

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

Valid XML Documents

A "Valid" XML document is a "Well Formed" XML document, which also conforms to the rules of a Document Type Definition (DTD):

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE note SYSTEM "Note.dtd">

<note>

<to>Tove</to>

<from>Jani</from>

<heading>Reminder</heading>

<body>Don't forget me this weekend!</body>

</note>
```

The DOCTYPE declaration in the example above, is a reference to an external DTD file. The content of the file is shown in the paragraph below.

XML DTD

The purpose of a DTD is to define the structure of an XML document. It defines the structure with a list of legal elements:

```
<!DOCTYPE note

[

<!ELEMENT note (to,from,heading,body)>

<!ELEMENT to (#PCDATA)>

<!ELEMENT from (#PCDATA)>

<!ELEMENT heading (#PCDATA)>

<!ELEMENT body (#PCDATA)>

]>
```

XML Schema

W3C supports an XML-based alternative to DTD, called XML Schema:

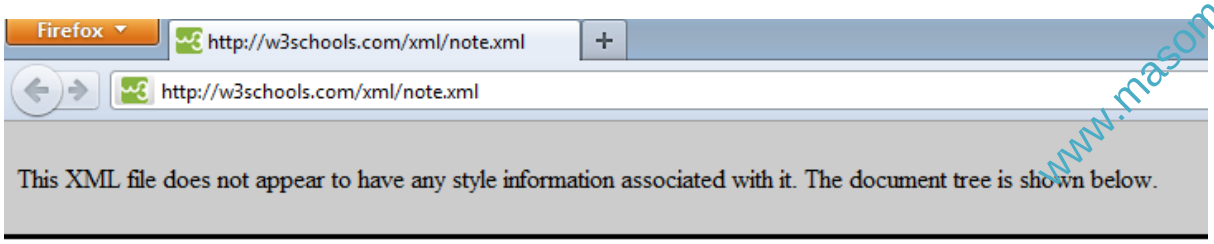
```
<xs:element name="note">
<xs:complexType>
  <xs:sequence>
    <xs:element name="to" type="xs:string"/>
    <xs:element name="from" type="xs:string"/>
    <xs:element name="heading" type="xs:string"/>
    <xs:element name="body" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
</xs:element>
```

4.7 Viewing XML Files

Raw XML files can be viewed in all major browsers. Don't expect XML files to be displayed as HTML pages.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
 - <note>
      <to>Tove</to>
      <from>Jani</from>
      <heading>Reminder</heading>
      <body>Don't forget me this weekend!</body>
   </note>
```

The following image is the display of above xml file on a web browser.




```
<!-- Edited by XMLSpy® -->
-<note>
    <to>Tove</to>
    <from>Jani</from>
    <heading>Reminder</heading>
    <body>Don't forget me this weekend!</body>
  </note>
```

The XML document will be displayed with colour-coded root and child elements. A plus (+) or minus sign (-) to the left of the elements can be clicked to expand or collapse the element structure. To view the raw XML source (without the + and - signs), select "View Page Source" or "View Source" from the browser menu.

Chapter Review Questions

1. What is XML?
2. differentiate between XML and HTML
3. How is XML Used?
4. explain the XML tree giving an example

Suggested Further Reading

1. Mark S et al (1996), Special Edition using internet HTML Que Publishing Co ltd
2. Alex H et all(2000),, Professional active server Wrox publishers programmer to programmer series
3. http://www.w3schools.com

CHAPTER FIVE: DYNAMIC CONTENT - INTRODUCTION TO DHTML

Learning Objectives:

By the end of this chapter the learner shall be able to;

i. Explain the dynamic websites

ii. Explain DHTML

iii. Create DHTML pages

5.1 Dynamic websites

With dynamic web sites, each request, the page (the HTML) is constructed from information stored in files (such as images), information stored in databases (textual content) and programming logic (both server side such as PHP, Java, ASP etc and with "Web 2.0" more client side - Javascript). A dynamic web page is a kind of web page that has been prepared with fresh information (content and/or layout), for each individual viewing. It is not static because it changes with the time (e.g. a news content), the user (e.g. preferences in a login session), the user interaction (e.g. web page game), the context (parametric customization), or any combination thereof.

DHTML is the art of combining HTML, JavaScript, DOM, and CSS. DHTML is NOT a language or a web standard. According to the World Wide Web Consortium (W3C): "Dynamic HTML is a term used by some vendors to describe the combination of HTML, style sheets and scripts that allows documents to be animated."

1. *HTML*

The W3C HTML 4 standard has rich support for dynamic content:

- HTML supports JavaScript
- HTML supports the Document Object Model (DOM)
- HTML supports HTML Events
- HTML supports Cascading Style Sheets (CSS)

DHTML is about using these features, to create dynamic and interactive web pages.

2. *JavaScript*

JavaScript is the most popular scripting language on the internet, and it works in all major browsers. DHTML is about using JavaScript to control, access and manipulate HTML elements.

3. *HTML DOM*

The HTML DOM is a W3C standard. It describes the Document Object Model for HTML. The HTML DOM defines a standard way for accessing and manipulating HTML documents. DHTML is about using the DOM to access and manipulate HTML elements.

4. *HTML Events*

HTML events are a part of the HTML DOM. DHTML is about creating web pages that reacts to (user)events.

5. *CSS*

CSS defines how to display HTML elements. DHTML is about using JavaScript and the HTML DOM to change the style and positioning of HTML elements.

Three types of dynamic web sites

1. *Client-side scripting and content creation*

Using client-side scripting to change interface behaviours within a specific web page, in response to mouse or keyboard actions or at specified timing events. In this case the dynamic behaviour occurs within the presentation.

Such web pages use presentation technology called rich interfaced pages. Client-side scripting languages like JavaScript or ActionScript, used for Dynamic HTML (DHTML) and Flash technologies respectively, are frequently used to orchestrate media types (sound, animations, changing text, etc.) of the presentation. The scripting also allows use of remote scripting, a technique by which the DHTML page requests additional information from a server, using a hidden Frame, XMLHttpRequests, or a Web service.

2. *Server-side scripting and content creation*

A program running on the web server (server-side scripting) is used to change the web content on various web pages, or to adjust the sequence of or reload of the web pages. Server responses may be determined by such conditions as data in a posted HTML form, parameters in the URL, the type of browser being used, the passage of time, or a database or server state.

Such web pages are often created with the help of server-side languages such as ASP, ColdFusion, Perl, PHP, and other languages. These server-side languages often use the Common Gateway Interface (CGI) to produce dynamic web pages. Two notable exceptions are ASP.NET and JSP, which reuse CGI concepts in their APIs but actually dispatch all web requests into a shared virtual machine.

3. *Combining client and server side*

Ajax is a web development technique for dynamically interchanging content with the server-side, without reloading the web page. Google Maps is an example of a web application that uses Ajax techniques and database.

Main advantages of a dynamic website

- Much more functional website
- Much easier to update
- New content brings people back to the site and helps in the search engines
- Can work as a system to allow staff or users to collaborate

Main disadvantages of a dynamic website

- Slower to develop
- Expensive to develop
- Hosting costs a little more

5.2 Dynamic Hypertext Markup Language (DHTML)

Dynamic HTML, or DHTML, is an umbrella term for a collection of technologies used together to create interactive and animated web sites by using a combination of a static markup language (such as HTML), a client-side scripting language (such as JavaScript), a presentation definition language (such as CSS), and the Document Object Model. DHTML allows scripting languages to change variables in a web page's definition language, which in turn affects the look and function of otherwise "static" HTML page content, after the page has been fully loaded and during the viewing process. Thus the dynamic characteristic of DHTML is the way it functions while a page is viewed, not in its ability to generate a unique page with each page load.

Uses of DHTML

DHTML allows authors to add effects to their pages that are otherwise difficult to achieve. For example, DHTML allows the page author to:

- Animate text and images in their document, independently moving each element from any starting point to any ending point, following a predetermined path or one chosen by the user.
- Embed a ticker that automatically refreshes its content with the latest news, stock quotes, or other data.
- Use a form to capture user input, and then process and respond to that data without having to send data back to the server.
- Include rollover buttons or drop-down menus.

5.3 DHTML - JavaScript

JavaScript can create dynamic HTML content.

JavaScript Alone

In JavaScript, the statement: document.write(), is used to write output to a web page.

The following example uses JavaScript to display the current date and time on a page:

Example

```
<html>

<body>

<script type="text/javascript">

document.write(Date());

</script>

</body>

</html>
```

JavaScript and the HTML DOM

A JavaScript can also be used to change the content or attributes of HTML elements. To change the content of an HTML element:

document.getElementById(id).innerHTML=new HTML

To change the attribute of an HTML element:

document.getElementById(id).attribute=new value

JavaScript and HTML Events

A JavaScript can also be executed when an event occurs, like when a user clicks on an HTML element. To execute code when a user clicks on an element, use the following event attribute:

onclick=JavaScript

JavaScript and CSS

A JavaScript can also change the style of HTML elements. To change the style of an HTML element:

document.getElementById(id).style.property=new style

5.4 DHTML - HTML DOM

The HTML DOM is:

- A Document Object Model for HTML
- A standard programming interface for HTML
- Platform- and language-independent
- A W3C standard

The HTML DOM defines the objects and properties of all HTML elements, and the methods (interface) to access them. In other words: The HTML DOM is a standard for how to get, change, add, or delete HTML elements.

Change an HTML Element

The following example changes the content of an h1 element:

Example

<html>

<body>

<h1 id="header">Old Header</h1>

```
<script type="text/javascript">
document.getElementById("header").innerHTML="New Header";
</script>

</body>
</html>
```

Example explained:

- The HTML document above contains an h1 element with id="header"
- We use the HTML DOM to get the element with id="header"
- A JavaScript changes the content (innerHTML) of that element

Change an HTML Attribute

The following example changes the src attibute of an img element:

Example

```
<html>
<body>
<img id="image" src="smiley.gif">
<script type="text/javascript">
document.getElementById("image").src="landscape.jpg";
</script>
</body>
</html>
```

Example explained:

- The HTML document above contains an img element with id="image"
- We use the HTML DOM to get the element with id="image"

- A JavaScript changes the src attribute of that element from "smiley.gif" to "landscape.jpg"

5.5 DHTML - HTML Events

HTML events can trigger actions in a browser. Every element on an HTML page has events which can trigger a JavaScript. For example, we can use the onClick event of a button element to indicate that a function will run when a user clicks on the button. We define the events in the HTML tags.

Examples of events:

- A mouse click
- A web page or an image loading
- Mousing over a hot spot on the web page
- Selecting an input field in an HTML form
- Submitting an HTML form
- A keystroke

In the following example, the content of the h1 element will change when a user clicks on it:

Example

```
<html>
<body>
<h1 onclick="this.innerHTML='Ooops!'">Click on this text</h1>
</body>
</html>
```

You can also add the script in the head section, and then call a function from the event handler:

Example

```
<html>
<head>
<script type="text/javascript">
```

```
function changetext(id)
{
id.innerHTML="Ooops!";
}
</script>
</head>
<body>
<h1 onclick="changetext(this)">Click on this text</h1>
</body>
</html>
```

5.6 DHTML - CSS

JavaScript and the HTML DOM can be used to change the style of any HTML element.

