INF-3016

GAUHATI UNIVERSITY
Centre for Distance and Online Education

Third Semester
(Under CBCS)

M.Sc.-IT

Paper: INF 3016
WEB PROGRAMMING TECHNOLOGIES

Contents: **Page No.**

SLM Development Team:

HoD, Department of Computer Science, GU
Programme Coordinator, M.Sc.-IT, GUCDOE
Prof. Shikhar Kr. Sarma, Department of IT, GU
Dr. Khurshid Alam Borbora, Assistant Professor, GUCDOE
Dr. Swapnanil Gogoi, Assistant Professor, GUCDOE
Mrs. Pallavi Saikia, Assistant Professor, GUCDOE
Dr. Rita Chakraborty, Assistant Professor, GUCDOE
Mr. Hemanta Kalita, Assistant Professor, GUCDOE

Contributors:

Mr. Hemanta Kalita	(Block 1 : Units- 1,2,3,4,8 & 9)
Asstt. Prof., GUCDOE	(Block 2 : Units- 1 & 9)
Dr. Swapnanil Gogoi	(Block 1 : Units- 5,6 & 7)
Asstt. Prof., GUCDOE	(Block 2 : Unit- 4)
Mrs. Chayanika Sarma	(Block 1 : Units- 10 & 11)
Asstt. Prof., S.B. Deorah College, Guwahati	(Block 2 : Units- 7 & 8)
Mr. Hem Chandra Das	(Block 2 : Units- 2 & 3)
Asstt. Prof., Dept. of Computer Science & Technology, Bodoland University	
Dr. Ridip Dev Choudhury	(Block 2 : Unit- 5)
Associate Professor, HCB School of Science & Technology, KKHSOU	
Mrs. Pallavi Saikia	(Block 2 : Unit- 6)
Asstt. Prof., GUCDOE	

Course Coordination:

Dr. Debahari Talukdar	Director, GUCDOE
Prof. Anjana Kakoti Mahanta	Programme Coordinator, GUCDOE Dept. of Computer Science, G.U.
Dr. Khurshid Alam Borbora	Assistant Professor, GUCDOE
Dr. Swapnanil Giogoi	Assistant Professor, GUCDOE
Mrs. Pallavi Saikia	Assistant Professor, GUCDOE
Dr. Rita Chakraborty	Assistant Professor, GUCDOE
Mr. Hemanta Kalita	Assistant Professor, GUCDOE
Mr. Dipankar Saikia	Editor SLM, GUCDOE

Content Editor:

Dr. Khurshid Alam Borbora	Assistant Professor GUCDOE

Cover Page Designing:

Bhaskar Jyoti Goswami	GUCDOE
Nishanta Das	GUCDOE

ISBN: 978-81-982028-7-1
October, 2024

Published on behalf of Gauhati University Centre for Distance and Online Education by the Director, and printed at Gauhati University Press, Guwahati-781014.

BLOCK- I

UNIT-1
INTERNET BASICS

UNIT STRUCTURE

1.1 INTRODUCTION

Internet is the massive collection of computer networks that connect number of computers, people, databases and files that interact with each other. The word internet comes from the word **inter**connection + **net**work.The computer network defines the connection between the computer with wires or wireless according to the requirement. The computer networks are categorized as Local Area Network (LAN), Metropolitan Area Network (MAN) and Wide Area Network (WAN). LAN is a computer network which connects the computers in a small geographical area and MAN connects the computer which is larger than LAN but smaller than the WAN. WAN is a computer network which connects the computers in large geographical area in the world. We are familiar with the telephone system, where one can call any person using this system as in post office also one letter is dropped in drop box can

reach any address in the world. So in case of computer, one is connected with the internet can communicate with any computer in the world. Here in this unit we will discuss about the history and basic idea of internet. Also in this unit we will discuss the different types of internet services, such as, telnet, ftp and WWW(World Wide Web).

1.2 OBJECTIVES

After going through this unit, learner will able to:

- Learn about the history of the Internet.
- Understand the basic idea of Internet.
- Understand the concept of WWW.
- Learn about the concept of telnet and ftp.
- Understand the concept of e-mail.

1.3 HISTORY OF INTERNET

There was no any master plan or idea for starting internet. As required to establish the communication between the people and the various devices in the world it has started to developed gradually.

1.3.1 ARPANET:

ARPANET is the precursor to the internet. In 1957, US government formed Advanced Research Projects Agency(ARPA) which is a part of Department of Defense to ensure the strength in science and technology in their military application. In 1969, ARPA came with the idea of distributed network called Advanced Research Projects Agency Network (ARPANET) to need the military communications and alerting the nuclear attack. ARPANET was a network that communicated major computers at the University of California at Los Angeles, the University of California at Santa Barbara, Stanford Research Institute and the University of Utah. After two years, many universities and research institute joined in ARPANET. In 1975, ARPANET was declared operational and was used develop further communication technology. In that time several computers of other countries also connect with the ARPANET using satellite link. ARPANET was developed in such a manner that it could continue its process if one or more sites were

destroyed.Many universities could not connect with ARPANET due to the standardization. So, ARPANET required a standard platform to communicate the computer in different regions. As a result the TCP/IP was developed as protocol to communicate the networks in different region. The main aim of ARPANET was to communicate multiple users in the same time for sending and receiving messages. The network operated with the technique called packet switching technique. In packet switching technique, network is communicated with common set of rules called protocols, which connected the computer, understand the messages and reply them according to the requirement.

Stop to Consider

In, Packet switching technique the total message is divided into smaller chunks. These message chunks are routed into best possible path to reach its destination.

1.3.2 World Wide Web (WWW)

Before WWW, all the communications are text communications only. The graphics portion is implemented with the help of a language called Hypertext Markup Language (HTML). The mechanism from which an end user can view the graphic file is called the WWW. The World Wide Web allows the internet user to view multimedia document which includes text, graphics, audio/video etc. almost in every subjects. WWW was developed by British computer scientist Tim Berners-Lee in 1989. In 1990,Tim Berners-Lee explained three fundamental technologies which are also the base of the today's internet world. These three technologies are:

i. HTML → The language for the web.

ii. URL→ Uniform Resource Locator. The unique address to identify the resources in the web.

iii. HTTP→ Hypertext Transfer Protocol used for transmitting the HTML document.

By the end of 1990, the first web page was served and finally in 1991 people were invited to join the new web community.As internet user increases day by day Berners-Lee directs the World Wide Web Consortium (W3C), a group of industries and universities representatives that inspects the standards of Web technology. The growth of internet from the early days is extraordinary. The main reason of this growth is that internet has endless application for many companies, industries and individuals in daily life. For many internet users, postal services are totally replaced by the e-mail service. The most important part of the internet service is the WWW that provides the facilities to send and receive information on a single command or a single click of mouse. The evolution of internet creates a new era and brought different applications such as net-banking, e-commerce, e-governanceetc. and technical products like dedicated computers, Internet television, smart phone, intelligent home appliances etc.

1.4 OWNERS OF THE INTERNET

Whenever we think about internet,the question is always come to our mind that "Who is the owner of the internet?" The answer in a single word is nobody; Governments, corporations, universities, commercial companies own the internet jointly. The assignment of the IP addresses and domain names are oversees by the nonprofit organization, Internet Corporation for Assigned Name and Numbers (ICANN). Internet Service Provider provides the registration service for each new connected network. The Internet Society(ISOC) has a significant control over the internet. The main objective of ISOC is to promote global information exchange over the internet. The Internet Architecture Board (IAB) is registered as a component of ISOC in 1992. The main task of IAB is to control the evolution of protocol over the world wide. IAB decided which network standard is necessary in different situations.

The responsibilities of IAB are:

(i) It provides oversight of the architecture for the protocols and procedures used by the internet.
(ii) It supervised the process used to create Internet Standards.

(iii) It acts as a mediator between the Internet society and the other organizations concerned with the network standard and other technical issues relevant to the Internet.

Check Your Progress I

1. Multiple Choice Questions

(i) ARPANET is the

(a) Precursor to the internet (b) Internet protocol

(c) ISP (d) DNS

(ii) The unique address to identify the resources in the web is called

(a) ISP (b) DNS

(c) URL (d)CNN

(iii) HTTP is used for

(a) Connect to internet (b) giving IP address

(c)transmitting the HTML document (d) send message

(iv) Registration of each new connection is controlled by

(a) ISP (b) DNS

(c) CNN (d) URL

(v) Assignment IP address done by

(a) ICANN (b) Wide World Web

(c) DNS (d) URL

1.5 ANATOMY OF INTERNET

Initially internet was used as small and in confined geographically area but now internet is used by millions of people from all over the world. At present internet is used to play games,

financial transactions, check the local weather, for research and many more.

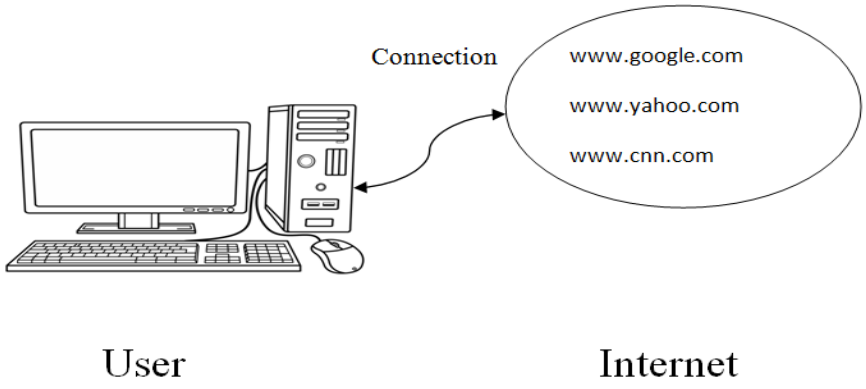


Figure 1.1: Anatomy of Internet

Figure 1.1 depicts the small portion of an internet. Actually internet is like a bookshelf which consists of millions of books and each book has different chapters. Internet consists of a large numbers of networks which are interconnected to each other. There are four basic building blocks of internet and these are:

(i) Hosts: Host is a node or computer in the network in which client can log into or use resources from this node.
(ii) Routers: These are used connect one network with other networks in the world.
(iii) Clients: User's computer is termed as the client and is always request for resources
(iv) Connections: The connection is concern with how user can connect with one point to the other point in the network.

The internet is the client-server system.In client-server system, the client is always request for resources from the server and server responses according to the requirement of the clients. The web browser interprets the server responses as data and displays it on the user's computer screen.

1.6 WORKING OF INTERNET

Internet is a combination of networks. The internet is not a single company or a group of companies or not a single network. Internet is a global network platform in which a single computer can be connected via telecommunication links with other networks. Internet allows sharing of resources from different hosts or server to the different clients.

The working idea of internet is very simple. In Figure 1.2 shows if we connect two computers with a single wire where one is sending a request to the other and other is responses according to the request.It is simple network connection between two computers.

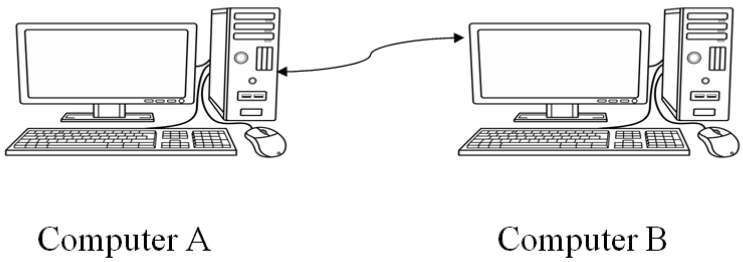


Figure: 1.2 Sample of Internet Connection

In Figure 1.3 shows multiple connections between two computers with multiple servers. In comparison to the previous connection this connection is more complex. When user looks a web page in the client's machine, many things happen along the way. There are many ways to connect your computer with the internet like dial-up, ISDN, DSL, cable modem, wireless, leased line etc. These all technical methods are used to provide the fast communication speed in the internet.

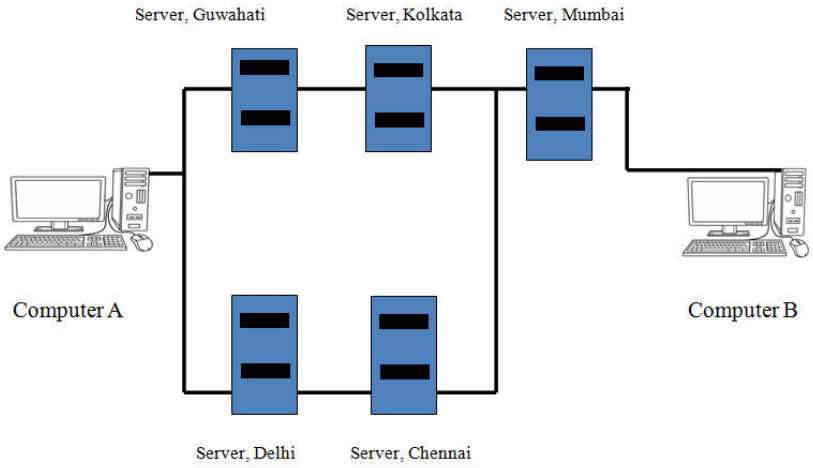


Figure 1.3: Sample Internet Connections

When a person sends a message to another person through internet, firstly this message is broken into some packets. These packets contain the address of the destination and the sender machine. Packets are travelled thousands of kilometres through different technologies. These packets are addressed the final destination through a special device called router. Router forwarded the packets to the destination machine and the paths of the packets are may not be same. Unless and until all the packets reached the destination, one cannot open the message. Once all the packets reached the destination these are reassembled into original message and one can see or open the message.

Stop to Consider

To reach a Web server, your Internet Service Provider (ISP) sends packets of data to another ISP, which may send them to another ISP.

The basic requirements for getting internet connection are as follows:

- Computer → A computer with the required installed software. It should have enough power and memory with multimedia feature. Though 128 MB RAM is sufficient it

is recommended 512 RAM or more for connecting the computer to the internet. At present devices like smart phones, pocket PC etc. are also used for surfing internet.

- Modem→Modem is stand for modulation/Demodulation. A modem can be connected internally or externally. Modem converts the digital data to analog that can be transmitted over a telephone line and vice versa.
- Besides of the above you also required an account with an Internet Service Provider (ISP). The ISP provides you an access to the internet for a monthly fee. ISP also provides you the username and password for connecting your computer with the internet. Modem converts digital computer information to analog which are travel through wired communication. The process of converting the digital data to analog is known as modulation. At the receiving end your ISP have another modem which will link you to internet.

  When you call an ISP to sign up for an Internet account, make sure you get the following information.

  - Username
  - Password
  - Access phone number
  - Your host name or domain name
  - Domain Name System (DNS) server address
- In addition to the above requirement, there are two sets of software required to connect to the internet. The first one is system software which is necessary for setting up and controlling the connection from your computer's operating system to the internet. This set of software must installed first before you installed the application software that means the second set of software. This type of software instructs the modem to dial the right phone number and to identify your machine to access the provider's system for access to your network. Once a connection is established TCP/IP network software allows your computer to communicate properly over the internet.

  The second set of software includes the internet application software such as web browsers or outlook express. These types of software allow you to view

information on the World Wide Web (WWW). End user communicates with the internet through a web browser. Some popular web browsers are Google chrome, Mozilla Firefox, Microsoft Internet Explorer etc. Search engines are the software that enables searching the content available in the internet. Actually search engine is an information retrieval system. Search engine reduce the time for searching any information in the internet. Search engine is the interface of group of words or content that allows to type any keyword, and on the basis of typing, the search engine extract several matching content from millions of web pages. Some of the popular search engines are Google, Alta Vista, Excite, Yahoo etc.

1.7 NET ETIQUETTES

Net etiquettes or netiquettes is a set of rules for behaving online. Netiquettes enhanced the quality of chatting or discussion in the internet. If you do not follow the netiquettes then you will get some critical messages from the internet. When you enter a new environment, you have to follow the rules and regulations of that environment. The cyberspace has its own culture so the internet user must follow this culture otherwise there is no any control over the internet. The rules for the internet users are as follows.

(i) When user communicates with others through internet, their messages, tone of the language must be understandable and appreciable. Make sure your communication must present you in the way that you wish to stand.

(ii) Communicate with online user in similar manner as you communicate in your real life.

(iii) Treat all the emails you receive confidential unless the senders give special permission to share it.

(iv) Respect the copyright and license agreement of the materials written or submitted by others. If you quote something from the internet, you must mention the sources.

(v) Do not participate in the chain letters.

(vi) Do not type your messages in all capital letters. This is bad as shouting. So, do not shout in the internet.

(vii) Do not assume any internet communication is completely secure. Verify all the suspected communication properly before reply it.

1.8 INTERNET SERVICES

There are many services provide by the internet for the necessities of the growth and development. Individual and many companies and different institutions uses internet in many ways. Some of the services provided by the internet are as follows.

1.8.1 Electronic-mail or e-mail

The electronic mail-system or e-mail works in the similar manner as the postal service. In the postal service, implementing the mail operation, lots of elements arerequired. These elements include mail box, letter and envelopes with addresses, Mail trunk, letter boxes and lots of people. But in e-mail service you can complete same things by sitting at your computer. You can compose andsend a letter to the corresponding addresses by giving simple commands on your mail screen.

The following steps are required for understanding the working of e-mail services.

Step 1: To begin with, you have to create an e-mail account. For this purpose, you can create a G-mail account or any other mail account.

Step 2: After creating your mail account successfully, you have to login your account by providing your mail address and password.

Step 3: Type a mail and then send it to the corresponding addresses.

Step 4: The message is sent to your Internet Service Provider's mail server.

Step 5: The mail server examinee the address, and take a decision how to route the sending message.

Step 6: The message travels over the internet, arriving the mail server of the recipient's Internet Service Provider, where it is held in an electronic mailbox.

Step 7: To read the messages, user in the receiving end logs onto the mail server and retrieves the new message from the inbox.

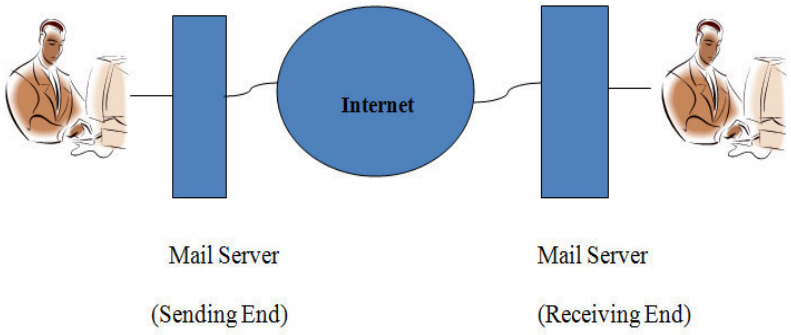


Figure 1.4: Working of e-mail service

E-mail protocols: In e-mail services, different complexities are arises for sending and receiving messages in different platform. For reducing these types of complexities, e-mail protocols are used. E-mail protocols are not belongs to any other companies or organizations. So these protocols can be used by anyone who wants to use it. The following are some e-mail protocols used in e-mail services.

- **SMTP(Simple Mail Transfer Protocol):**SMTP is a TCP/IP protocol that specifies how computer exchanges mails form sender to receiver. SMTP works together with POP(Post Office Protocol) and this is the reason the e-mail services works well in the internet. SMTP is used to upload the mails from the client's computer to Internet Service provider's computer server. The server computer also used SMTP to forward messages to their final destinations. SMTP is a simple ASCII protocol.
- **POP3 (Post Office Protocol):**POP3 is used to check any non-web based e-mail account. It is designed to support offline mail processing. The offline mode is a one kind of store-and-forward service which is intended to move mail from the mail server to the destination computer. The e-mail

client request new messages from e-mail server with the help of POP3 and the server extract all new messages out to the client.Once the connection is established, POP3 protocol goes through three states in the sequence provided as follows.

(i) Authorization –The authorization state deals with having the user log in.
(ii) Transaction – This state deals with the user collecting the e-mails and making them for deletion from the mailbox.
(iii) Update- The update state deals with the e-mails to be deleted.

- **IMAP4:**IMAP4 stands for Internet Message Access Protocol. It is a protocol for retrieving e-mail messages. The latest version of IMAP4 supports some additional features in comparison to POP3.It permits a client e-mail program to access remote message stores as if they were local. The advantage of IMAP4 is that any changes done to your message are stored in the server and it saves local disk space. But not all ISPs support IMAP.
- **MIME:**MIME stand for Multipurpose Internet Mail Extensions. It is used to specify encoding for different types of information into text, making it possible to send as e-mail messages. The standard MIME is also used in other types of communication. The basic idea of MIME is to continue to use the RFC 822 format. MIME defines five new message headers. These headers are as follows.
  - MIME version→It is used for identifying MIME version.
  - Content-Description→Content-Description describes the content of the message.
  - Content-Id→ It defines unique identifier.
  - Content-Transfer-Encoding→ It means how the body is arranged for transmission.
  - Content-Type→ It defines the type and format of the content.

1.8.2 FTP (File Transfer Protocol)

FTP is used for transferring files and folders between computers on the internet. It is used to allow user to get access to the files which is stored in the directory of a remote computer connected

to the internet. Files can be transferred in either direction. The downloading refers the transfer of a file from remote computer to your own computer and uploading refers the transfer of a file from your computer to the remote computer. For uploading and downloading files the following steps are required.

- Log in to a remote computer which configured as FTP server.
- For gaining access,users have to submit the username and password.
- Change the path in the remote computer from where users wish to upload or download the files.
- Transfer the files to or from the system.

There are two types of FTP connections available on the internet-one is anonymous and other is non-anonymous. Anonymous FTP is most widely used. In an anonymous FTP, any user can gain access to the FTP directory, because the username is here anonymous and password is always in the user's email address. But in the non-anonymous FTP,user requires a unique username and password for getting the access to the FTP directory. FTP runs on typical client-server architecture. The FTP server looks after the file security, file organization and the transfer control. FTP servers are judged by their performance. This depends upon how the servers are configured or the features of the operating system. Sometimes the FTP client built into the browser or used some specialized programs.

Most of the operating system comes with built-in FTP client that is accessed from the console application. Steps required for connecting FTP site using console application are as follows.

1. **Open a console window:**For opening a console application, first open command prompt or MS-DOS prompt.

2. **Connect to a Remote FTP site:**When you enter your own login name and password for the remote machine it will returns a prompt "ftp>" which will permit you to access the remote machine. You can access the remote machine with some FTP commands. The following is a typical result of the help command running on a console or in the command prompt

The descriptions of some FTP commands are summarized in the Table 1.1.

Table 1.1: Description of some FTP commands

Command	Description
?	To request help
append	Add or append data to an existing file
ascii	To set mode of file transfer to ascii
bell	Beep when command is completed
binary	Set binary transfer type
bye	Terminate ftp session and exit
cd	Change remote working directory
close	Terminate ftp session
delete	Delete remote file in the current remote directory
help	To request a list all available command
lcd	Change local working directory
ls	List contents of remote directory
mdir	List contents of multiple remote directories
mput	Send multiple files
mget	Get multiple files
open	Connect to remote ftp
put	Send one file
pwd	Print working directory on remote machine
rename	Rename file
rmdir	Remove directory on the remote machine
verbose	Toggle verbose mode

1.8.3 TELNET

The word Telnet is abbreviated from TELecommunicationsNETwork. Using telnet, devices are connected anywhere on a network or in the internet without using wired connection. The computer that acts as a terminal is called Telnet client and the computer that act as the host is called the Telnet server. Telnet client must run a Telnet client application and the Telnet server must run the server application. The TCP/IP protocols are used for transmitting the information from Telnet client to Telnet server. Telnet uses TCP port protocol and port 23 to establish a connection with remote computers. Since Telnet is text-based, user only use keyboard for navigation purpose.

Telnet is placed at the application layer of TCP/IP protocol that provides remote terminal access to a host computer. The telnet protocols give the ability to connect to machine by giving commands and create an interactive session. The commands typed by the user are transmitted directly to the remote machine and responses are displayed in the client's screen. This interactive session is known as **Remote Login**.

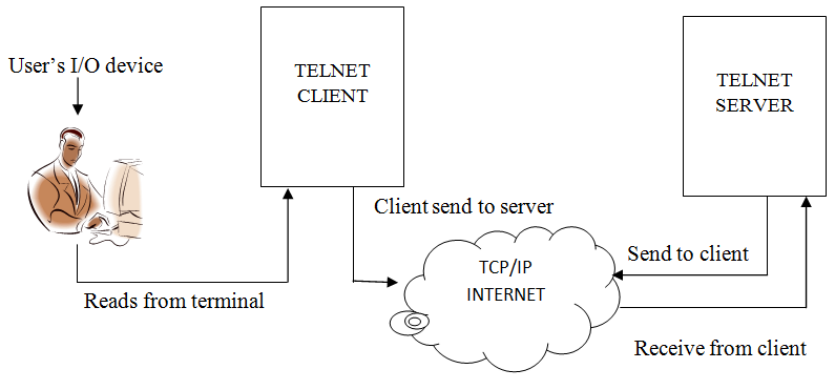


Figure 1.5: Establishing TELNET Connection

Check Your Progress II

2. Multiple Choice Questions

(i) user's computer is termed as

(a) server (b) client

(c) node (d) workstation

(ii) Internet is a combination of

(a) Computer (b) node

(c) networks (d) data packets

(iii) Device used to connect one network with other network is called

(a) Switch (b) Cable

(c) repeaters (d) Router

(iv) Set of rules behaving online is known as

(a) Net rules (b) web rules

(c) netiquettes (d) internet rules

(v) SMTP stands for

(a) Simple Mail Transfer Protocol (b) Single Mail Transfer Protocol

(c) Simple Mail Transfer Port (d) Simple Mail Transaction Port.

3. State True or False

(i) POP3 is an e-mail protocol.

(ii) Anonymous FTP is most widely used.

(iii) Mozila Firefox is an example of search engine.

(iv) Web browser is an example of system software.

(v) FTP runs on typical client-server architecture.

1.9 SUMMING UP

- Internet is the massive collection of computer networks that connect number of computers, people, databases and files that interact with each other.
- The evolution of internet creates a new era and brought different applications such as net-banking, e-commerce, e-governance etc. and technical products like dedicated computers, Internet television, smart phone, intelligent home appliances etc.
- LAN is a computer network which connects the computers in a small geographical area.

- The World Wide Web allows the internet user to view multimedia document which includes text, graphics, audio/video etc.
- Internet consists of a large numbers of networks which are interconnected to each other.
- There are many ways to connect your computer with the internet like dial-up, ISDN, DSL, cable modem, wireless, leased line etc.
- Search engines are the software that enables searching of the content available on the internet.
- Netiquettes enhanced the quality of chatting or discussion in the internet.
- FTP is used for transferring files and folders between computers on the internet.
- Using telnet, devices are connected anywhere on a network or in the internet.
- The telnet protocols give the ability to connect to machine by giving commands and create an interactive session which is known as Remote Login.

1.10 ANSWER TO CHECK YOUR PROGRESS

1. (i) (a) Precursor to the internet (ii) (c) URL (iii) (c)transmitting the HTML document
(iv) (a) ISP (v) (a) ICANN
2 (i) (b) client (ii) (c) networks (iii) (d) Router
(iv) (c) netiquettes
(v) (a) Simple Mail Transfer Protocol
3. (i) True (ii) True (iii) False (iv)False (v) True

1.11 POSSIBLE QUESTIONS

1. What is ARPANET?
2. Explain the role of WWW on internet.
3. Explain the responsibilities of Internet Architecture Board (IAB).
4. Explain the basic building blocks of Internet.
5. Describe the working of internet with a suitable diagram.
6. What are the basic requirements of getting internet connection?
7. What is netiquette? Give some examples.
8. Explain the working of e-mail.

9. What is SMTP? Explain the working of SMTP.
10. What is POP3? Explain the working of POP3.
11. What is IMAP4?
12. What is MIME?
13. What is FTP? Explain the working of FTP.
14. What is TELNET?

1.12 REFERENCES AND SUGGESTED READINGS

- Tanenbaum, A. S. (2003). *Computer networks*. Pearson Education India.
- Holzner, Steven. *HTML black book*. Paraglyph, Incorporated, 2002.
- Stallings, W. (2007). *Data and computer communications*. Pearson Education India.

---×---

UNIT-2
WEB PAGE DESIGN-I

UNIT STRUCTURE

2.1 INTRODUCTION

HTML (Hyper Text Markup Language) is a way of representing text, video and other kind of resources in the web. HTML describes the structure and the behavior of web pages. The web page is a structural document and the core elements of a web page are the text file which is written in HTML language. The WWW (World Wide Web) is one of the most exciting and useful aspect of the internet. There is a difference between the internet and the web. The internet is the medium for running the web. In this unit we will discuss some of the basic HTML tags which are used for designing a web page.

2.2 OBJECTIVES

After going through this unit, learner will able to:

- Learn the concept of HTML.
- Learn how to create and save an HTML document.
- Understand the structure of the HTML document.
- Understand the HTML elements.

2.3 HTML (HYPERTEXT MARKUP LANGUAGE)

HTML is a standard markup language used for creating web pages. It describes the structure of the web pages. An HTML page contains series of elements which tells the browser how to display it. All HTML documents run through any web browser. HTML tags are written with help of '<' and'>'. This is the basic building of HTML documents. Tags in an HTML document are not case sensitive.HTML is developed and maintained by World Wide Consortium (W3C). HTML is derived from the Standard Generalized Markup Language (SGML). The different versions of HTML are as follows

- **HTML 1.0 –** HTML 1.0 has been released in 1992 with very low specification.
- **HTML 2.0 –** It is updated version of HTML. This version has been released in 1995. This version included file uploading through a form, table and some client side image map.
- **HTML 3 –** This HTML 3 includes the addition of FIG, LISTs and LINK elements. It supports style sheet by including STYLE element and class attribute. This version was introduced on 1997.
- **HTML 4 –** This is the updated version of HTML 3 induced more attributes and elements in comparison of HTML 3. HTML 4 include the APPLET, EMBED and IMG elements and the feature of event capturing.
- **HTML 5 –** HTML 5 is the latest version of HTML, which was released in October 2009. It include new elements and attributes, such as AUDIO, VIDEOs and using these type of elements and attributes user can make the web pages more attractive and interactive.

2.4 CREATING AND SAVINGAN HTML DOCUMENT

Creating and saving an HTML page is very simple. Anyone can type their code or text in a text editor and save it with .html or .htm extension. For opening the html file user can select the file which user wish to view then **right click** on it then select the option **open with→** Choose the available web browser for viewing the file.

STOP TO CONSIDER

Remember the small things like missing < or typing (\) instead of (/) are enough to give errors in viewing the html file.

The basic structure of an HTML document is as follows.

```
<! DOCTYPE HTML>

<HTML>
        <HEAD>
                <TITLE>Page Title</TITLE>
        </HEAD>
        <BODY>

             <H1>Welcome to GUCDOE</H1>
             <P>M.Sc.-IT 3rd Semester</P>

        </BODY>
 </HTML>
```

The meanings of the different elements of the above structure are as follows.

- <! DOCTYPE HTML> -- This is the very first element in an HTML document. It provides the Document Type Definition declaration for the browsers to specify which version of the markup language is used in the HTML document. The "!" character in <! DOCTYPE HTML>indicates that DOCTYPE element is an SGML declaration.

- <HTML> is the starting or root element of the HTML document. It inform the browser that the page that is loaded written with the html code.
- **<**HEAD**>**element contains the information about the HTML document. The different numbers of tags can be placed between the <HEAD> and </HEAD> tags including <BASE>, <ISINDEX>, <LINK>, <META>, <SCRIPT>, <STYLE> and <TITLE>. The browser generally does not display the information except for the text which is written in between the <TITLE> and </TITLE> tags.
- <TITLE**>** element specifies a title of the HTML document. This title is displayed in the browser's title bar.
- **<**BODY**>** specifies the body of the document. This will contain all the visible contents in the HTML document. It is placed after the closing tag of the HEAD element in the HTML document.
- <H1**>** specifies the heading of the document.
- **<**P> defines a paragraph.

STOP TO CONSIDER

Unlike other HTML elements, the DOCTYPE element does not have any named attributes.

2.5 HTML ATTRIBUTES

HTML attributes provides the additional information about an HTML elements. All the HTML documents can have attributes. All attributes in an HTML document written with the help of starting tag.

The attributes associated with the <body> tag are:

(i) Background: This attribute is used for provide the background for the document.

Syntax

<body background=”URL or path/ filename”>

Document here

</body>

Example 2.1:

<! DOCTYPE HTML>

<HTML>

<HEAD>

<TITLE> background image</TITLE>

</HEAD>

<BODY background="E:/background.jpg">

<P>WELCOME TO GUCDOE</P>

</BODY>

</HTML>

(ii) Bgcolor: This attribute is used to setting of the background color for the document

Syntax:

<body bgcolor = “#rrggbb or color name” >

Document here

</body>

Here “#rrggbb” is a hexadecimal code used for the color red, blue and green. This code is starting from 0 to 255. The tag #FFFFFF is a hex code which designates the white background color because the FF is translated to 255. Hence the entire hex code for RGB=255 is #FFFFFF. The darkest color in hexadecimal code is ‘00’. The red color page will be “FF0000”, green color page will

“00FF00” and the bright blue color page will “0000FF”. In HTML, 16 colors can be called directly by their color names rather than using the hexadecimal values.

Example 2.2:

```
<! DOCTYPE HTML>


<HTML>
    <HEAD>
        <TITLE> background color</TITLE>
    </HEAD>
    <BODY bgcolor="red">
        WELCOME TO GUCDOE
    </BODY>
</HTML>
```

The example 2.2 can also be written as:

```
<! DOCTYPE HTML>
<HTML>
    <HEAD>
        <TITLE> background color</TITLE>
    </HEAD>
    <BODY bgcolor="#FF0000">
        WELCOME TO GUCDOE
    </BODY>
</HTML>
```

(iii) Text: The text attribute is used for altering the normal text color of the document.

Syntax:

```
<body Text= "#rrggbb" or colorname >
        Document here
</body>
```

Example 2.3:

```
<! DOCTYPE HTMl>

<HTML>
        <HEAD>
                <TITLE> background Text</TITLE>
        </HEAD>
        <BODY text="blue">
                 WELCOME TO GUCDOE
        </BODY>
</HTML>
```

The example 2.3 can also be written as:

```
<! DOCTYPE HTML>

<HTML>

        <HEAD>
                <TITLE> background Text</TITLE>
        </HEAD>
        <BODY text="#0000FF">
                 WELCOME TO GUCDOE
        </BODY>
</HTML>
```

2.6 PHYSICAL ELEMENTS INAN HTML DOCUMENT

Physical elements are also known as the character formatting. Whenever user wish to specify the text is bold, italic, underline, big etc. then user can use these physical elements.

The common physical elements are:

Bold: The bold elements specify that the text should be rendered in boldface.

Syntax: <B>……</B>

Italic: The italic elements specify that the text should be rendered in italic font.

Syntax: <I>…….</I>

Underline: The underline element specifies that the text written between underline tags should be rendered as underlined text.

Syntax: <U>……..</U>

Superscript: The superscript element specifies the enclosed text in superscript tag should be rendered as a superscript using a smaller font in compared with the normal font.

Syntax: ^{…….}

Subscript: The subscript element specifies the enclosed text in subscript tag should be rendered as a subscript using a smaller font in compared with the rest of the text.

Syntax: _{…….}

Big: The big element specifies the content written in between big tag should be displayed as big font in compared with the current font.

Syntax: <BIG>…….</BIG>

Small: The small element specifies the content written in between small tag should be displayed as small font in compared with the current font

Syntax: <SMALL>…….</SMALL>

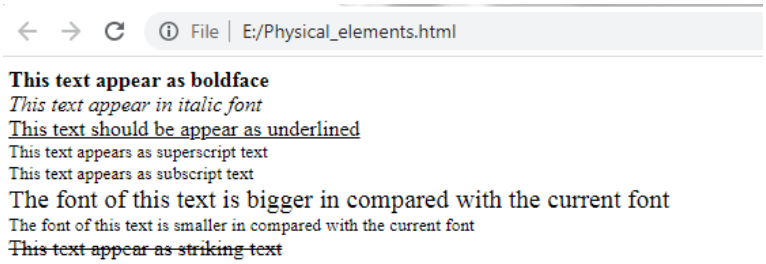
Strike: The strike element specifies that the content written in between strike tag should be rendered as the striking text.

Syntax: <STRIKE>…….</STRIKE>

Example 2.4: use of physical elements

```
<! DOCTYPE HTML>
<HTML>
    <HEAD>
        <TITLE> Using of Physical elements</TITLE>
    </HEAD>
    <BODY>
        <B>This text appear as boldface</B><BR>
        <I>This text appear in italic font</I><BR>
        <U>This text should be appear as
underlined</U><BR>
        <SUP>This text appears as superscript
text</SUP><BR>
        <SUB>This text appears as subscript
text</SUB><BR>
        <BIG>The font of this text is bigger in compared
with the current font</BIG><BR>
        <SMALL> The font of this text is smaller in
        compared with the current font</SMALL><BR>
        <STRIKE> This text appear as striking
text</STRIKE><BR>
    </BODY>
</HTML>
```

Output of example 2.4:



STOP TO CONSIDER

To insert returns or blank lines in a document, the
 element is used.
 is the empty element without a closing tag

2.7 LOGICAL ELEMENTS IN AN HTML DOCUMENT

Logical elements are used to alert the browser that what types of text written inside the tags. Some of the examples of the logical tags are citation, code, strong, etc.

The common logical elements are:

Cite: The enclosed text in citation elements are typically italics. It is used to specifying the external references or resources in a page.

Syntax: <CITE>……..</CITE>

Strong: The enclosed texts in strong elements are typically bold.

Syntax: <STRONG>……..</STRONG>

Code: The code element indicates an example of code. It is typically rendered as mono spaced font.

Syntax: <CODE>……..</CODE>

Variable: The variable element indicates a variable name. The variable name may be in a mathematical equation or in a computer programs.

Syntax:<VAR>……..</VAR>

Emphasis: The Emphasis element indicates typographic emphasis typically rendered as italics.
Syntax: <EM>……..</EM>

Keyboard Input: It defines the keyboard input.

Syntax: <KBD>……..</KBD>

Sample: The sample element indicates a sequence of literal characters. It defines the sample output text which rendered as monospaced.

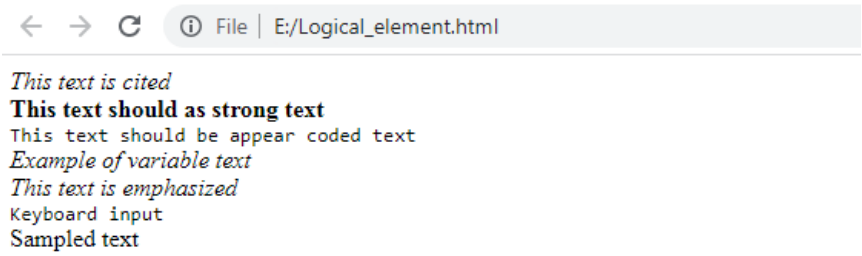
Syntax: <SAM>……</SAM>

Example 2.5: HTML document with the logical elements.

```
<! DOCTYPE HTML>
<HTML>
    <HEAD>
        <TITLE> Using of Physical elements</TITLE>
    </HEAD>
<BODY>
    <CITE>This text is cited</CITE><BR>
    <STRONG>This text should as strong
text</STRONG><BR>
    <CODE>This text should be appearingas coded
text</CODE><BR>
    <VAR>Example of variable text</VAR><BR>
    <EM>This text is emphasized</EM><BR>
```

```
<KBD>Keyboard input</KBD><BR>

<SAM>Sampled text</SAM><BR>

</BODY>

</HTML>
```

Outputof example 2.5:



Check Your Progress-I

1. Multiple Choice Questions

(i) HTML stands for_____.

a) HyperText Markup Language
b) HomeText Markup Language
c) Hyperlink Markup Language
d) HyperText Media Language

(ii) All HTML element run through_____.

(a) Web server
(b) WWW
(c) Web Protocol
(d) Web browser

(iii)Basic building block of an HTML document are_____.

(a) Browser
(b) Protocol
(c) Hyperlink
(d) Tags

(iv)The root element of an HTML document is_____.

(a) <BODY>
(b) <HEAD>
(c) <HTML>
(d)

(v) Choose the correct HTML tag to make a text bold.

(a) <BOLD>
(b) <B>
(c) <BLD>
(d)

2. State True or False.

(i) <HEAD> element contains the information about the HTML document.
(ii) HTML documents cannot have attributes.
(iii) The <KBD> tag defines the keyboard input.
(iv) The **bgcolor**attribute is used for provide the background for the document.
(v) Tags in an HTML document are case sensitive.

2.8 SECTION HEADINGS IN HTML

HTML headings defined with the <H1> to <H6> tags.

<H1> define most important heading and the <H6> defines the least important heading. The 'align' attribute can be used with the heading tag to position the heading on the browser screen. This align attribute can be set as left, right or center with the heading tags.

Example 2.6:

```
<! DOCTYPE HTML>
    <HTML>
        <HEAD>
            <TITLE> Heading Demo</TITLE>
        </HEAD>
        <BODY>
            <H1 align= "center" >Heading 1</H1>
            <H2 align= "left" >Heading 2</H2>
            <H3 align= "left">Heading 3</H3>
            <H4 align= "left" >Heading 4</H4>
            <H5 align = "left" >Heading 5</H5>
```

```
        <H6 align =”left”>Heading 6</H6>
    </BODY>
</HTML>
```

Output of example 2.6:

2.9 FORMATTING PARAGRAPH IN HTML

Paragraph in a document start with a new line and it is usually a block of text. <P> defines the paragraph in an HTML document. The align attribute with the values left, center and right can also be added with <P> element. The left setting is the default alignment.

Example 2.7

```
<! DOCTYPE HTML>
<HTML>
    <HEAD>
        <TITLE>Paragraph Examples</TITLE>
    </HEAD>
    <BODY>
```

```
        <P align="center">This is a paragraph. </P>
        <P align="left">This is a paragraph. </P>
        <P align="right">This is a paragraph. </P>
    </BODY>
</HTML>
```

2.10 FORMATTING PREFORMATTED TEXT IN HTML

The preformatted text element presents blocks of text in fixed-width font. The <PRE> element is used for preformatted text and it preserves both line breaks and the spaces in the document.The text written in <PRE> tag is displayed exactly as written in the HTML source code. The elements that define paragraph formatting must not be used with the <PRE> tag.

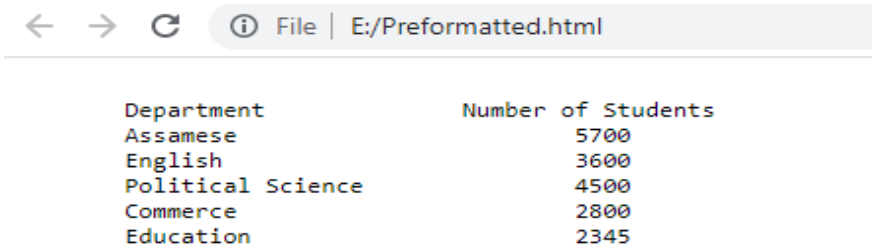
Example 2.8:HTML document with preformatted text.

```
<HTML>
    <HEAD>
        <TITLE> Preformatted text</TITLE>
    </HEAD>
    <BODY>
        <PRE>
            Department          Number of Students
            Assamese            5700
            English             3600
            Political Science   4500
            Commerce            2800
            Education           2345
        </PRE>
```

</BODY>

</HTML>

Output of example 2.8:



2.11 FORMATTING HORIZONTAL RULES IN HTML

The<HR> tag is used to specify the horizontal rule be drawn across the page. The recent browser added 5 attributes with the horizontal rule. Theseattributes with the horizontal rule are:

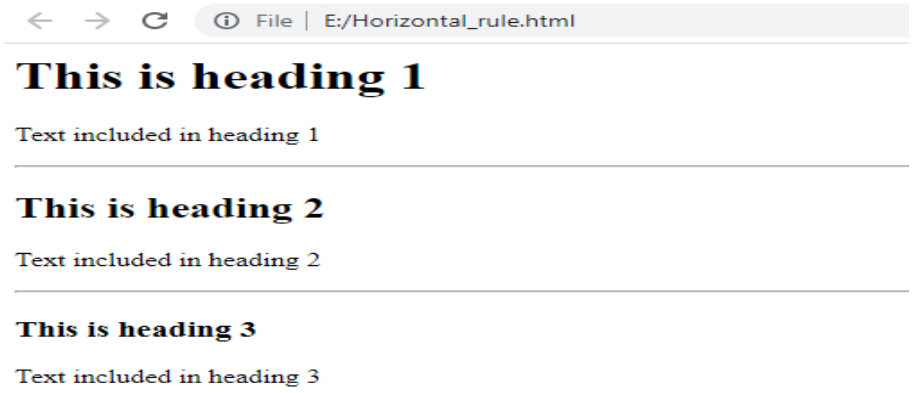
- **<HR size=”number”>:**Here the size specifies the thickness of the horizontal rule. The default size is 1.
- **<HR width= “number” or “percent”>:** The default width of the horizontal rule is as wide as the page width.Using the width attribute user can fixed the width of the horizontal rule or specifies the exact width in pixels or a relative width measured in percent of the page width.
- **<HR align= “left” or “right” or “center”>:** User can align the horizontal rule as left, right and center.
- **<HR noshade>:**‘noshade’ attribute specifies that the horizontal rule should not be shaded.
- **<HR color= “name” | “#rrggbb”>:** The color attribute specify the color of the horizontal rule or any “rrggbb” hexadecimal value.

Example 2.9

<! DOCTYPE HTML>

```
<HTML>
    <BODY>
        <H1>This is heading 1</H1>
        <P>Text included in heading 1</P>
        <HR>
        <H2>This is heading 2</H2>
        <P>Text included in heading 2</P>
        <HR>
        <H3>This is heading 3</H3>
        <P>Text included in heading 3</P>
</BODY>
</HTML>
```

Output of example 2.9

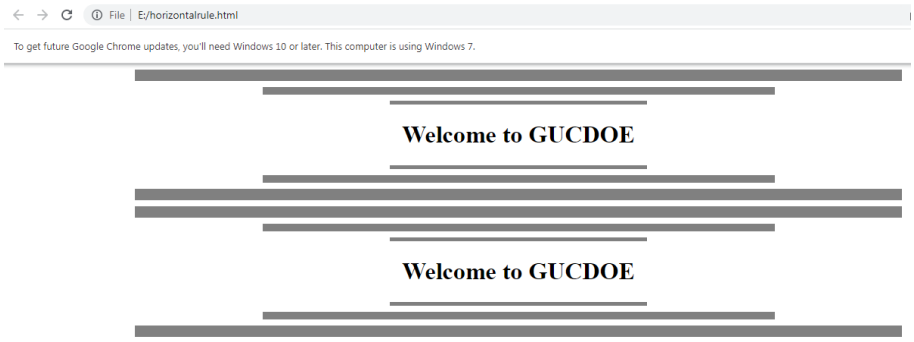


Example 2.10: HTML document with horizontal rule and its attributes.

```
<! DOCTYPE HTML>
```

```
<HTML>
    <HEAD>
        <TITLE> Horizontal Rule</TITLE>
    </HEAD>
    <BODY>
        <HR align="center" Noshade  size="15" width="75%">
        <HRalign="center" Noshade  size="10" width="50%">
        <HR align="center" Noshade  size="5"  width="25%">
        <H1 align="center"> Welcome to GUCDOE </H1>
        <HR align="center" Noshade  size="5" width="25%">
        <HR align="center" Noshade  size="10" width="50%">
        <HR align="center" Noshade  size="15"  width="75%">
        <HR align="center" Noshade  size="15" width="75%">
        <HR align="center" Noshade  size="10" width="50%">
        <HR align="center" Noshade  size="5"  width="25%">
        <H1 align="center"> Welcome to GUCDOE </h1>
        <HR align="center" Noshade  size="5" width="25%">
        <HR align="center" Noshade  size="10" width="50%">
        <HR align="center" Noshade  size="15"  width="75%">
    </BODY>
</HTML>
```

Output of example 2.10:



2.12. DIVISIONS

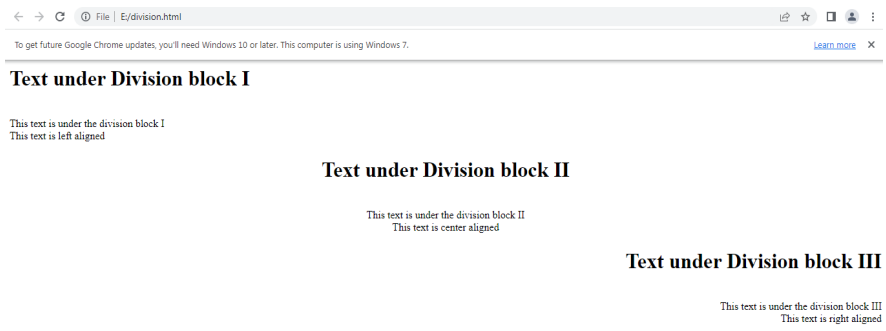
The division element is used to divide or section in an HTML document.It is implemented with the <DIV> tag. The align attribute with the values left, right and center can be used along with the <DIV> tag.

Example 2.10: HTML document with division element.

```
<! DOCTYPE HTML>


<HTML>
    <HEAD>
        <TITLE> Example of division element</TITLE>
    </HEAD>
<BODY>
    <DIV align="left">
        <H1> Text under Division block I </H1><BR>
            This text is under the division block I<BR>
            This text is left aligned
    </DIV>
```

```
<DIV align="center">
        <H1> Text under Division block II</H1><BR>
        This text is under the division block II<BR>
        This text is center aligned
</DIV>

</DIV>
<DIV align="right">
        <H1> Text under Division block III</H1><BR>
        This text is under the division block III<BR>
        This text is right aligned
</DIV>
</BODY>
</HTML>
```

Output of example 2.10:



2.13 LISTS

There are different types lists supported by HTML documents are:

2.13.1 Unordered or Bulleted list
2.13.2 Ordered list

2.13.3 Definition list

2.13.1 Unordered or bulleted list

The unordered list is starting with the <UL> and end with the </UL> tag. In the content of the <UL> tag the <LI> tag is used for listing the items. The type attribute can be used with the <UL> tag and values of type attribute are “disc”, “circle” and “square”.

Syntax:

<UL type= “list type”>

<LI> list item 1

<LI> list item 2

……..

……...

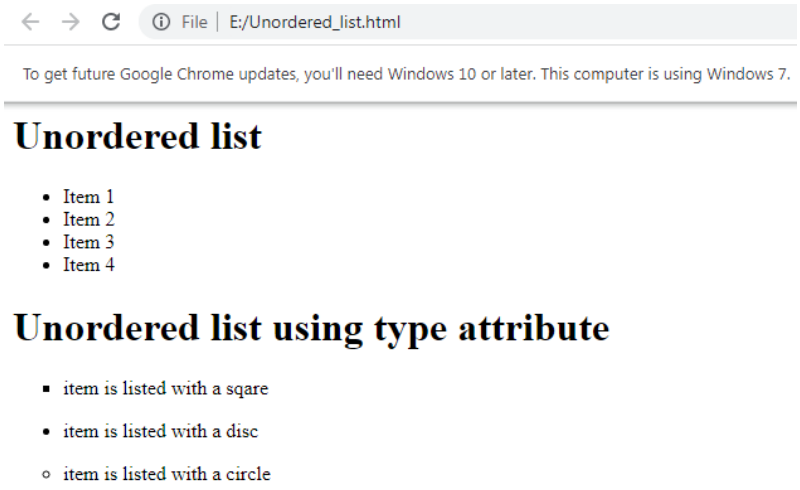
</UL>

Example 2.11

```
<! DOCTYPE HTML>
<HTML>
    <HEAD>
        <TITLE> Example of Unordered list</TITLE>
    </HEAD>
    <BODY>
        <H1> Unordered list</H1>
        <UL>
            <LI> Item 1
            <LI> Item 2
            <LI> Item 3
            <LI> Item 4
```

```
        </UL>
        <H1> Unordered list using type attribute</H1>
        <UL type="square">
            <LI> item is listed with a sqare
        </UL>
        <UL type="disc">
            <LI> item is listed with a disc
        </UL>
        <UL type="circle">
            <LI> item is listed with a circle
        </UL>
    </BODY>
</HTML>
```

Output of example 2.11



2. 13. 2 ORDERED LIST

Ordered list is implemented with the help of <OL> tag. The content under <OL> tag is listed orderly like 1,2,3,…….., etc.

The different types specified in <ol> tags are

(i) Type= “1” means the listed items are 1,2,3,4,…….

(ii) Type = “A” means the listed items are A,B,C,D,…….

(iii) Type = “a” means the listed items are a,b,c,d,……..

(iv) Type = “I” means the listed items are I,II,III,IV,…….

(v) Type = “i” means the listed items are i, ii, iii, iv,……..

By default the ordered list is stating with the number 1. If users wish to set the stating value, it can be implemented with the help of “start” attribute. Also each <LI> tag has also its local type of attribute which is set to 1, A, a, I or i. Once an <LI> item is set with a new type, then it will override the current numbering style for the rest of the list items.

Syntax:

<OL type= “list_type” start= “start_value”>

<LI> list item1

<LI> list item 2

<LI> list item3

<LI> list item 4

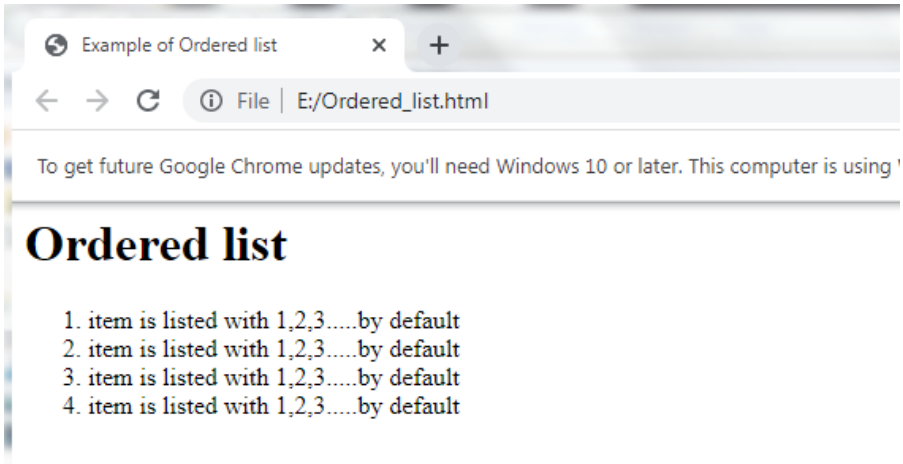
</OL>

Example 2.12

```
<! DOCTYPE HTML>
<HTML>
    <HEAD>
        <TITLE> Example of Ordered list</TITLE>
    </HEAD>
```

```
<BODY>
    <H1> Ordered list</H1>
    <OL>
        <LI> item is listed with 1,2,3.....by default
        <LI> item is listed with 1,2,3.....by default
        <LI> item is listed with 1,2,3.....by default
        <LI> item is listed with 1,2,3.....by default
    </OL>
</BODY>
</HTML>
```

Output of example 2.12



Example 2.13 HTML document with Ordered list using type attributes.

```
<! DOCTYPE HTML>
<HTML>
    <HEAD>
```

```
        <TITLE> Example of Ordered list</TITLE>
    </HEAD>
<BODY>
<H1> Ordered list using type attribute</H1>
    <OL type="A">
        <LI> item is listed with A,B,C,D.....
        <LI> item is listed with A,B,C,D.....
        <LI> item is listed with A,B,C,D.....
        <LI> item is listed with A,B,C,D.....
    </OL>
    <OL type="a">
        <LI> item is listed with a,b,c,d.....
        <LI> item is listed with a,b,c,d.....
        <LI> item is listed with a,b,c,d.....
        <LI> item is listed with a,b,c,d.....
    </OL>
    <OL type="I">
        <LI> item is listed with I,II,III,IV.....
        <LI> item is listed with I,II,III,IV.....
        <LI> item is listed with I,II,III,IV.....
        <LI> item is listed with I,II,III,IV.....
    </OL>
    <OL type="A" start="10">
        <LI> item is listed with J,K,L,M.....
        <LI> item is listed with J,K,L,M.....
```

<LI> item is listed with J,K,L,M.....

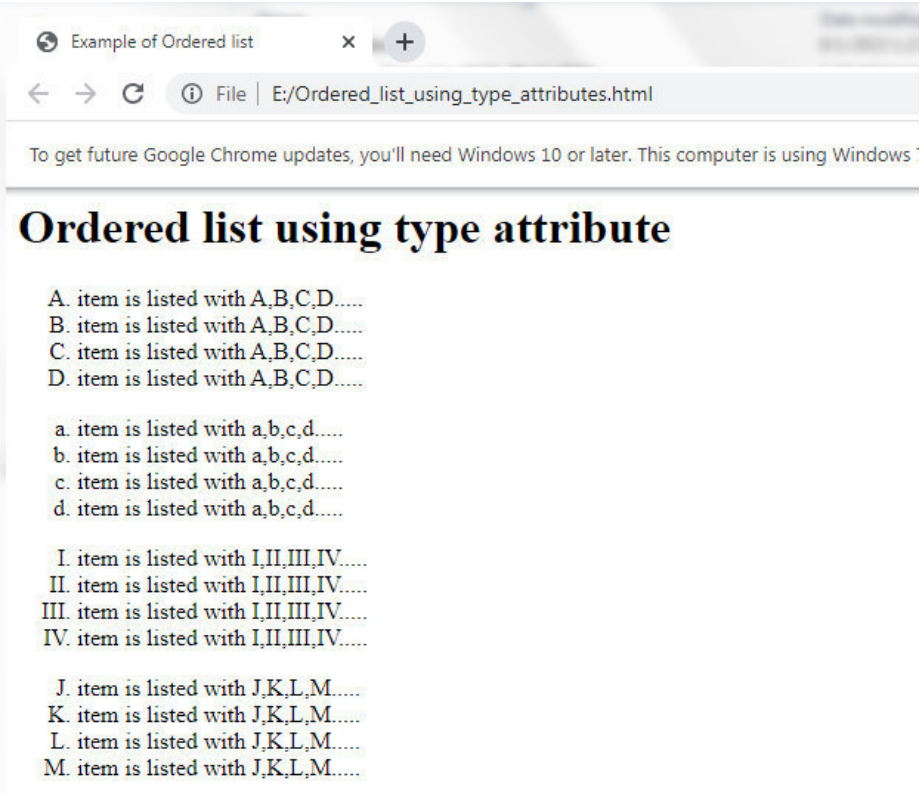
<LI> item is listed with J,K,L,M.....

</OL>

</BODY>

</HTML>

Output of example 2.13



2.13.3 DEFINITION LIST

The tag <DL> defines the definition list. A definition contains the definition and terms for a particular subject.

The tags associated with the definition list are:

<DL>: It defines the definition list in the HTML document.

<DT>: It defines the term in a definition list in the HTML document.

<DD>: It defines the definition in a definition list in the HTML document.

STOP TO CONSIDER

Use of the <DD>without <DT> is non-standard. Both the <DD> and <DT> tags does not require close tag but if you use long definition, using the close tag may be helpful.

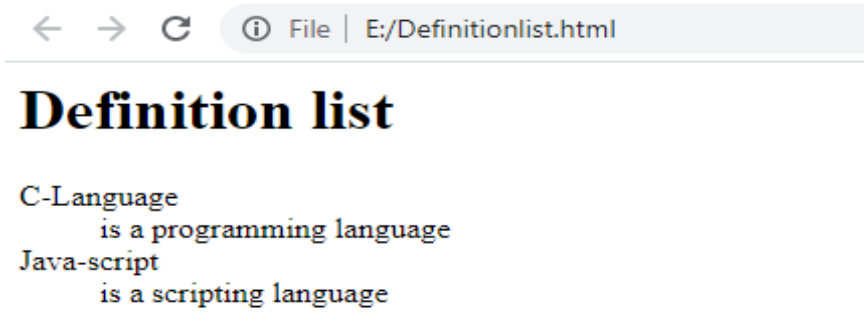
The compact attribute can be used with the definition list. The compact attribute defines the list should be smaller than the normal and it reduces the space between the list items and the indentation of the list.

Example 2.14

```
<! DOCTYPE HTML>
<HTML>
    <HEAD>
        <TITLE> Example of Definition list</TITLE>
    </HEAD>
    <BODY>
        <H1> Definition list</H1>
        <DL>
            <DT> C-Language
            <DD> is a programming language
            <DT> Java-script
            <DT> is a scripting language
        </DL>
```

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

Output of example 2.14:



Example 2.14:HTML document using Definition list with compact attribute.

```
<! DOCTYPE HTML>
<HTML>
    <HEAD>
        <TITLE> Example of Definition list with compact attribute</TITLE>
    </HEAD>
    <BODY>
        <H1> Definition list with compact attribute</H1>
        <DL compact>
            <DT> A
            <DD> is the first letter of the English alphabet.
            <DT> B
            <DD> is the second letter of the English alphabet.
        </DL>
    </BODY>
```

</HTML>

Output of example 2.14:

Check Your Progress II

3. Write HTML codes for the displaying the followings

(i) National Parks

A. Kaziranga National Park.
B. Manas National Park.
C. Dibru - Saikhowa National Park.
D. Nameri National Park.

(ii) National Parks

- Kaziranga National Park.
- Manas National Park.
- Dibru - Saikhowa National Park.
- Nameri National Park.

4. Write HTML code for displaying the following using Pre-formatted text.

Multimedia software and their Uses

Name of Software	Application
Adobe Photoshop	Graphics and Image editing
Adobe PageMaker	Text Editing Tool
Adobe Premiere	Video Editing

Hypertext link means the link provides a path that connect user from one part of an HTML document to another part of the same document. It also provides a path for connecting one HTML document to another HTML document and with different resources. Hypertext link is possible, as every document in the web has its unique address which is known as the URL (Uniform Resources

Locator). The anchor element **<A>** is used to define hypertext links. The link text is usually appears as blue and underlined text.

The **<A>** tag can be used for one of the two things one is for navigating external links and one is for navigating internal links. These external and internal links are depend on which attribute is used with the anchor element.

The attributes associated with the <A> tag are:

(i) Href:The href attribute is used to set the URL of the destination document.

Example:

<P><A href="https://www.w3schools.com/"> W3Schools.com</P>

Here,<A> defines the anchor tag

https://www.w3schools.com/→ defines the URL

w3Schools.com → defines the link text .

(ii) Name attribute: This attribute is used to create a link of one part of an HTML document with another part of the same document.

Syntax:

<A name = "Hello"> Text or Image </A>

<A href = "#Hello" > Click here to go the text or image </A>

Here the first one is the anchor tag and the second one is the 'link' tag. The link tag directs the browser to the name anchor. The symbol "#" character identifies the link which is a portion of a document.

Example 2.15:

```
<! DOCTYPE HTML>
<HTML>
    <HEAD>
        <TITLE> Hyperlink example</TITLE>
    </HEAD>
    <BODY>
        <H1>Example of hyperlink one page to another HTML page</H1>
        <A href="E:\homepage.html">Click here to go to home page.</A>

    </BODY>
</HTML>
```

Output of example 2.15:

In the output whenever user click on the text "**Click here to go home page**", this will open the home page.

Check your Progress III

5. Multiple Choice Questions

(i) Which of the following element define the paragraph in an HTML document?

(a)
(b) <P>
(c) <PRG>
(d) <bld>

(ii) The <PRE> element is used for_____.

(a) preformatted text
(b) Formatted text
(c) Previous text
(d) Performance text

(iii) Which of the following tag is used specify horizontal rule?

(a) <HG>
(b) <H1>
(c) <HV>
(d) <HR>

(iv) Ordered list is implemented with the help of____.

(a)
(b)
(c) <dl>
(d) <ot>

(v) href attribute is used to____.

(a) Link between two document
(b) Hypertext reference
(c) Insert an image
(d) set the URL of the destination document

6. State True or False

(i) The tag <dl> defines the definition list
(ii) The division element is used to form a paragraph in an HTML document.
(iii) Attributes are not allowed with the <hr> element.
(iv) The compact attribute can be used with the definition list.
(v) The <li> tag is used for listing the items.

2.15 SUMMING UP

- HTML (HyperText Markup Language) is a way of representing text, video and other kind of resources in web.
- An HTML page contains series of elements which tells the browser how to display it.
- All HTML files are saved with .htm or .html extension.
- HTML attributes provides the additional information about an HTML elements.
- An HTML element is written in between start and end tags.
- Text level elements are the elements that affect the text of the document.
- Block level elements define the structural content block, such as paragraph or lists.
- The unordered list is starting with the <ul> and end with the tag.
- Ordered list is implemented with the help of <ol> tag.
- A definition contains the definition and terms for a particular subject.
- Hypertext link means the link provides a path that connect user from one part of an HTML document to another part of the same document.

2.16 ANSWER TO CHECK YOUR PROGRESS

1. (i)(a) (ii)(d) (iii) (d) (iv) (c) (v) (b)

2. (i) True (ii)False (iii) True (iv) False (v) False

3. (i)

```
<html>
    <head>
        <title> Example of Ordered list</title>
    </head>
    <body>
        <h1> National Parks</h1>
        <ol type="A">
            <li> Kaziranga National Park
            <li> Manas National Park
            <li> Dibru-Saikhowa National Park
            <li> Nameri National Park
        </ol>
    </body>
</html>
```

(ii)

```
<html>
    <head>
        <title> Example of unordered
list</title>
    </head>
<body>
    <h1> National Parks</h1>
    <ul type="disc">
```

```
<li> Kaziranga National Park
<li> Manas National Park
<li> Dibru-Saikhowa National Park
<li> Nameri National Park
</ul>
</body>
</html>
```

4.

```
<html>
<head>
<title> Preformatted text</title>
</head>
<body>
<h1> Multimedia software and their Uses</h1>
<pre>
Name of Software Application

Adobe Photoshop Graphics and Image editing
Adobe PageMaker Text Editing Tool
Adobe Premiere Video Editing
</pre>
```

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

5. (i) (b) (ii)(a) (iii) (d) (iv) (a) (v) (d)

6. (i) True (ii) False (iii) False (iv) True (v)True

2.17 POSSIBLE QUESTIONS

1. What is HTML?
2. How can you create and save an HTML file?
3. What are HTML elements? Give examples.
4. What are HTML attributes?
5. Explain the attributes of <body> element with suitable examples.
6. Explain the structure of an HTML document.
7. Explain five physical elements with examples.
8. Explain five logical elements with examples.
9. Explain the use of section heading in an HTML document.
10. How can you format a paragraph in an HTML document?
11. Explain the use of preformatted text in an HTML document.
12. How horizontal rule helps in formatting an HTML document?
13. What is an unordered list? Explain different types of attributes of unordered list with examples.
14. What is an ordered list? How ordered list is helpful in formatting an HTML document?
15. What is Hypertext link? Explain the use of "href" attribute in an HTML document.

2.18 REFERENCES AND SUGGESTED READINGS

- https://www.w3schools.com/
- HTML 5 BLACK BOOK Published by dreamtech press.

---×---

UNIT-3
WEB PAGE DESIGN-II

UNIT STRUCTURE

3.1 INTRODUCTION

There are different types HTML elements which are frequently used for designing web pages. Some HTML tags using with different attributes gives tremendous effect in displaying web pages. HTML provides user the <TABLE> element to create a table. Here we will discuss how the TABLE element helps user to display the text in tabular format. HTML enables user to creating a form with the help of <FORM> element. Here in this unit we also discuss about different attributes associated to the<FORM> element. Also

we will discuss about some standard HTML elements used with their different attributes which are different from the basic HTML elements.

3.2 OBJECTIVES

After going through this unit learner will able to:

- Understand the different formatting styles of font and font sizes.
- Learn how to insert an image in HTML document.
- Learn about the TABLE element and its attributes.
- Learn about spanning the rows and columns of the table.
- Understand about the FORM element and its attributes.
- Learn about INPUT element and its types such as text, password, checkbox, radio button, submit button and reset button.
- Learn about some interactive element used in HTML for displaying web page.

3.3 FORMATTING STYLES

The following tags specify the different fonts and font sizes and how to add scrolling and blinking text in an HTML document.

3.3.1 Fonts and Font Sizes

The tag <FONT> is used to specify the changes the color, size and face of the font. The <font> tag has the following attributes:

(i) Color: This attribute sets the color of text that will appear on the screen. Color can be set by using one of the color name or as a hexadecimal 'rrggbb' triplet value.

Syntax:

<FONTcolor = "#rrggbb or color name">……..</FONT>

Example:

<FONTcolor= "0000FF">This text is blue color**.</FONT>**

Or

<FONTcolor= "blue" >This text is blue color**.</FONT>**

(ii) Face: This attribute specify the typeface of the font. The typeface displayed by the "face" attribute must installed on the user computer. If it is not installed on the computer, the text will display in the default type according to the browser preference setting.

Syntax:

< FONT face = " name [, name]" >…….</FONT>

Example:

<FONT face = "Times New Roman">This text will be displayed in Times New Roman.**</FONT>**

(iii) Size: The size attribute defines the size of the font, in a range from 1 to 7. The default size is 3.

Syntax:

<FONT size ="fontsize" >……..</FONT>

<FONT size = "7">This is the biggest font size</FONT>

Example 3.1:

```
<! DOCTYPE HTML>
<HTML>
    <HEAD>
        <TITLE> Specifying font tag with different attributes</TITLE>
```

```
</HEAD>
<BODY>
<H1>Example of font tag with different attributes</H1>
<FONTcolor="blue" face="Calibri" size="7">
This is blue color, Calibri  size 7 text
</FONT><BR>
<FONTcolor="green" face="Times New Roman" size="7">
This is green color,
Times New Roman
size 7 text
</FONT><BR>
<FONTcolor="red" face="Arial">
This is red color,
Arial
default size text
</FONT><BR>
</BODY>
</HTML>
```

Output of the example 3.1:

3.3.2Scrolling Text

The tag <MARQUEE> allows the user to create a scrolling text.

Syntax:
<MARQUEE>……..</MARQUEE>

Marquee tag has the following attributes.

- **Align:** The align attribute can be used with the **<MARQUEE>** attribute. User can set top, middle or bottom that specifies that the text around the marquee should align with the top, middle or bottom of the marquee.

- **Bgcolor:** Thisattribute specifies the background color of the text which is written in between the <MARQUEE> tag.The bgcolor can be set with the color name or hex triplet like 'rrggbb'.

- **Direction:**Thisattribute specifies the direction of the scrolling text. The direction of the scrolling text can be set as left or right.

- **Height:** It specifies the height of the marquee text .The height attribute can be set as either in pixel or as a percentage of the of the screen height.

- **Width:** The width specifies the width of marquee text. It can be set either in pixel value or as a percentage of the screen height.

Examples:

<MARQUEE> Welcome to GUCDOE.</MARQUEE>.

<MARQUEE height = "50%" width = "40%" bgcolor = "blue">Welcome to GUCDOE. </ MARQUEE>

3.3.3 HTML Comment Tag

User canadd comment in an HTML document by using the following tag.

<! - - - - - - -Write your comment here - - - - - - - >

Comments are used for documenting the HTML document.

Example 3.2:

```
<! DOCTYPE HTML>
    <HTML>
        <HEAD>
            <TITLE>Use of comment tag</TITLE>
        </HEAD>
        <BODY>
            <H1>Example of using the comment
tag</H1>
            <P> Browser is a program which allows user
to access
                    document on the WWW.
            </P>
            <!..... It is about the browser.....>
        </BODY>
    </HTML>
```

In the above example, the line “It is about the browser” written in the comment line and will not display in the output window.

3.4 IMPORTING IMAGES IN HTML DOCUMENT

The <IMG> tag is used to inserting an image in the html page. The <IMG> tag has the following attributes.

- **src**: The value of the src attribute is the path or URL of the image to be inserted.

Syntax: <IMGsrc= “flower.jpg”>

- **align:**The align attribute is used to align an image on the Web page. The values of the align attribute may be set as top, middle, and bottom. The two other possible values of align attribute can also be set as the “left” and “right”.
- **alt:** The alt attribute specifies the alternative text which is actually displayed if the selected image is not displayed.
- **Height and Width:**The height and width are represented in terms in the pixel values. Height specifies the height of the image and width specifies the width of the image.

Stop to Consider

It is good practicing that specifying the height and width of the image always. Otherwise the page might flicker when the image is load.

Example 3.3:

```
<! DOCTYPE HTML>
<HTML>
    <HEAD>
        <TITLE> Inserting an Image</TITLE>
    </HEAD>
    <BODY>
        <H1>Example of inserting an image</H1>
        <IMGsrc="rose.jpg" alt="Warning" width="200"
height="300">
    </BODY>
</HTML>
```

Output of the example 3.3:

Example of inserting an image

Example 3.4:

```
<! DOCTYPE HTML>
<HTML>
    <HEAD>
        <TITLE> Inserting an Image with align attribute</TITLE>
    </HEAD>
    <BODY>
        <H1>Example of inserting an image with align attribute</H1>
        <P><IMG align="top" src="rose.jpg" alt="Warning">
            Text is aligned at the top of the image
        <P><IMG align="middle" src="rose.jpg" alt="Warning">
            Text is aligned at the middle of the image
        <P><IMG align="bottom" src="rose.jpg" alt="Warning">
            Text is aligned at the bottom of the image
    </BODY>
</HTML>
```

Output of example 3.4:

Example of inserting an image with align attribute


Text is aligned at the top of the image


Text is aligned at the middle of the image


Text is aligned at the bottom of the image

Check Your Progress-I

1. Multiple Choice Questions

(i) The tag used to specify the changes the color, size and face of the font is___.
(a) <face>
(b) <font>
(c) <fontface>
(d) <fonttype>

(ii) Which of the following hexadecimal triplet is used for set color of text?
(a) rrggbb
(b) rgrgbb
(c) rgbrgb
(d) bgbgbg

(iii)The tag allows the user to create a scrolling text is_____.
(a) <scroll>
(b) <mark>
(c) <scrolltext>
(d) <marquee>

(iv)Comments are used for _____ document.
(a) Correcting HTML
(b) Coding HTML
(c) Documenting HTML
(d) Binding HTML

(v) The alt attribute specify the_____.
   (a) preformatted text
   (b) alternative text
   (c) automated text
   (d) scrolling text

2. State True or False

(i) The <face> attribute specify the typeface of the font.
(ii) The direction of the scrolling text cannot be set as left or right.
(iii) The <img> tag is used to inserting an image in the html page.
(iv) The align attribute cannot be used with the <marquee> tag.
(v) The value of the src attribute is the path or URL of the image to be inserted.