Change Style of the Current HTML Element.

To change the style of the current HTML element, use the following statement:

this.style.property=new style

Example

```
<html>
<body>
<h1 onclick="this.style.color='red'">Click Me!</h1>
</body>
</html>
```

Change Style of a Specific HTML Element

To change the style of a specific HTML element, use the following statement:

document.getElementById(id).style.property=new style

Example

```
<html>

<body>

<h1 id="h1" onclick="document.getElementById('h1').style.color='red'">Click Me!</h1>

</body>

</html>
```

Chapter Review Questions

1. what is a dynamic website
2. What is DHTML?
3. define three types of a dynamic website
4. explain how JavaScript can create Dynamic HTML content
5. give example of events in HTML documents

Suggested Further Reading

1. Mark S et al (1996), Special Edition using internet HTML Que Publishing Co ltd
2. Alex H et all(2000),, Professional active server Wrox publishers programmer to programmer series
3. http://www.w3schools.com

CHAPTER SIX: DYNAMIC WEBSITE – INTRODUCTION TO PHP

Learning Objectives:

By the end of this chapter the learner shall be able to;

iv. Explain what is PHP

v. Installing WAMP server

vi. Write and run PHP files

6.1 Introduction to PHP

PHP is a general-purpose scripting language originally designed for web development to produce dynamic web pages. For this purpose, PHP code is embedded into the HTML source document and interpreted by a web server with a PHP processor module, which generates the web page document. It also has evolved to include a command-line interface capability and can be used in standalone graphical applications. PHP can be deployed on most web servers and as a standalone interpreter, on almost every operating system and platform free of charge. A competitor to Microsoft's Active Server Pages (ASP) server-side script engine, PHP is installed on more than 20 million websites and 1 million web servers.

PHP was originally created by Rasmus Lerdorf in 1995. The main implementation of PHP is now produced by The PHP Group and serves as the de facto standard for PHP as there is no formal specification. PHP is free software released under the PHP License which is incompatible with the GNU General Public License (GPL) due to restrictions on the usage of the term PHP.

PHP (*Hypertext Preprocessor*) is a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML. PHP is a powerful tool for making dynamic and interactive Web pages. PHP is a server-side scripting language, like ASP. PHP scripts are executed on the server. PHP supports many databases (MySQL, Informix, Oracle, Sybase, Solid, PostgreSQL, Generic ODBC, etc.). PHP is open source software i.e. PHP is free to download and use

PHP File

- PHP files can contain text, HTML tags and scripts

- PHP files are returned to the browser as plain HTML
- PHP files have a file extension of ".php", ".php3", or ".phtml"

Reasons for using PHP

- PHP runs on different platforms (Windows, Linux, Unix, etc.)
- PHP is compatible with almost all servers used today (Apache, IIS, etc.)
- PHP is FREE to download from the official PHP resource: www.php.net
- PHP is easy to learn and runs efficiently on the server side

Main areas where PHP scripts are used

- Server-side scripting
- Command line scripting. You can make a PHP script to run it without any server or browser. You only need the PHP parser to use it this way. This type of usage is ideal for scripts regularly executed using cron (on *nix or Linux) or Task Scheduler (on Windows).
- Writing desktop applications

6.2 Getting started with PHP

PHP runs on the server. You need to download a server application that will run your PHP files from your Desktop or Laptop. You also need to down a PHP editor that will help you edit the PHP files, however you can use notepad but ensure that the file is saved with a .php file extension

WampServer

WampServer is a package of independently-created programs installed on computers that use a Microsoft Windows operating system. WAMP is an acronym formed from the initials of the operating system Microsoft Windows and the principal components of the package: Apache, MySQL and one of PHP. Apache is a web server. MySQL is an open-source database. PHP is a scripting language that can manipulate information held in a database and generate web pages dynamically each time content is requested by a browser. Other programs may also be included in a package, such as phpMyAdmin which provides a graphical user interface for the MySQL database manager. On the other hand we have LAMP which is same as WAMP but for Linux operating system and MAMP for MAC.

Downloading WampServer

To download WampServer go to the following link:

http://www.wampserver.com/en/download.php

Using the above link you will download WampServer 2.1a [24/12/10] which includes:

- Apache 2.2.17
- Php 5.3.3
- Mysql 5.1.53 (version 64 bits)
- Mysql 5.5.8 (version 32 bits)
- PhpMyadmin 3.2.0.1
- SQLBuddy 1.3.2

How to install WampServer

Double click on the downloaded file and just follow the instructions. Everything is automatic. The WampServer package is delivered with the latest releases of Apache, MySQL and PHP.

How to run WampServer

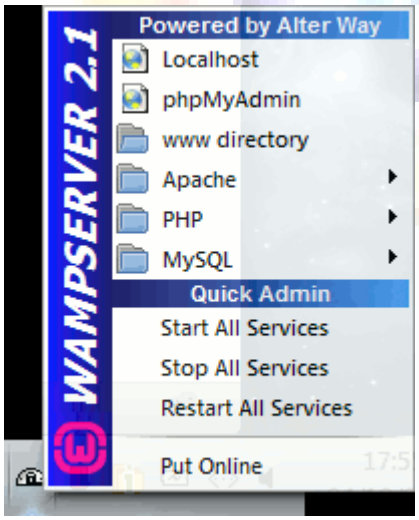
When you install WampServer an icon is created on the desktop and a menu is created on the programs menu on a windows operating system. To run WampServer you need to click on any of these.

Running the first PHP page

Create the following PHP file and save as “hello.php” at c:\wamp\www

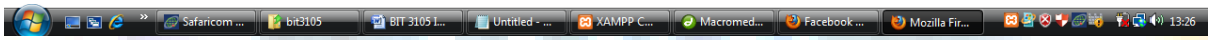
```
<html>
<body>
 <?php
echo "Hello World";
?>
</body>
</html>
```

After creating this file click on the link "Localhost" in the WampServer menu or open your browser and open the http://localhost/hello.php



WAMP sever menu found near the time on the task bar.

After running the first PHP file this will be the result on your browser:



6.3 PHP Syntax

PHP code is executed on the server, and the plain HTML result is sent to the browser. A PHP scripting block always starts with <?php and ends with ?>. A PHP scripting block can be placed anywhere in the document. On servers with shorthand support enabled you can start a scripting block with <? and end with ?>. For maximum compatibility, it's recommend that you use the standard form (<?php) rather than the shorthand form.

```
<?php
?>
```

A PHP file normally contains HTML tags, just like an HTML file, and some PHP scripting code. Below, we have an example of a simple PHP script which sends the text "Hello World" to the browser:

```
<html>
<body>
 <?php
echo "Hello World";
?>
</body>
</html>
```

Each code line in PHP must end with a semicolon. The semicolon is a separator and is used to distinguish one set of instructions from another. There are two basic statements to output text with PHP: echo and print. In the example above we have used the echo statement to output the text "Hello World".

Note: The file must have a .php extension. If the file has a .html extension, the PHP code will not be executed.

Comments in PHP

In PHP, we use // to make a single-line comment or /* and */ to make a large comment block.

```
<html>
<body>
<?php
//This is a comment

/*
This is
a comment
block
*/
?>
</body>
</html>
```

6.4 PHP Variables

A variable is used to store information. Variables are used for storing values, like text strings, numbers or arrays. When a variable is declared, it can be used over and over again in your script. All variables in PHP start with a $ sign symbol.

The correct way of declaring a variable in PHP:

```
$var_name = value;
```

New PHP programmers often forget the $ sign at the beginning of the variable. In that case it will not work. The following are a variable containing a string, and a variable containing a number:

```
<?php
$txt="Hello World!";
$x=16;
?>
```

PHP is a Loosely Typed Language

In PHP, a variable does not need to be declared before adding a value to it. In the example above, you see that you do not have to tell PHP which data type the variable is. PHP automatically converts the variable to the correct data type, depending on its value. In a strongly typed programming language, you have to declare (define) the type and name of the variable before using it. In PHP, the variable is declared automatically when you use it.

Naming Rules for Variables

- A variable name must start with a letter or an underscore "_"
- A variable name can only contain alpha-numeric characters and underscores (a-z, A-Z, 0-9, and _)
- A variable name should not contain spaces. If a variable name is more than one word, it should be separated with an underscore ($my_string), or with capitalization ($myString)

PHP String Variables

A string variable is used to store and manipulate text. String variables are used for values that contain characters. After we create a string we can manipulate it. A string can be used directly in a function or it can be stored in a variable. Below, the PHP script assigns the text "Hello World" to a string variable called $txt:

```
<?php

$txt="Hello World";

echo $txt;

?>
```

The Concatenation Operator

There is only one string operator in PHP. The concatenation operator (.) is used to put two string values together. To concatenate two string variables together, use the concatenation operator:

```
<?php

$txt1="Hello World!";

$txt2="What a nice day!";

echo $txt1 . " " . $txt2;
```

```
?>
```

The output of the code above will be:

Hello World! What a nice day!

If we look at the code above you see that we used the concatenation operator two times. This is because we had to insert a third string (a space character), to separate the two strings.

The strlen() function

The strlen() function is used to return the length of a string. Let's find the length of a string:

```
<?php
echo strlen("Hello world!");
?>
```

The output of the code above will be:

12

The length of a string is often used in loops or other functions, when it is important to know when the string ends. (i.e. in a loop, we would want to stop the loop after the last character in the string).

The strpos() function

The strpos() function is used to search for a character/text within a string. If a match is found, this function will return the character position of the first match. If no match is found, it will return FALSE. Let's see if we can find the string "world" in our string:

```
<?php
echo strpos("Hello world!","world");
?>
```

The output of the code above will be:

6

The position of the string "world" in the example above is 6. The reason that it is 6 (and not 7), is that the first character position in the string is 0, and not 1.

6.5 PHP Operators

Operators are used to operate on values.

Arithmetic Operators

Operator	Description	Example	Result
+	Addition	x=2 x+2	4
-	Subtraction	x=2 5-x	3
*	Multiplication	x=4 x*5	20
/	Division	15/5 5/2	3 2.5
%	Modulus (division remainder)	5%2 10%8 10%2	1 2 0
++	Increment	x=5 x++	x=6
--	Decrement	x=5 x--	x=4

Assignment Operators

Operator	Example	Is The Same As
=	x=y	x=y
+=	x+=y	x=x+y
-=	x-=y	x=x-y
*=	x*=y	x=x*y

/=	x/=y	x=x/y
.=	x.=y	x=x.y
%=	x%=y	x=x%y

Comparison Operators

Operator	Description	Example
==	is equal to	5==8 returns false
!=	is not equal	5!=8 returns true
<>	is not equal	5<>8 returns true
>	is greater than	5>8 returns false
<	is less than	5<8 returns true
>=	is greater than or equal to	5>=8 returns false
<=	is less than or equal to	5<=8 returns true

Logical Operators

Operator	Description	Example
&&	and	x=6 y=3 (x < 10 && y > 1) returns true
||	or	x=6 y=3 (x==5 || y==5) returns false
!	not	x=6 y=3 !(x==y) returns true

6.6 Conditional Statements

Conditional statements are used to perform different actions based on different conditions. Very often when you write code, you want to perform different actions for different decisions. You can use conditional statements in your code to do this. In PHP we have the following conditional statements:

- if statement - use this statement to execute some code only if a specified condition is true
- if...else statement - use this statement to execute some code if a condition is true and another code if the condition is false
- if...elseif....else statement - use this statement to select one of several blocks of code to be executed
- switch statement - use this statement to select one of many blocks of code to be executed

1. ***The if Statement***

Use the, if statement to execute some code only if a specified condition is true.