3.5 HTML TABLES

The html table contains the rows and columns. The rows in a table are logically related. The <TABLE> tag is used to generate a table in an html page. The basic tags associated with the table are as follows.

- **<TR>……..</TR>**: It creates a row in a table.
- **<TD>……..</TD>**: Defines the table data in a row. Cell is defined by a <td> tag.
- **<TH>……..</TH> : C**reates heading in a table.
- **<CAPTION>…….</CAPTION>** : It will places a title at the top of the table.

The following attributes can be used with the <TABLE> tag.

(i) **Align**: The align attribute can be used with the cell and the table itself. The value of align attribute can be set as the **left, right** and **center.**

(ii) **Width:** It specifies the width of the table in the browser window. The width attribute can be set in the pixel value

(iii) or to a percentage of the available width of the screen. When a fixed width is required the pixel value is preferable.

(iv) **Border:** This attribute specifies the width of a border around the table. The default border value is set to zero. The standard units for width specified using the suffixes are as follows

Suffix Name	**Unit Name**
pt	Point
cm	Centimeters
mm	Millimeters
px	Screen pixels

(v) **Cellpadding:** It determines the space between the edge and the content of the cell. It can be identified in pixel value.

(vi) **Cellspacing:** It determines the amount of space between cells. It can also be identified in pixel values.

(vii) **Colspan:** The colspan attribute is used to make a cell spanning over multiple column. The default colspan is 1. This attribute is used within <th> and <td> tags.

(viii) **Rowspan:** The rowspan attribute is used to make a cell spanning over multiple rows. The default rowspan is 1.This attribute is used within <th> and <td> tags.

The content of a cell within a table can have the following format.

- **Text:** It is obvious that text can be placed in a cell in different format.
- **Images:** Images can be placed in a cell by the <img> tag between the <td> and </td> tags.

- **Blank spaces:**A blank space can be placed by putting nothing in between the <td> and <./td> tags.
- **Other Tables:** One table can be embedded to the other table .Internet explorer or Netscape Navigator support tables within tables.

Example 3.5:

```
<! DOCTYPE HTML>
<HTML>
    <HEAD>
        <TITLE> Table example</TITLE>
    </HEAD>
<BODY>
<H1>Standard units for width specified using the suffixes</H1>
    <TABLE width="75%" border="5pt" align="center"
    cellspacing="5"cellpadding="10">
    <TR>
        <TH> Suffix Name </TH>
        <TH> Unit Name   </TH>
    </TR>
    <TR>
        <TD>pt</TD>
        <TD> Point</TD>
    </TR>
    <TR>
        <TD> cm </TD>
        <TD>Centimeters</TD>
```

```
        </TR>
        <TR>
            <TD> mm </TD>
            <TD>Millimeters</TD>
        </TR>
        <TR>
            <TD>px</TD>
            <TD> Screen pixels</TD>
        </TR>
</BODY>
</HTML>
```

Output of example 3.5:

Standard units for width specified using the suffixes

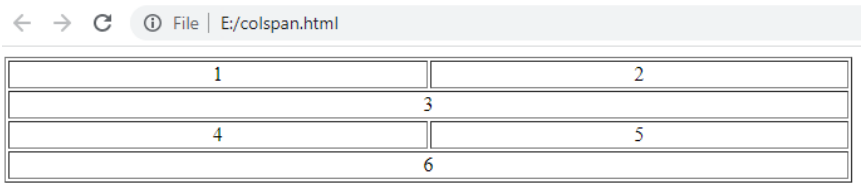
Suffix Name	Unit Name
pt	Point
cm	Centimeters
mm	Milimeters
px	Screen pixels

Example 3.6: Use of colspanattributes.

```
<! DOCTYPE HTML>
<HTML>
    <HEAD>
        <TITLE>Example of colspan attribute</TITLE>
```

```
</HEAD>
<BODY>
        <TABLE  width="50%" border="1">
        <TR align="center">
                <TD> 1</TD>
                <TD> 2</TD>
        </TR>
        <TR align="center">
                <TDcolspan="2">3</TD>
        </TR>
        <TR align="center">
                <TD>4</TD>
                <TD>5</TD>
        </TR>
        <TR align="center">
                <TDcolspan="2">6</TD>
        </TR>
        </TABLE>
</BODY>
</HTML>
```

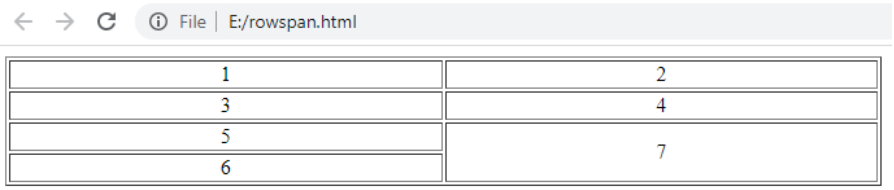
Output of example 3.6



Example 3.7: Use of rowspan attributes.

```
<HTML>
    <HEAD>
        <TITLE>Example of rowspan attributes</TITLE>
    </HEAD>
    <BODY>
        <TABLE width="50%" border="1">
            <TR align="center">
            <TD> 1</TD>
            <TD> 2 </TD>
            </TR>
            <TR align="center">
            <TD>3</TD>
            <TD> 4</TD>
            </TR>
            <TR align="center">
                <TD>5</TD>
                <TDrowspan="2">7</TD>
            </TR>
            <TR align="center">
                <TD>6</TD>
            </TR>
        </TABLE>
    </BODY>
</HTML>
```

Output of example 3.7



3.5.1 Inserting Image in a Table

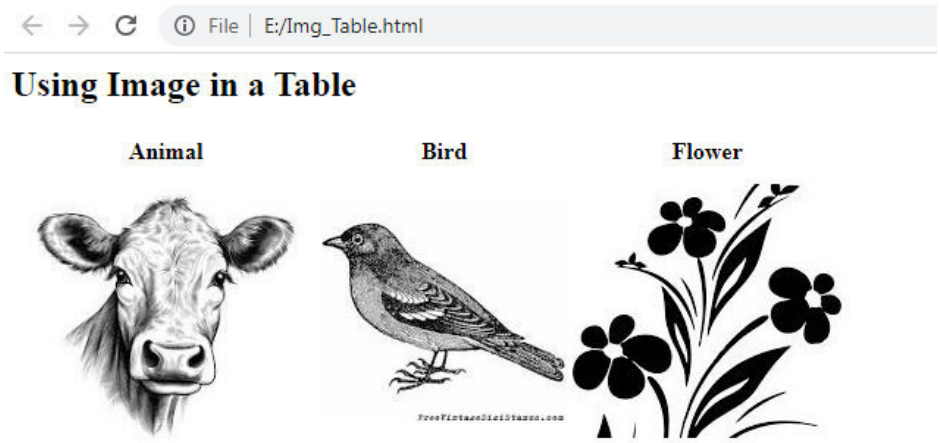
We can insert the images in a table with the help of <IMG> tag. User can insert different images in a table according to the requirements. Example 3.8 shows how image can be inserted in a table using the <IMG> tag with help of src attribute.

Example 3.8

```
<! DOCTYPE HTML>
<HTML>
    <HEAD>
        <TITLE> Inserting Image in a Table</TITLE>
    </HEAD>
    <BODY>
        <H2> Using Image in a Table</H2>
        <TABLE>
            <TR>
                <TH>Animal</TH>
                <TH> Bird</TH>
                <TH> Flower</TH>
            </TR>
            <TR>
```

```
            <TD><IMG src="animal.jpg" /></TD>
                <TD><IMG src="bird.jpg" /></TD>
            <TD><IMG src="flower.jpg" /></TD>
        </TABLE>
        </BODY>
    </HTML>
```

Output of Example 3.8



3.5.2 Creating Advanced Tables

As discussed, for creating a table, user have to use five tags which are <TABLE>,<CAPTION>,<TR>,<TH> and <TD>. HTML provides some additional elements for creating advanced table. These additional elements are:

(i) THEAD — It contains rows, heading, and cells for formation of head of a table.
(ii) TFOOT—It will create a footer in a table and appears in the end of the table.
(iii) TBODY—It contain the table data.

Creating a table using the THEAD, TFOOT, and TBODY elements is shown in the example 3.9.

Example 3.9:

```
<! DOCTYPE HTML>
<HTML>
        <HEAD>
                <TITLE>Creating an advanced
table</TITLE>
        </HEAD>
        <BODY>
                <TABLE border="1">
                <CAPTION> Details of Students
                </CAPTION>
                <THEAD>
                <TR>
                                <TH>Student_ID</TH>
                                <TH>Student_Name</TH>
                                <TH>Student_Address</TH>
                </TR>
                </THEAD>
                <TFOOT>
                <TR>
                        <TH colspan="3"> This table contains
                        the information about the
                        students.</TH>
                </TR>
                </TFOOT>
                <TBODY>
                <TR>
                        <TD>012024</TD>
                                <TD>SurajShirke</TD>
                                <TD>Mumbai</TD>
                </TR>
                <TR>
                        <TD>022024</TD>
                                <TD>SwapanGhosh</TD>
                                <TD>Kolkata</TD>
                </TR>
                        <TR>
                        <TD>032024</TD>
                                <TD>Amar Kalita</TD>
```

```
                    <TD>Assam</TD>
                </TR>
        </TBODY>
    </TABLE>
    </BODY>
    </HTML>
```

Output of example 3.9:

Details of Students

Student_ID	Student_Name	Student_Address
012024	Suraj Shirke	Mumbai
022024	Swapan Ghosh	Kolkata
032024	Amar Kalita	Assam
This table contains the information about the students.		

3.5.3 Nested Tables

In HTML, We can create a table inside another table. For example if we need to design a web page which will required two tables for storing the information separately. In this situation, we have to create the nested tables. The example 3.10 shows the creation of nested table for storing the information of some students.

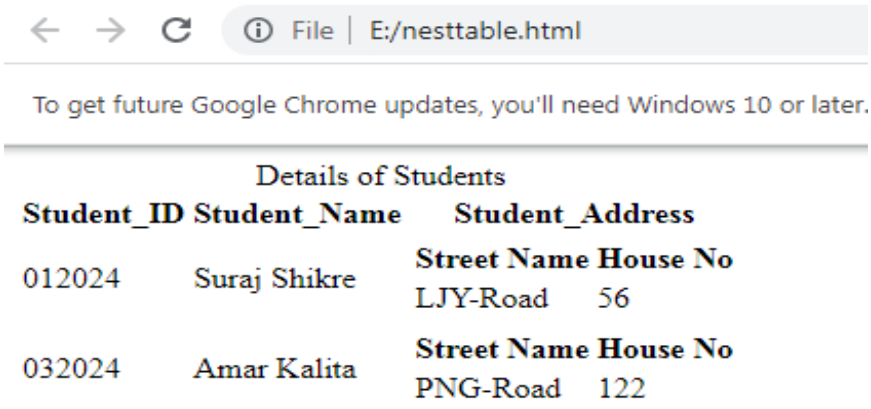
Example 3.10:

```
<! DOCTYPE HTML>
<HTML>
    <HEAD>
        <TITLE>Creating nested table</TITLE>
    </HEAD>
    <BODY>
        <TABLE>
```

```
<CAPTION> Details of Students  </CAPTION>

        <TR>
                <TH>Student_ID</TH>
                <TH>Student_Name</TH>
                <TH>Student_Address</TH>
        </TR>
        <TR>
                <TD>012024</TD>
                <TD>SurajShikre</TD>
        <TD>
        <TABLE>
                <TR>
                        <TH> Street Name</TH>
                        <TH>House No</TH>
                </TR>
                        <TR>
                                <TD> LJY-Road</TD>
                                <TD>56</TD>
                        </TR>
        </TABLE>
        </TD>
</TR>
        <TR>
                <TD>032024</TD>
```

```
<TD>Amar Kalita</TD>
        <TD>
        <TABLE>
            <TR>
                <TH> Street Name</TH>
                <TH>House No</TH>
            </TR>
            <TR>
                <TD> PNG-Road</TD>
                <TD>122</TD>
            </TR>
        </TABLE>
        </TD>
    </TR>
</TABLE>
</BODY>
</HTML>
```

Output of Example 3.10



3.6 HTML FORMS

Forms make a web page more interactive than any other document in the web pages. In a web page, user can submit their information and queries in a structured way by using form. This is the only way through which one can send a request in the web. Forms enables user in different applications in the web.

Form can be used for

- Customer and student registration.
- Compliance, financial and operational audits.
- Gather feedback from customer.
- Reporting workplace incident.
- Automate sales order, service requests and billing process.
- Track assets, stock levels, and manage inventory. etc.

An HTML form is created with the help of <form> tag. The content of the form are placed in between the start tag <form> and end tag </form>.

The form element has the following attributes.

- **Action:** The action attribute takes the URLs as its arguments. The action is set equal to the URL of the processing script so that the browser knows where to send the form data after it is entered.
- **Method:** The method attribute specifies how to send form data. The method attributes can have two values either GET or POST. GET method is less secure in comparison to the

post method. If user wants to bookmark the results then he or she can choose the get method. On the other hand post method does not cache the data it will directly send the form input to the server. Post request is secured because it does not exposed data in the URL bar.

The common tags that associated with the form are:

- **INPUT TAG**: The input tag is used to input something in the form and it defined by the <input> tag. It tells the server that what type of data to be submitted by the form. The attributes of the <input> tag are:
  1. **Type:** Type attribute specifies type of input control to create on the form. Type attributes can be written as the following format:
     Type= "text | password | checkbox | radio | submit | reset | button | file"

  2. **Name: T**he name attributes specifies a name to the control.
     Example: <input type= "text" name= "first_name">

  3. **Value:** This attribute specifies the value given to the control. This value attribute is optional except the radio button.
     Example: <input type= "radio" value= "rose">

  4. **Size:** These attribute species the size of the control. It will define the number characters that a text element can display without scrolling.
     Example : <input type= "text" size = "20">

  5. **Maxlength:** This attributes specifies the maximum number characters that can be allowed in a text box. It is used when the text type is text or password.
     Example : <input type= "text" maxlength= "8">

  6. **Checked :** The checked attribute is Boolean attribute. It specifies the input element is pre-selected when page is load. The checked attribute can be used with the checkbox and the radio button.

7. **Type**: It defines the type which is given in input control when creating a form. Type can take any of the following values:

   (i) **Text**: This will create a single line text box.
   <input type= "text">

   (ii) **Password:** This control is used to input a password that is it will hide the inputted characters. But whenever submitted the form, the hidden characters are saved as the inputted text.
   <input type= "password" >

   (iii) **Radio:** It will create a radio button on the form for single selection. Several radio buttons within the same name may appear in a form. But only one name and value pair is submitted finally.
   <input type= "radio" name= "rose" value= "yes">

   (iv) **Checkbox:** Using checkbox user can select zero or more options from a limited number of choices. The checkboxes value is only submitted with form when it is checked. Checkbox allows the user to select more than one value for a given property.
   <input type= "checkbox" name= "flower" value= "rose" >

   (v) **Image:** It will create a graphical submit button.
   <input type= "image" src = "submit_img.gif">

   (vi) **Submit:** It will create a submit button. The form is submitted in the respected URL which is specified in the action attribute after user click on this submit button.
   <input type= "submit" value= "submit the form">

(vii) **Reset:** It will create a reset button. Whenever user activated the reset button all the controls of the form reset to the initial control of the form.
<input type= "reset" value = "reset the form">

(viii) **File:** It is used for uploading a file. It will define a file selected field and create a browse button. By clicking on the browse button user can select the path for browsing the respected files.

3.7 TEXTAREA TAG

The <TEXTAREA> tag is similar to the input tag. It is used for creation of multiline text input control.

The default text is placed between the **<TEXTAREA>…….</TEXTAREA>** tags. It is basically used for collecting the user's input as review or comment. The attributes of the <TEXTAREA> tags are as follows.

(i) Name: Name attribute is used for assigning a name to the element.
(ii) Rows: It will specify the number of visible text lines.
(iii) Cols: It will specify the visible width in average character widths.

Syntax:

<TEXTAREA name="w3comment" rows="number of rows" cols="number of columns">

The default text goes here ……

</TEXTAREA>

Example 3.11

<! DOCTYPE HTML>

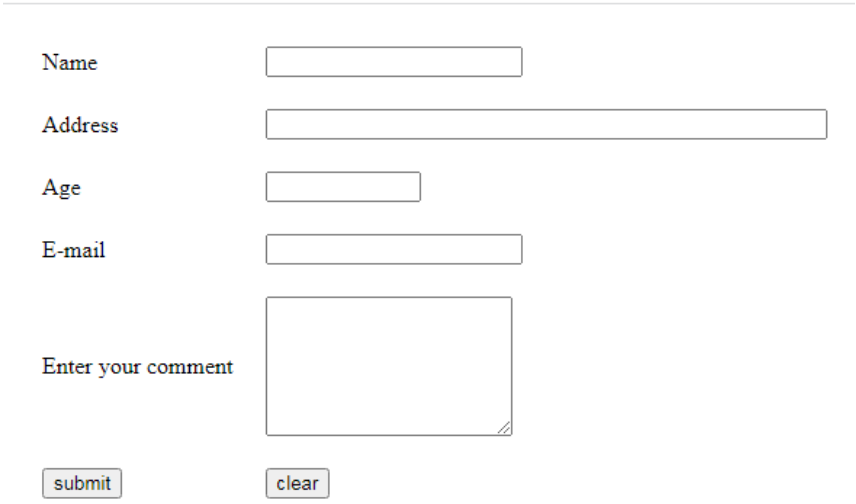
<HTML>

<HEAD>

```
            <TITLE> Form example</TITLE>
        </HEAD>
<BODY>
<TABLEcellspacing= "20">
<TR>
        <TD> Name </TD>
        <TD>
            <INPUT type="text" name="name" size="20"
            maxlength="20">
        </TD>
</TR>
<TR>
        <TD> Address </TD>
<TD>
        <INPUT type="text" name="name" size="50"
        maxlength="50">
        </TD>
</TR>
<TR>
        <TD> Age </TD>
        <TD>
        <INPUT type="text" name="name" size="10"
        maxlength="2">
        <TD>
</TR>
<TR>
```

```
        <TD> E-mail </TD>
<TD>
        <INPUT type="text" name="name" size="20"
        maxlength="20">
        </TD>
</TR>
<TR>
        <TD> Enter your comment </TD>
<TD>
        <TEXTAREA name="comments" rows="6">
        </TEXTAREA>
        </TD>
</TR>
<TR>
<TD>
        <INPUT type="submit" value="submit" name="send">
        </TD>
<TD>
        <INPUT type="reset" value="clear" name="reset">
        </TD>
</TR>
</BODY>
</HTML>
```

Output of example of 3.11:



3.8 THE <SELECT> AND <OPTION> TAGS

The<SELECT> tag is used to create a drop-down list. It is basically used for collect the user input.

The name attribute of the <SELECT> tag defines the name of the element. If user deletes the name attributes, no data will be selected from the drop-down list.

The <OPTION> inside the <SELECT> tag define the available options in the drop-down list.

The <OPTION> tag has the following two attributes

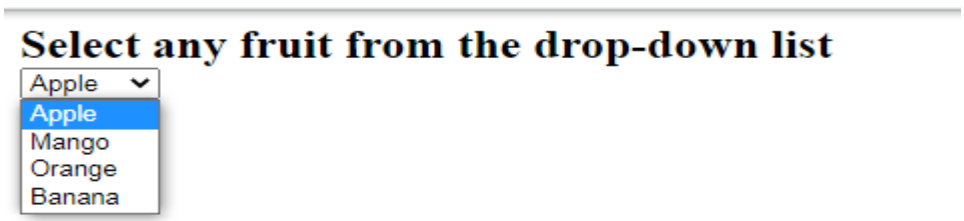
(i) **Selected:** when this attribute is selected or set it determines which option is selected by default.
(ii) **Value:** This attribute specifies the value to be submitted for the particular choice.

Example 3.12shows how a drop-down list is created using SELECT and OPTION elements.

Example 3.12:

```
<! DOCTYPE HTML>
        <HTML>
                <HEAD>
                        <TITLE> Example of Select and
option</TITLE>
                </HEAD>
        <BODY>
                <FORM>
                <H2> Select any fruit from the drop-down list<BR>
<SELECT>
                <OPTION value="apple">Apple</OPTION>
                <OPTION value="mango">Mango</OPTION>
                <OPTION value="orange">Orange</OPTION>
                <OPTION value="banana">Banana</OPTION>
</SELECT>
</FORM>
</BODY>
</HTML>
```

Output of 3.12



Check Your Progress-II

3. Multiple Choice Questions

(i) The tag used to formatting a table in an HTML document is ___.

(a) <TLB>
(b) <TABLE>
(c) <tableformat>
(d) <TBD>

(ii) The tag used to places a title at the top of the table is____.

(a) <DT>
(b) <DL>
(c) <TR>
(d) <CAPTION>

(iii) The space between the edge and the content of the cell is called____.

(a) CELLPADING
(b) CELLSPACING
(c) BORDER
(d) CAPTION

(iv) An HTML form is created with the help of _____ tag.

(a) <FRM> tag
(b) <FORM> tag
(c) <HTMFRM> tag
(d) < CAPTION>

(v) Method is an attribute of_____.

(a) TABLE element
(b) IMG element
(c) FORM element
(d) SRC element

5. Write HTML code for formatting the following.

Which of the following team won the 1983 world cup cricket?

- ○ **Australia**
- ○ **India**
- ○ **West Indies**
- ○ **Sri Lanka**

Submit

6. Write HTML code for formatting the following.

Which of the following team has won the ICC man's World Cup ?

- ☐ **Australia**
- ☐ **India**
- ☐ **West Indies**
- ☐ **Sri Lanka**

Submit

3.9 INTERACTIVE ELEMENTS

An interactive web page allows user to easily perform the task associated with that web page.

3.9.1 The DETAILS and SUMMARY elements

The DETAILS and SUMMARY elements newly introduced in HTML 5. The DETAILS element is used to display the additional details about a document. Suppose we want to display an article then DETAILS element is used to display publishing date, author name and SUMMARY element is used to provide the summary of the current document. Example of DETAILs and SUMMARY elements are as follows

<DETAILS>

<SUMMARY>This contains the summary of the document. </SUMMARY>

</DETAILS>

3.9.2 The MENU element

The MENU element is used to display menu items on a web page. The MENU item contains one or more menu items which is defined by the <LI> element. Two attributes **label** and **type** are used with the <MENU> element. Example for the use of <MENU> item is provided as follows.

```
<MENU>
  <LI><INPUT type= “radio”/> Apple</LI>
  <LI><INPUT type= “radio”/> Mango</LI>
</MENU>
```

3.9.3 The COMMAND element

The COMMAND element is used to execute commands with help of command button. The examples of the command button are radio button, checkbox or it may be a simple text, on a web page. The example 3.13 shows the use of the COMMAND element.

Example 3.13:

```
<!DOCTYPE HTML>
<HTML>
<HEAD>
<TITLE>Example of Command button </TITLE>
</HEAD>
<BODY>
  <H2>Using the command element</H2>
<MENU>
  <COMMAND onclick= "alert ("use command button")"> Click Here</COMMAND>
</MENU>
```

</BODY>

</HTML>

3.10 SUMMING UP

- The tag <FONT> is used to specify the changes the color, size and face of the font.
- The text Color can be set by using one of the color name or as a hexadecimal ‘rrggbb’ triplet value.
- The tag <MARQUEE> allows the user to create a scrolling text.
- The <IMG> tag is used to inserting an image in the html page.
- The value of the src attribute is the path or URL of the image to be inserted.
- The <TABLE> tag is used to generate a table in an html page.
- <TR>, <TD>, <TH>, <CAPTION> are the basic tags associated with the table.
- align, width, border, cellpadding, cellspacing, colspan, rowspan are the attributes associated with the <TABLE> tag.
- Forms make a web page more interactive than any other document in the web pages.
- Action and method are the two attributes associated with the <form> tag.
- The method attributes can have two values either GET or POST.
- POST method does not cache the data it will directly send the form input to the server.
- <input>,<textarea> are the common tags associated with the form element.
- Creating a drop-down list the <select> and <option> tag is used.
- The <DETAILS> and <SUMMARY> elements is newly introduced in HTML5.
- For displaying the menu items the <MENU> element is used.

- In HTML the COMMAND element is used to execute the command.

3.11 ANSWERS TO CHECK YOUR PROGRESS

1. (i) (b) (ii) (a) (iii) (d) (iv) (c) (v) (b)

2. (i) True (ii) False (iii) True (iv) False (v) True

3. (i) (b) (ii) (d) (iii) (a) (iv) (b) (v) (c)

4. (i) True (ii) False (iii) False (iv) True (v) True

5.

```
 <! DOCTYPE HTML>
<html>
  <head>
  <title>Creating Radio button</title>
</head>
  <body>
        <form method="post" action=" ">
<h1> Which of the following team won the 1983 world cup
cricket?<br><br>
<input type="radio" name="quiz" value="Australia">Australia<br>
<input type="radio" name="quiz" value="India">India<br>
<input type="radio" name="quiz" value="West Indies">West
Indies<br>
<input type="radio" name="quiz" value="Sri Lanka">Sri
Lanka<br><br>
<input type="submit" value="Submit" name="submit"><br>
</form>
</body>
</html>
```

6.

```
<! DOCTYPE HTML>

<html>
        <head>
        <title>Creating Check boxes</title>
        </head>
                <body>
                <form method="post" action=" ">

<h1> Which of the following team has won the ICC man's World
                                Cup ?<br><br>

<input type="checkbox" name="quiz1"
value="Australia">Australia<br>
<input type="checkbox" name="quiz 2" value="India">India<br>

<input type="checkbox" name="quiz 3" value="West
                        Indies">West Indies<br>

                        <input type="checkbox" name="quiz 4"
                        value="Sri Lanka">Sri Lanka<br><br>
<input type="submit" value="Submit" name="submit"><br>
</form>
</body>
</html>
```

3.12 POSSIBLE QUESTIONS

1. Explain the attributes associated with the <FONT> element.
2. What are the uses of <MARQUEE> element?
3. How can you insert a comment in an HTML document? Explain it with a suitable example.
4. What are the different attributes associated with the <IMG> element?
5. How table is inserted in an HTML document?
6. What are the uses of rowspan and colspan attributes in an HTML document? Explain with suitable examples.
7. Explain the <THEAD>,<TBODY> and <TFOOT> elements.

8. Explain the nesting table with the help of a suitable example.
9. How can you insert images in a table?
10. What is the use of form? Explain the action and method attribute of form element.
11. Explain the different attributes of <INPUT> tag.
12. Explain the <TEXTAREA> tag with an example.
13. Explain the <SELECT> and <OPTION> element with a suitable example.
14. Explain the Details and Summary element.
15. Differentiate between MENU and COMMAND element

3.13 REFERENCES AND SUGGESTED READINGS

- https://www.w3schools.com/
- HTML 5 BLACK BOOK Published by dreamtech press.

---×---

UNIT-4
CASCADING STYLE SHEET (CSS)

UNIT STRUCTURE

4.1 INTRODUCTION

Cascading Style Sheet(CSS) is a text file with the .css extension and commonly used to define the style on web pages written in HTML. It defines the styles in a web page like font size, text format, background color etc. Using CSS, user can use the same

style in multiple web pages. User can apply the styles in multiple components in a single web page. If a web page contains multiple tables then all the tables are styled writing just a single style code. CSS reduces the complexity and the redundancy of the code in a web page. The CSS file contains the style code for the structure of a web page and these styles can be modified according to the requirement.

4.2 OBJECTIVES

After going through this unit learner will able to:

- Understand the evolution of CSS.
- Learn how to useid and class selector in CSS.
- Learn about the different types of CSS.
- Understand the CSS colors and CSS fonts and font styles.
- Learn how to insert CSS tables and its different properties.
- Understand the CSS lists and forms.
- Understand the CSS counters, CSS Responsive and CSS website Layout

4.3 EVOLUTION OF CSS

CSS was introduced in 1996 on the recommendation of World Wide Web Consortium (W3C). CSS also used in the different languages like Extensible Markup Language(XML) and XHTML. The CSS1 was introduced in 1996 and after that CSS2 and CSS3 was introduced. CSS1 version includes the features like margin, borders, padding and positioning the elements. It also provides the text attributes, color to text and background elements. CSS2 version includes providing the styles for different media types, font properties such as shadows and the bidirectional text. CSS3 version includes the features like support more color for wider range, allows multiple background on a web page, multicolumn text without using table, provides CSS selectors, provide custom fonts etc.

4.4 SYNTAX OF CSS

CSS uses syntax to apply CSS rules in an HTML document. The Syntax of a CSS document is as follows.

Syntax:

Selector

{

1st property: value;

2nd property: value;

3rd property: value;

.....

……..

nth property: value;

}

Example:

h1{ color: blue ; font-size=20px;}

The selectors are normally an HTML element user want to style. The CSS rule has two different parts, one part is the selector and the other part is declaration. Each selector declaration has a property and a value. In the above example the text written under h1 have the blue color with 20 pixel font size.

The different types of selectors are as follows:

(i) **The Universal Selector:** The universal selector selects all the elements that are present in an HTML document. This selector is used to apply same rule to all the elements in the HTML document. The symbol asterisk (*) is used to represent universal selector:
The use of universal selector can be shown in the following manner:

*{

```
        Margin: 0;
        Padding: 0;
    }
```

In the preceding code, the margin and padding properties are set to 0 that means all the elements in the HTML document follow these properties.

Example 4.1:

```
<!DOCTYPE html>

<html>
    <head>
        <style type="text/css">
         *
        {
            margin: 0;
             padding: 0;
         }
        </style>

</head>

    <body>
        <h1>GUCDOE</h1>
        <h1>This paragraph is styled with CSS.</h1>
    </body>
</html>
```

(ii) The Type Selector: The type selector is used to select a specified element in an HTML document. The styles which are set to a particular type can only be effect to that type.

Example 4.2:

```
<!DOCTYPE html>
<html>
```

```
<head>
    <style type="text/css">
    h1
    {
    color:red;
     text-align:center;
     }
    </style>
</head>
<body>
    <p>GUCDOE</h1>
    <h1>This paragraph is styled with CSS.
</h1>
</body>
</html>
```

In the example 4.2 the properties set to h1 type selector this means the text written in between <h1> and </h1> only be changed according to the value.

(iii) The Class Selector: The class selector allows user to apply CSS rules to the element that carries a class. This allows user to set particular style for any HTML document with the same class. The class selector is defined with a ".".

Example 4.3:

```
<!DOCTYPE html>
<html>
    <head>
```

```
        <style type="text/css">
            .info
            {
            text-align:center;
            }
        </style>
    </head>
    <body>
        <h1 class="info">Center-aligned
      heading</h1>
        <p class="info">Center-aligned
    paragraph.</p>
    </body>
</html>
```

(iv) **The ID Selector:**The value of the ID element is unique within a document. The ID selector is applied to the content of the single element. The ID selector is defined with "#" symbol.

Example 4.4:

```
<!DOCTYPE html>

    <html>
        <head>
            <style type="text/css">
                #cdoe
                {

text-align:center;

color:red;

                }
        </style>
        </head>
        <body>

            <p id="cdoe">Gauhati University, Centre for
            Distance and Online Education</p>
```

```
        <p>This paragraph is not affected by the
    style.</p>
    </body>
</html>
```

4.5 TYPES OF CSS

CSS are categorized on the basis of CSS style applies on an HTML document. There are three ways to apply CSS style to an HTML document and these are:

(i) The internal style-sheet
(ii) The external style-sheet
(iii) The in-line style-sheet

(i) The internal style-sheet: When a single document required unique style the internal style-sheet is used. The internal style-sheet is written within HEAD element of the HTML document. The syntax of internal style sheet is written as

```
<style type= "text/css"
    selector { property: value: }
</style>
```

The following code snippet shows a sample CSS document written using the internal style sheet.

```
<head>
    <style type= "text/css">
        hr {color:blue;}
        p{margin-left-20px;}
    </style>
</head>
```

Advantages and Disadvantages of internal style sheet:

Advantages:

- Affect only in the page in which the styles are placed.
- Allows user to change the style of the same HTML in which they are working.

Disadvantages

- If users wish to use same style in different document, need to repeat them for every document.
- Increase the page load time.

(ii) **External Style-sheet:**In external style-sheet, the CSS file is written outside the HTML document and the reference for the external style-sheet is placed in the HTML document. In an external style sheet the styles are saved in single document with .css extension. Each page must link with the .css file using the <link> tag. The <link> element goes inside the head section. The attributes associated with the <link> element are rel, type and href .The <link> element goes inside the head section are as follows:

```
<head>
   <link rel= “stylesheet” type= “text/css” href= “hello.css”/>
</head>
```

Here the value of rel attribute is set to stylesheet, the value of type attribute is set to text/css, and that href attribute is set to hello.css.

Example 4.5:

```
<!DOCTYPE html>

 <html>
 <head>
 <link rel="stylesheet" type="text/css" href="hello.css" />
```

```
</head>
<body>
<p>GUCDOE
<hr>
<h1> Welcome to GUCDOE
</body>
</html>
```

Here the **hello.css** file is shown as follows

```
h1
{
        background-color:blue;
}
p
{
        background-color:red;
}
```

Now if we run the html file through the web browser, the style defined in the hello.css will change the background of the text of <p> and <h1> element respectively.

Advantages and disadvantages of external style sheet:
Advantages:

- There is no requirement of defining a specific style for every element.
- Allows user to group different styles in a more efficient manner.

Disadvantages:

- Increasing the downloading time as the entire CSS file has to be downloaded first.
- It will display the web page only after the entire style sheet is loaded.

(iii) **Inline Style-sheet:**The inline style sheet properties are written in single and separated by semicolons. The properties are placed inside the style attribute of the

HTMLelement. The example shows how to change the color and the left margin of a paragraph.

```
<p style="color:blue;margin-left:40px">This is a
paragraph.</p>
```

Advantages and disadvantages of inline style sheet:

Advantages:

- Inline style sheet has the highest precedence over the external and internal style sheet
- Provide quick and easy approaches to add styles in a web page.
- User need not to create the whole document for adding inline styles.

Disadvantages:

- Inline style must be applied to every element on which user want to apply a style.
- Inline style sheet does not allow the elements, which has some special effect.

4.6 CSS COLORS

CSS enables user to add different types of colors in their own web pages with different specifications. Color can be defined with the predefined color name or the special specifications. These specifications are RGB, RGBA, HSL and HSLA. Example 4.6 and example 4.7 shows how pre-defined color and RGB color changes the background of the web page.

Example 4.6

```
<!DOCTYPE html>
<html>
<body>
<h1 style="background-color:Tomato;">Tomato</h1>
<h2 style="background-color:Orange;">Orange</h2>
<h3 style="background-color:Gray;">Gray</h3>
<h4 style="background-color:SlateBlue;">SlateBlue</h4>
<h5 style="background-color:Violet;">Violet</h5>
</body>
</html>
```

Example 4.7

```
<!DOCTYPE html>
<html>
<body>

<h4>Specify colors using RGB values</h4>

<h1 style="background-color:rgb(0, 0, 0);">The background color
is black</h1>
<h2 style="background-color:rgb(0, 0, 255);">The background
color is blue</h2>
<h3 style="background-color:rgb(0, 255,0);">The background color
is green</h3>
<h4 style="background-color:rgb(255, 0, 0);">The background
color is red </h4>
<h5 style="background-color:rgb(255, 255, 255);">The background
color is white </h5>
</body>
</html>
```

The hexadecimal color specified with the #RRGGBB or #rrggbb and the HSL color can be defined with hue, saturation and lightness. Examples 4.8 shows how background color can be change with the help of hexadecimal color and the HSL colors.

Example 4.8:

```
<!DOCTYPE html>
<html>
<body>

<h1>background color using hexadecimal and HSL color</h1>
<h2 style="background-color:#000000;">The background color is
        black</h2>
<h2 style="background-color:#ff0000;">The background color is
        red</h2>
<h2 style="background-color:#00ff00;">The background color is
        green</h2>
<h2 style="background-color:#0000ff;">The background color is
        blue</h2>
<h2 style="background-color:hsl(0, 100%, 50%);">hsl(0, 100%,
        50%)</h2>
<h2 style="background-color:hsl(240, 100%, 50%);">hsl(240,
        100%, 50%)</h2>
<h2 style="background-color:hsl(147, 50%, 47%);">hsl(147, 50%,
        47%)</h2>
```

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

STOP TO CONSIDER

The size of a back-ground image cannot be defined in negative values.