Syntax

if (condition) code to be executed if condition is true; The following example will output "Have a nice weekend!" if the current day is Friday:

```
<html>
<body>
<?php
$d=date("D");
if ($d=="Fri") echo "Have a nice weekend!";
?>
</body>
</html>
```

Notice that there is no ..else.. in this syntax. The code is executed only if the specified condition is true.

2. ***The if...else Statement***

Use the, if....else statement to execute some code if a condition is true and another code if a condition is false.

Syntax

```
if (condition)
  code to be executed if condition is true;
else
  code to be executed if condition is false;
```

The following example will output "Have a nice weekend!" if the current day is Friday, otherwise it will output "Have a nice day!":

```
<html>
<body>
<?php
$d=date("D");
if ($d=="Fri")
  echo "Have a nice weekend!";
else
  echo "Have a nice day!";
?>
</body>
</html>
```

If more than one line should be executed if a condition is true/false, the lines should be enclosed within curly braces:

```
<html>
<body>
<?php
```

```
$d=date("D");
if ($d=="Fri")
  {
  echo "Hello!<br />";
  echo "Have a nice weekend!";
  echo "See you on Monday!";
  }
?>
</body>
</html>
```

3. *The if...else if....else Statement*

Use the, if....else if...else statement to select one of several blocks of code to be executed.

Syntax

```
if (condition)
  code to be executed if condition is true;
elseif (condition)
  code to be executed if condition is true;
else
  code to be executed if condition is false;
```

The following example will output "Have a nice weekend!" if the current day is Friday, and "Have a nice Sunday!" if the current day is Sunday. Otherwise it will output "Have a nice day!":

```
<html>
<body>
<?php
```

```
$d=date("D");
if ($d=="Fri")
  echo "Have a nice weekend!";
elseif ($d=="Sun")
  echo "Have a nice Sunday!";
else
  echo "Have a nice day!";
?>
</body>
</html>
```

4. PHP Switch Statement

Use the switch statement to select one of many blocks of code to be executed.

Syntax

```
switch (n)
{
case label1:
  code to be executed if n=label1;
  break;
case label2:
  code to be executed if n=label2;
  break;
default:
  code to be executed if n is different from both label1 and label2;
}
```

This is how it works: First we have a single expression n (most often a variable), that is evaluated once. The value of the expression is then compared with the values for each case in the structure. If there is a match, the block of code associated with that case is executed. Use break to prevent the code from running into the next case automatically. The default statement is used if no match is found.

```
<html>
<body>
<?php
switch ($x)
{
case 1:
  echo "Number 1";
  break;
case 2:
  echo "Number 2";
  break;
case 3:
  echo "Number 3";
  break;
default:
  echo "No number between 1 and 3";
}
?>
</body>
</html>
```

6.7 PHP Loops

Loops execute a block of code a specified number of times, or while a specified condition is true. Often when you write code, you want the same block of code to run over and over again in a row. Instead of adding several almost equal lines in a script we can use loops to perform a task like this. In PHP, we have the following looping statements:

- while - loops through a block of code while a specified condition is true
- do...while - loops through a block of code once, and then repeats the loop as long as a specified condition is true
- for - loops through a block of code a specified number of times
- for each - loops through a block of code for each element in an array

1. The while Loop

While loop executes a block of code while a condition is true.

Syntax

```
while (condition)
 {
 code to be executed;
 }
```

The example below defines a loop that starts with i=1. The loop will continue to run as long as i is less than, or equal to 5. i will increase by 1 each time the loop runs:

```
<html>
<body>
<?php
$i=1;
while($i<=5)
 {
 echo "The number is " . $i . "<br />";
```

```
 $i++;
 }
?>
</body>
</html>
```


2. *The do...while Statement*

The do...while statement will always execute the block of code once, it will then check the condition, and repeat the loop while the condition is true.

Syntax

```
do
 {
 code to be executed;
 }
while (condition);
```

The example below defines a loop that starts with i=1. It will then increment i with 1, and write some output. Then the condition is checked, and the loop will continue to run as long as i is less than, or equal to 5:

```
<html>
<body>
<?php
$i=1;
do
 {
 $i++;
 echo "The number is " . $i . "<br />";
```

```
 }
while ($i<=5);
?>
</body>
</html>
```

3. The for Loop

For loop is used when you know in advance how many times the script should run.

Syntax

```
for (init; condition; increment)
 {
 code to be executed;
 }
```

Parameters:

- **init**: Mostly used to set a counter (but can be any code to be executed once at the beginning of the loop)
- **condition**: Evaluated for each loop iteration. If it evaluates to TRUE, the loop continues. If it evaluates to FALSE, the loop ends.
- **increment**: Mostly used to increment a counter (but can be any code to be executed at the end of the loop)

Note: Each of the parameters above can be empty, or have multiple expressions (separated by commas).

The example below defines a loop that starts with i=1. The loop will continue to run as long as i is less than, or equal to 5. i will increase by 1 each time the loop runs:

```
<html>
<body>
```

```
<?php
for ($i=1; $i<=5; $i++)
 {
 echo "The number is " . $i . "<br />";
 }
?>
</body>
</html>
```

4. *The foreach Loop*

The foreach loop is used to loop through arrays.

Syntax

```
foreach ($array as $value)
 {
 code to be executed;
 }
```

For every loop iteration, the value of the current array element is assigned to $value (and the array pointer is moved by one) - so on the next loop iteration, you'll be looking at the next array value. The following example demonstrates a loop that will print the values of the given array:

```
<html>
<body>
<?php
$x=array("one","two","three");
foreach ($x as $value)
 {
```

```
 echo $value . "<br />";
 }
?>
</body>
</html>
```


6.8 PHP Arrays

An array stores multiple values in one single variable. A variable is a storage area holding a number or text. The problem is, a variable will hold only one value. An array is a special variable, which can store multiple values in one single variable. If you have a list of items (a list of car names, for example), storing the cars in single variables could look like this:

```
$cars1="Saab";
$cars2="Volvo";
$cars3="BMW";
```

However, what if you want to loop through the cars and find a specific one? And what if you had not 3 cars, but 300? The best solution here is to use an array. An array can hold all your variable values under a single name. And you can access the values by referring to the array name. Each element in the array has its own index so that it can be easily accessed. In PHP, there are three kind of arrays:

- Numeric array - An array with a numeric index
- Associative array - An array where each ID key is associated with a value
- Multidimensional array - An array containing one or more arrays

1. Numeric Arrays

A numeric array stores each array element with a numeric index. There are two methods to create a numeric array.

1. In the following example the index are automatically assigned (the index starts at 0):

   ```
   $cars=array("Saab","Volvo","BMW","Toyota");
   ```

2. In the following example we assign the index manually:

```
$cars[0]="Saab";

        $cars[1]="Volvo";

        $cars[2]="BMW";

        $cars[3]="Toyota";
```

In the following example you access the variable values by referring to the array name and index:

```
<?php

$cars[0]="Saab";

$cars[1]="Volvo";

$cars[2]="BMW";

$cars[3]="Toyota";

echo $cars[0] . " and " . $cars[1] . " are Swedish cars.";

?>
```

The code above will output:

Saab and Volvo are Swedish cars.

2. *Associative Arrays*

An associative array, each ID key is associated with a value. When storing data about specific named values, a numerical array is not always the best way to do it. With associative arrays we can use the values as keys and assign values to them.

In this example we use an array to assign ages to the different persons:

```
$ages = array("Peter"=>32, "Quagmire"=>30, "Joe"=>34);
```

This example is the same as example 1, but shows a different way of creating the array:

```
$ages['Peter'] = "32";

$ages['Quagmire'] = "30";

$ages['Joe'] = "34";
```

The ID keys can be used in a script:

```
<?php
$ages['Peter'] = "32";
$ages['Quagmire'] = "30";
$ages['Joe'] = "34";
echo "Peter is " . $ages['Peter'] . " years old.";
?>
```

The code above will output:

Peter is 32 years old.

3. Multidimensional Arrays

In a multidimensional array, each element in the main array can also be an array. And each element in the sub-array can be an array, and so on.

In this example we create a multidimensional array, with automatically assigned ID keys:

```
$families = array
 (
 "Griffin"=>array
 (
 "Peter",
 "Lois",
 "Megan"
 ),
 "Quagmire"=>array
 (
 "Glenn"
```

```
),
"Brown"=>array
(
"Cleveland",
"Loretta",
"Junior"
)
);
```

The array above would look like this if written to the output:

```
Array
(
[Griffin] => Array
 (
 [0] => Peter
 [1] => Lois
 [2] => Megan
 )
[Quagmire] => Array
 (
 [0] => Glenn
 )
[Brown] => Array
 (
 [0] => Cleveland
```

[1] => Loretta

[2] => Junior

)

)

Lets try displaying a single value from the array above:

echo "Is " . $families['Griffin'][2] .

" a part of the Griffin family?";

The code above will output:

Is Megan a part of the Griffin family?

6.9 PHP Functions

The real power of PHP comes from its functions. In PHP, there are more than 700 built-in functions. To keep the script from being executed when the page loads, you can put it into a function. A function will be executed by a call to the function. You may call a function from anywhere within a page.

Creating a PHP Function

A function will be executed by a call to the function.

Syntax

function functionName()

{

code to be executed;

}

PHP function guidelines:

- Give the function a name that reflects what the function does
- The function name can start with a letter or underscore (not a number)

A simple function that writes my name when it is called:

```
<html>
<body>
<?php
function writeName()
{
echo "Kai Jim Refsnes";
}
echo "My name is ";
writeName();
?>
</body>
</html>
```

Adding parameters

To add more functionality to a function, we can add parameters. A parameter is just like a variable. Parameters are specified after the function name, inside the parentheses.

The following example will write different first names, but equal last name:

```
<html>
<body>
<?php
function writeName($fname)
{
echo $fname . " Refsnes.<br />";
}
echo "My name is ";
```

```
writeName("Kai Jim");

echo "My sister's name is ";

writeName("Hege");

echo "My brother's name is ";

writeName("Stale");

?>

</body>

</html>
```

The following function has two parameters:

```
<html>

<body>

<?php

function writeName($fname,$punctuation)

{

echo $fname . " Refsnes" . $punctuation . "<br />";

}

echo "My name is ";

writeName("Kai Jim",".");

echo "My sister's name is ";

writeName("Hege","!");

echo "My brother's name is ";

writeName("Ståle","?");

?>

</body>
```

```
</html>
```

Return values

To let a function return a value, use the return statement.

Example

```
<html>

<body>

<?php

function add($x,$y)

{

$total=$x+$y;

return $total;

}

echo "1 + 16 = " . add(1,16);

?>

</body>

</html>
```

6.10 PHP Forms and User Input

The PHP $_GET and $_POST variables are used to retrieve information from forms, like user input.

PHP Form Handling

The most important thing to notice when dealing with HTML forms and PHP is that any form element in an HTML page will automatically be available to your PHP scripts. The example below contains an HTML form with two input fields and a submit button (save this page as welcome .html):

```
<html>

<body>

<form action="welcome.php" method="post">
```

Name: <input type="text" name="fname" />

Age: <input type="text" name="age" />

<input type="submit" />

</form>

</body>

</html>

When a user fills out the form above and click on the submit button, the form data is sent to a PHP file, called "welcome.php":

"welcome.php" looks like this (save this page as welcome .php):

<html>

<body>

Welcome <?php echo $_POST["fname"]; ?>!

You are <?php echo $_POST["age"]; ?> years old.

</body>

</html>

PHP $_GET Function

The built-in $_GET function is used to collect values from a form sent with method="get". Information sent from a form with the GET method is visible to everyone (it will be displayed in the browser's address bar) and has limits on the amount of information to send.

Example

<form action="welcome.php" method="get">

Name: <input type="text" name="fname" />

Age: <input type="text" name="age" />

<input type="submit" />

</form>

When the user clicks the "Submit" button, the URL sent to the server could look something like this:

http://www.w3schools.com/welcome.php?fname=Peter&age=37

The "welcome.php" file can now use the $_GET function to collect form data (the names of the form fields will automatically be the keys in the $_GET array):

Welcome <?php echo $_GET["fname"]; ?>.

You are <?php echo $_GET["age"]; ?> years old!

When to use method="get"?

When using method="get" in HTML forms, all variable names and values are displayed in the URL. This method should not be used when sending passwords or other sensitive information. However, because the variables are displayed in the URL, it is possible to bookmark the page. This can be useful in some cases. The get method is not suitable for very large variable values. It should not be used with values exceeding 2000 characters.

PHP $_POST Function

The built-in $_POST function is used to collect values from a form sent with method="post". Information sent from a form with the POST method is invisible to others and has no limits on the amount of information to send. However, there is an 8 Mb max size for the POST method, by default (can be changed by setting the post_max_size in the php.ini file).

Example

<form action="welcome.php" method="post">

Name: <input type="text" name="fname" />

Age: <input type="text" name="age" />

<input type="submit" />

</form>

When the user clicks the "Submit" button, the URL will look like this:

http://www.w3schools.com/welcome.php

The "welcome.php" file can now use the $_POST function to collect form data (the names of the form fields will automatically be the keys in the $_POST array):

Welcome <?php echo $_POST["fname"]; ?>!

You are <?php echo $_POST["age"]; ?> years old.

When to use method="post"

Information sent from a form with the POST method is invisible to others and has no limits on the amount of information to send. However, because the variables are not displayed in the URL, it is not possible to bookmark the page.

The PHP $_REQUEST Function

The PHP built-in $_REQUEST function contains the contents of both $_GET, $_POST, and $_COOKIE. The $_REQUEST function can be used to collect form data sent with both the GET and POST methods.

Example

Welcome <?php echo $_REQUEST["fname"]; ?>!

You are <?php echo $_REQUEST["age"]; ?> years old.

Chapter Review Questions

1. What is PHP?
2. Write a simple PHP code to display "hello world" on the browser
3. Explain why PHP is a loosely typed language
4. Giving examples name all the conditional statements found in PHP
5. Giving examples name all the loops found in PHP
6. differentiate between PHP $_GET and PHP $_POST

Suggested Further Reading

1. Mark S et al (1996), Special Edition using internet HTML Que Publishing Co ltd
2. Alex H et all(2000),, Professional active server Wrox publishers programmer to programmer series
3. http://www.w3schools.com

CHAPTER SEVEN: MYSQL DATABASE AND PHP

Learning Objectives:

By the end of this chapter the learner shall be able to;

vii. Explain what is MySQL

viii. Create database and tables using PHP

ix. Insert, edit and delete data in MySQL database

7.1 Introduction MySQL

MySQL is a relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation.

Free-software-open source projects that require a full-featured database management system often use MySQL. For commercial use, several paid editions are available, and offer additional functionality. Applications which use MySQL databases include: Joomla, WordPress, MyBB, phpBB, Drupal and other software built on the LAMP software stack. MySQL is also used in many high-profile, large-scale World Wide Web products, including Wikipedia, Google and Facebook.

MySQL is a database server that is ideal for both small and large applications. MySQL supports standard SQL and compiles on a number of platforms. MySQL is the most popular open-source database system. The data in MySQL is stored in database objects called tables. A table is a collection of related data entries and it consists of columns and rows. Databases are useful when storing information categorically. A company may have a database with the following tables: "Employees", "Products", "Customers" and "Orders".

Database Tables

A database most often contains one or more tables. Each table is identified by a name (e.g. "Customers" or "Orders"). Tables contain records (rows) with data. Below is an example of a table called "Persons":

LastName	FirstName	Address	City
Hansen	Ola	Timoteivn 10	Sandnes
Svendson	Tove	Borgvn 23	Sandnes
Pettersen	Kari	Storgt 20	Stavanger

The table above contains three records (one for each person) and four columns (LastName, FirstName, Address, and City).

Queries

A query is a question or a request. With MySQL, we can query a database for specific information and have a recordset returned. Look at the following query:

SELECT LastName FROM Persons

The query above selects all the data in the "LastName" column from the "Persons" table, and will return a recordset like this:

LastName
Hansen
Svendson
Pettersen

7.2 PHP MySQL Connect to a Database

The free MySQL database is very often used with PHP.

Create a Connection to a MySQL Database

Before you can access data in a database, you must create a connection to the database. In PHP, this is done with the mysql_connect() function.

Syntax

mysql_connect(servername,username,password);

Parameter	Description
servername	Optional. Specifies the server to connect to. Default value is "localhost:3306"

username	Optional. Specifies the username to log in with. Default value is the name of the user that owns the server process
password	Optional. Specifies the password to log in with. Default is ""

In the following example we store the connection in a variable ($con) for later use in the script. The "die" part will be executed if the connection fails:

```
<?php

$con = mysql_connect("localhost","peter","abc123");

if (!$con)

  {

  die('Could not connect: ' . mysql_error());

  }

// some code

?>
```

Closing a Connection

The connection will be closed automatically when the script ends. To close the connection before, use the mysql_close() function:

```
<?php

$con = mysql_connect("localhost","peter","abc123");

if (!$con)

  {

  die('Could not connect: ' . mysql_error());

  }
```

```
// some code

mysql_close($con);

?>
```

7.3 PHP MySQL Create Database and Tables

A database holds one or multiple tables.

Create a Database

The CREATE DATABASE statement is used to create a database in MySQL.

Syntax

```
CREATE DATABASE database_name
```

To get PHP to execute the statement above we must use the mysql_query() function. This function is used to send a query or command to a MySQL connection.

Example

The following example creates a database called "my_db":

```
<?php

$con = mysql_connect("localhost","peter","abc123");

if (!$con)

  {

  die('Could not connect: ' . mysql_error());

  }

if (mysql_query("CREATE DATABASE my_db",$con))

  {
```

```
 echo "Database created";
 }
else
 {
 echo "Error creating database: " . mysql_error();
 }
mysql_close($con);
?>
```

Create a Table

The CREATE TABLE statement is used to create a table in MySQL.

Syntax

```
CREATE TABLE table_name
(
column_name1 data_type,
column_name2 data_type,
column_name3 data_type,
....
)
```

We must add the CREATE TABLE statement to the mysql_query() function to execute the command.

The following example creates a table named "Persons", with three columns. The column names will be "FirstName", "LastName" and "Age":

```
<?php
$con = mysql_connect("localhost","peter","abc123");
if (!$con)
  {
  die('Could not connect: ' . mysql_error());
  }
// Create database
if (mysql_query("CREATE DATABASE my_db",$con))
  {
  echo "Database created";
  }
else
  {
  echo "Error creating database: " . mysql_error();
  }
// Create table
mysql_select_db("my_db", $con);
$sql = "CREATE TABLE Persons
(
FirstName varchar(15),
LastName varchar(15),
```

```
Age int

)";

// Execute query

mysql_query($sql,$con);

mysql_close($con);

?>
```

A database must be selected before a table can be created. The database is selected with the mysql_select_db() function. When you create a database field of type varchar, you must specify the maximum length of the field, e.g. varchar(15). The data type specifies what type of data the column can hold.

Primary Keys and Auto Increment Fields

Each table should have a primary key field. A primary key is used to uniquely identify the rows in a table. Each primary key value must be unique within the table. Furthermore, the primary key field cannot be null because the database engine requires a value to locate the record.

The following example sets the personID field as the primary key field. The primary key field is often an ID number, and is often used with the AUTO_INCREMENT setting. AUTO_INCREMENT automatically increases the value of the field by 1 each time a new record is added. To ensure that the primary key field cannot be null, we must add the NOT NULL setting to the field.

```
$sql = "CREATE TABLE Persons

(

personID int NOT NULL AUTO_INCREMENT,

PRIMARY KEY(personID),

FirstName varchar(15),

LastName varchar(15),
```

```
Age int

)";

mysql_query($sql,$con);
```

7.4 PHP MySQL Insert Into

The INSERT INTO statement is used to insert new records in a table. The INSERT INTO statement is used to add new records to a database table.

Syntax

It is possible to write the INSERT INTO statement in two forms.

1. The first form doesn't specify the column names where the data will be inserted, only their values:

   INSERT INTO table_name

   VALUES (value1, value2, value3,...)

2. The second form specifies both the column names and the values to be inserted:

   INSERT INTO table_name (column1, column2, column3,...)

   VALUES (value1, value2, value3,...)

To get PHP to execute the statements above we must use the mysql_query() function. This function is used to send a query or command to a MySQL connection.

Example

The following example adds two new records to the "Persons" table:

```
<?php

$con = mysql_connect("localhost","peter","abc123");

if (!$con)

 {

 die('Could not connect: ' . mysql_error());

 }
```

```
mysql_select_db("my_db", $con);

mysql_query("INSERT INTO Persons (FirstName, LastName, Age)

VALUES ('Peter', 'Griffin', '35')");

mysql_query("INSERT INTO Persons (FirstName, LastName, Age)

VALUES ('Glenn', 'Quagmire', '33')");

mysql_close($con);

?>
```

Insert Data From a Form Into a Database

Now we will create an HTML form that can be used to add new records to the "Persons" table. Here is the HTML form (save as ‘form.html’):

```
<html>

<body>

<form action="insert.php" method="post">

Firstname: <input type="text" name="firstname" />

Lastname: <input type="text" name="lastname" />

Age: <input type="text" name="age" />

<input type="submit" />

</form>

</body>

</html>
```

When a user clicks the submit button in the HTML form in the example above, the form data is sent to "insert.php". The "insert.php" file connects to a database, and retrieves the values from the form with the PHP $_POST variables. Then, the mysql_query() function executes the INSERT INTO statement, and a new record will be added to the "Persons" table.