4.7 CSS FONTS AND TEXT STYLE

Fonts represent the style and size of the text that is displayed in a web browser. The designing of a web page depends on what type of font, size and styles are chosen.

4.7.1 The Font-family Property

Fonts are categorized with different types of font families which are listed below.

(i) **Serif –**It refers to the font family in which the width of character is proportional and the font have a small strokes at the end of the each letter. Some of the fonts are included in this family are Times New Roman, Georgia, Book Antiqua, Bookman Old Style, Rockwell etc.

(ii) **Sans-serif**- It refers to the font family in which the width of characters is proportional but there is no small strokes attached with the letters. Some of the fonts that are included in this family are Microsoft sans Serif, Verdana, Tahoma, Arial, Arial Black etc.

(iii) **Cursive**- It refers to the font which is appears as human hand writing.Some of the fonts that are included in this family are Comic Sans Ms, Monotype Corsiva, French Script MT, Tempus Sans ITC, Vivaldi etc.

(iv) **Fantasy**-In this font family font characters cannot be characterized in asingle rule. The font characters in this font family are decorative. Some of the fonts included in this family are Impact, Papyrus, Copperplate Gothic Light, Copperplate Gothic Bold, Curiz MT, Rockwell Exra Bold etc.

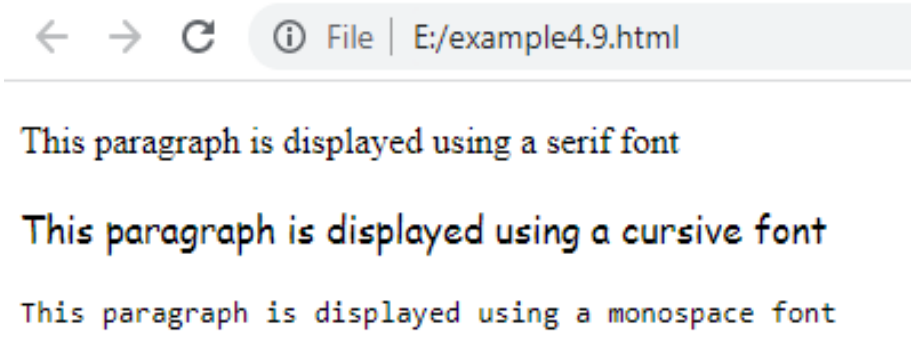
(v) **Monospace**- It refers to the font family in which displays the text written with a type writer. In this family the width of the characters are same. Some of the fonts of this family are Courier New, Lucida Console, Lucida Sans Typewriter, Andale Mono IPA etc.

Example 4.9 shows the creation of a web page using different font families.

Example 4.9

```
<!DOCTYPE html>
<html>
<head>
<title> Example of font-family</title>
</head>
<body>
<p style="font-family:serif">
      This paragraph is displayed using a serif font
</p>
<p style="font-family:cursive">
      This paragraph is displayed using a cursive font
</p>
<p style="font-family:monospace">
      This paragraph is displayed using a monospace font
</p>
</body>
</html>
```

Output of example 4.9



STOP TO CONSIDER

You can specify more than one font family in the font-family property. If one is not installed in the system then font-family property will take the second one.

4.7.2 The font-size Property

The font-size property is used to change the size of the text. In general, the value of these properties is specified in pixel. The font size can also be specified by the following three ways.

(i) **Using absolute values**—Absolute value refers the absolute size of the font. The absolute value can be categorized as small,medium,large,x-small,x-medium,x-large,xx-small,xx-medium,xx-large. The xx-large is the largest text size whereas the xx-small is the smallest text size.

(ii) **Using relative values**- The relative refers that the font-size is not fixed. This relative values are calculated on the basis of the current font values. Relative values can be set by specifying the font-size as larger or smaller depending on the requirement of the font size.

(iii) **Using percentage value**- User can change the font size by specifying percentage value in the font-size property.

The example 4.10 shows the font sizes using absolute values, relative values and percentage values.

Example 4.10

```
<!DOCTYPE HTML>
<html>
    <head>
    <title> Font-size</title>
    <style type="text/css">
   body
      {
       font-size:medium
       }
```

```
</style>

</head>

<body>

<h1> Working with font-size property using absolute values</h1>

<li><p style="font-size:xx-small">small font size</p></li>

<li><p style="font-size:medium">medium font size</p></li>

<li><p style="font-size:xx-large">large</p></li>

<h1> Working with font-size property using relative values</h1>

<li><p> Base font value</p></li>

<li><p style="font-size:larger"> This font is larger than base
font</p></li>

<li><p style="font-size:smaller"> This font is smaller than base
font</p></li>

<h1> Working with font-size property using percentage
values</h1>

<li><p style="font-size:100%"> This font size is same as the default
size</p></li>

<li><p style="font-size:200%"> This font size is double of the
default size</p></li>

<li><p style="font-size:50%"> This font size is half of the default
size</p></li>

</body>

</html>
```

Check Your Progress I

1. **State True or False**

   (i) CSS is a text with .css extension.
   (ii) CSS was introduced on the recommendation of W3C.
   (iii) CSS rules cannot be applied in HTML document.
   (iv) Selectors are normally an HTML element.
   (v) The Symbol # is used to represent universal selectors.

2. **Fill in the blanks**

   (i) The _____selector is defined with dot (.) operator.
   (ii) When a single document required unique style the _____ style-sheet is used.
   (iii) In _____ style sheet the CSS file written outside the HTML document.
   (iv) The _____color specified with #rrggbb.
   (v) The _____ property is used to change the size of the text.

4.8 CSS BORDER, PADDING AND MARGIN

The CSS border, padding and margin can be represented using box-model. The different areas that can be represented using box-model are shown in Figure 4.1.

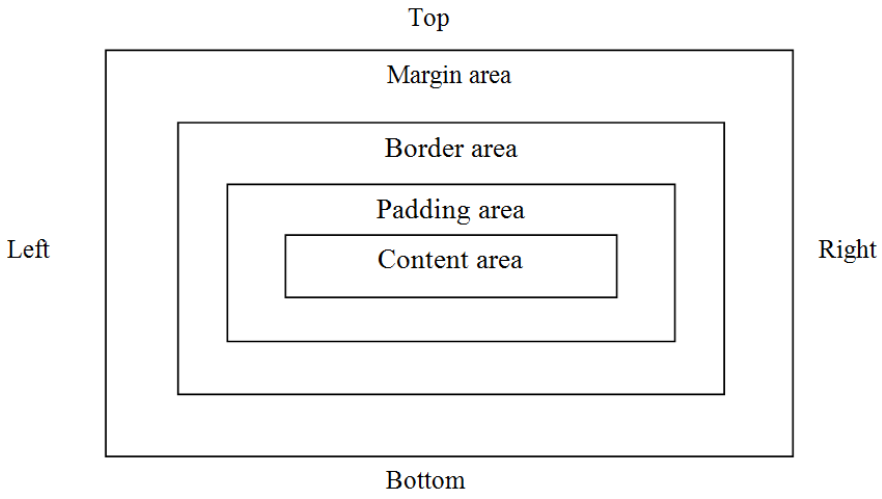


Figure 4.1: Displaying the Areas of Box Model

In Figure 4.1, the content area displays the content of a document. The padding area specifies the area around the content area, the border area specifies the area around the padding area and margin area specifies the area around the border area.

4.8.1 The Padding Properties

Padding specifies the distance between border and the content area. The negative value cannot be accepted by the padding property. The padding has the following properties

- padding-top
- padding-bottom
- padding-right
- padding-left

If the padding property has four values, then the values are applied to the top, right, bottom, and left padding respectively. The example 4.11 shows the padding properties top, bottom, left and right are set with the <p> element.

Example 4.11:

```
<!DOCTYPE HTML>

<html>

<head>
```

```
<style type="text/css">
p
{
 background-color:blue;
}
    p.padding
     {
       padding-top:20px;
       padding-bottom:20px;
       padding-right:10px;
       padding-left:10px;
      }
</style>
</head>
<body>
<p class="padding">padding with top and bottom are 20px,right and left are 10px</p>
</body>
</html>
```

4.8.2 The Border Properties

The border of a document specifies the space between the margin and the padding of a document. It defines the width, color and style of a border area of a box. The properties of border are as follows.

(i) **border -width**- It specifies the width of a border area. The possible values of a border-width are thin, medium and thick.

(ii) **border –color**- It specifies the color of a border. Border colors are defined using one of the following values.

- **Name**- It specifies the name of the color such as red, blue and green.
- **Red GreenBlue (RGB)-**Itdefines RGB value, such as rgb(255,0,0).
- **Hex**-It defines the Hex value such as #00ff00.

(iii) **Border-style**- It indicates the format of the border, such as solid, dashed, dotted or double. The border-style property can be used with the four directions, top, bottom, left and right.

Example 4.12 shows the uses of border property in CSS.

Example 4.12:

```
<!DOCTYPE HTML>
<html>
<head>
<style type="text/css">
    p.border1
    {
     border-style:double;
     border-width:10;
     border-color: blue
     }
</style>
</head>
<body>
<p class="border1">double line blue color border of width 10</p>
</body>
</html>
```

4.8.3 The Margin Properties

The margin area is used to create an extra space around an element. It does not contain any background color. It is also used to

determine the spacing around different elements. The margin property is used to set all the sides of an element such as top, bottom, right and left. The margin property can be changed independently by using different properties, such as margin-top, margin-right, margin-bottom and margin-left. Example 4.13 shows the working of the margin property.

Example 4.13

```
<!DOCTYPE HTML>

<html>

<head>

<style type="text/css">

p.margin

{

margin:100px 50px;

 }

</style>

</head>

<body>

<p class="margin">margin with top and bottom have 100px and left and right have 50 px</p>

</body>

</html>
```

4.9 CSS LISTS

CSS allows user to set various list-style-property to define the style of the list items. The type of list marker for an unordered list as follows

- **None** - It does not specify a marker for a list item.
- **Disc** - It specifies a filled circle as a marker of the list item.
- **Circle** -It specifies a circle as a marker of the list
- **Square**- it specifies a square as a marker of the list.

For an ordered list the type of list marker are as follows.

- **lower-roman**-It specifies that a marker is displayed in the lower-roman style, such as i ,ii, iii,iv,……
- **upper-roman**-It specifies that a marker is displayed in the upper-roman style, such as I,II,III,IV,……
- **lower-latin**- It specifies that a marker is displayed in the lower-latin style, such as a,b,c,d,…..
- **upper-latin**- It specifies that a marker is displayed in the upper-latin style, such as A,B,C,D,…..
- **decimal** - It specifies that a marker is displayed as decimal number.
- **lower-alpha**-It specifies that a marker is displayed in the lower-alpha style, such as a,b,c,d,…….
- **Upper-alpha**- It specifies that a marker is displayed in the upper-alpha style, such as A,B,C,D…….

Example 4.14 shows the code to display unordered and ordered list item marker.

Example 4.14

```
<!DOCTYPE HTML>
    <html>
    <head>
    <style type="text/css">
        ul.m{list-style-type:circle;}
        ol.n{list-style-type:lower-roman;}
    </style>
    </head>
    <body>
    <p> unordered list marker</p>
    <ul class="m">
    <li>item 1</li>
    <li>item 2</li>
```

```
<li>item 3</li>
</ul>
<p>ordered list marker</p>
<ol class="n">
<li>item 1</li>
<li>item 2</li>
<li>item 3</li>
</ol>
</body>
</html>
```

4.10 CSS TABLES

You have already learned to create tables using the table element of HTML. You can also customize the layout and the appearance of the table using a single CSS file. There are different types of table properties of CSS which are as follows

4.10.1 The Table-Layout Property and Caption-Side Property

The table-layout property specifies the way that a table should be displayed in a web browser. It will set the position of the table that means according to the requirement the table can be moved to different locations. The table-layout-property has the following values:

(i) **auto** – It specifies the layout which the web browser automatically resizes a table according to the content of the table. It is the default value of the table-layout property.

(ii) **fixed**- In this layout the web-browser does not resize the table according to the content. This value reads only the first row of a table to determine the final layout of the table.

The caption-side property displays the position of the caption in a table. The table caption is the small description of a table. The values of the caption-side propertyare top and bottom. Top defines a caption on the top of a table and bottom defines a caption on the bottom of a table.Example 4.15 shows the working with table layout and caption-side properties in CSS.

Example 4.15

```
<!DOCTYPE HTML>
<html>
<head>
<style type="text/css">
    table,td,th
      {
        border:1px solid black;
      }
    .hello
      {
        table-layout:fixed
      }
     caption
      {
       caption-side:top;
      }
</style>
</head>
<body>
<table class="hello">
<caption> Working with table layout and caption</caption>
<tr>
<th>column heading 1</th>
<th>column heading 2</th>
```

```
<th>column heading 3</th>
</tr>
<tr>
<td>Content 1</td>
<td>Content 2</td>
<td>Content 3</td>
</tr>
</table>
</body>
</html>
```

4.10.2 The Border-Collapse Property, Border-SpacingProperty and Empty-Cell Property

The border-collapse property allows determining the border should be displayed around a table. User need to set the proper value for border-collapse property for displaying the table in different style. The values of border-collapse property are as follows.

(i) **collapse** - It will set a common border to all the cells of a table.
(ii) **separate**- It will set a separate border for each cell in a table.
(iii) **inherit**- It inherit the value of the property from the parent element.

The border-spacing determine the space between the borders of two adjacent table cells. The values of border-spacing property are as follows.

(i) **length** -This value specifies the horizontal and vertical space between the border of table cells. If you specify two length values then it will take first the horizontal value and then the vertical value.

(ii) **inherit**- It will determine the border-spacing property should be inherited from the parent element.

The empty-cell property determines a cell which does not contain any content. It will determine the border of an empty cell. The possible values of empty-cell property are show, hide and inherit. In empty-cell property, if you set the value as show it will display a border around an empty cell. Again if you set the value as hide, it will hide the border of an empty cell.Finally setting the value as inherit specifies that the value of the empty-cells property of an element should inherit the value of the same property from its element.The uses of border-collapse property and border-spacing property are shown in the Example 4.16.

Example4.16:

```
<!DOCTYPE HTML>
<html>
<head>
<style type="text/css">
    .tableproperty1
      {
        border-collapse:separate;
        border-spacing:20px;
      }
    table,td,th
      {
         border:1px solid black;
       }
    .tableproperty2
      {
       border-collapse:separate;
```

```
        border-spacing:20px 50px;
       }
</style>
</head>
<body>

<caption> Working with border-collapse and border-spacing
properties</caption>
<table class="tableproperty1" border="1">
<tr>
<th>Student_Name</th>
<th>Student_Address</th>
<th>Student_Age</th>
</tr>
<tr>
<td>Gagan Choudhury</td>
<td>Guwahati</td>
<td>25</td>
</tr>
</table>
<table class="tableproperty2" border="1">
<tr>
<th>Student_Name</th>
<th>Student_Address</th>
<th>Student_Age</th>
</tr>
```

```
<tr>
<td>Tony Bayan</td>
<td>Kolkata</td>
<td>24</td>
</tr>
</table>
</body>
</html>
```

4.11 CSS FORMS

We have already learned the different elements uses with the HTML form. The format of an HTML form greatly improved with CSS. In CSS the different attributes that can be used with the <input> element are as follows.

(i) **input [type=text]**- It will select the text filed.
(ii) **input[type=password]**- It will select the password field.
(iii) **input[type=number]**- It will select the number field.

4.11.1 Padded Input, Bordered Input and Colored Input in CSS:

The padding property is used to add space inside the text field. The border property is used to change the border color and sizes. You can use the border-radius property to add rounded corner. In case of colored input, the back-ground color will change with the back-ground property and text color will change with color property. Example 4.17 shows the working of different input element in a CSS file.

Example 4.17

```
<!DOCTYPE html>
<html>
<head>
```

```
<style type="text/css">
input[type=text]
{
 width: 100%;
padding: 12px 20px;
 margin: 8px 0;
 box-sizing: border-box;
color:blue;
}
</style>
</head>
<body>
<h2>working with input boxes</h2>
<form>
<label >First Name</label>
<input type="text" id="fname" name="fname">
<label >Last Name</label>
<input type="text" id="lname" name="lname">
</form>
</body>
</html>
```

4.11.2 Styling Textarea, Menus and Input Buttons

The textarea often used with the form for collecting the comments or remarks. The textarea can take number of characters which are in fixed width. Menus and buttons are the essential part in building a form. CSS menus can be styles with the help of select element. Input button can be designed with input element by

specifying the type as button, submit and reset. Styling of CSS textarea and styling menu are shown in Example 4.18 and Example 4.19 respectively.

Example 4.18

```
<!DOCTYPE html>
<html>
<head>
<style>
textarea {
width: 100%;
 height: 150px;
 padding: 12px 20px;
 border: 2px solid black
border-radius: 4px;
background-color: #f8f8f8;
font-size: 16px;
}
</style>
</head>
<body>

<h2>example of the styling textarea</h2>
<form>
<textarea>Please enter some text here...</textarea>
</form>
</body>
```

```
</html>
```

Example 4.19

```
<!DOCTYPE html>
<html>
<head>
<style>
select {
 width: 50%;
 padding: 20px 10px;
border: dashed;
 border-radius: 4px;
}
</style>
</head>
<body>
<h2>Example of a select menu</h2>
<form>
<select id="game" name="game">
<option value="foot">Football</option>
<option value="ck">Cricket</option>
<option value="hk">hockey</option>
</select>
</form>
</body>
</html>
```

For styling different buttons you can use different styles for each button. If all the buttons required a single style then this can be implemented by writing a single code for all the buttons. Example 4.20 shows how three buttons can be styled writing a single code.

Example 4.20

```
<!DOCTYPE html>
<html>
<head>
<style>
 input[type=button], input[type=submit], input[type=reset]
{
 background-color: green;
 border: none;
color: black;
padding: 20px 25px;
 margin: 4px 2px;
 cursor: pointer;
}
</style>
</head>
<body>
<h2>Example of styling form buttons</h2>
<input type="button" value="Button">
<input type="reset" value="Reset">
<input type="submit" value="Submit">
</body>
```

</html>

4.12 CSS COUNTERS, CSS RESPONSIVE AND CSS WEBSITE LAYOUT

CSS counters are like variable whose values can be incremented according to the CSS rules.Working with the CSS counters the following properties are required.

(i) **Counter-reset**- It will create or reset a counter.
(ii) **Counter-increment**- It will increment a counter value.
(iii) **Content**-It will insert a generated content
(iv) **Counter( )** function – Add the value of a counter to an element.

The Example 4.21 creates a counter for the page and then increment the counter for each<h1> element and adds “Heading” at the beginning of each<h1> element.

Example 4.21

```
<!DOCTYPE html>

<html>

<head>

<style>

body {

counter-reset: heading;

}

h1::before

                        {

counter-increment: heading;

content: "Heading " counter(heading) ": ";

}

</style>
```

```
</head>

<body>

<h1>example of using CSS Counters</h1>

<h2>Internet Basics</h2>

<h2>Web Page Design</h2>

<h2>CSS</h2>

<h2>Javascript basics</h2>

<h2>XML</h2>

</body>

</html>
```

Responsive web-design is a web design approach to design a web page which is well and good for all types screen. A web-page can be viewed in any types of screen like desktop, tablet, and phone. In responsive web design the designing web-page must be easy to access and look good regardless of the device.

CSS website layout can be divided into the following parts:

(i) Header
(ii) Menu
(iii) Content page
(iv) footer

Different website has different layout according to their requirement. But most of the websites are designed including the above parts. Header part is always in the top of the website and it generally display the title or logo of the web site. Example4.22 shows how header can be designed in a website.

Example4.22

```
<!DOCTYPE html>

<html>

<head>

<title>example of website layout</title>
```

```
<style>
        body
          {
           margin: 1;
          }
   .header {
          background-color: gray;
          padding: 20px;
          text-align: center;
          }
</style>
</head>
<body>
<div class="header">
<h1>Header</h1>
</div>
</body>
</html>
```

Navigating menu helps the visitor to navigating through the web site. Example4.23 shows the creation of the navigating menu in a web site.

Example4.23

```
<!DOCTYPE html>
<html>
<head>
<title>Example of CSS Navigation bar</title>
```

```
<style>
* {
 box-sizing: border-box;
 }
body {
 margin: 0;
}
.header
{
background-color: #f4f4f4;
padding: 20px;
 text-align: center;
}
.top {
overflow: hidden;
 background-color: green;
}

.top a {
 float: left;
 display: block;
color: black;
 text-align: center;
 padding: 14px 16px;
 }
```

```
.top a:hover {
background-color: gray;
color: black;
}
</style>
</head>
<body>
<div class="header">
<h1>Header</h1>
</div>
<div class="top">
<a href="#">Link</a>
<a href="#">Link</a>
<a href="#">Link</a>
</div>
</body>
</html>
```

The content of a websites are designed according to the requirement of the target user. In most of the website content are divided into some columns. The columns are may be equal or unequal according to the requirement. The Example4.24 shows the creation of a website layout including, header, navigating menu and content in column format.

Example 4.24

```
<!DOCTYPE html>
<html>
<head>
```

```
<title>Example of CSS content page</title>
<style>
 * {
box-sizing: border-box;
 }
body
{
margin: 0;
    }
  .header
        {
           background-color: gray;
           padding: 20px;
           text-align: center;
         }
   .top
     {
       overflow: hidden;
       background-color: green;
     }
  .top a
        {
          display: block;
          color: gray;
          text-align: center;
```

```
        padding: 14px 16px;
      }
.top a:hover
        {
          background-color: #ddd;
          color: white;
         }
.column
{
 float: left;
width: 50%;
padding: 15px;
    }
</style>
</head>
<body>
<div class="header">
<h1>Header</h1>
</div>
<div class="top">
<a href="#">Link</a>
<a href="#">Link</a>
<a href="#">Link</a>
</div>
<div class="column">
```

```
<h2>Column</h2>
</div>
<div class="column">
<h2>Column</h2>
</div>
</body>
</html>
```

Check Your Progress II

3. State True or False

(i) The CSS border, padding and margin can be represented using box-model.
(ii) The negative value accepted by padding property.
(iii) None list marker does not specify a marker for a list item.
(iv) Fixed is the default value of the table-layout property.
(v) The caption-side property displays the position of the caption in a table.

4. Fill in the blanks

(i) The _____ property determines a cell which does not contain any content.
(ii) The _____property is used to change the border color and sizes.
(iii) The _____ is used with the form for collecting the comments or remarks.
(iv) _____ will create or reset a counter.
(v) CSS menus can be styles with the help of _____ element.

4.13 SUMMING UP

- CSS was introduced in 1996 on the recommendation of World Wide Web Consortium (W3C).
- The CSS file commonly writtento define the style on web pages.
- User can use the same style for multiple pages with the help of CSS.
- The CSS code can be modified according to the requirement of the user.
- The CSS rule has two different parts, one part is the selector and the other part is declaration.
- The internal style-sheet is written within HEAD element of the HTML document.
- In an external style sheet the styles are saved in single document with .css extension.
- The properties are written in single and separated by semicolons in inline style sheet.
- CSS enables user to add different types of colors with different specification in their own web pages.
- The designing of a web page depends on what type of font, size and styles are chosen.
- The CSS border, padding and margin can be represented using box-model.
- CSS allows user to set various list-style-property to define the style of the list item.
- The format of an html form greatly improved with CSS.
- CSS counters are like variable whose values can be incremented according to the CSS rules.
- User can design a website with different layout according to their requirement in CSS.

4.14 ANSWERS TO CHECK YOUR PROGRESS

1. (i) True (ii) True (iii) False (iv) True (v) False

2. (i) class (ii) inline (iii) external (iv)hexadecimal (v) font-size

3. (i) True (ii) False (iii) True (iv) False (v) True

4. (i) empty-cell (ii) border (iii) textarea (iv) Counter-reset (v) Select

4.15 POSSIBLE QUESTIONS

1. What is CSS? Explain the evolution of CSS.

2. What are the selectors in CSS? How selectors are declared in a CSS?

3. Explain the universal and type selectors with examples.

4. Explain the Id selector with a suitable example.

5. Explain the different ways to apply CSS styles to an HTML document.

6. How CSS colors can be added to a web page?

7. What are the uses of font-size property in a web page? Explain with an example.

8. Explain CSS border, padding and margin.

9. How CSS allows user to set various list-style-properties in a web page?

10. Explain the different types of table properties of CSS.

11. Explain the textarea, menus and input buttons in CSS with examples.

12. What is a CSS counter? Explain it with an example.

13. How can you categorize the CSS website layout?

14. What is CSS responsive?

4.16 REFERENCES AND SUGGESTED READINGS

- https://www.w3schools.com/
- HTML 5 BLACK BOOK Published by dreamtech press

---×---

UNIT- 5
JAVASCRIPT BASICS

UNIT STRUCTURE

5.1 INTRODUCTION

In the earlier units, designing web pages with Hypertext Markup Language (HTML) is already discussed. It is observed that HTML has some limitations. The web pages created with only HTML are completely static in nature that means these pages don't have the ability to present dynamic contents with dynamic updates and real-time user interactions.We can create different forms using HTML but form validations cannot be performed by using only HTML.HTML doesn't offer to perform any calculation and manipulation of data on the client side. HTML also doesn't deliver any mechanism for asynchronous communication with servers. So, if only HTML is used to build a web application then its responsiveness will be limited.JavaScript can be used with HTMLas a solution to remove these limitations.JavaScript is a scripting language and it plays a very important role to make web pages interactive, attractive and dynamic. In this unit, we are going to learn the basics of JavaScript.

5.2 OBJECTIVES

After going through this chapter, we will be able to learn:

- What is JavaScript?
- How to write a JavaScript and execute it.
- About variables, constants and data types in JavaScript
- Different operators available in JavaScript.
- Different types of expressions and statements in JavaScript.

5.3 INTRODUCTION TO JAVASCRIPT

JavaScript was introduced by Brendan Eich in 1995. JavaScript is anobject-oriented scripting language. JavaScript offers different objects for different purposes where each objectprovides different properties and methods. But JavaScript is not a pure Object Oriented Programming (OOP) language like Java because it doesn't support class concept. JavaScript isused along with HTML to develop dynamic, interactive and attractive web pages.A JavaScript program can bewritten within a HTML document or written in an external file and that file is included in a HTML document. JavaScript programs are embedded within HTML documents to adddynamic and interactive properties to the associated web pages. For example, if a form is created in a web page using HTML then automatic data validations for data entered by users to the form can be offered by using a JavaScript program.For example, if user enters an invalid date or mobile number or name then an error or alarm is presented in front of the user.Data manipulation and different types of calculations can be performed in a web page by JavaScript programs. Simple interactive facility for users can also be offered in a web page by JavaScript programs. For example: On mouse click, a particular mathematical operation can be selected by users to perform a particular mathematical calculation.

In general, JavaScript is used as a client-side scripting language. A client-side JavaScript program is executed in the web browser of the user. JavaScript can be referred as a cross-platform scripting language as it is supported by all main web browsers.Asynchronous communication with servers can be implemented by using JavaScript code in a HTML document.

5.3.1Writing JavaScript Code

We can write JavaScript code in two ways. We can write JavaScript code within a HTML document by using <script> tag in the <head> or <body> section of the HTML file. For example: An HTML file with JavaScript code is presented as follows.

Script 5.1(a): HTML file with a JavaScript within the <head> section

```
<html>
        <head>
                <title>First JavaScript Code</title>
<script type="text/javascript">
                        document.write("Welcome to GUCDOE");

</script>

</head>
        <body>

        </body>
</html>
```

Script 5.1(b): HTML file with a JavaScript within the <body> section

```
<html>
        <head>
                <title>First JavaScript Code</title>

        </head>
<body>
<script type="text/javascript">
                        document.writeln(" Welcome to Gauhati
University ");
                        document.writeln(" Welcome to GUCDOE
                ");
```

```
</script>

</body>
</html>
```

Secondly, we can write and save JavaScript code in an external file where in general, the extension of the external file will be .js and it is called as JavaScript source file. Then this external file is included in the associated HTML file using <script> tag with 'src' attribute. 'src' attribute is used to specify the Uniform Resource Locator(URL) of an external JavaScript source file.For example:

Script 5.2(a): JavaScript code written in an external file, "FirstJavaScript.js"

```
document.write(" Welcome to GUCDOE ");
```

Script 5.2(b): HTML file where an external file "FirstJavaScript.js" containing JavaScript code is included

```
<html>
    <head>
    <title>  First JavaScript Code  </title>
<script type= "text/javascript" src= "FirstJavaScript.js">
</script>

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

In case of writing a short JavaScript code, it is easier to write the code within a HTML file. Otherwise, if a JavaScript code is a long code then it is more convenient to write it in a .js external file. JavaScript code written in a .js file can be shared among multiple web pages.

‘write( )’ and ‘writeln( )’ method:‘write( )’ and ‘writeln( )’ method are available under document object in JavaScript. Detailed discussion on document object will be provided in later unit. ‘write( )’ and ‘writeln( )’ method are used to include a new text to a web page. The most common argument to both of these methods is text contained within single or double quotation marks.Numbers, Booleans and variables can also be passed as parameters to both of these methods.The difference between ‘write( )’ and ‘writeln( )’ is that ‘writeln( )’ method appends a line break after the argument.

‘write( )’ method is already used in the Script 5.1(a). The output of the HTML file in Script 5.1(a) will be a text string as shown below.
Welcome to GUCDOE

‘writeln( )’ method is also already used in the Script 5.1(b). The output of the HTML file in Script 5.1(b) will be two text strings as shown below.
Welcome to Gauhati University
Welcome to GUCDOE

Some other examples of JavaScript statements with ‘write( )’ and ‘writeln( )’ methods are presented as follows.

```
document.write("JavaScript is a scripting language");
document.writeln("JavaScript is a scripting language");
document.write('JavaScript is a scripting language');
document.writeln('JavaScript is a scripting language');
document.write(2024);
document.writeln(2024);

var  temp = 2024;
document.write(temp);
document.writeln(temp);
document.write(true);
document.writeln(true);
```

5.3.2 Execution of JavaScript Code

JavaScript is an interpreted scripting language. Nowadays, the JavaScript interpreter is available in all web browsers. So, when a web page is loaded in a browser, the embedded JavaScript code in the page is executed with the help of the JavaScript interpreter that is available in the web browser. We can also run a JavaScript code directly in the developer console of any web browser. In Google Chrome browser, following steps are performed to open the console and execute JavaScript code directly.

1. Open the browser.
2. Right-click on the page displayed in the browser(Figure 5.1(a)).
3. Select "Inspect" (Figure 5.1(a)).
4. Click on the "Console" tab(Figure 5.1(b))
5. Type the JavaScript code directly into the console(Figure 5.1(b))
6. Execute the code by pressing‘Enter’.

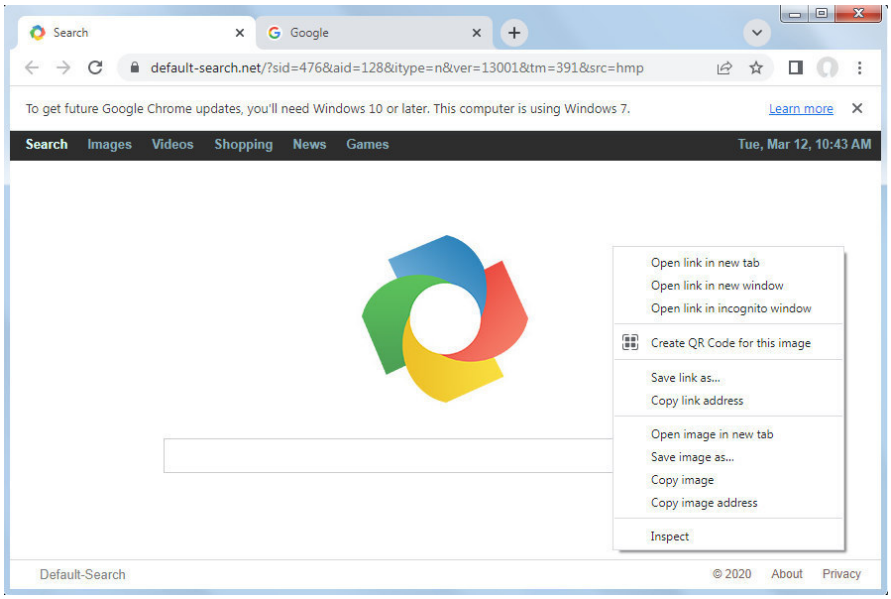


Figure 5.1(a): Open Google Chrome Browser

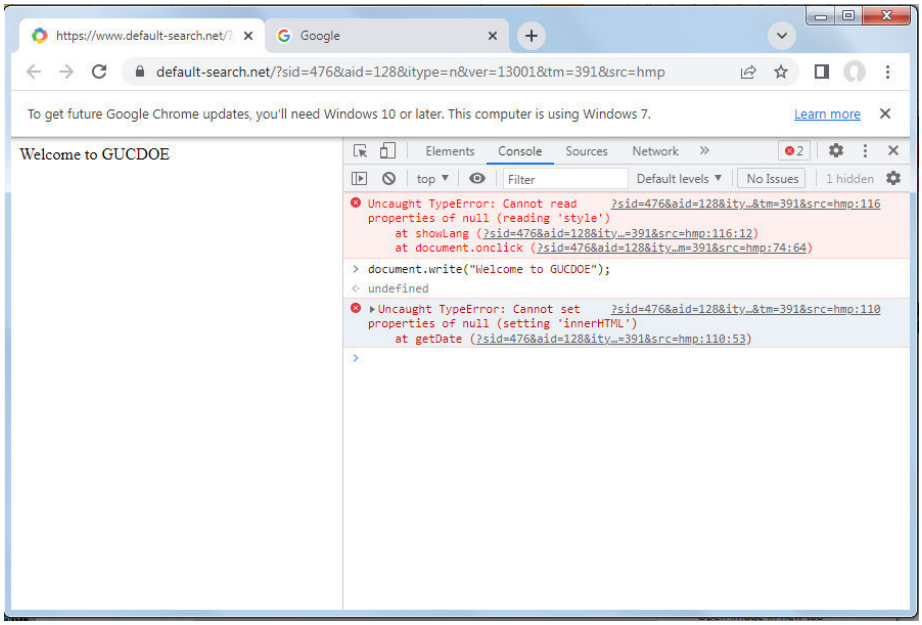


Figure 5.1(b): Console in Google Chrome to write and execute JavaScript code

There are different ways to execute server-side JavaScript code. One way to execute server-side JavaScript code is byusing Node.js. Node.js is a JavaScript runtime environment. The JavaScript code is written in a .js file and it can be executed in the terminalusing the Node.js.

5.4 VARIABLES AND CONSTANTS

Variables and constants are the memory locations to store and manipulate data. The values stored in variables can be changed as per requirement during the execution of a JavaScript code. On the other hand, in case of constants, values cannot be changed after initialization.

Variable declaration in JavaScript can be performed in three ways that are:

a. In JavaScript, variables are declared automatically when first used.
   For example:
   varNum = 50;

In the above statement, ‘varNum’ will be automatically declared as variable and it will be initialized with a value 50.

b. In the second way, variables can be declared using ‘var’ keyword. The syntax to declare a variable using ‘var’ in JavaScript is presented as follows.

```
var variable_name;
```

Example of variable declaration and initialization using ‘var’:

```
var basicSalary = 25000 ;
```

In the above statement, ‘basicSalary’ is a variable and it is initialized to the value 25000.

c. Finally, variables can also be declared using ‘let’ keyword. If the JavaScript code is required to be executed in old browsers then ‘var’ should be used for variable declaration. The syntax to declare a variable using ‘let’ in JavaScript is presented as follows.

```
let variable_name ;
```

Example of variable declaration and initialization using ‘let’:

```
letbasicSalary = 25000 ;
```

In the above statement, ‘basicSalary’ is a variable and it is initialized to the value 25000.

The syntax to declare a constant in JavaScript is presented as follows.

```
constconstant_name;
```

‘const’ is a reserved word in JavaScript and ‘constant_name’ is the name of the declared constant.

Example of constant declaration and initialization:

```
conststudentRegistration = 7701 ;
```

In the above statement, ‘studentRegistration’ is a constant and it is initialized to the value 7701.

In JavaScript, to select a name for a variable, some rules as mentioned below has to be followed.

- Reserved words of JavaScript cannot be used for naming a variable.
- The name of a variable cannot start with a number. It must start with an uppercase or lowercase English alphabet or underscore (_) or dollar sign($).
- The name of a variable cannot contain spaces.

Reserved words in JavaScript are presented as follows.

abstract,break,byte,case,catch, char,class,const, continue,debugger,default,delete,do, double,else,enum,export,extends,false,finally,float,for, function,goto,if,implements,import, in,instanceof,int, let, long,native,new, null,package,private, public,return,short, super,switch,synchronized, this,throw,throws,transient, true,try,typeof,var, void,volatile,while,with.

5.5 DATA TYPES

There are different types of data types available in JavaScript. Broadly, these data types are divided into two categories that are primitive data types and composite data types.

Primitive data types in JavaScript:

Number: Assigned for a variable to storenumbers with or without decimal places.

Example:

```
var temp1 = 14 ;
var temp2 = 7.20 ;
var temp3 = - 20 ;
var temp3 = 25e5 ;
```

String: Assigned fora variable to store text (collection of characters).

Example:

```
var string1 = “Welcome to GUCDOE” ;
```

Boolean: Assigned for a variable to store Boolean values (true or false).

Example:

```
var  temp1 = 14 ;
var  temp2 = 7 ;
var  temp3 = (temp1 = = temp2) ;
```

Undefined: Assigned for a variable where no value is assigned or declared

Example:

```
var  temp1;
```

Null: Assigned for an empty value

Example:

```
var temp = null ;
```

BigInt: It is a new data type in JavaScript. It is assigned to store big numbers that cannot be stored in Number type variable.

Example:

```
var temp=
BigInt("9812674586234572901236728123 5");
```

Composite data types in JavaScript:

Array: Assigned for storing arrays.

Example:

```
var Num = [45,78,90,12,14,7];
```

Object: Assigned for storing objects.

Example:

```
var student = {SName:"Ridip", Roll_No:50, Percentage: 78.5};
```

CHECK YOUR PROGRESS

1. Fill in the blanks

(a) JavaScript is an _____ scripting language.

(b) JavaScript code can be written directly within a HTML document using _____tag.

(c) JavaScript interpreter is available in _____.

(d) ____ data type is used to store big numbers that cannot be stored in Number type variable.

(e) In JavaScript, _____ data type is assigned for a variable to store only 'true' or 'false'.

5.6 OPERATORS

JavaScript supports different types of operators which are presented as follows.

1. Arithmetic operators: Arithmetic operators are used for mathematical calculations. Different arithmetic operators available in JavaScript are presented as follows.
   - Addition operator (+): '+' is used for addition of two operands.
   - Subtraction operator (-): '-' is used for subtraction of one operand from another operand.
   - Multiplication operator (*) : '*' is used for multiplication of two operands.
   - Division operator (/): '/' is used for division of one operand by another operand.
   - Modulus operator (%): '%' is used to estimate the remainder when one operand is divided by another operand.
   - Negation operator (-): Negation operator is used to get the negation of an operand.
   - Increment operator (++): '++' is used to increase the value of an operand by one.
   - Decrement operator (--): '--' is used to decrease the value of an operand by one.

- Exponentiationoperator (**): '**' is used to raise the first operand to the power of the second operand.

STOP TO CONSIDER

** operator is a new operator and it was introduced in ECMAScript 2016. It may not be useful in some browsers.

Script 5.3: JavaScript code to demonstrate the use of Arithmetic operators

```
<html>
        <head>
                <title>Use of Arithmetic Operators</title>
<script type="text/javascript">
                        let num1 = 10, num2 = 20;
                        let sum, sub, mult, div, mod,pow,ne;

                        document.writeln("Initial value in num1 = ",num1);
                        document.writeln("Initial value in num2 = ",num2);

                        sum = num1 + num2;
                        document.writeln("num1 + num2 = ",sum);

                        sub = num2 - num1;
                        document.writeln("num2 - num1 = ",sub);

                        mult = num1 * num2;
                        document.writeln("num1 * num2 = ",mult);
                        div = num2/num1;
                        document.writeln("num2/num1 = ",div);

                        mod = num1 % num2;
                        document.writeln("num1 % num2 = ",mod);

                        ne = - num1;
```

```
                document.writeln(ne);

                num1++;
                document.writeln("num1 = ",num1);

                num2--;
                document.writeln("num2 = ",num2);

                pow= num1**2;
                document.writeln("num1**2 = ",pow);
</script>

</head>
    <body>

    </body>
</html>
```

The output of the above code will be:

```
Initial value in num1 = 10
Initial value in num2 = 20
num1 + num2 = 30
num2 - num1 = 10
num1 * num2 = 200
num2/num1 = 2
num1 % num2 = 10
-10
num1 = 11
num2 = 19
num1**2 = 121
```

2. **Assignment operators:** Assignment operators are used to assign a value to a variable or constant. Different assignment operators available in JavaScript are presented as follows.
   - =
   - +=
   - -=
   - *=
   - /=
   - %=

Script 5.4:JavaScript code to demonstrate the use of Assignment operators

```
<html>
    <head>
        <title>Use of Assignment Operator</title>
    <script type="text/javascript">
            let num1 = 10, num2 = 20;
        document.writeln("Initial value in num1 = ",num1);
            document.writeln("Initial value in num2 = ",num2);
            num1 += num2; // Meaning: num1 = num1 + num2
            document.writeln("num1 = ",num1);
            num1 -= num2;// Meaning: num1 = num1 - num2
            document.writeln("num1 = ",num1);
            num1 *= num2; // Meaning: num1 = num1 * num2
            document.writeln("num1 = ",num1);
            num1/=num2; // Meaning: num1 = num1 / num2
            document.writeln("num1 = ",num1);
            num1 %= num2; // Meaning: num1 = num1 % num2
            document.writeln("num1 = ",num1);
        </script>

        </head>
    <body>

    </body>
</html>
```

The output of the above code will be:

```
Initial value in num1 = 10
Initial value in num2 = 20
num1 = 30
num1 = 10
```

num1 = 200
num1 = 10
num1 = 10

3. **Comparison operators:** Comparison operators are used to compare two operands and it returns a Boolean value of true or false depending upon the operator and values of the operands. Different comparison operators available in JavaScript are presented as follows.
   - ==operator compare two operands and returns true if the operands are equal.
   - ===operator compare two operands and returns true if the operands are equal and the type of both operands are same.
   - !=operator compare two operands and returns true if the operands are not equal.
   - !== operator compare two operands and returns true if the operands are not equal or the types of the operands are not same.
   - >operator compare two operands and returns true if value of the left operand is greater than the value of the right operand.
   - <operator compare two operands and returns true if value of the left operand is smaller than the value of the right operand.
   - >=operator compare two operands and returns true if value of the left operand is larger than or equal to the value of the right operand.
   - <=operator compare two operands and returns true if value of the left operand is smaller than or equal to the value of the right operand.

Script 5.5:JavaScript code to demonstrate the use of Comparison operators

```
<html>
    <head>
        <title>Use of Comparison Operator</title>
    <script type="text/javascript">
            let num1 = 10, num2 = 20;
```

```
                let result;
                document.writeln("Initial value in
                num1 = ",num1);
                document.writeln("Initial value in
                num2 = ",num2);
                result = num1== num2;
                document.writeln(result);
                result = num1 === num2;
                document.writeln(result);
                result = num1 != num2;
                document.writeln(result);
                result = num1!== num2;
                document.writeln(result);
                result = num1 > num2;
                document.writeln(result);
                result = num1 < num2;
                document.writeln(result);
                result = num1 >= num2;
                document.writeln(result);
                result = num1 <= num2;
                document.writeln(result);
            </script>

        </head>
        <body>

        </body>
</html>
```

The output of the above code is:

```
Initial value in num1 = 10
Initial value in num2 = 20
false
false
true
true
false
true
false
true
```

4. Logical operators: Logical operators are used on Boolean operands. Different logical operators available in JavaScript are presented as follows.
    - && operator returns true if the Boolean values of both left and right expression are true.
    - || operator returns false if the Boolean values of both left and right expression are false otherwise it will return true.
    - ! operator returns the opposite Boolean value to the Current Boolean value of the expression(operand).

Script 5.6:JavaScript code to demonstrate the use of logical operators

```
<html>
    <head>
        <title>Use of Logical Operator</title>
    <script type="text/javascript">
            let num1 = 10, num2 = 20;
            let result;
            document.writeln("Initial value in num1 = ",num1);
            document.writeln("Initial value in num2 = ",num2);
            result = (num1 == num2) && (num1 < num2);
            document.writeln(result);
            result =(num1 == num2) || (num1 < num2);
            document.writeln(result);
            result = !(num1 == num2);
            document.writeln(result);
        </script>
    </head>
    <body>

    </body>
</html>
```

The output of the above code is:

```
Initial value in num1 = 10
Initial value in num2 = 20
false
true
true
```

5. **Bitwise operators:**Bitwise operators are used toimplement bitwise operations on operands. Different bitwise operators available in JavaScript are presented as follows.
   - &
   - |
   - ~
   - ^
   - <<
   - >>
   - >>>

6. **Conditional operator:?:**

   The syntax to use conditional operator is:

   ```
   Variable = (condition) ? Expression1:Expression2
   ```

   If the result of the condition is true then Expression1 will be assigned to Variable otherwise Expression2 will be assigned to Variable. For example: Let us consider the following JavaScript statements.

   ```
   var  num1 = 10;
   var num2 = 20;
   temp = (num1 == num2) ? num1 : num2 ;
   document.write(temp);
   ```

   From the above statements, it is observed that the value in num1 is not equal to the value in num2. So, the condition in the conditional operator evaluates to false and as a result, the value in num2 will be assigned to temp. The output of the above statements will be 20.

7. **Type operators:**
   - 'typeof' operator: It is used to find out the data type of a variable.
   - 'instanceof' operator: It is used to find out whether an object is an instance of a specified object type. It will return a Boolean value of 'true' or 'false'.

8. **String operator:**'+' operator is used for string concatenation in JavaScript. For example consider the following JavaScript code.

```
var string1 = " Gauhati ";
var string2 = "University" ;
var string3 = string1 + string2;
```

It is already stated that '+' operator is also used for addition of two numbers. So it will be interesting to explore how it will work with both strings and numbers together in JavaScript. Consider the following JavaScript code to find out the answer of this question.

```
var string1 = "Guwahati-" +781028;
var string2 = "JavaScript" + 2 + 3 ;
var string3=  2 + 3 + "JavaScript";
document.writeln(string1);
document.writeln(string2);
document.writeln(string3);
```

Output of the above code is:

```
Guwahati-781028
JavaScript23
5JavaScript
```

In JavaScript, if '+' operator is used to add a string and a number then the number is treated as a string. For this reason the value in string1 is displayed as "Guwahati-781028". Here '781028' is treated as a string.

In JavaScript, the associativity of '+' operator is left to right. So, JavaScript expressions with multiple '+' operator are evaluated from left to right. For this reason, in the third expression, 2 and 3 are treated as numbers and with the first '+' operator from left, the result will be 5. As the

fourth operand in that expression is a string, so after the evaluation of the first '+' operator from left, 5 will be treated as string with the second '+'operator from left. The final value of string3 will be "5JavaScript".

9. **'new' operator:**In JavaScript, if we have to create instances of user-defined objects or built-in constructor functions then 'new' operator can be used to do it.

10. **'delete'operator:**In JavaScript, if we require to delete a property from an object then 'delete' operator can be used to do it.

5.7 EXPRESSIONS

We have already learnt about variables, constants, data types and different operators supported by JavaScript in earlier sections. In this section, we are going to learn about JavaScript expressions. JavaScript expressions are the groupings of literal values, variables, constants, operators and functions. Each expression in JavaScript produces an output in JavaScript. We have already used different simple JavaScript expressions in the earlier sections. Examples of different JavaScript expressions are presented as follows.

```
var num1 = 30;
var num2 = 40;
var sum = num1 + num2;
num1 == num2;
var num3 = num1 > num2 ? num1 : num2;
var string1 = "Gauhati University";
var string2 = "CDOE";
var string3 = string1 + string2;
var arrNum = new Array();
arrNum[0] =23;
arrNum[1] = 45;
 var flag = (arrNum[0] ==arrNum[1]) && ( arrNum[0] == 23);
```

5.8 STATEMENTS

Statements available in JavaScript can be divided into different types which are declarative statements, arithmetic statements, assignment statements, conditional statements, iterative statements, jump statements, function call and function definition statements. In JavaScript, a statement is ended with a semicolon (;) and a group of multiple JavaScript statements can be created by using curly brackets ({ }).

Declarative statements: Statements used to declare variables or constants in JavaScript are called as declarative statements.

For example:

```
var basicSalary;
```

Arithmetic statements: Statements used to perform arithmetic operations in JavaScript are called as arithmetic statements.

For example:

```
grossSalary= basicSalary+ HRA + DA ;
```

Assignment statements: Statements used to assign values to variables or constants or properties in JavaScript are called as assignment statements.

For example:

```
var basicSalary = 50000;
```

Conditional statements: Statement used to execute one or a block of statements based on the result of a condition is called conditional statement. The result of a condition in a conditional statement will be true or false.In JavaScript, following conditional statements are available.

'if' statement:

Syntax:

```
if (condition)
{
        // One or Block of JavaScript statements //
}
```

If the condition in the 'if' statement evaluates to 'true' then the JavaScript statements within 'if' will be executed.

'if….else' statement:

Syntax:

```
if(condition)
{
        // One or Block of JavaScript statements //
}
else
{
        // One or Block of JavaScript statements //
}
```

If the condition in the 'if' statement evaluates to 'true' then the JavaScript statements within 'if' will be executed. On the other hand if the condition in the 'if' statement evaluates to 'false' then statements within 'else' will be executed.

Script 5.7: Write a JavaScript to input an integer number and using 'if….else' statement check whether it is an odd or even number.

```
<html>
<head>
        <title>Odd or Even</title>
<script type="text/javascript">
        var num1 = prompt("Enter a number=");
        var num2 = parseInt(num1);// Convert string to integer number
        document.writeln("The input number is=",num2);
        if(num2 % 2 == 0)

                document.writeln(num2," is an even number");
        else
                document.writeln(num2," is an odd number");
```

```
</script>

</head>
    <body>

    </body>
    </html>
```

In the above JavaScript code, prompt() method is used to provide a dialog box with a text, a text box, a OK button and a Cancel button to the user so that the user can input a text in the text box(Figure 5.2). If the user clicks on the OK button then the user input is returned as a string and if user clicks on the Cancel button then null is returned. If we input an integer number in the text box available in the dialog box and click on the OK button then actually a string will be returned. So, parseInt() method is used in the above code to convert the string returned by the prompt() method to an integer number.

Figure 5.2: Dialog Box by prompt() Method

'if….else if….else' statement:

Syntax:

```
if(condition1)
{
        // One or Block of JavaScript statements //
}
else if(condition2)
{
        // One or Block of JavaScript statements //
```

```
}
else
{
        // One or Block of JavaScript statements //
}
```

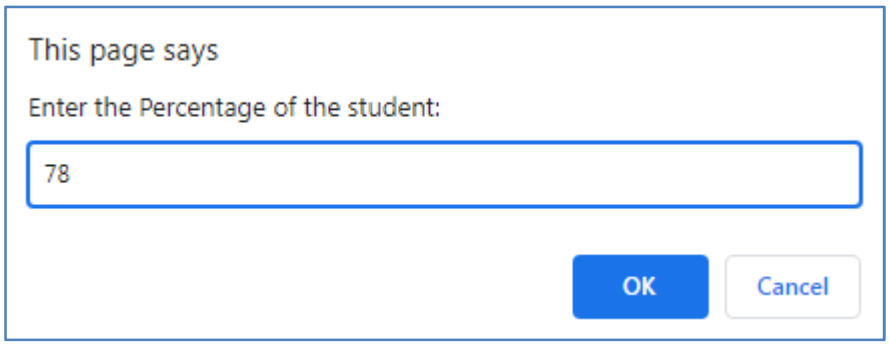
If different blocks of statements should be executed based on different conditions then ‘if….else if….else’ statement can be used in JavaScript. As shown in the syntax, if ‘condition1’ evaluates to ‘true’ then JavaScript statements within ‘if’ will be executed. If ‘condition1’ evaluates to ‘false’ then ‘condition2’ in ‘else if’ will be evaluated and if it is ‘true’ thenJavaScript statements within ‘else if’ will be executed. On the other hand if ‘condition2’ also evaluates to ‘false’ then JavaScript statements within ‘else’ will be executed. If more conditions are required to be evaluated for more number of statement blocks then more number of ‘else if’ can be used depending upon the number of conditions.

Script 5.8: Write a JavaScript code to find out the division of a student based on the percentage obtained.

```
<html>
<head>
        <title>Estimate Division</title>
<script type = "text/javascript" >
        var perString = prompt("Enter the Percentage of the
        student:");
          var per = parseInt(perString);

        document.writeln("The Percentage of the student
        is:",per);
        if( per>=60 )
        document.writeln(" 1St Division");
         else if( per>=45 && per<60)
        document.writeln("2nd Division");
         else if( per>=30 && per< 45)
        document.writeln("3rd Division");
         else
        document.writeln("Fail");
</script>
```

```
</head>
        <body>

        </body>
        </html>
```

The output of the above code is:



The Percentage of the student is:78
1St Division

'switch' statement:

Syntax:

```
switch( JavaScript expression)
{
        case result1:
                // One or Block of JavaScript statements //
                break;
        case result2:
                // One or Block of JavaScript statements //
                break;
        case result3:
                // One or Block of JavaScript statements //
                break;
        default:
                // One or Block of JavaScript statements //

}
```

In case of ‘switch’ statement, JavaScript statements will be executed depending upon the value of the JavaScript expression. As shown in the syntax, if the value of the JavaScript expression is ‘result1’ then JavaScript statements under ‘case result1:’ will be executed. If the value of the JavaScript expression is ‘result2’ then JavaScript statements under ‘case result2:’ will be executed. If the value of the JavaScript expression is ‘result3’ then JavaScript statements under ‘case result3:’ will be executed. But if the value of the JavaScript expression is not matched with any of these values (‘result1’,‘result2’ and ‘result3’) then JavaScript statements under ‘default:’ will be executed. More number of ‘case’ statements can be included in ‘switch’ depending upon the requirement. ‘break’ statement is used to exit from the ‘switch’ statement after the execution of JavaScript statements available under a ‘case’ statement. If ‘break’ statement is not usedin a ‘case’ then the execution of JavaScript statements available in the next ‘case’ or ‘default’ will be continued.

Script 5.9: Write a JavaScript to find out the capital of a country where the country must be one of the top 10 countries based on its area in the world (Use ‘switch’ statement).

```
<html>
<head>
<title>Capital of a Country</title>
<script type="text/javascript">
        var country = prompt("Enter Country Name:");
document.writeln("Country Name is:",country);

switch(country.toUpperCase())/*toUpperCase() method is used to convert a string to uppercase letters.*/
        {
        case "RUSSIA": document.writeln("Capital is Moscow");
                        break;
         case "CANADA": document.writeln("Capital is Ottawa");
                        break;
        case"CHINA":  document.writeln("Capital is Beijing");
                        break;
        case "UNITED STATES": document.writeln("Capital is Washington D.C");
                        break;
```

```
            case "BRAZIL" :document.writeln("Capital is Brasilia");
                    break;
            case "AUSTRALIA": document.writeln("Capital is
            Canberra");
                    break;
case "INDIA": document.writeln("Capital is New Delhi");
                    break;
 case "ARGENTINA": document.writeln("Capital is Buenos
Aires");
                    break;
case "KAZAKHSTAN": document.writeln("Capital is Astana");
                    break;
 case "ALGERIA": document.writeln("Capital is Algiers");
                    break;
 default: document.writeln("\n Wrong Input");
 }
</script>

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

The output of the above code is:

Country Name is:India
Capital is New Delhi

Iterative statements: Statement used to execute one or block of statements repeatedly as long asa certain condition evaluates to 'true'. In JavaScript, following iterative statements are available.

'while' statement:

Syntax:

```
while (condition)
{
        // One or Block of JavaScript statements //
}
```

As long as 'condition' in 'while' evaluates to 'true', the statements available within 'while' will be executed repeatedly.

Script 5.10: Write a JavaScript to display a string 10 times using 'while' statement.

```
<html>
<head>
<title>Iterative Statement</title>
<script type = "text/javascript" >
        i = 1;
        while(i <= 10)
        {
                document.writeln("\n Welcome to GUCDOE");
                i++;
        }
</script>
</head>
<body>

</body>
</html>
```

In the above code, the initial value of i is 1 and it is incremented by 1 in each iteration of the 'while' statement. The statements within 'while' will be executed repeatedly as long as the value of i is less than and equal to 10.

The output of the above code is:

Welcome to GUCDOE

Welcome to GUCDOE

Welcome to GUCDOE

Welcome to GUCDOE

Welcome to GUCDOE

Welcome to GUCDOE

Welcome to GUCDOE

Welcome to GUCDOE

Welcome to GUCDOE

Welcome to GUCDOE

'do…while' statement:

Syntax:

```
do
{
        // One or Block of JavaScript statements //
} while (condition);
```

As long as 'condition' in 'do…while' evaluates to 'true', the statements available within 'do…while' will be executed repeatedly.

Script 5.11:Write a JavaScript to display all the even numbers from 1 to 100 using 'do…while' statement.

```
<html>
<head>
<title>Iterative Statement</title>
<script type = "text/javascript" >
        i = 1;
document.writeln("\n Even numbers from 1 to 100 are:\n");
        do
        {
                if(i%2 == 0)
                        document.write(" ", i);
                i++;
        }while(i <= 100);

</script>
```

```
</head>
<body>

</body>
</html>
```

The output of the above code is:
Even numbers from 1 to 100 are:

2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94 96 98 100

'for' statement:

Syntax:

```
for(initialization; condition; increment or decrement)
{
    // One or Block of JavaScript statements //
}
```

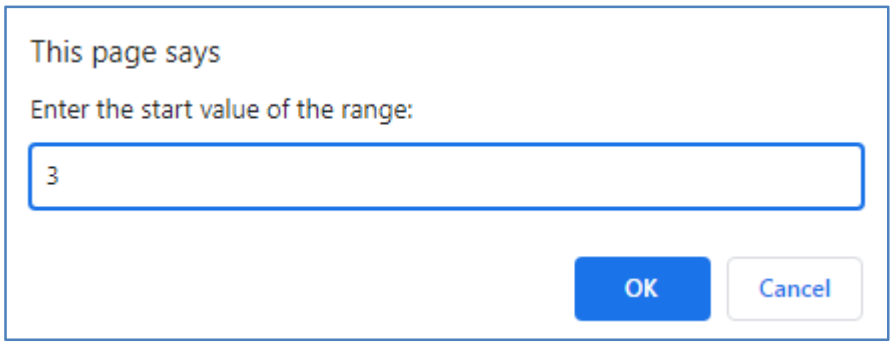
As shown in the syntax, the first statement in a 'for' loop is an initialization statement where an initial value is assigned to a counter variable. The second statement is a condition that must evaluates to 'true' for the execution of the statements available within the 'for' loop. Finally the third statement is used to increment or decrement the counter variable.

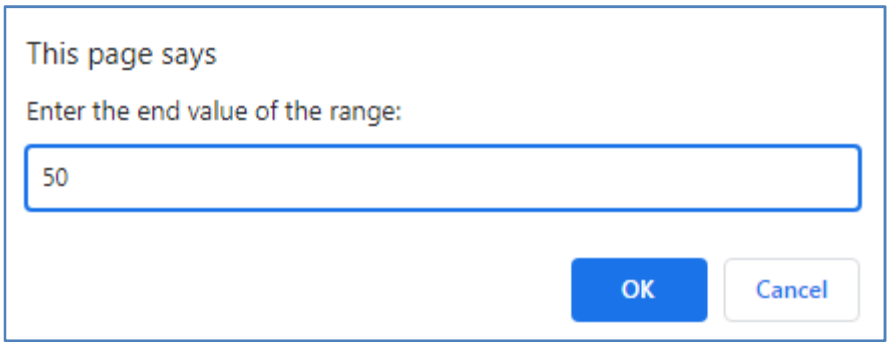
Script 5.12: Write a JavaScript to display all the prime numbers within an input range of integer numbers using 'for' statement.

```
<html>
<head>
<title>Iterative Statement</title>
<script type = "text/javascript" >
    var temp = prompt("Enter the start value of the range:");
    var start = parseInt(temp);
    var temp = prompt("Enter the end value of the range:");
    var end = parseInt(temp);
```

```
document.writeln("Prime numbers from ",start," to ",end," are:\n");
        if(start == 1)
           start++;
for( i = start; i <= end; i++)
        {
          flag = 1;
        for(j = 2; j<= i/2; j++)
          {
if(i % j == 0)
                    {
                        flag = 0;
                        break;
                    }
              }
if(flag == 1)
document.write("   ",i);

        }
</script>
</head>
<body>

</body>
</html>
```

The output of the above code is:





Prime numbers from 3 to 50 are:

3 5 7 11 13 17 19 23 29 31 37 41 43 47

Jump statements: Statements used to change the actual flow of statements execution in a JavaScript program by moving to another part of the program and start executing it,arereferred as jump statements. There are four jump statements available in JavaScript. These are 'break', 'continue', 'return' and 'throw'.

'break' statement: 'break' statement is used to exit from a loop or a switch block without executing statements available inside the loop or the 'switch' block after the 'break statement. If there are more statements available outside the loop or the 'switch' block then these statement will be executed after exiting from the loop or the 'switch' block due to 'break'statement.

'continue' statement:'continue' statement is used to stop executing the current iteration of a loop and start executing the next iteration.

'return' statement:'return' statement is used to exit from a function and after executing a 'return' statement, a value may be returned depending upon the job of that function.

'throw' statement:'throw' statement is used to throw an exception when an error occurred in a try block.

Function call and function definition statements: Statement used to define a function is known as function definition statement in JavaScript. On the other hand, statement used to call an already defined function is known as function call statement. More detailed discussion on this topic will be presented in the later unit.

Comments can also be included in JavaScript programs. Comments are not treated as statements and part of JavaScript programs. Single line comments can be included using '//' in JavaScript. A single line comment is started with '//'as shown in **Script 5.7**.On the other hand multi-line comments can be included in a JavaScript program by using '/* */'. Multi-line comments are begun with'/*' and end with '*/' as shown in**Script 5.9**.

5.9 ARRAY IN JAVASCRIPT

Array is a data structure used to store more than one value within a single variable. In JavaScript, an array can store values with different data types like Number, String, Object, Array.We can declare an array in JavaScript in three ways as shown below with three examples.

```
var array1 = [ ];
var array2 = new Array();
var array3 = new Array(10);
```

Above JavaScript statements will declare empty arrays. The third statement will declare an empty array with the capacity to hold 10 elements.

In JavaScript, initialization of anarray can be performed in different ways as shown with the following examples.

```
var intNumber = [ 25 , 56 , 78 , 42 , 14];

var arrayName = new Array(4);
arrayName[0] = "Pranab";
arrayName[1] = "Pradip";
arrayName[2] = "Ankur";
arrayName[3] = "Naba";

var arrayNum = new Array(10,20,30,40,50);
```

Each element available in an array can be accessed with the array name andan index within [ ]. In JavaScript, the index of the first element of an array is zero. In the above example, the statement 'arrayName [0] = "Pranab";' will assign "Pranab" as the first element in the array 'arrayName'.

Property ‘length’ can be used to find out the size of an array as shown in the following example.

```
var arrayNum = new Array(100, 200, 300, 400, 500);
var arrayLength = arrayNum.length ;
```

Variable ‘arrayLength’ will hold the size of the array ‘arrayNum’.

Some of the important methods used in JavaScript array manipulation are presented as follows.

(a) push():push() method is used to insert a new element into an array at the last position. For example:

```
var arrayNum = new Array(100, 200, 300, 400, 500);
arrayNum.push(600);
```

From the above statements, 600 will be inserted into the array ‘arrayNum’ after the value 500.

(b) unshift(): unshift() method is used to insert one or more than one new elements into an array at the beginning. For example:

```
var arrayNum = new Array(100, 200, 300, 400, 500);
arrayNum.unshift(10,50);
```

From the above statements, 10 and 50 will be inserted into the array ‘arrayNum’ at the beginning and as a result the value 100 becomes the third element in that array.

(c) concat():concat() method is used to produce a new array by combining two arrays. For example:

```
var arrayNum1 = new Array(10, 20, 30, 40, 50);
var arrayNum2 = new Array(100, 200, 300, 400, 500);
var arrayNum3 = arrayNum1.concat(arrayNum2);
```

From the above statements, the array ‘arrayNum2’ is combined at the end of the array ‘arrayNum1’ and so the content of the array ‘arrayNum3’ will be “10,20,30,40,50,100,200,300,400,500”.

(d)shift():shift() method is used to delete the first element from an array and then it moves each of the remaining elements to the next lower index. For example:

```
var arrayNum = new Array(100, 200, 300, 400, 500);

arrayNum.shift();
```

From the above statements, the value 100 is deleted from the array 'arrayNum' and the value 200 becomes the first element of that array.

(e) pop():pop() method is used to delete the last element of an array. For example:

```
var arrayNum = new Array(100, 200, 300, 400, 500);

arrayNum.pop();
```

From the above statements, the value 500 will be deleted from the array 'arrayNum'.

(f) splice():splice() method can be used to both insert and delete elements from an array.

```
var arrayNum = new Array(100, 200, 300, 400, 500);

arrayNum.splice( 1, 2, 50, 60);
```

The first parameter in splice(), denotesthe index from where new elements will be inserted or existing elements will be deleted. The second parameter denotes the number of elements to be deleted. The remaining parameters are the new elements to be inserted into the array. So from the above statements, it is observed that the index to start insertion and deletion of array elements will be 1 and the number of elements to be deleted is 2. So, 200 and 300 will be deleted from the array 'arrayNum'. 50 and 60 are the new values that will be inserted from the index 1 in the array 'arrayNum'. So finally, the content of the array 'arrayNum' will be "100,50,60,400,500".

(g)sort(): sort() method is used to sort an array alphabetically. For example:

```
var arrayName = new Array("Pranab" , "Pradip" , "Ankur" , "Naba");
arrayName.sort();
```

From the above statements, names stored in the array 'arrayName' will be sorted in alphabetical order.

(h) reverse(): reverse() method is used to reverse the array elements in an array. For example:

```
var arrayName = new Array(“Pranab” , “Pradip” , “Ankur” , “Naba”);
arrayName.reverse();
```

From the above statements, names stored in the array ‘arrayName’ will be reversed in order.

(i) toString():toString() method is used to transform an array to a string of values. These values are the array elements and these are separated with commas in the output string. For example:

```
var arrayNum = new Array(100, 200, 300, 400, 500);
arrayNum.toString();
```

From the above statements, the array ‘arrayNum’ will be converted to string of values that are 100, 200, 300, 400 and 500.

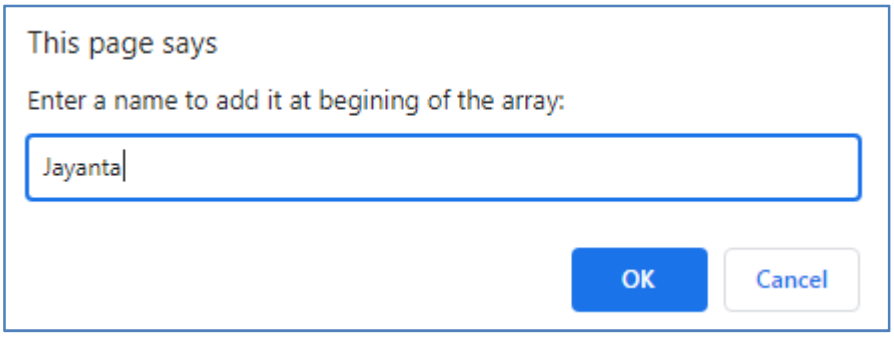
Script 5.13: Write a JavaScript to create an array to store names. Insert new names at the beginning and at the end of the array. Sort the array in alphabetical order.

```
<html>
<head>
<title>Array</title>
<script type = "text/javascript" >
    var arrayNames = new Array("Pranab","Ankur","Pranjal","Prahelika","Gita");
document.writeln("The initial array of names is:");
document.writeln(arrayNames);
    var tempName = prompt("Enter a name to add it at begining of the array:");
arrayNames.unshift(tempName);
document.writeln("After insertion of the new name, the array is:");
document.writeln(arrayNames);
    var tempName = prompt("Enter a name to add it at the end of the array:");
arrayNames.push(tempName);
document.writeln("After insertion of the new name, the array is:");
```

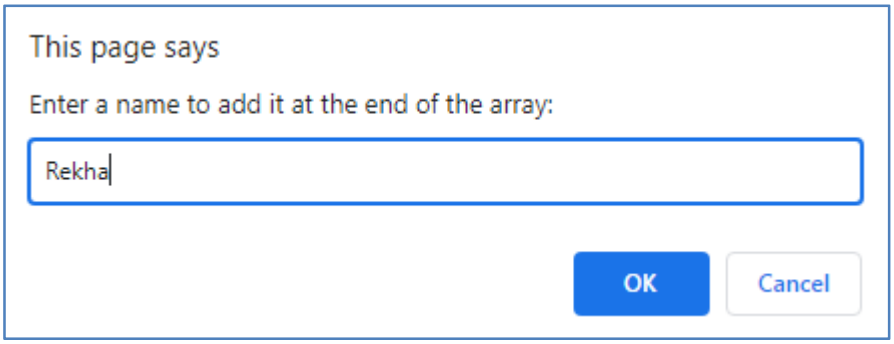
```
document.writeln(arrayNames);
arrayNames.sort();
document.writeln("After sorting in alphabetical order, the array is:");
document.writeln(arrayNames);
</script>
</head>
<body>
</body>
</html>
```

The output of the above code is:

The initial array of names is:
Pranab,Ankur,Pranjal,Prahelika,Gita



After insertion of the new name, the array is:
Jayanta,Pranab,Ankur,Pranjal,Prahelika,Gita



After insertion of the new name, the array is:
Jayanta,Pranab,Ankur,Pranjal,Prahelika,Gita,Rekha

After sorting in alphabetical order, the array is:
Ankur,Gita,Jayanta,Prahelika,Pranab,Pranjal,Rekha

Script 5.14: Write a JavaScript to create two arrays of integer numbers. Display all the odd numbers available in the first array. Add these odd numbers to the second array.

```
<html>
<head>
<title>Array</title>
<script type = "text/javascript" >
        var firstArray = new Array(34,7,14,13,21,72,11,67,55,4);
        var secondArray = new Array(89,1,66,37,71,9,8);
document.writeln("The first array is:");
document.writeln(firstArray);
document.writeln("The second array is:");
document.writeln(secondArray);
        var arrLength = firstArray.length;
document.writeln("The odd numbers available in the first array are:");
for(i = 0 ; i<arrLength ; i++)
        {
            if(firstArray[i] %2 != 0)
            {
document.writeln("   ",firstArray[i]);
secondArray.push(firstArray[i]);// Add odd number to the second array
            }
        }
document.writeln("After insertion of odd numbers, the second array is:");
document.writeln(secondArray);
</script>
</head>
<body>

</body>
</html>
```

The out of the above code is:

```
The first array is:
 34,7,14,13,21,72,11,67,55,4
```

The second array is:
89,1,66,37,71,9,8
The odd numbers available in the first array are:
7
13
21
11
67
55
After insertion of odd numbers, the second array is:
89,1,66,37,71,9,8,7,13,21,11,67,55

CHECK YOUR PROGRESS

2. Choose the correct option

(a) Which of the following statement is not a jump statement in JavaScript?

(i) break
(ii) switch
(iii) continue
(iv) return

(b) Which of the following is an arithmetic operator in JavaScript?

(i) %
(ii) %=
(iii) = = =
(iv) Both (i) and (ii)

(c) _____ is a logical operator.

(i) &
(ii) ++
(iii) !
(iv) None of the above

(d) _____ is a conditional operator.

(i) ?:
(ii) ?=
(iii) ?;

(iv) None of the above

(e) ____ is a conditional statement.

(i) switch
(ii) if
(iii) for
(iv) Both (i) and (ii)

(f) Which of the following method can be used to add an element to an array?

(i) shift()
(ii) push()
(iii) add()
(iv) insert()

5.10 SUMMING UP

- JavaScript is an object-oriented scripting language. It is used along with HTML to develop dynamic, interactive and attractive web pages. In general, JavaScript is used as a client-side scripting language.
- We can write JavaScript code within a HTML document by using <script> tag in the <head> or <body> section of the HTML file. We can also write and save JavaScript code in an external file where in general, the extension of the external file will be .js and it is called as JavaScript source file. Then this external file is included in the associated HTML file using <script> tag with 'src' attribute.
- JavaScript is an interpreted scripting language. The JavaScript interpreter is provided in all web browsers.
- The syntax to declare a variable in JavaScript is presented as follows.

  var variable_name ;
- The syntax to declare a constant in JavaScript is presented as follows.

  constconstant_name ;
- The categories of data types available in JavaScript are primitive data types and composite data types.