Here is the "insert.php" page(save as ‘insert.php):

```
<?php
$con = mysql_connect("localhost","peter","abc123");
if (!$con)
  {
  die('Could not connect: ' . mysql_error());
  }
mysql_select_db("my_db", $con);
$sql="INSERT INTO Persons (FirstName, LastName, Age)
VALUES
('$_POST[firstname]','$_POST[lastname]','$_POST[age]')";
if (!mysql_query($sql,$con))
  {
  die('Error: ' . mysql_error());
  }
echo "1 record added";
mysql_close($con)
?>
```

7.5 PHP MySQL Select

The SELECT statement is used to select data from a database.

Syntax

```
SELECT column_name(s)
FROM table_name
```

To get PHP to execute the statement above we must use the mysql_query() function. This function is used to send a query or command to a MySQL connection. The following example selects all the data stored in the "Persons" table (The * character selects all the data in the table):

```
<?php
$con = mysql_connect("localhost","peter","abc123");
if (!$con)
  {
  die('Could not connect: ' . mysql_error());
  }
mysql_select_db("my_db", $con);
$result = mysql_query("SELECT * FROM Persons");
while($row = mysql_fetch_array($result))
  {
  echo $row['FirstName'] . " " . $row['LastName'];
  echo "<br />";
  }
mysql_close($con);
?>
```

The example above stores the data returned by the mysql_query() function in the $result variable. Next, we use the mysql_fetch_array() function to return the first row from the recordset as an array. Each call to mysql_fetch_array() returns the next row in the recordset. The while loop loops through all the records in the recordset. To print the value of each row, we use the PHP $row variable ($row['FirstName'] and $row['LastName']).

Display the Result in an HTML Table

The following example selects the same data as the example above, but will display the data in an HTML table:

```
<?php
$con = mysql_connect("localhost","peter","abc123");
```

```
if (!$con)
  {
  die('Could not connect: ' . mysql_error());
  }
mysql_select_db("my_db", $con);
$result = mysql_query("SELECT * FROM Persons");
echo "<table border='1'>
<tr>
<th>Firstname</th>
<th>Lastname</th>
</tr>";
while($row = mysql_fetch_array($result))
  {
  echo "<tr>";
  echo "<td>" . $row['FirstName'] . "</td>";
  echo "<td>" . $row['LastName'] . "</td>";
  echo "</tr>";
  }
echo "</table>";
mysql_close($con);
?>
```

7.6 PHP MySQL The Where Clause

The WHERE clause is used to filter records. The WHERE clause is used to extract only those records that fulfill a specified criterion.

Syntax

SELECT column_name(s)

FROM table_name

WHERE column_name operator value

To get PHP to execute the statement above we must use the mysql_query() function. This function is used to send a query or command to a MySQL connection. The following example selects all rows from the "Persons" table where "FirstName='Peter':

```
<?php
$con = mysql_connect("localhost","peter","abc123");
if (!$con)
  {
  die('Could not connect: ' . mysql_error());
  }

mysql_select_db("my_db", $con);
$result = mysql_query("SELECT * FROM Persons
WHERE FirstName='Peter'");
while($row = mysql_fetch_array($result))
  {
  echo $row['FirstName'] . " " . $row['LastName'];
  echo "<br />";
  }
?>
```

7.7 PHP MySQL The ORDER BY Keyword

The ORDER BY keyword is used to sort the data in a recordset. The ORDER BY keyword sort the records in ascending order by default. If you want to sort the records in a descending order, you can use the DESC keyword.

Syntax

```
SELECT column_name(s)
FROM table_name
ORDER BY column_name(s) ASC|DESC
```

The following example selects all the data stored in the "Persons" table, and sorts the result by the "Age" column:

```
<?php
$con = mysql_connect("localhost","peter","abc123");
if (!$con)
 {
 die('Could not connect: ' . mysql_error());
 }
mysql_select_db("my_db", $con);
$result = mysql_query("SELECT * FROM Persons ORDER BY age");
while($row = mysql_fetch_array($result))
 {
 echo $row['FirstName'];
 echo " " . $row['LastName'];
 echo " " . $row['Age'];
 echo "<br />";
 }
```

```
mysql_close($con);

?>
```

Order by Two Columns

It is also possible to order by more than one column. When ordering by more than one column, the second column is only used if the values in the first column are equal:

```
SELECT column_name(s)

FROM table_name

ORDER BY column1, column2
```

7.8 PHP MySQL Update

The UPDATE statement is used to modify data in a table. The UPDATE statement is used to update existing records in a table.

Syntax

```
UPDATE table_name

SET column1=value, column2=value2,...

WHERE some_column=some_value
```

Note: Notice the WHERE clause in the UPDATE syntax. The WHERE clause specifies which record or records that should be updated. If you omit the WHERE clause, all records will be updated!

To get PHP to execute the statement above we must use the mysql_query() function. This function is used to send a query or command to a MySQL connection.

The following example updates some data in the "Persons" table:

```
<?php

$con = mysql_connect("localhost","peter","abc123");

if (!$con)

  {

  die('Could not connect: ' . mysql_error());
```

```
 }

mysql_select_db("my_db", $con);

mysql_query("UPDATE Persons SET Age = '36'

WHERE FirstName = 'Peter' AND LastName = 'Griffin'");

mysql_close($con);

?>
```

7.9 PHP MySQL Delete

The DELETE statement is used to delete records in a table. The DELETE FROM statement is used to delete records from a database table.

Syntax

```
DELETE FROM table_name

WHERE some_column = some_value
```

Note: Notice the WHERE clause in the DELETE syntax. The WHERE clause specifies which record or records that should be deleted. If you omit the WHERE clause, all records will be deleted!

To get PHP to execute the statement above we must use the mysql_query() function. This function is used to send a query or command to a MySQL connection.

Example

The following example deletes all the records in the "Persons" table where LastName='Griffin':

```
<?php

$con = mysql_connect("localhost","peter","abc123");

if (!$con)

 {

 die('Could not connect: ' . mysql_error());

 }

mysql_select_db("my_db", $con);
```

```
mysql_query("DELETE FROM Persons WHERE LastName='Griffin'");

mysql_close($con);

?>
```


Chapter Review Questions

1. What is MySQL
2. Write PHP code to connect to a database called “MKU_DB” in the server named “www.mku.ac.ke” username is "admin" and the password is "MKU"
3. Write PHP code to close a database connection

Suggested Further Reading

1. Mark S et al (1996), Special Edition using internet HTML Que Publishing Co ltd
2. Alex H et all(2000),, Professional active server Wrox publishers programmer to programmer series
3. http://www.w3schools.com

CHAPTER EIGHT: ADVANCED PHP

Learning Objectives:

By the end of this chapter the learner shall be able to;

x. Explain the date function

xi. use PHP include file and PHP file handling

xii. Create cookies and sessions

8.1 PHP Date() Function

The PHP date() function is used to format a time and/or date. The PHP date() function formats a timestamp to a more readable date and time. A timestamp is a sequence of characters, denoting the date and/or time at which a certain event occurred.

Syntax

date(format,timestamp)

Parameter	**Description**
format	Required. Specifies the format of the timestamp
timestamp	Optional. Specifies a timestamp. Default is the current date and time

PHP Date() - Format the Date

The required format parameter in the date() function specifies how to format the date/time. Here are some characters that can be used:

- d - Represents the day of the month (01 to 31)
- m - Represents a month (01 to 12)
- Y - Represents a year (in four digits)

Other characters, like"/", ".", or "-" can also be inserted between the letters to add additional formatting:

```
<?php

echo date("Y/m/d") . "<br />";

echo date("Y.m.d") . "<br />";

echo date("Y-m-d");

?>
```

PHP Date() - Adding a Timestamp

The optional timestamp parameter in the date() function specifies a timestamp. If you do not specify a timestamp, the current date and time will be used. The mktime() function returns the Unix timestamp for a date. The Unix timestamp contains the number of seconds between the Unix Epoch (January 1 1970 00:00:00 GMT) and the time specified.

Syntax for mktime()

mktime(hour,minute,second,month,day,year,is_dst)

To go one day in the future we simply add one to the day argument of mktime():

```
<?php

$tomorrow = mktime(0,0,0,date("m"),date("d")+1,date("Y"));

echo "Tomorrow is ".date("Y/m/d", $tomorrow);

?>
```

8.2 PHP Include File

Server Side Includes (SSI)

You can insert the content of one PHP file into another PHP file before the server executes it, with the include() or require() function. The two functions are identical in every way, except how they handle errors:

- include() generates a warning, but the script will continue execution
- require() generates a fatal error, and the script will stop

These two functions are used to create functions, headers, footers, or elements that will be reused on multiple pages. Server side includes saves a lot of work. This means that you can create a standard

header, footer, or menu file for all your web pages. When the header needs to be updated, you can only update the include file, or when you add a new page to your site, you can simply change the menu file (instead of updating the links on all your web pages).

PHP include() Function

The include() function takes all the content in a specified file and includes it in the current file. If an error occurs, the include() function generates a warning, but the script will continue execution.

Example 1

Assume that you have a standard header file, called "header.php". To include the header file in a page, use the include() function:

```
<html>

<body>

<?php include("header.php"); ?>

<h1>Welcome to my home page!</h1>

<p>Some text.</p>

</body>

</html>
```

Example 2

Assume we have a standard menu file, called "menu.php", that should be used on all pages:

```
<a href="/default.php">Home</a>

<a href="/tutorials.php">Tutorials</a>

<a href="/references.php">References</a>

<a href="/examples.php">Examples</a>

<a href="/about.php">About Us</a>

<a href="/contact.php">Contact Us</a>
```

All pages in the Web site should include this menu file. Here is how it can be done:

```
<html>

<body>

<div class="leftmenu">

<?php include("menu.php"); ?>

</div>

<h1>Welcome to my home page.</h1>

<p>Some text.</p>

</body>

</html>
```

If you look at the source code of the page above (in a browser), it will look like this:

```
<html>

<body>

<div class="leftmenu">

<a href="/default.php">Home</a>

<a href="/tutorials.php">Tutorials</a>

<a href="/references.php">References</a>

<a href="/examples.php">Examples</a>

<a href="/about.php">About Us</a>

<a href="/contact.php">Contact Us</a>

</div>

<h1>Welcome to my home page!</h1>

<p>Some text.</p>
```

</body>

</html>

PHP require() Function

The require() function is identical to include(), except that it handles errors differently. If an error occurs, the include() function generates a warning, but the script will continue execution. The require() generates a fatal error, and the script will stop.

Error Example include() Function

```
<html>

<body>

<?php

include("wrongFile.php");

echo "Hello World!";

?>

</body>

</html>
```

Error message:

Warning: include(wrongFile.php) [function.include]:

failed to open stream:

No such file or directory in C:\home\website\test.php on line 5

Warning: include() [function.include]:

Failed opening 'wrongFile.php' for inclusion

(include_path='.;C:\php5\pear')

in C:\home\website\test.php on line 5

Hello World!

Notice that the echo statement is executed! This is because a Warning does not stop the script execution.

Error Example require() Function

Now, let's run the same example with the require() function.

```
<html>
<body>
<?php
require("wrongFile.php");
echo "Hello World!";
?>
</body>
</html>
```

Error message:

Warning: require(wrongFile.php) [function.require]:
failed to open stream:
No such file or directory in C:\home\website\test.php on line 5

Fatal error: require() [function.require]:
Failed opening required 'wrongFile.php'
(include_path='.;C:\php5\pear')
in C:\home\website\test.php on line 5

The echo statement is not executed, because the script execution stopped after the fatal error. It is recommended to use the require() function instead of include(), because scripts should not continue after an error.

8.3 PHP File Handling

The fopen() function is used to open files in PHP.

Opening a File

The fopen() function is used to open files in PHP. The first parameter of this function contains the name of the file to be opened and the second parameter specifies in which mode the file should be opened:

```
<html>

<body>

<?php
$file=fopen("welcome.txt","r");
?>

</body>

</html>
```

The file may be opened in one of the following modes:

Modes	**Description**
r	Read only. Starts at the beginning of the file
r+	Read/Write. Starts at the beginning of the file
w	Write only. Opens and clears the contents of file; or creates a new file if it doesn't exist
w+	Read/Write. Opens and clears the contents of file; or creates a new file if it doesn't exist
a	Append. Opens and writes to the end of the file or creates a new file if it doesn't exist
a+	Read/Append. Preserves file content by writing to the end of the file
x	Write only. Creates a new file. Returns FALSE and an error if file already exists
x+	Read/Write. Creates a new file. Returns FALSE and an error if file already exists

Note: If the fopen() function is unable to open the specified file, it returns 0 (false).

Example

The following example generates a message if the fopen() function is unable to open the specified file:

```
<html>

<body>

<?php

$file=fopen("welcome.txt","r") or exit("Unable to open file!");

?>

</body>

</html>
```

Closing a File

The fclose() function is used to close an open file:

```
<?php

$file = fopen("test.txt","r");

//some code to be executed

fclose($file);

?>
```

Check End-of-file

The feof() function checks if the "end-of-file" (EOF) has been reached. The feof() function is useful for looping through data of unknown length.

Note: You cannot read from files opened in w, a, and x mode!

```
if (feof($file)) echo "End of file";
```

Reading a File Line by Line

The fgets() function is used to read a single line from a file.

Note: After a call to this function the file pointer has moved to the next line.

Example

The example below reads a file line by line, until the end of file is reached:

```
<?php
$file = fopen("welcome.txt", "r") or exit("Unable to open file!");
//Output a line of the file until the end is reached
while(!feof($file))
  {
  echo fgets($file). "<br />";
  }
fclose($file);
?>
```

Reading a File Character by Character

The fgetc() function is used to read a single character from a file.

Note: After a call to this function the file pointer moves to the next character.

Example

The example below reads a file character by character, until the end of file is reached:

```
<?php
$file=fopen("welcome.txt","r") or exit("Unable to open file!");
while (!feof($file))
  {
  echo fgetc($file);
  }
fclose($file);
?>
```

8.4 PHP File Upload

With PHP, it is possible to upload files to the server.

Create an Upload-File Form

To allow users to upload files from a form can be very useful. Look at the following HTML form for uploading files:

```
<html>

<body>

<form action="upload_file.php" method="post"

enctype="multipart/form-data">

<label for="file">Filename:</label>

<input type="file" name="file" id="file" />

<br />

<input type="submit" name="submit" value="Submit" />

</form>

</body>

</html>
```

Notice the following about the HTML form above:

- The **enctype attribute** of the <form> tag specifies which content-type to use when submitting the form. "multipart/form-data" is used when a form requires binary data, like the contents of a file, to be uploaded
- The **type="file" attribute** of the <input> tag specifies that the input should be processed as a file. For example, when viewed in a browser, there will be a browse-button next to the input field

Note: Allowing users to upload files is a big security risk. Only permit trusted users to perform file uploads.

Create The Upload Script

The "upload_file.php" file contains the code for uploading a file:

```
<?php
```

```
if ($_FILES["file"]["error"] > 0)
  {
  echo "Error: " . $_FILES["file"]["error"] . "<br />";
  }
else
  {
  echo "Upload: " . $_FILES["file"]["name"] . "<br />";
  echo "Type: " . $_FILES["file"]["type"] . "<br />";
  echo "Size: " . ($_FILES["file"]["size"] / 1024) . " Kb<br />";
  echo "Stored in: " . $_FILES["file"]["tmp_name"];
  }
?>
```

By using the global PHP $_FILES array you can upload files from a client computer to the remote server. The first parameter is the form's input name and the second index can be either "name", "type", "size", "tmp_name" or "error". Like this:

- $_FILES["file"]["name"] - the name of the uploaded file
- $_FILES["file"]["type"] - the type of the uploaded file
- $_FILES["file"]["size"] - the size in bytes of the uploaded file
- $_FILES["file"]["tmp_name"] - the name of the temporary copy of the file stored on the server
- $_FILES["file"]["error"] - the error code resulting from the file upload

This is a very simple way of uploading files. For security reasons, you should add restrictions on what the user is allowed to upload.

Restrictions on Upload

In this script we add some restrictions to the file upload. The user may only upload .gif or .jpeg files and the file size must be under 20 kb:

```
<?php
if (($_FILES["file"]["type"] == "image/gif")
|| ($_FILES["file"]["type"] == "image/jpeg")
|| ($_FILES["file"]["type"] == "image/pjpeg"))
&& ($_FILES["file"]["size"] < 20000))
 {
 if ($_FILES["file"]["error"] > 0)
   {
   echo "Error: " . $_FILES["file"]["error"] . "<br />";
   }
 else
   {
   echo "Upload: " . $_FILES["file"]["name"] . "<br />";
   echo "Type: " . $_FILES["file"]["type"] . "<br />";
   echo "Size: " . ($_FILES["file"]["size"] / 1024) . " Kb<br />";
   echo "Stored in: " . $_FILES["file"]["tmp_name"];
   }
 }
else
 {
 echo "Invalid file";
 }
```

```
?>
```

Note: For IE to recognize jpg files the type must be pjpeg, for FireFox it must be jpeg.

Saving the Uploaded File

The examples above create a temporary copy of the uploaded files in the PHP temp folder on the server. The temporary copied files disappears when the script ends. To store the uploaded file we need to copy it to a different location:

```
<?php
if ((($_FILES["file"]["type"] == "image/gif")
|| ($_FILES["file"]["type"] == "image/jpeg")
|| ($_FILES["file"]["type"] == "image/pjpeg"))
&& ($_FILES["file"]["size"] < 20000))
 {
 if ($_FILES["file"]["error"] > 0)
  {
  echo "Return Code: " . $_FILES["file"]["error"] . "<br />";
  }
 else
  {
  echo "Upload: " . $_FILES["file"]["name"] . "<br />";
  echo "Type: " . $_FILES["file"]["type"] . "<br />";
  echo "Size: " . ($_FILES["file"]["size"] / 1024) . " Kb<br />";
  echo "Temp file: " . $_FILES["file"]["tmp_name"] . "<br />";
  if (file_exists("upload/" . $_FILES["file"]["name"]))
    {
```

```
    echo $_FILES["file"]["name"] . " already exists. ";
    }
  else
    {
    move_uploaded_file($_FILES["file"]["tmp_name"],
    "upload/" . $_FILES["file"]["name"]);
    echo "Stored in: " . "upload/" . $_FILES["file"]["name"];
    }
  }
 }
else
 {
 echo "Invalid file";
 }
?>
```

The script above checks if the file already exists, if it does not, it copies the file to the specified folder.

Note: This example saves the file to a new folder called "upload"

8.5 PHP Cookies

A cookie is often used to identify a user. A cookie is often used to identify a user. A cookie is a small file that the server embeds on the user's computer. Each time the same computer requests a page with a browser, it will send the cookie too. With PHP, you can both create and retrieve cookie values.

How to Create a Cookie?

The setcookie() function is used to set a cookie.

Note: The setcookie() function must appear BEFORE the <html> tag.

Syntax

setcookie(name, value, expire, path, domain);

Example 1

In the example below, we will create a cookie named "user" and assign the value "Alex Porter" to it. We also specify that the cookie should expire after one hour:

```
<?php
setcookie("user", "Alex Porter", time()+3600);
?>
<html>
.....
```

Note: The value of the cookie is automatically URLencoded when sending the cookie, and automatically decoded when received (to prevent URLencoding, use setrawcookie() instead).

Example 2

You can also set the expiration time of the cookie in another way. It may be easier than using seconds.

```
<?php
$expire=time()+60*60*24*30;
setcookie("user", "Alex Porter", $expire);
?>
<html>
.....
```

In the example above the expiration time is set to a month (60 sec * 60 min * 24 hours * 30 days).

How to Retrieve a Cookie Value?

The PHP $_COOKIE variable is used to retrieve a cookie value.

In the example below, we retrieve the value of the cookie named "user" and display it on a page:

```
<?php
```

```
// Print a cookie

echo $_COOKIE["user"];

// A way to view all cookies

print_r($_COOKIE);

?>
```

In the following example we use the isset() function to find out if a cookie has been set:

```
<html>

<body>

<?php

if (isset($_COOKIE["user"]))

  echo "Welcome " . $_COOKIE["user"] . "!<br />";

else

  echo "Welcome guest!<br />";

?>

</body>

</html>
```

How to Delete a Cookie?

When deleting a cookie you should assure that the expiration date is in the past.

Delete example:

```
<?php

// set the expiration date to one hour ago

setcookie("user", "", time()-3600);

?>
```

What if a Browser Does NOT Support Cookies?

If your application deals with browsers that do not support cookies, you will have to use other methods to pass information from one page to another in your application. One method is to pass the data through forms. The form below passes the user input to "welcome.php" when the user clicks on the "Submit" button:

```
<html>

<body>

<form action="welcome.php" method="post">

Name: <input type="text" name="name" />

Age: <input type="text" name="age" />

<input type="submit" />

</form>

</body>

</html>
```

Retrieve the values in the "welcome.php" file like this:

```
<html>

<body>

Welcome <?php echo $_POST["name"]; ?>.<br />

You are <?php echo $_POST["age"]; ?> years old.

</body>

</html>
```

8.6 PHP Sessions

A PHP session variable is used to store information about, or change settings for a user session. Session variables hold information about one single user, and are available to all pages in one application.