- Primitive data types in JavaScript are Number, String, Boolean, Undefined, Null and BigInt.
- Composite data types in JavaScript are Array and Object.
- Operators supported by JavaScript are presented as follows.
    1. Arithmetic operators: Addition operator (+), Subtraction operator (-), Multiplication operator (*), Division operator (/), Modulus operator (%), Negation operator (-), Increment operator (++), Decrement operator (--), Exponentiation operator (**).
    2. Assignment operators: = , += , -= , *= , /= , %=
    3. Comparison operators: = = , = = = , != , != = = , > , < , >= , <=
    4. Logical operators: && , || , !
    5. Bitwise operators:& , | , ~ , ^ , << , >> , >>>
    6. Conditional operator: ?:
    7. Type operators: 'typeof' operator , 'instanceof' operator.
    8. String operator: '+' operator is used for string concatenation in JavaScript.
    9. 'new' operator
    10. 'delete' operator
- JavaScript expressions are the groupings of literal values, variables, constants, operators and functions. Each expression in JavaScript produces an output in JavaScript.
- Different types of statements available in JavaScript are declarative statements, arithmetic statements, assignment statements, conditional statements, iterative statements, jump statements, function call and function definition statements.
- Conditional statements are used to execute one or a block of statements based on the result of a condition. The result of a condition in a conditional statement will be true or false. Conditional statements in JavaScript are (a) 'if' statement , (b) 'if….else' statement , (c) 'if….else if….else' statement , (d) 'switch' statement.
- Iterative statements areused to execute one or block of statements repeatedly as long as a certain condition evaluates to 'true'. Iterative statements in JavaScript are (a) 'while' statement , (b) 'do…while' statement , (c) 'for' statement

- Jump statements are usedto change the actual flow of statements execution in a JavaScript program by moving to another part of the program and start executing it. Jump statements available in JavaScript are ‘break’, ‘continue’, ‘return’ and ‘throw’.
- Single line comments can be included using ‘//’ in JavaScript and multi-line comments can be included in a JavaScript program by using ‘/* */’.
- Array is a data structure which can store multiple values within a single variable.

5.11 ANSWERS TO CHECK YOUR PROGRESS

1.(a) interpreted , (b) <script> , (c) web browsers , (d) BigInt , (e) Boolean

2. (a) (ii) switch, (b) (i) % ,(c) (iii) ! , (d)(i) ?: , (e) (iv) Both (i) and (ii) , (f) (ii) push()

5.12 POSSIBLE QUESTIONS

1. Explain the importance of JavaScript.
2. Explain different types of data types available in JavaScript.
3. Explain different types of operators available in JavaScript.
4. Why conditional statement is important in JavaScript?
5. Write a JavaScript to display all prime numbers from 1 to 200.
6. Write down the difference between ‘break’ statement and ‘continue’ statement.

5.13 REFERENCES AND SUGGESTED READINGS

- Harvey, Harvey Deitel, and Abbey Deitel. "Internet and World Wide Web How To Program." (2011).
- Gosselin, Don. *Javascript*. Course Technology Press, 2004.
- Pollock, John. *JavaScript, A Beginner's Guide*. McGraw Hill Professional, 2009.
- Thau, Dave, and Dave Thau. *The Book of JavaScript: A Practical guide to interactive Web pages*. No Starch Press, 2000.

---×---

UNIT- 6
JAVASCRIPT OBJECTS

UNIT STRUCTURE:

6.1 INTRODUCTION

We have already learnt about the basics of JavaScript in the previous unit. In this unit, we are going to learn about JavaScript objects. JavaScript objects play a very important role in writing efficient JavaScript code to make web pages interactive, attractive and dynamic. A JavaScript object is a group of functions or methods and properties.In JavaScript, objects can also be stated as key-value pairs as these are the groups of some unique identifiers and each identifier is related to a value. In this unit, different important concepts of JavaScript object will be discussed with examples.

6.2 OBJECTIVES

After going through this chapter, we will be able to learn:

- How to create and display JavaScript object?
- About JavaScript object properties and methods.
- About JavaScript object Constructors.
- About JavaScript object Prototypes.
- About some of important JavaScript objects with their important properties and functions.

6.3 CREATE AND DISPLAY JAVASCRIPT OBJECTS

A JavaScript object can be created using different ways. For example:

(a)
```
let  student ={
        sName: “Pranjal Das”,
        sRoll: 45,
        sCourse: “MSc.IT”,
        sPercentage: 78,
        display :function(){
                document.writeln("Student
        name:",student.sName);
                document.writeln("Student Roll
        Number:",student.sRoll);
                document.writeln("Student
        Course:",student.sCourse);
                document.writeln("Student
        Percentage:",student.sPercentage);
         }


};
student.display();
```

In the above example, a JavaScript object is created by using object literal notation. In this method, properties and functions are added to the new object within curly braces ‘{ }’. ‘:’ is used to associate a value with a property in an object. In the above example, a new object ‘student’ is created by adding four properties that are ‘sName’, ‘sRoll’, ‘sCourse’ and ‘sPercentage’. Each of these properties is associated with a value.sName’ is associated with the string “Pranjal Das”. Similarly, ‘sRoll’ is associated with 45, ‘sCourse’ is associated with “MSc.IT”, and ‘sPercentage’ is associated with 78. A function, ‘display ()’ is also added to the newly created object.

The output of the above JavaScript code will be:

```
Student name:Pranjal Das
Student Roll Number:45
Student Course:MSc.IT
```

Student Percentage:78

(b) let student = new Object();
student.sName = “Pranjal Das”;
student.sRoll = 45;
student.sCourse = “MSc.IT”;
student.sPercentage = 78;
student.display=function(){
document.writeln("Student name:",student.sName);
document.writeln("Student Roll Number:",student.sRoll);
document.writeln("Student Course:",student.sCourse);
document.writeln("Student Percentage:",student.sPercentage);
};
student.display();

In the above example, ‘new’ keyword is used to create a JavaScript object ‘student’. In this method, ‘dot’ (.) operator is used for adding properties and functions to the newly created object as shown in the above example.

(c) let studentPrototype = {
display: function(){
document.writeln("Student name:",student.sName);
document.writeln("Student Roll Number:",student.sRoll);
document.writeln("Student Course:",student.sCourse);
document.writeln("Student Percentage:",student.sPercentage);
}
};

let student =Object.create(studentPrototype);
student.sName = "Pranjal Das";
student.sRoll= 45;
student.sCourse= "MSc.IT";

```
student.sPercentage= 78;
student.display();
```

In the above example, Object.create() method is used to create an object ‘student’ based on the prototype object ‘studentPrototype’. We will learn about object prototypes with more details in a later section of this unit.

We can display a JavaScript object using different methods. For example, consider the following JavaScript code to create an object, ‘car’.

```
car = new Object();
car.companyName= "ABC";
car.model= "M1";
car.variant= "V1";
car.price= 780000;
```

Now, we are going to display the above JavaScript object, ‘car’ by using following methods.

(a) A JavaScript object can be displayed by simply displaying properties of that object as a string by using property name,dot (.) notation orbracket ([ ]) notation and ‘+’ operator. ‘+’ operator is used for string concatenation. For example, we can display the object, ‘car’ by using following JavaScript code.

```
document.writeln(car.companyName+ "," +  car.model+ "," 
+ car['variant'] + "," + car['price']);
```

The output of the above JavaScript code will be:

ABC,M1,V1,780000

(b)We can display a JavaScript object by using Object.values() method. In this approach, Object.values() is used to convert a JavaScript object to an array and then that array can be displayed.For example, we can display the object, ‘car’ by using following JavaScript code.

```
let obArray= Object.values(car);
document.writeln(obArray);
```

The output of the above JavaScript code will be:

ABC,M1,V1,780000

(c) We can display a JavaScript object by using JSON.stringify() method. In this approach, JSON.stringify() is used to convert a JavaScript object to a string and then that string cam be displayed.For example, we can display the object, 'car' by using following JavaScript code.

```
let obString= JSON.stringify(car);
document.writeln(obString);
```

The output of the above JavaScript code will be:

{"companyName":"ABC","model":"M1","variant":"V1","price":780000}

(d) We can also construct a string by joining all the properties of a JavaScript object and then we can display that string. Now, we can use '*for loop'* control to access each property of an object and concatenate it using '+' operator to a string one by one. For example, we can display the object, 'car' by using following JavaScript code.

```
let i;
let obString =" ";
for(i in car)
{
        obString += car[i]+" ";
};
document.writeln(obString);
```

In the above example, 'i' in the '*for loop'*, represents each property of the object, 'car' in corresponding iterations of the loop. On the other hand, car[i] gives the property value of the property represented by 'i'.

The output of the above JavaScript code will be:

ABC M1 V1 780000

So, we can also display the object with both properties and corresponding property values using *'for loop'* as shown in the following JavaScript code.

```
 let i;
for(i in car)
{
        document.writeln(i,":", car[i]);
};
```

The output of the above JavaScript code will be:

companyName:ABC
model:M1
variant:V1
price:780000

6.4 JAVASCRIPT OBJECT PROPERTIES AND METHODS

We have already learnt to create a JavaScript object with properties and functions in the introduction of this unit. In this section, we are going to learn about object properties and methods in more detail.

First of all, we have to know the meaning of property of a JavaScript object. Property of a JavaScript object is a variable available in that object. The properties of an object can be accessed publically by using the object and dot (.) notation or bracket ([ ]) notation. For example, consider the following JavaScript code.

```
let  employee ={
        eName: "Pranjal Das",
        eDesignation: "Manager",
        eAddress: " Guwahati, Assam",
        eSalary: 70000,
        eContact: 9786489071,
};

document.writeln("\n Employee name is:",
employee.eName);
document.writeln("\n Contact number of the Employee is:",
employee['eContact']);
```

In the above JavaScript code, the object, ‘employee’ is created with five properties that are ‘eName’, ‘eDesignation’, ‘eAddress’, ‘eSalary’ and ‘eContact’. As shown in the code, the property, ‘eName’ is accessed by using dot(.) notation and the

property, ‘eContact’ is accessed by using bracket([ ]) notation. So the output of the above code will be:

Employee name is: Pranjal Das
Contact number of the Employee is: 9786489071

In JavaScript, we can add new properties to an existing object and delete existing properties. We can includea new property to an already available object by assigning a value to the new property with the help of dot(.) notation or bracket([ ]) notation. We can remove an already available property of an object by using ‘delete’ keyword. We can also modify existing properties of a JavaScript object byassociating new values to the existing properties. For example consider the following JavaScript code.

```
let  employee ={
                eName: "Pranjal Das",
                eDesignation: "Manager",
                eQualification: "MBA",
                eAddress: " Guwahati, Assam",
                eSalary: 70000,
                eContact: 9786489071,
};

let obString = JSON.stringify(employee);
document.writeln("The object ‘employee’ is:");
document.writeln(obString);


employee.eDepartment = "IT";
employee['eAge'] = 45;
obString = JSON.stringify(employee);
document.writeln("\nAfter adding new properties(eDepartment and
eAge), the object is:");
document.writeln(obString);


delete employee.eQualification;
obString = JSON.stringify(employee);
document.writeln("\nAfter removing ‘eQualification’ property, the
object is:");
```

```
document.writeln(obString);


employee.eSalary = 100000;
obString = JSON.stringify(employee);
document.writeln("\nAfter modifying ‘eSalary’ property, the object is:");
document.writeln(obString);
```

The output of the above code will be:

```
The object ‘employee’ is:
{"eName":"Pranjal
Das","eDesignation":"Manager","eQualification":"MBA","eAddress":" Guwahati, Assam","eSalary":70000,"eContact":9786489071}

After adding new properties(eDepartment and eAge), the object is:
{"eName":"Pranjal
Das","eDesignation":"Manager","eQualification":"MBA","eAddress":" Guwahati,
Assam","eSalary":70000,"eContact":9786489071,"eDepartment":"IT","eAge":45}

After removing ‘eQualification’ property, the object is:
{"eName":"Pranjal Das","eDesignation":"Manager","eAddress":" Guwahati,
Assam","eSalary":70000,"eContact":9786489071,"eDepartment":"IT","eAge":45}

After modifying ‘eSalary’ property, the object is:
{"eName":"Pranjal Das","eDesignation":"Manager","eAddress":" Guwahati,
Assam","eSalary":100000,"eContact":9786489071,"eDepartment":"IT","eAge":45}
```

A JavaScript object can contain one or multiple methods to provide different functionalities depending upon the requirements. The methods available in an object can access and modify the properties of that object.A method of an object can also call other

methods of that object and methods of other objects. To invoke a method of an object, dot(.) notation is used. Now consider the following JavaScript code to clearly understand JavaScript object methods.

```
let  student ={
sName: "Sarat Deka",
sRoll: 51,
sCourse: "MSc.IT",
   markP1: 87,
   markP2: 73,
   markP3: 90,
   markP4: 67,

display : function(){
document.writeln("Student name:",student.sName);
document.writeln("Student Roll Number:",student.sRoll);
document.writeln("Course name :",student.sCourse);
document.writeln("Mark obtained in Paper1
:",student.markP1);
document.writeln("Mark obtained in Paper2
:",student.markP2);
document.writeln("Mark obtained in Paper3
:",student.markP3);
document.writeln("Mark obtained in Paper4
:",student.markP4);

   },

percentage: function(){

        let per =(student.markP1+
        student.markP2+student.markP3+student.markP4)/4;
        document.writeln("Percentage of ",student.sName,"
        is = ",per);
 }

};
student.display();
student.percentage();
```

In the above JavaScript code, it is observed that two methods (‘display()’ and ‘percentage()’) are included in the object, ‘student’. ‘display()’ method is defined to display all the values of the properties available in the object and ‘percentage()’ is defined to estimate the percentage of a student.

The output of the above code will be:

```
Student name :Sarat Deka
Student Roll Number :51
Course name :MSc.IT
Mark obtained in Paper1 :87
Mark obtained in Paper2 :73
Mark obtained in Paper3 :90
Mark obtained in Paper4 :67
Percentage of  Sarat Deka is = 79.25
```

6.5 JAVASCRIPT OBJECT CONSTRUCTORS

An object constructor in JavaScript is a function and it is used to initialize the object.Object constructor can be used to create multiple objects of the same type.When JavaScript objects are created based on an object constructor then they inherit all the variables and statements of that object constructor. In JavaScript, any JavaScript function can be used as an object constructor.So, we can define a function using ‘function’ keyword and that can be used as an object constructor. For example, consider the following JavaScript code.

```
function Student(sName,sRoll,sCourse,sPer)   // Object
constructor
{
  this.sName = sName;
  this.sRoll = sRoll;
  this.sCourse = sCourse;
  this.sPer = sPer;
}

let student1 = new Student("Arup Saikia",101,"M.Sc.IT",78);
```

```
let i;
for(i in student1)
{
        document.writeln(i," : ", student1[i]);
};
```

In the above JavaScript code, a function, 'Student ( )' is defined with four parametersso that it can be used as an object constructor. Then, using 'new' keyword, an object, 'student1' is created by calling the object constructor that is 'Student()'.

So, the output of the above code will be:

```
sName : Arup Saikia
sRoll : 101
sCourse : M.Sc.IT
sPer : 78
```

We can also use class syntax to create an object constructor in JavaScript. For example, consider the following JavaScript code.

```
class Student
{
  constructor(sName,sRoll,sCourse,sPer)  //Object constructor
   {
        this.sName = sName;
        this.sRoll = sRoll;
        this.sCourse = sCourse;
        this.sPer = sPer;
   }
}

let student1 = new Student("Pranjal Deka",102,"Assamese",65);

let i;
for(i in student1)
{
    document.writeln(i," : ", student1[i]);
};
```

In the above JavaScript code, a constructor function is defined within the class, ‘Student’. Then the object, ‘student1’ of the class, ‘Student’ is created using ‘new’ keyword.

So, the output of the above code will be:

sName :Pranjal Deka
sRoll : 102
sCourse : Assamese
sPer : 65

STOP TO CONSIDER

‘this’ is a JavaScript keyword. It is used inside an object constructor to refer the new object. When it is used inside a method of an object then it refers the object. When it is used outside of any function and inside of a function which is not a method of any JavaScript object then it refer the global object like ‘window’ in case of the web browser environment.

6.6 JAVASCRIPT OBJECT PROTOTYPES

In JavaScript, prototype can be referred as a mechanism by which objects can inherit properties and methods from other objects and functions. So, a JavaScript object can inherit properties and methods from a prototype. For example, a Date object is inherited from Date.prototype. ‘prototype’ is a built-in property available in every JavaScript objects and functions. This property actually an object in JavaScript and we can add properties and methods to this object. When we add properties and methods to ‘prototype’ property of an object then it allows other objects to inherit these properties and methods.So, every JavaScript object is related to a ‘prototype’ property.Now, when an object inherits properties and methods from the related prototype and again that prototype can inherits properties and methods of its related prototype then it forms a prototype chain. Now consider the following JavaScript code to understand the creation of a new object from a prototype.

```
let studentPrototype = {              // Prototype object
        sName:"Arnab Sarma",
```

```
        sRoll: 109,
        sCourse: "BCA",
        sPercentage: 81


};

let student  =Object.create(studentPrototype);
student.display= function(){
        document.writeln("Student
name:",student.sName);
        document.writeln("Student Roll
Number:",student.sRoll);
        document.writeln("Student
Course:",student.sCourse);
        document.writeln("Student
Percentage:",student.sPercentage);
 }

student.display();
```

In the above JavaScript code, the object, 'student' is created by using Object.create( ) method by inheriting the properties of the prototype object, 'studentPrototype'.

The output of the above code will be:

```
Student name:Arnab Sarma
Student Roll Number:109
Student Course:BCA
Student Percentage:81
```

We can also add properties and methods to the prototype property of a constructor function so that these can be inherited by all objects that are created with that constructor function using 'new' operator. For example, consider the following JavaScript code.

```
function Student(sName, sRoll, sCourse, sPer)
{
this.sName = sName;
this.sRoll = sRoll;
this.sCourse = sCourse;
```

```
this.sPer = sPer;
this.display = function(){
document.writeln("Student name :",this.sName);
document.writeln("Student Roll Number :",this.sRoll);
document.writeln("Course name :",this.sCourse);
document.writeln("Percentage :",this.sPer);
        }


}

let student1 = new Student("Rajib Deka",22,"M.Com.",91);
student1.display();
```

In the above JavaScript code, ‘Student()’ is a constructor and ‘student1’ is an object created using ‘new’ operator based on the constructor, ‘Student()’. 'student1’ inherits all the properties and one method from ‘Student()’.

The output of the above code will be:

```
Student name :Rajib Deka
Student Roll Number :22
Course name :M.Com.
Percentage :91
```

The property, ‘prototype’ can also be used to include new properties and methods to an object constructor. For example, consider the following JavaScript code.

```
function Employee(Name,Designation,Address,Salary){
this.eName=Name;
        this.eDesignation=Designation;
        this.eAddress=Address;
        this.eSalary=Salary;


}

Employee.prototype.eContact = 78654895431;
Employee.prototype.display = function() {
document.writeln(" Employee name is:", this.eName);
document.writeln(" Employee Designation is:",
this.eDesignation);
```

```
    document.writeln(" Employee Address is:", this.eAddress);
    document.writeln(" Employee Salary is:", this.eSalary);
    document.writeln(" Employee Contact no. is:",
this.eContact);
    };

    let employee1=new Employee("Tilok
Bora","Manager","Guwahati",120000);
    employee1.display();
```

In the above JavaScript code, The property, 'prototype' is used to add the property, 'eContact' and the method, 'display()' to the object constructor, 'Employee'. Then the object, 'emplyee1'is created using 'new' operator.

The output of the above code will be:

```
Employee name is:Tilok Bora
Employee Designation is:Manager
Employee Address is:Guwahati
Employee Salary is:120000
Employee Contact no. is:78654895431
```

6.7 IMPORTANT JAVASCRIPT OBJECTS

The following presents some important JavaScript objects with their important properties and methods.

(i) String: String object is used to work with strings.It provides methods to manipulate strings in JavaScript.

Property:

- **length** is a property of Sting object that returns the length of the string.

Methods:

- **charAt(pos)** returns the character available at the index position, 'pos' in a string.
- **charCodeAt(pos)** returns the Unicode of the character available at the index position, 'pos' in a string.

- **concat(str)**returns a string which is the result of concatenation of the string, ‘str’ at the end of the string which invokes it.
- **indexOf(str,pos)**returns the index position of the first occurrence of the substring, ‘str’ in a string. ‘pos’ is the starting index position from where the search operation will start. If the substring is not available in the string then it will return -1.
- **lastIndexOf(str,pos)** returns the index position of the last occurrence of the substring, ‘str’ in a string. ‘pos’ is the starting index position from where the search operation will startbut the direction of the searching is performed towards 0 index position of the string. If the substring is not available in the string then it will return -1.
- **slice(startIndex,endIndex)**returns a substringof a string where ‘startIndex’ is the index position in the string from where the substring starts and the end index position of the substring will be one less than the index,‘endIndex’. It means the character available at the index, ‘endIndex’ will not be included in the substring. If ‘**startIndex’** is greater than **‘endIndex’** in**slice()** then it will return an empty string.
- **split(splitchars)**splits a string into array of substrings. ‘splitchars’ is a character or group of characters which indicate the end of each substring in the string.
- **substr(startIndex,len)**returns a substring of a string from the index position ‘startIndex’ and the length of the substring will be ‘len’.
- **substring(startIndex,endIndex)**works similar to **slice(startIndex , endIndex).**ButIf ‘**startIndex’** is greater than **‘endIndex’** in**substring( )**then it will swap the two parameters and provide a substring depending upon ‘**startIndex’ and ‘endIndex’**.
- **toLowerCase()** returns the string after converting its all uppercase characters to lowercase letters.

- **toUpperCase()** returns the string after converting its all lowercase characters to uppercase letters.

(ii) Number: Number object provides properties andmethods for mathematical operation to manipulate numbers.

Properties:

- **MAX_VALUE**is a property of Number object that indicates the biggest number possible in JavaScript programming.
- **MIN_VALUE**is a property of Number object that indicates the smallest number possible in JavaScript programming.
- **NEGATIVE_INFINITY** is a property of Number object that indicates –Infinity (Negative Infinity).
- **POSITIVE_INFINITY** is a property of Number object that indicates Infinity.
- **NaN**is a property of Number object that indicates "Not-a-Number" value.

Methods:

- **toString()**returns the string representation of a Number object.
- **toExponential()** returns a number in exponential notation for a Number object.
- **valueOf()**returns the numeric value of a Number object.

(iii) Boolean: Boolean object provides methods to manipulate Boolean values.

Methods:

- **toString()**returns the string representation of a Boolean object.
- **valueOf()**returns true or false depending upon the Boolean object.

(iv) Math: Math object provides methods to do different mathematical calculations.

Methods:

- **max(num1,num2)**returns the bigger value between num1 and num2.
- **min(num1,num2)** returns the smaller value between num1 and num2.
- **exp(num)**returns e^{num} . E is the Euler's number.
- **pow(a,b)**returns a^b .
- **sqrt(num)**returns square root of num.
- **abs(num**) returns absolute value of num.
- **ceil(num)**returns the smallest integer that is not less than num.
- **floor(num)**returns the largest integer that is not greater than num.
- **round(num)**returns closest integer to num.
- **log(num)**returns natural logarithm(base e) of num.
- **cos(num)** returns trigonometric cosine of num.
- **sin(num)**returns trigonometric sine of num.
- **tan(num)**returntrigonometric tangent of num.

(v) Date: Date object provides methods to manipulate date and time in JavaScript.

Methods:

- **getDate()**returns a number (1 to 31) which indicates the day of the month in local time.
- **getMilliseconds()** returns a number (0 to 999) which indicates milliseconds of the time in local time.
- **getSeconds()**returns a number (0 to 59) which indicates seconds of the time in local time.
- **getMinutes()**returns a number (0 to 59) which indicates minutes of the time in local time.
- **getHours()**returns a number (0 to 23) which indicates hours of the time in local time.

- **getDay()**returns a number (0 to 6) which indicates the day of the week in local time.
  0 for Sunday
  1 for Monday
  2 for Tuesday
  3 for Wednesday
  4 for Thursday
  5 for Friday
  6 for Saturday
- **getMonth()**returns a number (0 to 11) which indicates the month of the year in local time.
  0 for January
  1 for February
  2 for March
  3 for April
  4 for May
  5 for June
  6 for July
  7 for August
  8 for September
  9 for October
  10 for November
  11 for December

- **getFullYear()**returns the year in local time.
- **getTime()**returns the number of milliseconds between midnight of1st January 1970, and a particular date object.
- **setDate(x)**isused to set the day of the month in local time. Value for the argument, x should be from 1 to 31.
- **setMilliseconds(msec)**isused to set milliseconds of a Date object in local time.
- **setSeconds(sec,msec)**is used to set seconds of a Date object in local time. The first argument is used to set seconds and the second argument can be used to set milliseconds. The second argument is optional.
- **setMinutes(min, sec,msec)**is used to set minutes of a Date object in local time. The first argument is used to set minutes. The second argument can

be used to set seconds and the third argument can be used to set milliseconds. The second and third arguments are optional.

- **setHours(hr , min , sec ,msec)**isused to set minutes of a Date object in local time. The first argument is used to set minutes. The second argument can be used to set seconds and the third argument can be used to set milliseconds. The second and third arguments are optional.
- **setMonth(mo, da)**isused to set month of a Date object in local time. The first argument is used to set the month and the second argument can be used to set the date. The second argument is optional.
- **setFullYear(yr , mo, da)**is used to set the year of a Date object in local time. The first argument is used to set the year. The second argument can be used to set the month and the third argument can be used to set the date. The second and third arguments are optional.
- **toLocaleString() c**onverts a Date object to a string andreturns the date and time specific to the computer's locale.
- **toString()**convert a Date object to a string and returns the string. This string provides the date and time specific to the computer's locale.
- **valueOf()**returns the number of milliseconds since midnight of 1st January,1970.

(vi) document: document object is used to manipulate the document available in the browser window. We have already used two methods of document object that are write() and writeln().

Properties:

- **cookie**is a property of document objectthat provides the values of the cookies associated with the present document.
- **lastModified**is a property of the document object that gives the date and time of the last modification performed on the present document.

(vii) window:window object provides methods and properties to manipulate the browser window.

Properties:

- **document** is a property that represents the document object available in a window object.
- **closed** is a property that returns true if the window is closed.
- **opener** is a property that returns the window object that opened the current window.

Methods:

- **open(u ,wname,fe)**isused to create a new window. 'u' represents the URL of the window, 'wname' represents the name and 'fe' is a string that represents the visible features.
- **prompt(pr,de)**isused to display a dialog box. We have already used it in the earlier chapter.
- **close()**isused to close the current window and its object is released from the memory.
- **focus()**isused to set focus to the current window.
- **blur()**isused to get rid of the focus from the current window.

Now, consider the following JavaScript code to understand the use of the properties and methods of String object.

```
let str = new String("Welcome to Gauhati University CDOE");
document.writeln("String in str object is= ",str);
document.write("Length of the String in str object is=");
document.writeln(str.length);
document.write("Character available at index position 5 is=");
document.writeln(str.charAt(5));
document.write("Unicode of the character available at index
position 5 is=");
document.writeln(str.charCodeAt(5));
let str2nd=new String(",Assam");
document.write("After concatenation of str and str2nd, the string
is=")
```

```
document.writeln(str.concat(str2nd));
document.write("Index position of the first occurrence of Gauhati
is=");
document.writeln(str.indexOf("Gauhati",0));
document.write("Index position of the last occurrence of Gauhati
is=");
document.writeln(str.lastIndexOf("Gauhati"));
document.write("Substring extracted from the string with start index
5 and end index 15:");
document.writeln(str.slice(5,15));
document.write("Array of substrings where each substring is end
with a blank space=");
document.writeln(str.split(" "));
document.writeln("String after converted all characters to
lowercase:",str.toLowerCase());
document.writeln("String after converted all characters to
uppercase:",str.toUpperCase());
```

The output of the above code will be:

```
String in str object is= Welcome to Gauhati University CDOE
Length of the String in str object is=34
Character available at index position 5 is=m
Unicode of the character available at index position 5 is=109
After concatenation of str and str2nd, the string is=Welcome to
Gauhati University CDOE,Assam
Index position of the first occurrence of Gauhati is=11
Index position of the last occurrence of Gauhati is=11
Substring extracted from the string with start index 5 and end index
15:me to Gauh
Array of substrings where each substring is end with a blank
space=Welcome,to,Gauhati,University,CDOE
String after converted all characters to lowercase:welcome to
gauhati university cdoe
String after converted all characters to uppercase:WELCOME TO
GAUHATI UNIVERSITY CDOE
```

So, as shown in the above JavaScript code, we can use the properties and methods of any JavaScript object depending upon our requirements.

CHECK YOUR PROGRESS

1. _____method is used to create an object based on a prototype object.

(a) Object.create()
(b) Object constructor
(c) new
(d) None of the above

2. Which of the following is used to convert a JavaScript object to a string?

(a) Object.create()
(b) JSON.stringify()
(c) toObjectString()
(d) None of the above

3. What will be the output of the following JavaScript code?

```
Employee = new Object();
Employee.eName= "Pallab Das";
Employee.eAddress= "Guwahati";
Employee.eContact= 76547894321;
Employee.eSalary= 780000;

let i;
let obStr =" ";
for(i in Employee)
{
        obStr += Employee[i]+" ";
};
document.writeln(obStr);
```

(a)PallabDas , Guwahati , 76547894321 , 780000
(b) Pallab Das Guwahati 76547894321 780000
(c)PallabDasGuwahati76547894321780000
(d) None of the above

4. In JavaScript, the properties of an object can be accessed publically by using the object and _____.

(a) dot (.) notation
(b) bracket ([ ]) notation.

(c) :notation

(d) Both (a) and (b).

5. In JavaScript,_____ is used to initialize an object.

(a) Prototype object

(b) new

(c) object constructor

(d) None of the above

6. _____ indicates "Not-a-Number" value in case of a Number object.

(a) NaN

(b) NaM

(c) NuN

(d) None of the above

7. _____returns a number (1 to 31) that indicates the day of the month in local time.

(a) getDate()

(b) getDay()

(c) getTime()

(d) None of the above

8. Which of the following JavaScript object is used to manipulate the document available in the browser window?

(a) window object

(b) Math object

(c) document object

(d) String object

6.8 SUMMING UP

- A JavaScript object is a set of functions or methods and properties.
- A JavaScript object can be created using different ways. A JavaScript object can be created by using object literal notation. 'new' keyword can also be used to create a JavaScript object.
- JSON.stringify() is used to convert a JavaScript object to a string.

- Property of a JavaScript object is a variable available in that object. The properties of an object can be accessed publically by using the object and dot (.) notation or bracket ([ ]) notation.
- In JavaScript, we can add new properties to an existing object and delete existing properties.
- To invoke a method of an object, dot(.) notation is used.
- An object constructor in JavaScript is a function which initializes the object.
- Object constructor can be used to create multiple objects of the same type. When JavaScript objects are created based on an object constructor then they inherit all the variables and statements of that object constructor.
- In JavaScript, any JavaScript function can be used as an object constructor.
- In JavaScript, prototype can be referred as a mechanism by which objects can inherit properties and methods from other objects and functions.
- Examples of some important JavaScript objects are String, Number, Boolean, Math, Date, Window etc.

6.9 ANSWERS TO CHECK YOUR PROGRESS

1. (a) Object.create() ,2.(b) JSON.stringify() ,
3. (b) Pallab Das Guwahati 76547894321 780000, 4. (d) Both (a) and (b) ,
5. (c) object constructor , 6. (a) NaN,
7. (a) getDate() , 8. (c) document object

6.10 POSSIBLE QUESTIONS

1. Write down the different ways to create and display objects in JavaScript.
2. What is JavaScript object property? Give example.
3. Write a short note on JavaScript object constructor.
4. Explain JavaScript object prototype with example.
5. Write down some important methods provided by Math object.

6.11 REFERENCES AND SUGGESTED READINGS

- Harvey, Harvey Deitel, and Abbey Deitel. "Internet and World Wide Web How To Program." (2011).
- Gosselin, Don. *Javascript*. Course Technology Press, 2004.
- Pollock, John. *JavaScript, A Beginner's Guide*. McGraw Hill Professional, 2009.
- Thau, Dave, and Dave Thau. *The Book of JavaScript: A Practical guide to interactive Web pages*. No Starch Press, 2000.

---×---

UNIT-7
JAVASCRIPT FUNCTIONS

UNIT STRUCTURE

7.1 INTRODUCTION

We have already learnt about the basics of JavaScript and important concepts of JavaScript objects in the earlier chapters. In this unit, we are going to learn about functions in JavaScript. We can add specific functionalities by forming groups of JavaScript statements and these groups are called as functions. We have already studied different built-in functions available under different JavaScript objects. In this chapter, we will learn about some global functions available in JavaScript. In JavaScript, we can also execute a specific function on the occurrence of a specific event in a webpage. In this chapter, this event handling mechanism will also be discussed. Finally, client-side form validation can be possible by using the event handling mechanism and different functions and properties available in different JavaScript objects. For this purpose, we can also define our own functions.

7.2 OBJECTIVES

After going through this chapter, we will be able to learn:

- About JavaScript functions.
- How to define user-defined functions in JavaScript.
- About global functions in JavaScript.
- About recursive function.
- About JavaScript events.
- About client-side form validation in JavaScript.

7.3 INTRODUCTION TO JAVASCRIPT FUNCTIONS

A JavaScript function is a group of JavaScript statements which accomplish a definite job or provide a specific output. In JavaScript, different built-in functions are already available. We have already learnt about different built-in functions that are available under different JavaScript objects in the earlier chapters. JavaScript programmers can also define their own JavaScript functions to perform specific jobs depending upon their requirements and these functions are called User-Defined functions. A JavaScript function can be used to perform its job by using function call. A function call can be placed by the function name followed by parentheses (()) and one or more arguments (if any or if required) can be passed within the parentheses.

7.3.1DefiningUser-Defined Function

A JavaScript programmer can define a function using 'function' keyword. The syntax to define a User-Defined function is presented as follows.

```
function function_Name( Parameter1 , Parameter2 , Parameter3 ,……)
{
        JavaScript Statements
}
```

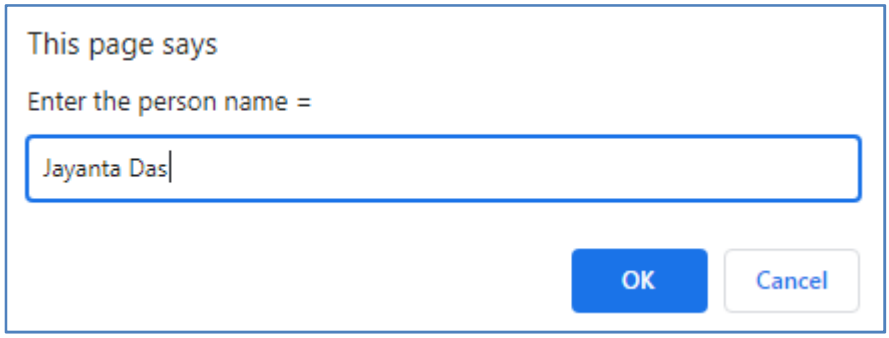
From the above syntax, it is observed that a function name must be provided after the keyword, 'function' to define a

JavaScript User-Defined function. The group of JavaScript statements of the function are added within the curly braces ({ }). A list of parameters is optional in a function definition. If the function requires any information to perform its job then that information is passed through the parameters to the function. The programmer can declare variables within a function definition depending upon the requirements and these variables are called local variables of that function. 'return' statement can be used to return values from functions. An example of User-Defined JavaScript function is presented in the following JavaScript code.