PHP Session Variables

When you are working with an application, you open it, do some changes and then you close it. This is much like a Session. The computer knows who you are. It knows when you start the application and when you end. But on the internet there is one problem: the web server does not know who you are and what you do because the HTTP address doesn't maintain state. A PHP session solves this problem by allowing you to store user information on the server for later use (i.e. username, shopping items, etc). However, session information is temporary and will be deleted after the user has left the website. If you need a permanent storage you may want to store the data in a database. Sessions work by creating a unique id (UID) for each visitor and store variables based on this UID. The UID is either stored in a cookie or is propagated in the URL.

Starting a PHP Session

Before you can store user information in your PHP session, you must first start up the session.

Note: The session_start() function must appear BEFORE the <html> tag:

```
<?php session_start(); ?>

<html>

<body>

</body>

</html>
```

The code above will register the user's session with the server, allow you to start saving user information, and assign a UID for that user's session.

Storing a Session Variable

The correct way to store and retrieve session variables is to use the PHP $_SESSION variable:

```
<?php

session_start();

// store session data

$_SESSION['views']=1;

?>

<html>
```

```
<body>

<?php

//retrieve session data

echo "Pageviews=". $_SESSION['views'];

?>

</body>

</html>
```

In the example below, we create a simple page-views counter. The isset() function checks if the "views" variable has already been set. If "views" has been set, we can increment our counter. If "views" doesn't exist, we create a "views" variable, and set it to 1:

```
<?php

session_start();

if(isset($_SESSION['views']))

$_SESSION['views']=$_SESSION['views']+1;

else

$_SESSION['views']=1;

echo "Views=". $_SESSION['views'];

?>
```

Destroying a Session

If you wish to delete some session data, you can use the unset() or the session_destroy() function. The unset() function is used to free the specified session variable:

```
<?php

unset($_SESSION['views']);

?>
```

You can also completely destroy the session by calling the session_destroy() function:

```
<?php

session_destroy();

?>
```

Note: session_destroy() will reset your session and you will lose all your stored session data.

8.7 PHP Sending E-mails

PHP allows you to send e-mails directly from a script.

The PHP mail() Function

The PHP mail() function is used to send emails from inside a script.

Syntax

mail(to,subject,message,headers,parameters)

Parameter	Description
to	Required. Specifies the receiver / receivers of the email
subject	Required. Specifies the subject of the email. **Note:** This parameter cannot contain any newline characters
message	Required. Defines the message to be sent. Each line should be separated with a LF (\n). Lines should not exceed 70 characters
headers	Optional. Specifies additional headers, like From, Cc, and Bcc. The additional headers should be separated with a CRLF (\r\n)
parameters	Optional. Specifies an additional parameter to the sendmail program

Note: For the mail functions to be available, PHP requires an installed and working email system. The program to be used is defined by the configuration settings in the php.ini file. Read more in our PHP Mail reference.

PHP Simple E-Mail

The simplest way to send an email with PHP is to send a text email. In the example below we first declare the variables ($to, $subject, $message, $from, $headers), then we use the variables in the mail() function to send an e-mail:

```
<?php

$to = "someone@example.com";
```

```
$subject = "Test mail";

$message = "Hello! This is a simple email message.";

$from = "someonelse@example.com";

$headers = "From:" . $from;

mail($to,$subject,$message,$headers);

echo "Mail Sent.";

?>
```

PHP Mail Form

With PHP, you can create a feedback-form on your website. The example below sends a text message to a specified e-mail address:

```
<html>

<body>

<?php

if (isset($_REQUEST['email']))

//if "email" is filled out, send email

  {

  //send email

  $email = $_REQUEST['email'] ;

  $subject = $_REQUEST['subject'] ;

  $message = $_REQUEST['message'] ;

  mail("someone@example.com", "$subject",

  $message, "From:" . $email);

  echo "Thank you for using our mail form";

  }
```

```
else

//if "email" is not filled out, display the form

 {

 echo "<form method='post' action='mailform.php'>

 Email: <input name='email' type='text' /><br />

 Subject: <input name='subject' type='text' /><br />

 Message:<br />

 <textarea name='message' rows='15' cols='40'>

 </textarea><br />

 <input type='submit' />

 </form>";

 }

?>

</body>

</html>
```

This is how the example above works:

- First, check if the email input field is filled out
- If it is not set (like when the page is first visited); output the HTML form
- If it is set (after the form is filled out); send the email from the form
- When submit is pressed after the form is filled out, the page reloads, sees that the email input is set, and sends the email

Chapter Review Questions

1. Explain what is a PHP session
2. Explain what is a PHP cookie

3. Write a code to create a PHP cookie
4. Write a code to upload a file to a server
5. write the code to include "functions.php" into "home.php" page

Suggested Further Reading

1. Mark S et al (1996), Special Edition using internet HTML Que Publishing Co ltd
2. Alex H et all(2000),, Professional active server Wrox publishers programmer to programmer series
3. http://www.w3schools.com

CHAPTER NINE: MULTIMEDIA WEBSITES

Learning Objectives:

By the end of this chapter the learner shall be able to;

i. Explain the voice and video multimedia formats

ii. adding multimedia to a webpage

iii. explain the windows multimedia formats

9.1 Introduction to Multimedia Websites

Multimedia is pictures, sound, music, animations, and videos. Modern web browsers have support for many multimedia formats. Multimedia is everything you can hear or see: texts, books, pictures, music, sounds, CDs, videos, DVDs, Records, Films, and more. Multimedia comes in many different formats. On the Internet you will find many of these elements embedded in web pages, and today's web browsers have support for a number of multimedia formats.

Browser Support

The first Internet browsers had support for text only, and even the text support was limited to a single font in a single color, and little or nothing else. Then came web browsers with support for colors, fonts and text styles, and the support for pictures was added. The support for sounds, animations and videos is handled in different ways by different browsers. Some elements can be handled inline, some requires a plug-in and some requires an ActiveX control.

9.2 Multimedia Formats

Multimedia elements (like sounds or videos) are stored in media files. The most common way to discover the media type is to look at the file extension. When a browser sees the file extensions .htm or .html, it will assume that the file is an HTML page. The .xml extension indicates an XML file, and the .css extension indicates a style sheet. Picture formats are recognized by extensions like .gif and .jpg. Multimedia elements also have their own file formats with different extensions.

Multimedia Sound Formats

Sound can be stored in the following formats:

1. ***The MIDI Format*** - The MIDI (Musical Instrument Digital Interface) is a format for sending music information between electronic music devices like synthesizers and PC sound cards.The MIDI format was developed in 1982 by the music industry. The MIDI format is very flexible and can be used for everything from very simple to real professional music making. MIDI files do not contain sampled sound, but a set of digital musical instructions (musical notes) that can be interpreted by your PC's sound card. The downside of MIDI is that it cannot record sounds (only notes). Or, to put it another way: It cannot store songs, only tunes. The upside of the MIDI format is that since it contains only instructions (notes), MIDI files can be extremely small. The MIDI format is supported by many different software systems over a large range of platforms. MIDI files are supported by all the most popular Internet browsers. Sounds stored in the MIDI format have the extension .mid or .midi.

2. ***The RealAudio Format*** - The RealAudio format was developed for the Internet by Real Media. The format also supports video. The format allows streaming of audio (on-line music, Internet radio) with low bandwidths. Because of the low bandwidth priority, quality is often reduced. Sounds stored in the RealAudio format have the extension .rm or .ram.

3. ***The AU Format*** - The AU format is supported by many different software systems over a large range of platforms. Sounds stored in the AU format have the extension .au.

4. ***The AIFF Format*** - The AIFF (Audio Interchange File Format) was developed by Apple. AIFF files are not cross-platform and the format is not supported by all web browsers. Sounds stored in the AIFF format have the extension .aif or .aiff.

5. ***The SND Format*** - The SND (Sound) was developed by Apple. SND files are not cross-platform and the format is not supported by all web browsers. Sounds stored in the SND format have the extension .snd.

6. ***The WAVE Format*** - The WAVE (waveform) format is developed by IBM and Microsoft. It is supported by all computers running Windows, and by all the most popular web browsers (except Google Chrome). Sounds stored in the WAVE format have the extension .wav.

7. ***The MP3 Format (MPEG)*** - MP3 files are actually MPEG files. But the MPEG format was originally developed for video by the Moving Pictures Experts Group. We can say that MP3 files are the sound part of the MPEG video format. MP3 is one of the most popular sound formats for music recording. The MP3 encoding system combines good compression (small files) with high quality. Sounds stored in the MP3 format have the extension .mp3, or .mpga (for MPG Audio).

How to choose the sound format to use

The WAVE format is one of the most popular sound format on the Internet, and it is supported by all popular browsers. If you want recorded sound (music or speech) to be available to all your visitors, you should use the WAVE format. The MP3 format is the new and upcoming format for recorded music. If your website is about recorded music, the MP3 format is the choice of the future.

Multimedia Video Formats

Video can be stored in the following formats:

1. ***The AVI Format*** - The AVI (Audio Video Interleave) format was developed by Microsoft. The AVI format is supported by all computers running Windows, and by all the most popular web browsers. It is a very common format on the Internet, but not always possible to play on non-Windows computers. Videos stored in the AVI format have the extension .avi.

2. ***The Windows Media Format*** - The Windows Media format is developed by Microsoft. Windows Media is a common format on the Internet, but Windows Media movies cannot be played on non-Windows computer without an extra (free) component installed. Some later Windows Media movies cannot play at all on non-Windows computers because no player is available. Videos stored in the Windows Media format have the extension .wmv.

3. ***The MPEG Format*** - The MPEG (Moving Pictures Expert Group) format is the most popular format on the Internet. It is cross-platform, and supported by all the most popular web browsers. Videos stored in the MPEG format have the extension .mpg or .mpeg.

4. ***The QuickTime Format*** - The QuickTime format is developed by Apple. QuickTime is a common format on the Internet, but QuickTime movies cannot be played on a Windows computer without an extra (free) component installed. Videos stored in the QuickTime format have the extension .mov.

5. ***The RealVideo Format*** - The RealVideo format was developed for the Internet by Real Media. The format allows streaming of video (on-line video, Internet TV) with low bandwidths. Because of the low bandwidth priority, quality is often reduced. Videos stored in the RealVideo format have the extension .rm or .ram.

6. ***The Shockwave (Flash) Format***- The Shockwave format was developed by Macromedia. The Shockwave format requires an extra component to play. This component comes preinstalled with the latest versions of Netscape and Internet Explorer. Videos stored in the Shockwave format have the extension .swf.

9.3 Playing Sounds on a Web Site

Sounds can be played "inline" or by a "helper", depending on the HTML element you use.

1. ***Inline Sound***

When sound is included in a web page, or as part of a web page, it is called inline sound. If you plan to use inline sounds in your web applications, be aware that many people find inline sound annoying. Also note that some users might have turned off the inline sound option in their browser. Include inline sound only in web pages where the user expects to hear the sound. An example of this is a page which opens after the user has clicked on a link to hear a recording.

2. ***Using A Helper (Plug-In)***

A helper application is a program that can be launched by the browser to "help" playing sound. Helper applications are also called Plug-Ins. Helper applications can be launched using the <embed> or the <object> tag. One great advantage of using a helper application is that you can let some (or all) of the player settings be controlled by the user.Most helper applications allows manually (or programmed) control over the volume settings and play functions like rewind, pause, stop and play.

a. ***Using The <embed> Tag -*** The purpose of the <embed> tag is to embed multimedia elements in web page. The following code fragment displays a MIDI file embedded in a web page.

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

Note: The <embed> tag is deprecated. The World Wide Web Consortium (W3C) recommend using the <object> tag instead.

b. ***Using The <object> Tag*** - The purpose of the <object> tag is to embed multimedia elements in web page. The following code fragment displays a WAVE file embedded in a web page.

```
<object classid="clsid:22D6F312-B0F6-11D0-94AB-0080C74C7E95">

<param name="FileName" value="liar.wav" />

</object>
```

c. ***Using A Hyperlink -*** If a web page includes a hyperlink to a media file, most browsers will use a "helper application" to play the file. The following code fragment displays a link to a MIDI file. If a user clicks on the link, the browser will launch a helper application, like Windows Media Player to play the MIDI file:

```
<a href="beatles.mid">Play the Beatles</a>
```

9.4 Playing Videos on a Web Site

Videos can be played "inline" or by a "helper", depending on the HTML element you use.

1. ***Inline Videos*** **-** When a video is included in a web page it is called inline video. If you plan to use inline videos in your web applications, be aware that many people find inline videos annoying. Also note that some users might have turned off the inline video option in their browser. Instead include inline videos only in web pages where the user expects to see a video. An example of this is a page which opens after the user has clicked on a link to see the video.

2. ***Using A Helper (Plug-In)*** **-** A helper application is a program that can be launched by the browser to "help" playing a video. Helper applications are also called Plug-Ins. Helper applications can be launched using the <embed> or the <object> tag.One great advantage of using a helper application is that you can let some (or all) of the player settings be controlled by the user. Most helper applications allow manual (or programmed) control over the volume settings and play functions like rewind, pause, stop and play.

   a. ***Using The <embed> Tag*** - The purpose of the <embed> tag is to embed multimedia elements in web page. The following code fragment displays an AVI file embedded in a web page:

      <embed src="video.avi" />

   b. ***Using The <object> Tag*** - The purpose of the <object> tag is to embed multimedia elements in web page. The following code fragment displays an AVI file embedded in a web page: <

      object data="video.avi" type="video/avi" />

   c. ***Using A Hyperlink*** - If a web page includes a hyperlink to a media file, most browsers will use a "helper application" to play the file.The following code fragment displays a link to an AVI file. If a user clicks on the link, the browser will launch a helper application, like Windows Media Player to play the AVI file:

      Play a video file

9.5 The Object Element

The object element supports many different media types, like:

- Pictures

- Sounds
- Videos
- Other Objects

Displaying A Picture

You can display a picture as an object:

```
<object height="100%" width="100%"
type="image/jpeg" data="audi.jpeg">
</object>
```

Displaying A Web Page

You can display a web page as an object:

```
<object type="text/html" height="100%" width="100%"
data="http://www.w3schools.com">
</object>
```

Displaying A Sound

You can display a sound as an object:

```
<object
classid="clsid:22D6F312-B0F6-11D0-94AB-0080C74C7E95">
<param name="FileName" value="liar.wav" />
</object>
```

Displaying A Video

You can display a video as an object:

```
<object
classid="clsid:22D6F312-B0F6-11D0-94AB-0080C74C7E95">
<param name="FileName" value="3d.wmv" />
```

</object>

Displaying A Calendar

You can display a calendar as an object:

<object width="100%" height="80%"

classid="clsid:8E27C92B-1264-101C-8A2F-040224009C02">

<param name="BackColor" value="14544622">

<param name="DayLength" value="1">

</object>

Displaying Graphics

You can display graphics as an object:

<object width="200" height="200"

classid="CLSID:369303C2-D7AC-11D0-89D5-00A0C90833E6">

<param name="Line0001"

value="setFillColor(255, 0, 255)">

<param name="Line0002"

value="Oval(-100, -50, 200, 100, 30)">

</object>

Displaying Flash

You can display a flash animation as an object:

Example

<object width="400" height="40"

classid="clsid:d27cdb6e-ae6d-11cf-96b8-444553540000"

codebase="http://fpdownload.macromedia.com/

```
pub/shockwave/cabs/flash/swflash.cab#version=8,0,0,0">

<param name="SRC" value="bookmark.swf">

<embed src="bookmark.swf" width="400" height="40"></embed>

</object>
```

9.6 Windows Multimedia Formats

Windows media files have the extensions: .asf, .asx, .wma, and .wmv.

1. ***The ASF Format***

The ASF format (Advanced Streaming Format) is specially designed to run over the Internet. ASF files can contain audio, video, slide shows, and synchronized events. ASF files can be highly compressed and can be delivered as a continuous flow of data (on-line TV or radio). Files can be of any size, and can be compressed to match many different bandwidths (connection speeds).

2. ***The ASX Format***

ASX (Advanced Stream Redirector) files are not media files, but metafiles.Metafiles provides information about files. ASX files are plain text files used to describe multimedia content:

```
<ASX VERSION="3.0">
  <Title>Holiday 2001</Title>
  <Entry><ref href="holiday-1.avi"/></Entry>
  <Entry><ref href="holiday-2.avi"/></Entry>
  <Entry><ref href="holiday-2.avi"/></Entry>
</ASX>
```

The file above describes three multimedia files. When the ASX file is read by a player, the player can play the files described.

3. ***The WMA Format***

The WMA (Windows Media Audio) format is an audio format developed by Microsoft. WMA is designed to handle all types of audio content. The files can be highly compressed and can be delivered as a continuous flow of data (on-line radio). WMA files can be of any size, and be compressed to

match many different bandwidths (connection speeds). The WMA format is similar to the ASF format.

4. ***The WMV Format***

The WMV (Windows Media Video) format is a video format developed by Microsoft. WMV is designed to handle all types of video content. The files can be highly compressed and can be delivered as a continuous flow of data (on-line radio). WMV files can be of any size, and be compressed to match many different bandwidths (connection speeds). The WMV format is similar to the ASF format.

5. **Other Windows Media Formats**

WAX (Windows Media Audio Redirector) files are much the same as ASX files, but intended to describe audio files (.wma files). WMP (Windows Media Player) files and WMX are reserved file types for future use by Windows.

9.7 Playing QuickTime Movies

The object element can play QuickTime movies.

The QuickTime Format

The QuickTime format is developed by Apple. Videos stored in the QuickTime format have the extension .mov. QuickTime is a common format on the Internet, but QuickTime movies cannot be played on a Windows computer without an extra (free) component installed. With the object element, code that will play a QuickTime movie can easily be added to a web page. The object can be set to automatically install a QuickTime player if it is not already installed on the users computer.

This is the code required to play a QuickTime movie:

```
<object width="160" height="144" classid="clsid:02BF25D5-8C17-4B23-BC80-D3488ABDDC6B"
codebase="http://www.apple.com/qtactivex/qtplugin.cab">
    <param name="src" value="sample.mov">
    <param name="autoplay" value="true">
    <param name="controller" value="false">
    <embed src="sample.mov" width="160" height="144"
    autoplay="true" controller="false"
```

```
        pluginspage="http://www.apple.com/quicktime/download/">
        </embed>
</object>
```

The object Element

The width and height attributes of the object element should match the size of the movie in pixels. The *classid attribute* uniquely identifies the player software to use. It must be set to "clsid:02BF25D5-8C17-4B23-BC80-D3488ABDDC6B". This unique code identifies an ActiveX control that must be installed on the users PC before the movie can be played. If the user does not have the ActiveX control installed, the browser can automatically download and install it.The *codebase attribute* specifies the base path used to resolve relative URIs specified by the classid, data, and archive attributes. When absent, its default value is the base URI of the current document. Note: Internet Explorer uses this attribute to specify a location from where the player can be downloaded. It must be set to "http://www.apple.com/qtactivex/qtplugin.cab". This location will always contain the latest version of the QuickTime player. The *src parameter* should point to the movie file. The *autoplay parameter* should have the value "true" if you want the movie to play automatically. The *controller parameter* should have the value "false" if you don't want the control buttons to show.

The embed Element

The embed element is supported by old browsers, like Netscape 4 and 5. The width and height attributes of the embed element should match the size of the movie in pixels. The autoplay and controller attributes of the embed element should be set to the same values as for the parameters in the object element. The pluginspage attribute defines the players download path. It must be set to "http://www.apple.com/quicktime/download/".

9.8 Playing Real Video Movies

The object element can play Real Video movies.

The Real Video Format

The RealVideo format is developed by Real Media. Videos stored in the Real Video format have the extension .rm or .ram. The format allows streaming of video (on-line video, Internet TV) with low bandwidths. Because of the low bandwidth priority, quality is often reduced.

This is the code required to play a Real Video movie:

```
<object width="320" height="240"
```

```
classid="clsid:CFCDAA03-8BE4-11cf-B84B-0020AFBBCCFA">
        <param name="controls" value="ImageWindow" />
        <param name="autostart" value="true" />
        <param name="src" value="male.ram" />
</object>
```

The object Element

The width and height attributes of the object element should match the size of the movie in pixels. The *classid attribute* uniquely identifies the player software to use. It must be set to "clsid:CFCDAA03-8BE4-11cf-B84B-0020AFBBCCFA". This unique code identifies an ActiveX control that must be installed on the users PC before the movie can be played. If the user does not have the ActiveX control installed, the browser can automatically download and install it. The *param elements* supply additional information to the player. The *src parameter* should point to the movie (or audio) file. The *autostart parameter* should have the value "true" if you want the movie to play automatically. The *controls parameter* should have the value "ImageWindow" if you don't want the control buttons to show, or "All" if you want all the controls to show.

Controls Parameter Values

Value	**Displays**
All	Displays a full player with all controls
InfoVolumePanel	Title, author, and copyright and volume slider
InfoPanel	Title, author, and copyright
ControlPanel	Position slider, play, pause, and stop buttons
StatusPanel	Messages, current time position, and clip length
PlayButton	Play and pause buttons
StopButton	Stop button
VolumeSlider	Volume slide
PositionField	Position and clip length
StatusField	Messages
ImageWindow	The video image
StatusBar	Status, position and channels

Chapter Review Questions

1. Explain the sound multimedia formats
2. Explain the video multimedia formats
3. using the Object element can you write a code to display the following in a website
    a. Pictures
    b. Sounds
    c. Video
    d. Web page
4. Explain the windows multimedia formats

Suggested Further Reading

1. Mark S et al (1996), Special Edition using internet HTML Que Publishing Co ltd
2. Alex H et all(2000),, Professional active server Wrox publishers programmer to programmer series
3. http://www.w3schools.com

SAMPLE PAPERS