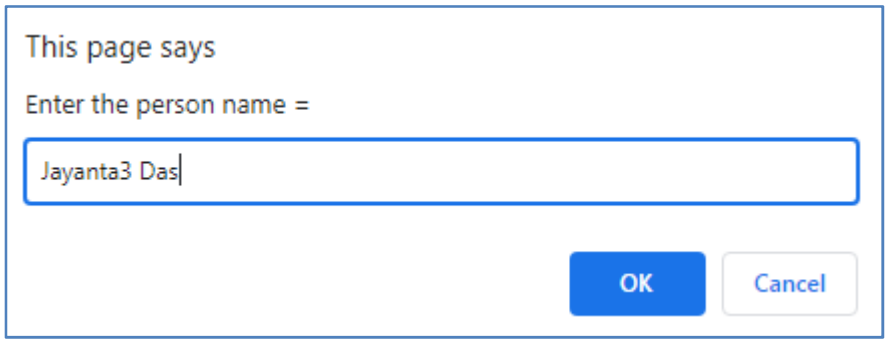
```
<html>
<head>
<title>Name Validation</title>
<script type = "text/javascript" >
      function isValidName(personName)  // User-Defined function
to validate a name
      {
for(i =0;i<personName.length;i++)
          {
              co = personName.charCodeAt(i);
              if(!(((co>=97) && (co<=122)) || ((co>=65) &&
(co<=90)) || (co==32)))
              {
                   return "Invalid Name";

              }
          }
           return "Valid Name";
      }

     var pName = prompt("Enter the person name =");
document.writeln(isValidName(pName));

</script>

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

In the above JavaScript code, a function, 'isValidName(personName)' is defined to validate a name. A person name is passed as a parameter to this function.

The output of the above code will be:



After clicking 'OK' button, the result will be:
Valid Name



After clicking 'OK' button, the result will be:
Invalid Name

7.4GLOBAL FUNCTIONS IN JAVASCRIPT

In JavaScript, if a function is defined outside any other function or block then it is referred as a global function. A global function can be called from any part of a JavaScript code. In JavaScript, some useful built-in global functions are available. These functions are available globally in JavaScript that mean these can be called from any part of any JavaScript code. Some important built-in global functions available in JavaScript are presented as follows.

parseInt(): The **parseInt()** function can have two arguments. The first argument is a string and

the function try to convert it into an integer number. If the function cannot convert the string into an integer number then it returns NaN. The second argument to this function is optional and it specifies the base of the number. So, if the second argument is 2 then it means that the number obtained from the string passed as the first argument is in binary format.

parseFloat(): The **parseFloat()** function try to convert a string into a floating point number. A string is passed as argument to this function. If the function cannot convert the string into a floating point value then it will return NaN.

eval(): The **eval()** function execute JavaScript code which is passed as a string to it.

isFinite(): The **isFinite()** function is used to check whether a number is finite. It returns true if the argument is not NaN, Number.POSITIVE_INFINITYorNumber.NEGATIVE_INFINITY.

isNaN(): The **isNaN()** function is used to check whether its argument is NaN. So, it returns true if its argument is not a number.

escape(): A string is passed as an argument to the **escape()** function. The **escape()** function returns a string that is created from the string argument by encodingit's all spaces, punctuation, accent characters and the characters that are not in the ASCII character set. This encoding is performed in a hexadecimal format.

unescape(): The **unescape()** function returns a string in which all the characters previously encoded by the escape() function are decoded.

7.5RECURSIVE FUNCTION

A function which calls itself is refered as a recursive function. A recursive function calls itself repeatedly until the required solution of a computational problem is attained. A

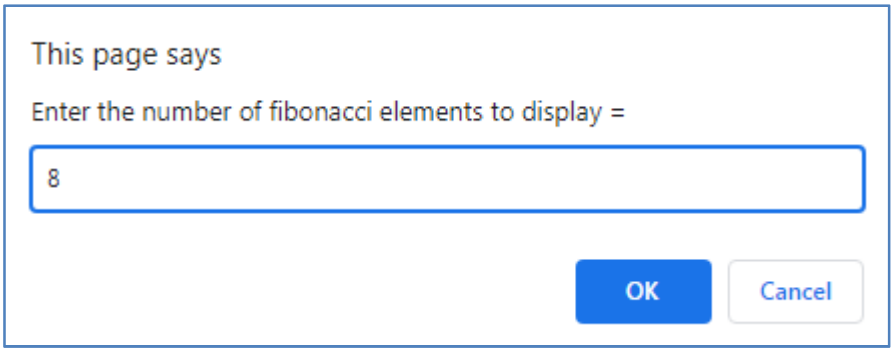
recursive function must have a condition to stop calling itself and this condition is depended upon the desired solution of the problem. For example, consider the following JavaScript code to understand the use of recursive function.

```
<html>
<head>
<title>Fibonacci Series</title>
<script type = "text/javascript" >
function fibonacci(T)  // Recursive function to display fibonacci series
      {
         if(T<2)
         {
                return T;
          }
         else
         {
                return (fibonacci(T-1) + fibonacci(T-2));  // Function calling itself
          }

      }

      var temp = prompt("Enter the number of fibonacci elements to display =");
      var no_Of_Elem = parseInt(temp);
      if(no_Of_Elem<=0)
document.writeln("Wrong input, a positive integer number should be entered");
      else
{
document.writeln("\n Required Fibonacci Series is::");
for(var i=1; i<=no_Of_Elem;i++)
document.writeln(fibonacci(i));
      }
</script>

</head>
<body>
</body>
```

</html>

In the above code, fibonacci() is a recursive function used to display the Fibonacci series with specified number of elements.

The output of the above code will be:



After clicking on OK button, the result will be:

Required Fibonacci Series is::
1
1
2
3
5
8
13
21

CHECK YOUR PROGRESS

1. Fill in the blanks

(a) A function which calls itself is refereed as a _____ function
(b) _____ convertsa string into an integer number.
(c) The isFinite() function is used to check _____.
(d) _____ statement can be used in a user-defined JavaScript function to return values from that function.
(e) A JavaScript programmer can define a function using _____ keyword.

7.6EVENT HANDLING IN JAVASCRIPT

When the state of an HTML object is changed due to any action accomplished by the user or the browser then it is refereed as an Event.In JavaScript, we can write code to perform specific task on the occurrence of a specific event. This process can be referred as Event Handling in JavaScript and the code can be called as Event handler. For example, using JavaScript code,a specific message can be displayed on the browser when click operation performed on a specific button available in a HTML document.

The syntax to handle event in JavaScript is presented as follows.

< HTML-Element HTML-Event= 'Event Handler (JavaScript Function)'>

Some important HTML events with their meanings are presented as follows.

- **onload :** Loading of an HTML element is completed successfully.
- **onunload :** A page is set to unload.
- **onfocus :** An HTML element becomes focused.
- **onblur :**Focus from an HTML element is removed.
- **onclick :** The user clicks a HTML element with the mouse.
- **onmousemove :**The mouse cursor moves over an HTML element. This event continuously active till the mouse cursor moves within the boundaries of the HTML element.
- **onmouseover :**The mouse cursor moves over a HTML element. This event is triggered only once when the mouse cursor just enters inside the boundary of the HTML element.
- **onmouseout :** The mouse cursor leaves a HTML element.
- **onsubmit :** The user clicks the submit button of a HTML form or press the Enter key.
- **onreset :**The user reset a HTML form.

- **onkeydown**:Any key of the keyboard is pressed down.
- **onkeypress :** Any key that produces a character is pressed down. So if the user pressed down the keys like ‘Ctrl’, ‘Shift’, ‘Alt’ etc. then this event will not triggered.
- **onkeyup :**The user releases a key of the keyboard after being pressed.
- **onchange :** An HTML element is changed.

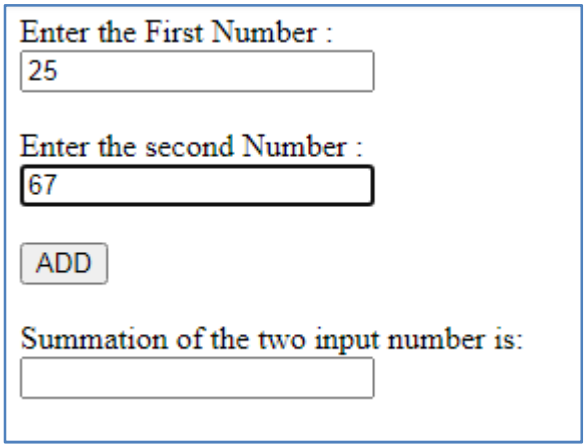
Now to understand the use of HTML events by event handling in JavaScript, consider the following HTML document with JavaScript code.

```
<html>
<head>
<title></title>
<script type = "text/javascript" >
      function add()
      {
 var first = document.getElementById("t1").value; /* Get value
            available in the first text box */

first_n = parseInt(first);  // Converted into integer number
            var second = document.getElementById("t2").value;/* Get value
            available in the second text box */

second_n = parseInt(second); // Converted into integer number
document.getElementById("t3").value = first_n + second_n;

      }

</script>

</head>
<body>
<label for="first">Enter the First Number  :</label><br>
```

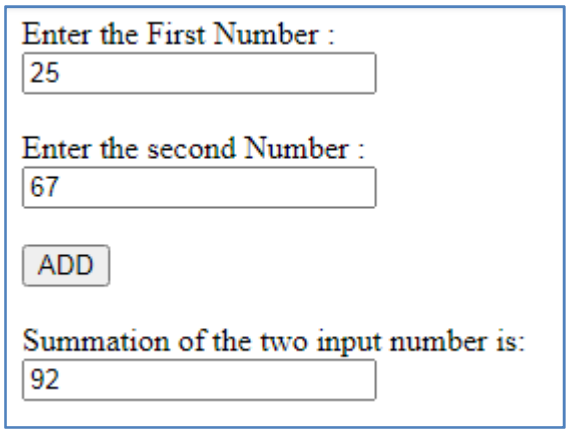
```
<input type="text" id="t1"/><br><br>
<label for="second">Enter the second Number :</label><br>
<input type="text" id="t2"/><br><br>
<input type="button" value="ADD" onclick="add()" /><br><br>
<label for="result">Summation of the two input number is:</label><br>
<input type="text" id="t3"/><br><br>

</body>
</html>
```

In the above HTML document, the use of 'onclick' event is presented. Here two text boxes are provided to input two numbers. In the JavaScript part, the function add() is defined and it will be called when onclick event is triggered on the button(ADD) available in the HTML body.The add() function will convert the string values received from the two text boxes into integer numbers using parseInt() method. Then the summation of these two numbers is placed on the third text box available in the HTML body.

The output of the above code will be:



Now on clicking the button(ADD), the following output will be displayed.



7.7CLIENT-SIDE FORM VALIDATION

We have already learnt in earlier chapters about client-side form design in HTML.When different types of data are inserted in the different controls or input elements available in a HTML form then these data must be in proper format. Before form submission, it must be validated that all necessary form controls are filled with data in proper format. This process is called as client-side form validation. For example, when a name of a person is inserted into a text box then it must be validated that the name must not contain any character other than the alphabets.

Some examples of data validation are presented as follows:

- **Validation of name:** The HTML form control to store a name must not be empty and the entered name must not contain any character other than the alphabets.
- **Validation of Email:** The entered Email must be a valid Email.
- **Validation of Contact Number:** The entered contact number must be a valid contact number and it must contain only digits.
- **Validation of Date:** The entered date must be a valid date.
- **Validation of Password:** The HTML form control to input the password must not be empty and the entered password must contain at least one digit, one uppercase letter, one lowercase letter and one special character.

When data validation is not successful on any HTML control depending upon the input data and the corresponding validation criteria then an alert message is displayed and the focus

is set back to that control. If all the data validations available in a HTML form are successfully completed on submitting the form then true is returned that means form submission is successful and it allows form data to be submitted to the server so that it can be stored in a database.

There are two ways to validate data in a HTML form control. We can use validation attributes on form controls to validate data. Different validation attributes are presented as follows.

- **'required'** attribute make sure that a HTML form control must be filled with data before submitting the corresponding HTML form.

  Example:<input type="text" name="contact_no" required>

- **'minlength'** attribute is used to set the minimum length of the string which will be the input to a HTML form control.

  Example: <input type="text" name="address" minlength="4">

- **'maxlength'** attribute is used to set the maximum length of the string which will be the input to a HTML form control.

  Example:<input type="text" name="address" maxlength="50">

- **'min'** attribute is used to set the minimum numeric value that can be entered into a HTML numeric input field.

  Example:<input type="number" name="percentage" min="30">

- **'max'** is used to set the maximum numeric value that can be input to a HTML numeric input field.

  Example: <input type="number" name="percentage" max="100">

- **'type'** attribute is used to specify the type of the data that will be input to a HTML form control.

  Example:<input type="number" name="age">

- **'pattern'** attribute is used to specify a regular expression that ensures the pattern of the input data.

  Example:<input type="text" name="person_name" pattern= "[A-Za-z\s]+">

We can also write our own JavaScript codes for different data validations on different HTML form controls and these can be used by using event handling mechanism.

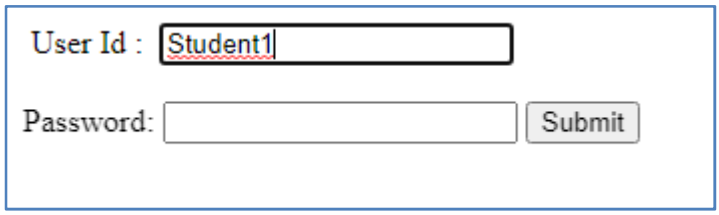
Example:

```
<html>
<head>
<title>Data Validation</title>
<script type = "text/javascript" >
      function validate()   /* Function for data validation
invoked when onclick
                                    event is triggered on the submit
button */
          {

var valSuccess= true;
var userid = document.getElementById("user").value;
          var passwd =
document.getElementById("pass").value;

          if(userid==""){
alert("Please input User-Id");
valSuccess= false;

          }
          if(passwd==""){

alert("Please input Password");
valSuccess= false;


          }

return valSuccess;
```

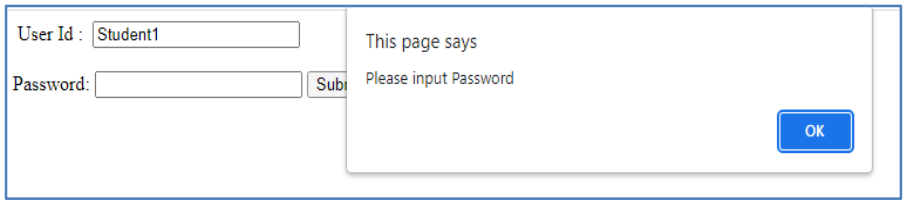
```
        }
        function focus_User()
        {
    document.getElementById("user").focus();
        }

    </script>

    </head>
    <body onload=focus_User()>
    <form >

     <label for="User">User Id :</label>

     <input type="text" id="user" name="userId"><br><br>
    <label for="password">Password:</label>

    <input type="text" id="pass" name="password">
    <input type="submit" onclick="validate()"> /* validate() is
invoked on

clicking the submit button*/

    </form>

    </body>
    </html>
```

With the above HTML code, a form is designed with two text boxes where the first text box is used to input user id and the second text box is used to input password. A submit button is also provided in the form to submit the input data. Now, when onclick event is triggered on the submit button then a JavaScript function, validate() is invoked. The purpose of validate() is to check whether the text boxes are left blank. If the text boxes are blank then on clicking the submit button, alert messages will be displayed.

The output of the above code:



Now on clicking the Submit button:



Now consider the following HTML code where a simple form is designed to input student data. Here, necessary data validations are performed by using both built-in validation attributes and user-defined JavaScript function.

```
<html>
<head>
<title>Data Validation</title>
<script type = "text/javascript" >
      function validate()  // Function for data validation
      {

         var val_Success=true;

            // Validation for Student's name

         var sName = document.getElementById("sname").value;

for(i = 0 ; i<sName.length ; i++)
          {
            co = sName.charCodeAt(i);
            if(!(((co>=97) && (co<=122)) || ((co>=65) && (co<=90))
      || (co==32)))
            {
          alert(" Enter a valid Name");
```

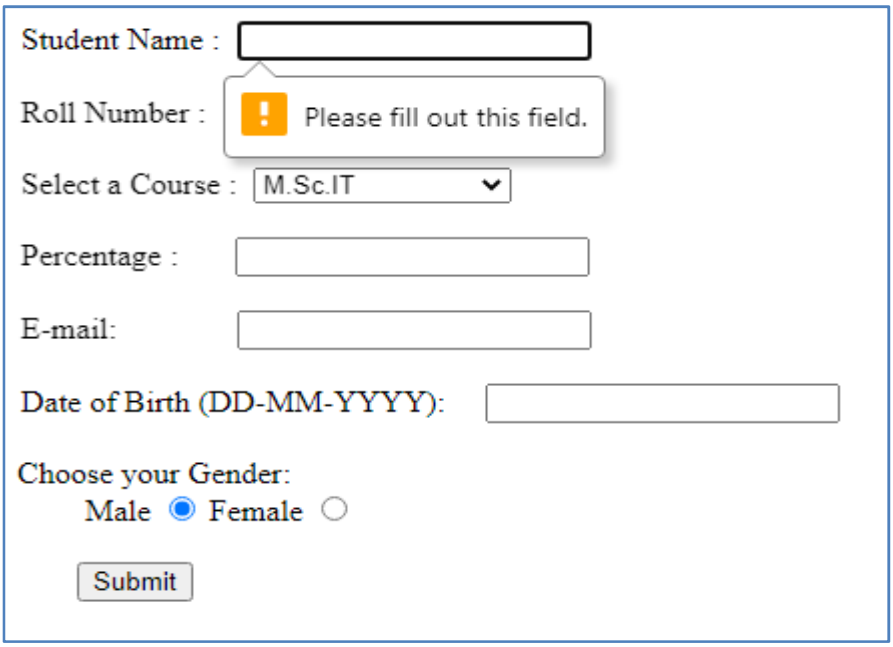
```
val_Success=false;

            }
          }

            // Validation for Date

          var sdob = document.getElementById("dob").value;

          if (!/^\d{2}-\d{2}-\d{4}$/.test(sdob))
          {

alert(" Enter a valid Date in DD-MM-YYYY format");
val_Success = false;
              return val_Success;

          }

splitD = sdob.split("-");
constsday = parseInt(splitD[0], 10);
constsmonth = parseInt(splitD[1], 10);
constsyear = parseInt(splitD[2], 10);

          if (smonth< 1 || smonth> 12)
          {

alert(" Enter a valid Month in Date");
val_Success=false;
          }

          if (sday< 1 || sday> 31)
          {

alert(" Enter a Valid Day in Date");
val_Success=false;
          }

          else
          {

             if (smonth == 2)
```

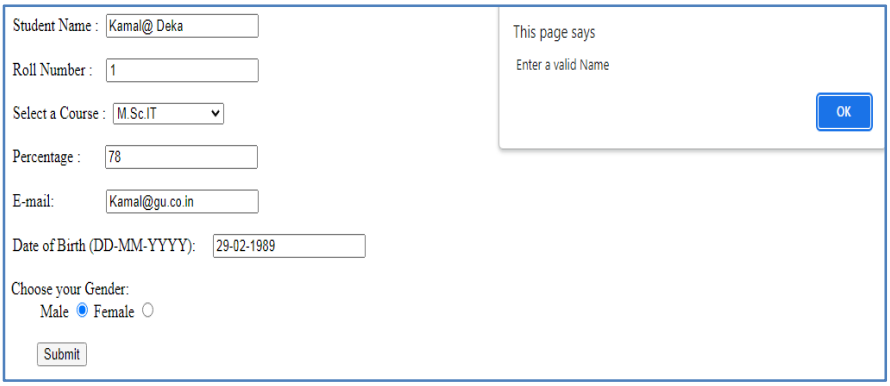
```
            {
              if(sday>29)
              {
alert(" Enter a Valid Day in Date");
val_Success=false;
              }
              else
              {
if(!((syear % 4 == 0 &&syear % 100 !== 0) || (syear % 400 == 0)))
                {

if(sday == 29)
                    {
alert(" Enter a valid Day in Date (Not a Leap Year)");
val_Success=false;
                    }
                }
              }
            }

            if (smonth == 4 &&sday==31)
            {

alert(" Enter a valid Day in Date");
val_Success=false;
            }
            if (smonth == 6 &&sday==31)
            {

alert(" Enter a valid Day in Date");
val_Success=false;
            }
            if (smonth == 9 &&sday==31)
            {

alert(" Enter a valid Day in Date");
val_Success=false;
            }
            if (smonth == 11 &&sday==31)
            {
```

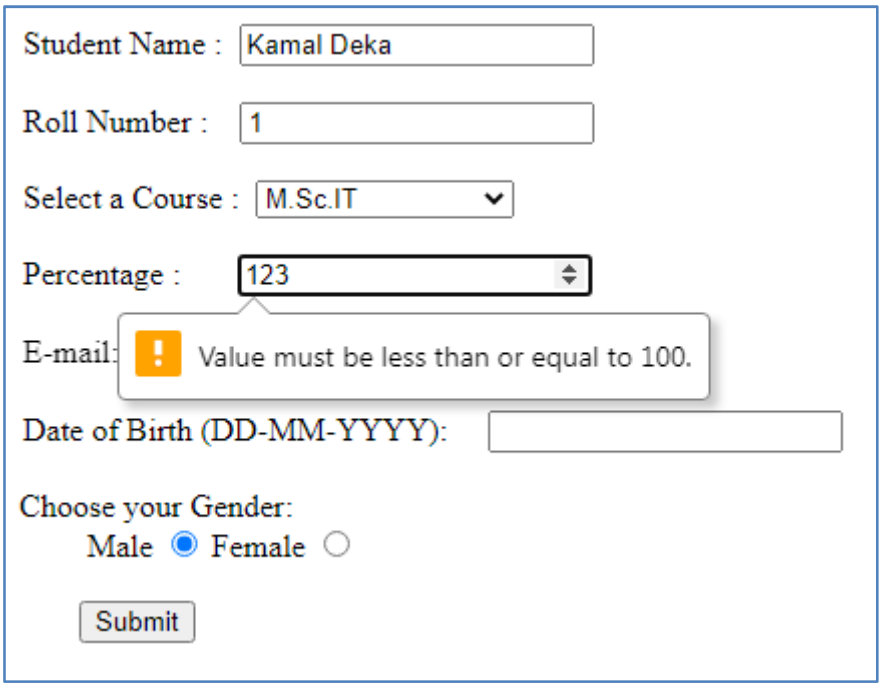
```
alert(" Enter a valid Day in Date");
val_Success=false;
                }
          }

          return val_Success;


      }


 function focus_User()
{
document.getElementById("user").focus();
 }

</script>

</head>
<body onload=focus_User()>
<form >

 <label for="Student">Student Name :</label>
                // 'required' attribute is used for validation
 <input type="text" id="sname" name="name"
        required><br><br>
 <label for="Roll_No">Roll Number  :</label>

   <input type="number" id="roll" name="rollno"
        required><br><br>

 <label for="course">Select a Course :</label>


 <select name="courses" id="course">
<option value="MScIT">M.Sc.IT</option>
<option value="Assamese">Assamese</option>
<option value="Political Science">Political Scienece</option>
<option value="English">English</option>
</select><br><br>
```

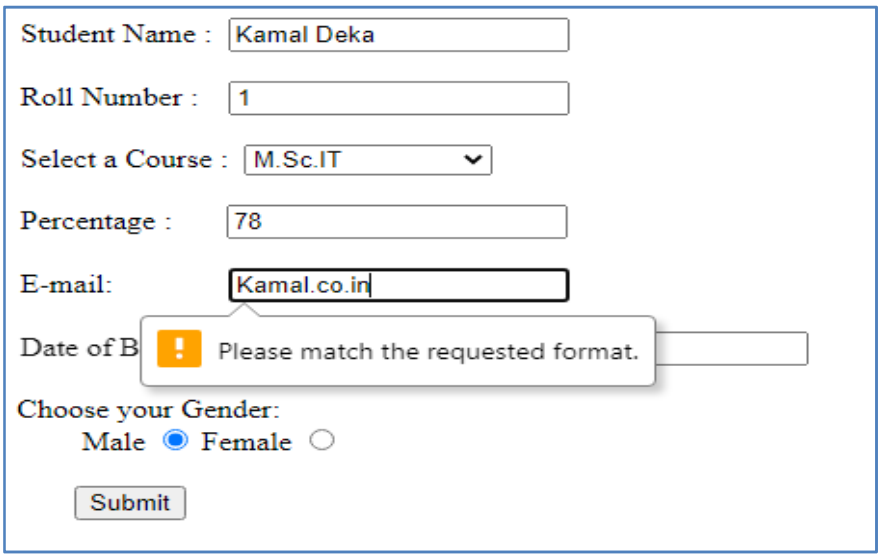
```
 <label for="percentage">Percentage :</label>

      // 'step', 'max' and 'required' attributes are used for
      validation of percentage

      <input type="number"
      id="per" name="spercentage" step=0.01 max=100
      required><br><br>
 <label for="email">E-mail:</label>

            // 'pattern' attribute is used for validation of E-mail
          &
      nbsp   <input type="text" id="email"
      name="semail" pattern="[a-z0-9._%+-]+@[a-z0-9.-]+\.[a-
      z]{2,}$"><br><br>
 <label for="date of birth">Date of Birth (DD-MM-
      YYYY):</label>

    <input type="text" id="dob"
      name="sdob" required><br><br>
<legend>Choose your Gender:</legend>
         <label
      for="male">Male</label>
<input type="radio" name="gender" id="male" value="male"
      checked>
<label for="female">Female</label>
<input type="radio" name="gender" id="female"
      value="female"><br><br>
        <input
      type="submit" onclick="validate()">

</form>

</body>
</html>
```

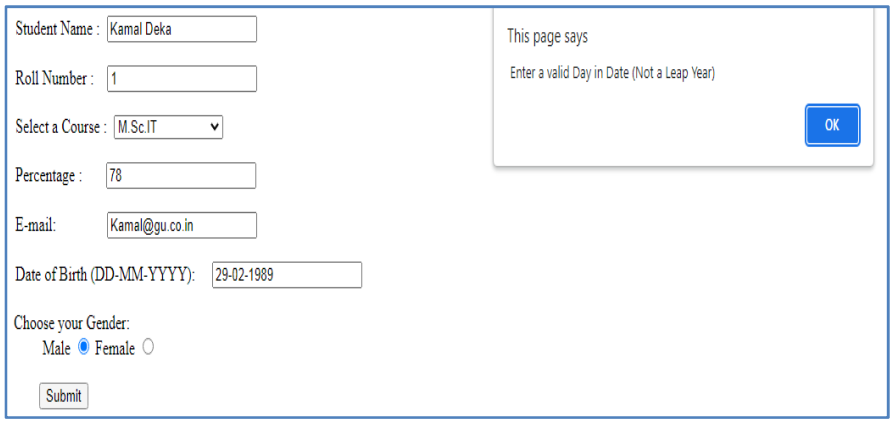
After executing the above HTML file in the browser, we will observe different outputs depending upon our input to the different HTML controls available in the designed form. Some of the outputs are presented as follows.


Student Name :
Roll Number :
Please fill out this field.
Select a Course : M.Sc.IT
Percentage :
E-mail:
Date of Birth (DD-MM-YYYY):
Choose your Gender:
Male
Female
Submit


Student Name : Kamal@ Deka
Roll Number : 1
Select a Course : M.Sc.IT
Percentage : 78
E-mail: Kamal@gu.co.in
Date of Birth (DD-MM-YYYY): 29-02-1989
Choose your Gender:
Male
Female
Submit
This page says
Enter a valid Name
OK


Student Name : Kamal Deka
Roll Number : 1
Select a Course : M.Sc.IT
Percentage : 123
E-mail:
Value must be less than or equal to 100.
Date of Birth (DD-MM-YYYY):
Choose your Gender:
Male Female
Submit


Student Name : Kamal Deka
Roll Number : 1
Select a Course : M.Sc.IT
Percentage : 78
E-mail: Kamal.co.in
Date of B
Please match the requested format.
Choose your Gender:
Male Female
Submit



7.8SUMMING UP

- A JavaScript function is a collection of JavaScript statements which perform a definite job or provide a specific output.
- Different built-in functions are already available in JavaScript. User-defined JavaScript functions are defined by the programmers to perform specific jobs depending upon their requirements.
- A JavaScript programmer can define a function using 'function' keyword. The syntax to define a User-Defined function is presented as follows.

function function_Name(Parameter1 , Parameter2 , Parameter3 ,……)

{

JavaScript Statements

}

- In JavaScript, if a function is defined outside any other function or block then it is referred as a global function. A global function can be called from any part of a JavaScript code.
- A function which calls itself is refereed as a recursive function. A recursive function calls itself repeatedly until the required solution of a computational problem is attained.
- When the state of an HTML object is changed due to any action accomplished by the user or the browser then it is refereed as an Event.

- In JavaScript, Event Handling is performed by writing JavaScript code to perform specific task on the occurrence of a specific event and that code can be called as Event handler.
- When different types of data are inserted in the different controls or input elements available in a HTML form then these data must be in proper format. Before form submission, it must be validated that all necessary form controls are filled with data in proper format. This process is called as client-side form validation.
- There are two ways to validate data in a HTML form control. We can use validation attributes on form controls to validate data. Secondly, we can also write our own JavaScript codes for different data validations on different HTML form controls and these can be used by using event handling mechanism.

CHECK YOUR PROGRESS

2. Choose the correct option

(a) Which of the following event is triggered when an HTML element becomes focused?
  (i) onload
  (ii) onfocus
  (iii) onblur
  (iv) None of the above

(b) 'onreset' event is triggered when _____.
  (i) user reset a HTML form.
  (ii) user reset a HTML button.
  (iii) user reset a page
  (iv) None of the above

(c) Which of the following event is triggered when an HTML element is changed?
  (i) onclick
  (ii) onreplace
  (iii) onchange
  (iv) None of the above

(d) Which of the following validation attribute is used to make sure that a HTML form control must be filled with data?
   (i) required
   (ii) fill
   (iii) set
   (iv) None of the above

(e) Which of the following is not a validation attribute in HTML?
   (i) required
   (ii) type
   (iii) pattern
   (iv) onsubmit

7.9ANSWERS TO CHECK YOUR PROGRESS

1. (a) Recursive
   (b) parseInt()
   (c) whether a number is finite
   (d) return
   (e) function

2. (a) (ii) onfocus
   (b) (i) user reset a HTML form.
   (c) (iii) onchange
   (d) (i) required
   (e) (iv) onsubmit

7.10POSSIBLE QUESTIONS

1. What is user-defined function? How user-defined function can be created in JavaScript? Give example.
2. What is global function? Give example of any three global JavaScript functions.
3. What is recursive function? Write down a recursive JavaScript function.

4. Explain the event-handling mechanism in JavaScript. Give example.
5. What is client-side form validation? Give any three examples of data validations.
6. Explain the ways to validate data available in HTML form controls. Give examples.

7.11REFERENCES AND SUGGESTED READINGS

- Harvey, Harvey Deitel, and Abbey Deitel. "Internet and World Wide Web How To Program." (2011).
- Gosselin, Don. *Javascript*. Course Technology Press, 2004.
- Pollock, John. *JavaScript, A Beginner's Guide*. McGraw Hill Professional, 2009.
- Thau, Dave, and Dave Thau. *The Book of JavaScript: A Practical guide to interactive Web pages*. No Starch Press, 2000.

---×---

UNIT- 8
MARKUP LANGUAGE BASICS

UNIT STRUCTURE

8.1 INTRODUCTION

SGML(Standard Generalized Markup Language) is a language defining markup languages. Here in this unit, we will discuss about different components of SGML. The initial component of SGML is the SGML declaration. SGML declaration specifies how the characters and delimiters may appear in the application. In this unit, we will discuss the structure, application and restriction of DTD(Document Type Definition) in different markup language. Here in this unit we also discuss about the declaration of entities and its attributes in DTD.

8.2 OBJECTIVES

After going through this unit learner will able to:

- Understand the concept of SGML.
- Learn the application of SGML.
- Understand the concept of DTD.
- Learn the concept of DTD elements.

- Understand the concept of content models of DTD.
- Learn the different attribute and entities of DTD.

8.3 WHAT IS SGML?

SGML basically derived from GML (Generalized Markup Language). The GML allows user to work on standard formatting styles for any electronic document. It was developed by ISO (International Organization for Standards) in 1986. It specifies the rules for formatting elements and tags.

8.4 APPLICATION OF SGML

The most widely used markup languages are SGML, HTML and XML. Markup language is basically used to tell the browser how the web page is set. Markup language is the powerful tools for creating and formatting a web page. It specifies the different types of character and other structuring element may appear and changing the structure of a webpage. In SGML the elements can be declared with the help of start and end tag. The major application of SGML is HTML.SGML uses the principle of logical document markup, and applying this principle in the form of definition of a generalized markup language. In a single sentence we can say that SGML is a Meta language.

Now let us discuss some HTML syntaxes.

(i) HTML elements:

HTML element defines the structure of a document. An HTML element generally consists of three parts namely a start tag, content, and an end tag. Every element has a start and stop tags. The start tag is written as <element-name> where element-name is the name of the element. The end tag of an element is written with a slash before the element name as </element-name>.

For example: <pre> The content is a pre formatted text</pre>

The SGML definition of HTML specifies that some HTML elements are not required the end tag. In the

reference manual, it indicates that the tag requires an end tag or not. Some element has no content in an HTML document. For example the element BR is use only for the line break. Such element in HTML is called the empty element. The empty element never has an end tags. The definition of each element in the reference manual specifies that the tag is empty or non-empty.

(ii) Attributes:

The properties of an element are called attributes. An attribute of an element can take a value. The value of an attribute appear before the ">" of an element start tag. In an element any number of attribute can appear in the start tag. The attributes may appear in any order. For example align attribute is set as "center" in the H1 element as follows

<H1 align= "center">
This is a centered aligned heading
</H1>

SGML requires user to delimit all attribute value using either single quotation mark (') or double quotation mark (") . In some situation, it is possible in HTML to specify the value of an attribute without any quotation marks. But it is recommended to use the quotation mark even when it is possible to eliminate them. Attribute names are always case insensitive.

(iii) HTML comments:

HTML comments have the following syntax:

<!----- This is an example of comment---->
<!---The comment line also
occupies more than one line--->

Different types of Markup:

In text processing, the markup gives a very good job of defining the specific physical representation of a document. For a certain kind of document it is more convenient to obtain a given layout for specific controlling. For preparing a document, it is

mandatory to describe the logical structure of the document by deleting every reference to physical representation. This is generally referred as the generic markup. In generic markup, the logical function of all elements of a document title, sections, paragraphs, tables, bibliographic references and the structural relations between them must be clearly mentioned or defined. The following shows different mark-up of some text.

Script

.pa

.bd chapter 10: Title of the chapter

.sp

Generic logical markup

LATEX

\chapter {Title of chapter}

\par

HTML (SGML)

<h1> Title of the chapter</h1>

<p>

Several document instances can belong to the same document class that means they are described by the same DTD. Let us consider two source text of an article as follows.

Article A	Article B
Title	Title
Section 1	**Section 1**
Subsection 1.1	
Subsection 1.1	

Subsection 1.2
Subsection 1.2

Section 2
Subsection 1.3

Section 3
Subsection 1.4

Subsection 3.1 **Section 2**

Subsection 3.2
Subsection 2.1

Subsection 3.3
Subsection 2.2

Bibliography **Bibliography**

The above specific structure looks different, but logical structure is built according to the same pattern, a title followed by one or more sections, each one is subdivided into zero or more subsections and put a bibliography at the end. So, we can say that document instances belong to the document class "article". To describe the formal structure of all documents of type "article", one has to construct the DTD. A DTD is expressed in a language defined by the SGML standard and identified all the elements belonging to the document class (sections, sub-sections etc.). For defining the DTD for the "Article A" and "Article B", one can use the short names for defining DTD. For example "sec" can be written instead of section and "ssec" can be written instead of subsection. The DTD can be defined as follows.

<Article>

<Title> An introduction to SGML</Title>

<sec> SGML: What is SGML? </sec>

<p>

<ssec> What is DTD?</ssec>

<p>

…….

……

<Bibliography>…….</Bibliography>

</Article>

Stop To Consider

The relations between elements do not always have to be hierarchical, DTD use element attributes to express this kind of hierarchy.

CHECK YOUR PROGRESS I

1. **Multiple Choice Questions**

(i) The initial component of the SGML is the _____.

(a) SGML coding
(b) SGML declaration
(c) SGML structure
(d) SGML tags

(ii) SGML basically derived from _____.

(a) Generalized Markup Language
(b) Hyper Text Markup Language
(c) Object oriented language
(d) Web based language

(iii) SGML was developed by _____.

(a) www
(b) Microsoft
(c) DreamTech
(d) ISO

(iv) The major application of SGML is _____.

(a) DTD
(b) GML
(c) HTML
(d) HTTP

(v) The empty element in HTML never has a _____.

(a) start tag
(b) scroll tag
(c) blank tag
(d) end tag

2. State True or False

(i) SGML specifies the rules for formatting elements and tags.

(ii) SGML is a Meta language.

(iii) An attribute of an element cannot take values

(iv) Attribute names are always case sensitive

(v) Markup language is the powerful tools for creating and formatting a web page.

8.5 DESCRIBING DTD

DTD provides a framework to validate XML document. DTD defines the structure of an XML document which contains some legal elements. User can include DTD as inline or as external references.

In DTD, for declaring elements in XML, we can use the following code.

```
<!ELEMENT element-name category>
or
<!ELEMENT element-name (element-content)>
```

For declaring empty element, we have to put the word "empty" in the category section as shown below.

```
<!ELEMENT  element-name empty>
```

For example :

```
<!ELEMENT br empty>
```

Elements with only the parsed character, data is declared with #PCDATA inside parentheses.

<!ELEMENT element-name (#PCDATA)>

For example :

<!ELEMENT to (#PCDATA)

Elements with any combinational parsable data can be declared by writing‘ANY’in the category part in the element declaration as shown below.

<!ELEMENT element-name ANY>

For example :

<!ELEMENT email ANY>

The element with child elements are declared with the name of the child elements inside the parentheses.

<!ELEMENT element-name (child_element1)>
or
<!ELEMENT element-name
(child_element1,child_element2,….. ....)>

For example:

<!ELEMENT email (to, from, body)>

The following code shows how to declare inline DTD in an XML file.

```
<?xml version=”1.0” ?>

<! DOCTYPE  email

[

<!ELEMENT email (to, from, body)>

<!ELEMENT to  (#PCDATA)>

<!ELEMENT  from  (#PCDATA)>

<!ELEMENT  body  (#PCDATA)>

]>
```

<email>

<to> student </to>

<from>GUCDOE </from>

<body> Welcome to our centre </body>

</email>

The following structure shows how to provide an external reference of DTD in an XML document.

<? Xml version = “1.0”?>

<!DOCTYPE email SYSTEM “info.dtd”>

<email>

<to> student </to>

<from> GUCDOE </from>

<body> Welcome to our centre </body>

</email>

Here an external reference of DTD file info.dtd is provided by using the <!DOCTYPE> tag. The “info.dtd” file contains the following codes.

<?xml version=”1.0” ?>

<!ELEMENT email (to, from, body)>

<!ELEMENT to (#PCDATA)>

<!ELEMENT from (#PCDATA)>

<!ELEMENT body (#PCDATA)>

8.5.1 Defining DTD for a Single Element

User can define DTD for a single element. The DTD for a single element does not contain any text or other element. The following code shows for how to define DTD for a single element.

<?xml version= “1.0” encoding = “UTF-8”?>

<!ELEMENT PRODUCTDATA (PRODUCT+)>

Here the (PRODUCT+) expression indicates that PRODUCTDATA element contains one or more elements of type <PRODUCT>. If user do not use the plus (+) sign, the DTD definition conveys to the parser that the PRODUCTDATA element contains a single PRODUCT element. Some DTD qualifiers that can be used for DTD definition are as follows.

- ? (Question mark)→ it is applies to zero or one element.
- (Asterisk) → it is applies to zero or more elements.
- + (Plus sign) → it is applies to one or more elements.

8.5.2 Defining DTD for Nested Element

We can define DTD for nested element also. For defining DTD for the nested element, the definition of the DTD informs the parser which elements can contain nested element. The following code shows how to define DTD for nested elements.

<!ELEMENT EMPLOYEEDATA (EMPLOYEE+)>

<!ELEMENT EMPLOYEE (EMPLOYEENAME, DESIGNATION)>

<!ELEMENT EMPLOYEENAME (#PCDATA)>

<!ELEMENT DESIGNATION (#PCDATA)>

In the above code, the DTD inform the parser that the <EMPLOYEE> element must contain the elements <EMPLOYEENAME> and <DESIGNATION>. The <EMPLOYEENAME> and <DESIGNATION>elements are defined to contain the data of type Parsed Character Data (PCDATA). The # symbol indicates that the keyword used after this symbol is a special word rather than an element name.

DTD can be defined using the OR condition. The following code define DTD using the OR condition.

<!ELEMENT item (#PCDATA | item)* >

Here the item element can contain either PCDATA or an item. The Symbol "*" (asterisk) at the end of the code specified that the element can occur zero or more number of times. Therefore item element can contain mixed types contents.

8.5. 3 Attributes in DTD

An XML element can also have some attributes. Defining attributes in a DTD specifies the characteristics of the element within the specified XML document. The syntax for defining attributes in a DTD is as follows.

<!ATTLIST element-name attribute-name attribute-type default-value>

The components of the syntax are as follows.

- **element-name** specifies the element for which the attributes are defined.
- **attribute-name** refers to the attribute of an element.
- **attribute-type** specifies the types of data that can be stored by the attribute. The different types of attributes are as follows.
  - CDATA- It specifies unparsed character data (a text string).
  - ID - It specifies a unique name for an attribute that no other ID attributes share.
  - IDREF- It specifies a reference to an ID defined in the DTD document.
  - IDREFS - It specifies a space separated list containing one or more ID references. That is a list of references to multiple elements.
  - NMTOKEN - It specifies a valid XML name composed of letters, numbers, underscore, hyphens and colons.
  - NMTOKENS - It specifies a space separated list of names.
  - NOTATION - It specifies the name of a DTD-specified notation.

- **default value** specifies the default value of an attribute. Default value can be categorized as follows.
  - value –Specifies a default value for an attribute.

- #REQUIRED – Specifies that the value of an attributes that must be provided.
- #IMPLIED – Specifies that the value of an attribute is optional.
- #FIXED value – Specifies that the value of an attribute is fixed.

The syntax for #REQUIRED is as follows.

<!ATTLIST element-name attribute-name attribute-type #REQUIRED>

For example :

<!ATTLIST employee phoneno CDATA #REQUIRED> for this DTD declaration the valid xml file will be as follows

<employee phoneno="1234567892" />

The #REQUIRED keyword is used when user don't have an option for a default value, but still want to force the attribute to be present.

The syntax for #IMPLIED is as follows.

<!ATTLIST element-name attribute-name attribute-type #IMPLIED>

For example:

<!ATTLIST employee phoneno CDATA #IMPLIED> for this DTD declaration the valid xml file will be as follows.

<employee phoneno="1234667788" />or <employee/>

The #IMPLIED keyword is used when you don't want to force the author to include an attribute, and you don't have an option for a default value.

The syntax for #FIXED is as follows.

<!ATTLIST element-name attribute-name attribute-type #FIXED "value">

For example:

<!ATTLIST product company CDATA #FIXED "TATA"> for this DTD declaration the valid xml file will be as follows.

<product company="TATA" />

The #FIXED keyword is used if the attribute value is fixed. If the author places another value, it will give an error message.

Example 8.1

```
<! DOCTYPE Book [

<!ELEMENT Book (Name, Author, Category, Price)>

<!ELEMENT Name (#PCDATA)>

<!ELEMENT Author  (#PCDATA)>

<!ELEMENT Category  Empty>

<!ELEMENT Price (#PCDATA)>

<!ATTLIST Category Ctype CDATA #REQUIRED>

]>
```

Example 8.1 shows the external DTD file. The Book element contains the Name, Author, Category and Price element. The Category element is empty and it defines an attribute as Ctype, which is mandatory with the Category element. Example 8.2 shows the XML file with reference to the external DTD.

Example 8.2: Referencing an External DTD

```
<? xml version= "1.0" ?>
<!DOCTYPE Book  "Mybook.dtd">
<Book>
        <Name>Programming with Java </Name>
        <Author> E. Balagurusamy </Author>
        <Category Ctype= " Programming" />
        <Price> Rs 588 </Price>
</Book>
```

8.6 DEFINING ENTITIES IN THE DTD

In an XML document, entity can be defined internally or it can be defined in an external document. User can define entity using the entity tag. The following syntax shows how to define entities in an external DTD.

<!ENTITY entity-name SYSTEM “URI/URL”>

Here in the above syntax the entity-name parameter specifies the name of the entity and the URL parameter specifies the URL of the DTD or XML document that contains the definition of the entity.

Entity can be defined internally by the following code.

<!ENTITY entity-name “entity definition”>

Here the string “entity definition” replaces the name of the entity whenever it is referenced in the XML document.

Now let us write the following codes defining two entities author and copyright are as follows.

<!ENTITY author “E. Balagurusamy”>

<!ENTITY copyright “copyright 2008 by PHI”>

Now by adding these two codes and using it in an XML document we will get the following code.

<publisher>&author;&copyright;</publisher>

In the above code, we have defined copyright information of a particular publisher created by an organization. But in the later time if another organization acquires the copyright of the book. The copyright information need to be changed only at one place. In our case the copyright entity need to be changed in the first code snippet. The updated copyright information is reflected in the entire XML document. For referencing external entities we need to use the SYSTEM or PUBLIC identifier to reference external entities defined in DTD or an XML. The following code shows how to refer an external entity.

<!DOCTYPE book SYSTEM “books.dtd” [

<!ENTITY author “E. Balagurusamy”>

<!ENTITY copyright SYSTEM “ bookinfo.xml”>

] >

Example 8.3 shows the uses of entity and attributes in a dtd file.

Example 8.3

```
<!DOCTYPE JOURNAL [

<!ELEMENT JOURNAL (ARTICLE+)>
<!ELEMENT ARTICLE
(INTRODUCTION,BODY,METHOD,CONCLUSION)>
<!ELEMENT INTRODUCTION (#PCDATA)>
<!ELEMENT BODY (#PCDATA)>
<!ELEMENT METHOD (#PCDATA)>
<!ELEMENT CONCLUSION (#PCDATA)>


<!ATTLIST ARTICLE AUTHOR CDATA #REQUIRED>
<!ATTLIST ARTICLE EDITOR CDATA #IMPLIED>
<!ATTLIST ARTICLE DATE CDATA #IMPLIED>
<!ATTLIST ARTICLE EDITION CDATA #IMPLIED>

<!ENTITY JOURNAL "COMPUTER INFORMATIVE">
<!ENTITY PUBLISHER "ELECTRICAL SOCIETY">
<!ENTITY COPYRIGHT "Copyright 2024informativepress">

]>
```

CHECK YOUR PROGRESS – II

3. Multiple Choice Questions

(i) DTD provides a framework to validate _____.

(a) XML document
(b) HTML codes
(c) external tags
(d) internal tags

(ii) DTD qualifier "?" is used for_____.

(a) zero or one element.
(b) One or more element
(c) Zero or more element
(d) More than one element

(iii) The child elements are declared with the name in a _____.

(a) Start and stop tag
(b) Parentheses
(c) Plus sign
(d) Tab order

(iv) CDATA specifies _____.

(a) Parsed Character data
(b) Unparsed Character data
(c) Only Character data
(d) Both parsed and unparsed character data.

(v) #REQUIRED specifies _____.

(a) Specifies a default value for an attribute.
(b) Specifies that the value of an attribute is optional.
(c) Specifies that the value of an attribute is fixed.
(d) Specifies that the value of an attributes that must be provided.

4.**State True or False**

(i) User cannot define DTD for a single element.
(ii) User can define DTD for nested element.
(iii) In an XML document entity can be defined internally only.
(iv) The IDREF attribute specifies a reference to an ID defined in the DTD document.
(v) Referencing external entities we need to use the SYSTEM or PUBLIC identifier.

8.7 SUMMING UP

- SGML basically derived from the Generalized Markup Language.
- SGML declaration specifies how the characters and delimiters may appear in the application.
- The most widely used markup languages are SGML, HTML and XML.
- Markup language is basically used to tell the browser how the web page is set and it is a powerful tool for creating and formatting a web page.
- SGML uses the principle of logical document markup, and applying this principle in the form of definition of a generalized markup language.
- The properties of an element are called attributes and an attribute of an element can take a value.
- SGML requires user to delimit all attribute value using either single quotation mark (‘) or double quotation mark (“) .
- DTD defines the structure of an XML document which contains some legal elements.
- User can include DTD as inline or as external references.
- DTD can be defined for a single element as well as for nested elements.
- Defining attributes in a DTD, specifies the characteristics of the element within the specified XML document.
- In an XML document entity can be defined internally or in an external document.

8.8 ANSWER TO CHECK YOUR PROGRESS

1. (i) (b) (ii) (a) (iii) (d) (iv) (c)
   (v) (d)

2. (i) True (ii)True (iii) False (iv) False
   (v)True

3. (i) (a) (ii) (a) (iii) (b) (iv) (b)

(v) (d)

4. (i) False (ii) True (iii) False (iv) True

(v) True

8.9 POSSIBLE QUESTIONS

1. What is SGML (Standard Generalized Markup Language)?

2. What are the applications of SGML?

3. Explain the different types of markup?

4. What is DTD (Document Type of Definition)?

5. How can you declare inline DTD in an XML file? Explain it with a suitable example.

6. How can you define DTD for a single element? Explain.

7. Explain the uses of DTD qualifiers.

8. How can you define DTD for nested element?

9. Explain the different types of attributes of an XML file.

10. Explain the way of defining entities in DTD.

8.10 REFERENCES AND SUGGESTED READINGS

- https://www.w3schools.com/
- HTML 5 BLACK BOOK Published by dreamtech press.
- Goldfarb, C. F. (1990). *The SGML handbook* (Vol. 10). Oxford University Press.

---×---

UNIT-9
EXTENSIVE MARKUP LANGUAGE (XML)

UNIT STRUCTURE:

9.1INTRODUCTION

XML is a markup language based on easy and platform independent rules. XML uses the user-defined tag to format and display textual information. XML is a standard format to exchange the different textual messages among the different application. In this unit we will discuss about the structure of an XML document alongwiththe different attributes and elements of XML. Finaly we will discuss about DTD (Document Type Definition) and the uses of XSL (Extensive Style sheet language).

9.2 OBJECTIVES

After going through this unit learner will be able to

- Understand the structure of an XML document.
- Displaying the content of an XML document.
- Understand the creation of an XML document
- Learn about the XML elements
- Describe the DTD
- Learn the attributes of DTD.
- Understand the concept of XSL

9.3 STRUCTURE OF AN XML DOCUMENT

There are certain rules for defining the structure of an XML document. The syntax for creating an XML document is called the markup syntax. While creating an XML document one must follow the following points

- XML document must have starting and end tags.
- The tags used for creating an XML document are case sensitive.
- XML document must have one root element.
- The elements in XML document must be properly nested.
- The values of an XML attribute must be enclosed in double quotes.
- XML is designed to carry data not to display data
- XML is designed to be self-descriptive

The structure of an XML document is given below

<?xml version= “1.0” encoding= “ISO-8859-1”?>

<!---This is a student element---- >

<Student>

<Roll_No>0424001</Roll_No>

<FirstName> Dipak </FirstName>

<MiddleName>Kumar</MiddleName>

<LastName>Kalita</LastName>

</Student>

In the above structure the XML document has the following sections

(i) XML declaration
(ii) XML elements
(iii) XML attributes
(iv) XML tree
(v) XML comments

(i) XML declaration

The XML declaration is used to identify that the specified document is an XML document. XML declaration is the first line in the document and it defines the version and the character encoding. XML declaration start with <?xml and end with ?>. Here in the above structure version attribute provide the version of the XML document. Encoding specifies the character encoding which is used by the document. This is the optional attribute in the XML declaration.

STOP TO CONSIDER

XML declaration must be at the beginning of the line and version, encoding attributes must be in the order.

(ii) XML elements

An XML element is the basic building block of an XML document. An element in an XML document must start with start tag (e.g. <Student>) and end with end tag (e.g.</Student>). An XML document must have a single root element. The root element is always in the top of the document and it contains the child elements. A root element does not have a parent element but it can have one or more child element.

The following rules must be considered while defining a XML element

- The element name cannot start with numbers or other punctuation mark but it can start with letters or underscore (_).
- Element names cannot contain spaces

- Element names cannot start with the word “xml”
- Element name cannot contain the “:” character.

Element names are divided into two categories namely empty and nested elements. The empty element does not contain any content or any other element within it. The element under another element is called the nested element. In an XML document the nested element must be properly nested. The parent element must be properly opened before the child element and must be closed after the child element.

(iii) XML attributes

The XML attributes provide the additional information for that particular element. The value of an attributes must be encoded with either a single quotation or double quotation. The example of XML attribute is as follows

```
<book category="WEB">
   <title lang="en">Learning XML</title>
   <author>Erik T. Ray</author>
</book>
```

Here the category and lang are the example of attributes where “WEB” and “en” are the values of these attributes.

(iv) XML tree

XML tree defines the elements in the tree structure. The XML tree contain the following elements

- Root element- it contains the other elements which means the child elements. An XML document can have only one root element.
- Parent element- A parent element can have multiple child elements.
- Child element- Child element is always contained in the parent element. A child element cannot have multiple parent elements. It is only under a single parent element.
- Sibling- It refers to the child element which is under a single parent.

(v) XML comments

XML comments are used for documenting or to give remarks. XML comments are written in the following format <! -------This information about a particular student --------->. Comment can be written anywhere in the document. But it cannot be written inside a tag or another comment.

9.4 CREATINGAN XML DOCUMENT

For Creating an XML document you have to declare the version and encoding part sequentially in the first line. In the second line write down the root element and then parent and child elements according to the requirement. Example 1 shows the creation of an XML document.

Example 1

```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
<book category="BIOGRAPHY">
<title lang="en">Wings of Fire</title>
<author>Dr. APJ Abdul Kalam</author>
<year>1999</year>
<price>198.00</price>
</book>
<book category="KIDS">
<title lang="en">Perfect Parenting</title>
<author>Sushant Kalra</author>
<year>2021</year>
<price>235.00</price>
</book>
<book category="WEB">
<title lang="en">Learning XML</title>
<author>Erik T. Ray</author>
<year>2001</year>
<price>2865.00</price>
</book>
<book category="NOVEL">
<title lang="en">All the Light We Cannot See</title>
<author>Anthony Doerr</author>
```

```
<year>2014</year>
<price>375.00</price>
</book>
</bookstore>
```

Output of Example 1

```
<?xml version="1.0" encoding="UTF-8" ?>
- <bookstore>
  - <book category="BIOGRAPHY">
      <title lang="en">Wings of Fire</title>
      <author>Dr. APJ Abdul Kalam</author>
      <year>1999</year>
      <price>198.00</price>
    </book>
  - <book category="KIDS">
      <title lang="en">Perfect Parenting</title>
      <author>Sushant Kalra</author>
      <year>2021</year>
      <price>235.00</price>
    </book>
  - <book category="WEB">
      <title lang="en">Learning XML</title>
      <author>Erik T. Ray</author>
      <year>2001</year>
      <price>2865.00</price>
    </book>
  - <book category="NOVEL">
      <title lang="en">All the Light We Cannot See</title>
      <author>Anthony Doerr</author>
      <year>2014</year>
      <price>375.00</price>
    </book>
</bookstore>
```

STOP TO CONSIDER

The output of a valid XML document displayed in Internet Explorer 8 web browser because all web browser do not support the XML format.

The XML tree structure for the Example1 is shown in the Fig 1.

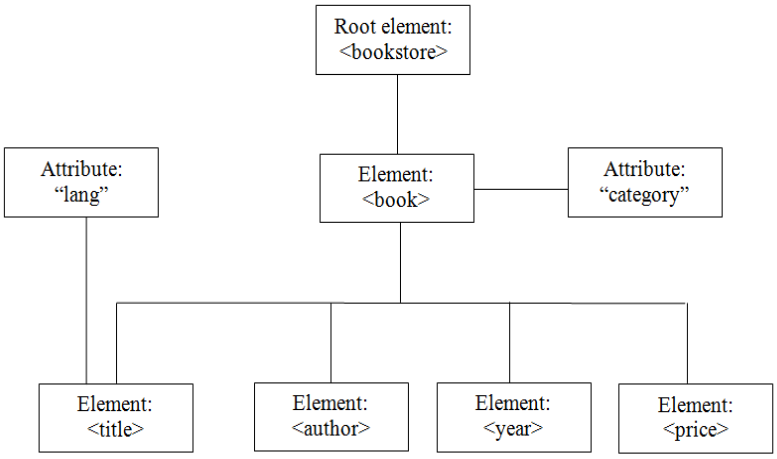


Fig.1 XML tree structure

Check Your Progress I

1. State True or False

(i) XML uses the user-defined tag.
(ii) The tags uses for creating an XML document are not case sensitive.
(iii) XML is designed to carry data.
(iv) XML document can have multiple root elements.
(v) Element names of XML cannot contain spaces.

2. Fill in the blanks

(i) The ____ attribute must specify in the declaration of an XML document.
(ii) The element under another element is called the _____ element.
(iii) The XML ____ provide the additional information for that particular element.
(iv) An XML document can have only one ___ element.
(v) _____ element always contain in the parent element.

9.5 WHAT IS DTD?

DTD provide the common rules for defining the structure of an XML document. It provides a framework to validate XML document. You can include DTD in an XML document either by inline declaration or as an external reference. The external reference of DTD is typically with a file extension .dtd. The example 2 shows the declaration of internal DTD in an XML document.

Example 2

```
<?xml version= "1.0" ?>
    <!DOCTYPE book [
        <!ELEMENT book (title,author) >
        <!ELEMENT title (#PCDATA) >
        <!ELEMENT author (#PCDATA) >
        ]>
    <book>
        <title>Computer Networks</title>
        <author>Andrew S. Tanenbaum</author>
    </book>
```

In inline DTD, the DTD code is written using the <?DOCTYPE> tag to define the structure of book, title, author elements. The Example 3 shows an external reference of DTD in an XML document.

Example 3

```
<?xml version= "1.0" ?>
<!DOCTYPE book SYSTEM "book.dtd">
<book>
    <title>Computer Networks</title>
    <author>Andrew S. Tanenbaum</author>
</book>
```

Example 3 have a DTD contained in a file, book.dtd, with the following structure:

<!ELEMENT book (title, author) >

<!ELEMENT title (#PCDATA) >

<!ELEMENT author (#PCDATA) >

The book.dtd file is contained in the same directory as the XML document. The character content of elements is termed PCDATA, meaning parsed character data. The character content of attributes is termed as CDATA, meaning character data. In an XML document we can use some DTD qualifiers which have different meaning in a document. These DTD qualifiers are as follows

Question mark (?) → ? applies to zero or one element.

Asterisk (*) →* applies to zero or more element.

Plus (+) → + applies to one or more element.

The example 4 shows how to create an internal DTD using with some DTD qualifiers.

Example 4

```
<?xml version="1.0"?>

<!DOCTYPE document [

<!ELEMENT document (employee)*>

<!ELEMENT employee (name, hiredate, projects)>

<!ELEMENT name (lastname, firstname)>

<!ELEMENT lastname (#PCDATA)>

<!ELEMENT firstname (#PCDATA)>

<!ELEMENT hiredate (#PCDATA)>

<!ELEMENT projects (project)*>

<!ELEMENT project (product,id,price)>

<!ELEMENT product (#PCDATA)>

<!ELEMENT id (#PCDATA)>

<!ELEMENT price (#PCDATA)>

] >
```

```
<document>
<employee>
<name>
<lastname>Das</lastname>
<firstname>Prabin</firstname>
</name>
<hiredate>October 15, 2005</hiredate>
<projects>
<project>
        <product>Printer</product>
        <id>111</id>
        <price>$111.00</price>
</project>
<project>
        <product>Laptop</product>
        <id>222</id>
        <price>$989.00</price>
</project>
</projects>
</employee>
<employee>
        <name>
        <lastname>Tanti</lastname>
        <firstname>Kiran</firstname>
        </name>
        <hiredate>October 20, 2005</hiredate>
        <projects>
```

<project>

<product>Desktop</product>

<id>333</id>

<price>$2995.00</price>

</project>

<project>

<product>Scanner</product>

<id>444</id>

<price>$200.00</price>

</project>

</projects>

</employee>

<employee>

<name>

<lastname>Chakraborty</lastname>

<firstname>Barun</firstname>

</name>

<hiredate>October 25, 2005</hiredate>

<projects>

<project>

<product>Keyboard</product>

<id>555</id>

<price>$129.00</price>

</project>

<project>

<product>Mouse</product>

<id>666</id>

<price>$25.00</price>

</project>

</projects>

</employee>

</document>

9.6 XML NAMESPACE

XML Namespaces provide a method to avoid element name conflicts. Since element names in XML are not predefined, a name conflict will occur when two different documents use the same element names. Suppose an XML document carries information of a table as follows

<table>

<tr>

<td> Mango</td>

<td> Apples</td>

</tr>

</table>

Again another XML document carries information about a table (a piece of furniture) as follows:

<table>

<name> coffee table </name>

<width>40</width>

<length>70</length>

</table>

If the above two XML documents were merged together, there would be an element name conflict because both documents contain

a <table> element with different content and definition. For solving the name conflicts we have to use a prefix to get the following two tables.

<h:table>

<h:tr>

<h:td> Mango</h:td>

<h:td> Apples</h:td>

</h:tr>

</h:table>

<f:table>

<f:name> coffee table </f:name>

<f:width>40</f:width>

<f:length>70</f:length>

</f:table>

Here the name conflict is totally avoided because two tables have different name prefixes.

(<h:table> and <f:table>)

9.7 XML ENTITY REFERENCES

XML provides the entity references to insert the symbols that have a specific purpose in xml document. The & lt; and & gt; are the some examples of entity references. The & lt; is used for less than symbol(<) and the & gt; is used for greater than(>) symbols. The Example 5 shows the uses of entity references.

Example 5

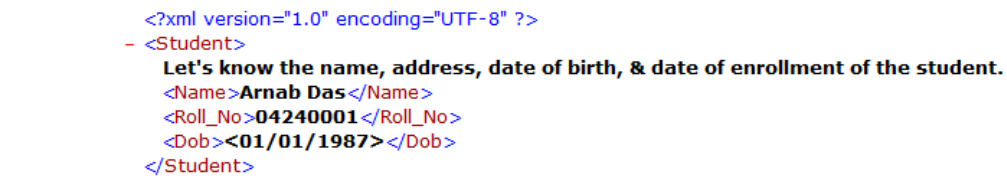
<?xml version="1.0" encoding="UTF-8"?>

<Student>

Let & apos; s know the name, address, date of birth, &
date of enrollment of the student.

```
        <Name> Arnab Das</Name>

        <Roll_No>04240001</Roll_No>

        <Dob>&lt;01/01/1987&gt;</Dob>

</Student>
```

Output of Example 5



Here & apos; is used for adding apostrophe s, & is used to add & character in between the sentence. The entity references & lt; and & gt; are used to display the date of birth of a student.

9.8 XML SCHEMAS

XML schemas define the structure of an XML document and it is alternative to DTD. XML schema validates the structure of an XML document with the help of XML parser. The main difference between DTD and the XML schema is that XML schema uses XSD file which is an XML based language, to describe the structure of an XML. The difference between the DTD file and XML schema are as follows

- DTD defines the elements, attributes and entities whereas XML schema defines the structure of an element with more features.
- DTD file does not support namespaces but XML schema provides support for namespaces.
- DTD does not follow the XML syntax but XML schema follows the XML syntax.

The XML documents that refer to a single XML schema are known as instances of the schema. The syntax for defining the XML schema is as follows

<xs:element name= “name” type= “type” />

The syntax includes two attributes, name and type. Here the name specifies the name of the element that appears in the XML document and the type attribute specifies the data type of that element. XML schema document contain two data type namely pre-defined and user-defined. The pre-defined data types available in XML schema are as follows

- xs:string
- xs:date
- xs:numeric
- xs:misc

The user-defined data-type can be categorized as simple and complex types. The simple type contains only the text and the complex type contain child element and its attributes. The syntax of declaring a simple type element is as follows

<xs:element name= “x” type= “y”/>

Here “x” is the name of the element and “y” is the data type of the element.

Consider the following XML document

<student> Kiran</student>

<Rollno>012401</Rollno>

The definition of the simple type element for the XML element is as follows

<xs:element name= “student” type= “xs:string”/>

<xs:element name=” “Rollno” type= “xs:integer”/>

The complex type elements contain child elements, attributes and text. They also contain mixed type content. For understanding the complex type element, consider the following XML document

<student>

<firstname>Kiran</firstname>

<lastname>Das</lastname>

</student>

The definition of the complex type element for the XML element is as follows

<xs: element name= “student”>

<xs:complextype>

<xs:sequence>

<xs: element name= “firstname” type= “xs:string”/>

<xs:element name= “lastname” type= “xs:string”/>

</xs:sequence>

</xs:complextype>

</xs:element>

9.9 INTRODUCTION TO XSLT

Extensible Stylesheet Language Transformation (XSLT) processor is used to transfer xml document to different format. XSLT document provides the instruction for transforming XML document into different format according to the requirement. XSLT uses some APIs to convert the xml document to other format. XSLT data model consists of various elements such as

(i) <xsl: stylesheet>

XSLT stylesheet starts with the stylesheet element. The syntax of XSLT stylesheet is as follows

<xsl:stylesheetxmlns:xsl= http://www.w3.org/1999/xsl/Transform
version= “1.0”>

Here xmlns is the attribute of stylesheet element which provides a Uniform Resource Identifier (URI) of XSLT. You have to set URI as http://www.w3.org/1999/xsl/Transform to check elements and attributes that are specified according to specification defined by W3C.

(ii) <xsl: value-of>
The value-of element specifies the value of an element or attribute. The syntax of value-of element is as follows
<xsl:value-of select= “element_name/@attribute_name”>

In the above code the value-of element indicates the name of an element which will display its value on a web browser.

The attribute name is always prefix by the @ symbol

For implementation of value-of element in XSL, we take one xml files(save as book.xml) which is shown in Example 6.

Example 6

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

<?xml-stylesheet type="text/xsl" href="hello.xsl"?>

<bookstore>

<book>

<title>Wings of Fire</title>

<author>Dr. APJ Abdul Kalam</author>

<year>1999</year>

<price>198.00</price>

</book>

<book category="KIDS">

<title lang="en">Perfect Parenting</title>

<author>Sushant Kalra</author>

<year>2021</year>

<price>235.00</price>

</book>

<book>

<title>Learning XML</title>

<author>Erik T. Ray</author>

<year>2001</year>

<price>2865.00</price>

</book>
```

```
<book>
<title>All the Light We Cannot See</title>
<author>Anthony Doerr</author>
<year>2014</year>
<price>375.00</price>
</book>
</bookstore>
```

In Example 6 the reference file is stated as “hello.xsl”. So we have to create the “hello.xsl” file. The hello.xsl file is shown in Example 7.

Example 7

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
<body>
<h2>My Book collection</h2>
<table border="1">
<tr bgcolor="#9acd32">
        <th>Title</th>
        <th>Author</th>
</tr>
<xsl:for-each select="bookstore/book">
        <tr>
<td><xsl:value-of select="title"/></td>
<td><xsl:value-of select="author"/></td>
        </tr>
```

</xsl:for-each>

</table>

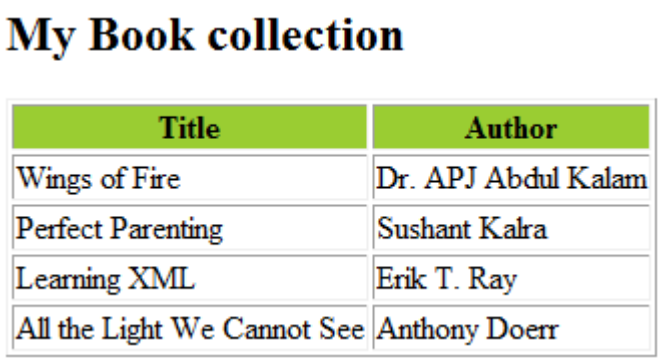
</body>

</html>

</xsl:template>

</xsl:stylesheet>

The output of Example 7 is as follows



My Book collection

Title	Author
Wings of Fire	Dr. APJ Abdul Kalam
Perfect Parenting	Sushant Kalra
Learning XML	Erik T. Ray
All the Light We Cannot See	Anthony Doerr

(iii) <xsl: for-each>

The "for-each" element of XSLT processes data in each time for specified pattern occurs in the XML document. The for-each element displays a pattern repeatedly which is similar to the looping construct in C or C++ languages.

The syntax of for-each element is as follows

<xsl:for-each select= "pattern">

Data to be processed here

</xsl:for-each>

(iv) <xsl: sort>

The "sort" element of XSLT arranges data in ascending or descending order. There are four attributes of sort element namely select, order, case-order and data-type. The syntax of sort element is as follows.

<xsl:sort select= "element_name/expression"

order= "ascending/descending"

case-order= "upper-first/lower-first"

data-type= “text/number” />

In the above syntax

- The select attribute indicates the expression on which the sorting operation can be done.
- The order attribute specifies whether the sorting operation is ascending or descending.
- The case-order attribute specifies the order of sorting for the uppercase and lowercase characters.
- The data-type attribute specifies the expression is sorted according to the given datatypes.

For implementing for-each and sort element the xml(save as book.xml) is shown in Example 8.

Example 8

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

<?xml-stylesheet type="text/xsl" href="sortauthor.xsl"?>

<bookstore>

<book>

<title>Wings of Fire</title>

<author>Dr. APJ Abdul Kalam</author>

<year>1999</year>

<price>198.00</price>

</book>

<book category="KIDS">

<title lang="en">Perfect Parenting</title>

<author>Sushant Kalra</author>

<year>2021</year>

<price>235.00</price>

</book>

<book>

<title>Learning XML</title>
```

<author>Erik T. Ray</author>

<year>2001</year>

<price>2865.00</price>

</book>

<book>

<title>All the Light We Cannot See</title>

<author>Anthony Doerr</author>

<year>2014</year>

<price>375.00</price>

</book>

</bookstore>

Now the xsl file (save as sortauthor.xsl) is shown in Example 9.

Example 9

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
<html>
<body>
<h2>My book Collection</h2>
<table border="1">
<tr bgcolor="#9acd32">
<th>Title</th>
<th>Author</th>
</tr>
<xsl:for-each select="bookstore/book">
```

```
                    <xsl:sort select="author"/>
<tr>
<td><xsl:value-of select="title"/></td>
<td><xsl:value-of select="author"/></td>
</tr>
</xsl:for-each>
</table>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

The output of Example 9

My book Collection

Title	Author
All the Light We Cannot See	Anthony Doerr
Wings of Fire	Dr. APJ Abdul Kalam
Learning XML	Erik T. Ray
Perfect Parenting	Sushant Kalra

Check Your Progress II

3. State True or False

(i) DTD provides a framework to validate an XML document.

(ii) You can not include DTD in an XML document.

(iii) The external reference of DTD is typically with a file extension .dtd.

(iv) The character content of attributes is termed PCDATA

(v) The DTD qualifier? is applies to zero or one element.

4. Fill in the blanks

(i) The XML _____ provide a method to avoid element name conflicts.

(ii) The &lt is the examples of _____.

(iii) XSLT stylesheet starts with the_____ element .

(iv) The _____ element specifies the value of an element or attribute.

(v) The _____ element of XSLT arranges data in ascending or descending order.

9.10 SUMMING UP

- XML is a markup language based on easy and platform independent rules.
- XML uses user defined tags.
- The syntax for creating an XML document is called the markup syntax
- The tags uses for creating an XML document are case sensitive.
- The XML declaration is used to identify that the specified document is an XML document.
- XML declaration start with <?xml and end with ?>.
- An XML element is the basic building block of an XML document.
- An XML document must have a single root element.
- Element names in XML are divided into two categories namely empty and nested elements.
- The XML attributes provide the additional information for that particular element.
- XML tree defines the elements in the tree structure.
- XML comments are used for documenting or to give remarks.

- DTD provide the common rules for defining the structure of an XML document.
- The external reference of DTD is typically with a file extension .dtd.
- XML Namespaces provide a method to avoid element name conflicts.
- XML provides the entity references to insert the symbols that have a specific purpose in xml document.
- XML schemas define the structure of an XML document.
- XML schema is alternative to DTD.
- XSLT document provide the instruction for transform XML document into different format according to the requirement.

9.11 ANSWER TO CHECK YOUR PROGRESS

1.(i) True (ii) False (iii)True (iv) False (v)True

2. (i) version (ii) nested (iii) attributes (iv) root (v) child

3. (i) True (ii) False (iii) True (iv) False (v) True

4. (i) Namespaces (ii) entity reference (iii) stylesheet

(iv) value-of (v) sort

9.12 POSSIBLE QUESTIONS

1. What is an XML? Explain the structure of an XML document.

2. Explain the various rules for defining an XML document.

3. What is XML declaration?

4. What is XML element? Explain the root, parent and child elements.

5. How can you define XML attributes? Explain it with a suitable example.

6. What are the uses of XML comment?

7. What is DTD? Differentiate between external and internal reference of DTD.

8. What are DTD qualifiers? Explain it with an example.

9. What is XSLT?

10. What is XML schema? Differentiate between XML schema and the DTD.

11. Explain the different types of data types available in XML schema document.

12. Explain the different XSLT element with examples.

9.13REFERENCESAND SUGGESTED READINGS

- https://www.w3schools.com/
- HTML 5 BLACK BOOK Published by dreamtech press

---×---

UNIT- 10
BASICS OF WEB-BROWSER

UNIT STRUCTURE:

10.0 INTRODUCTION

A browser is a software program that is used to explore, retrieve, and display the information available on the World Wide Web. This information may be in the form of pictures, web pages, videos, and other files that all are connected via hyperlinks and categorized with the help of URLs (Uniform Resource Identifiers). For example, you are viewing this page by using a browser.

10.1 OBJECTIVES

After going through this unit, you willbe able to know about-

- Web browser and its functions.
- Working principle of web browser.
- Plug-ins and other helper applications of a web browser.

10.2 WEB BROWSER

A browser is a software client program as it runs on a user computer or mobile device and contacts the webserver for the information requested by the user. The web server sends the data back to the browser that displays the results on internet supported devices. On behalf of the users, the browser sends requests to web servers all over the internet by using HTTP (Hypertext Transfer Protocol). A

browser requires a smartphone, computer, or tablet and internet to work.

History of Web Browser

- The **WorldWideWeb** was the first web browser. It was created by W3C Director Tim Berners-Lee in **1990**. Later, it was renamed **Nexus** to avoid confusion caused by the actual World Wide Web.
- The **Lynx** browser was a text-based browser, which was invented in **1992**. It was not able to display the graphical content.
- Although, the first graphical user interface browser was NCSA Mosaic. It was the first most popular browser in the world, which was introduced in **1993**.
- In **1994**, there were some improvements occurred in Mosaic and came to Netscape Navigator.
- In **1995,** Microsoft introduced the **Internet Explorer** It was the first web browser developed by Microsoft.
- A research project started on Opera in **1994**. Later, it was publicly introduced in 1996.
- **Apple's Safari** browser was introduced in **2003**. It was specifically released for Macintosh computers.
- In **2004**, Mozilla introduced **Firefox** as Netscape Navigator.
- In **2007**, a browser **Mobile Safari** was released as Apple mobile web browser.
- The popular browser **Google Chrome** was launched in **2008**.
- The fast-growing mobile-based browser **Opera Mini** was released in **2011**.
- The Microsoft **Edge** browser was launched in 2015.

Features of Web Browser

Most Web browsers offer common features such as:

1. **Refresh button:** Refresh button allows the website to reload the contents of the web pages. Most of the web browsers store local copies of visited pages to enhance the

performance by using a caching mechanism. Sometimes, it stops you from seeing the updated information; in this case, by clicking on the refresh button, you can see the updated information.

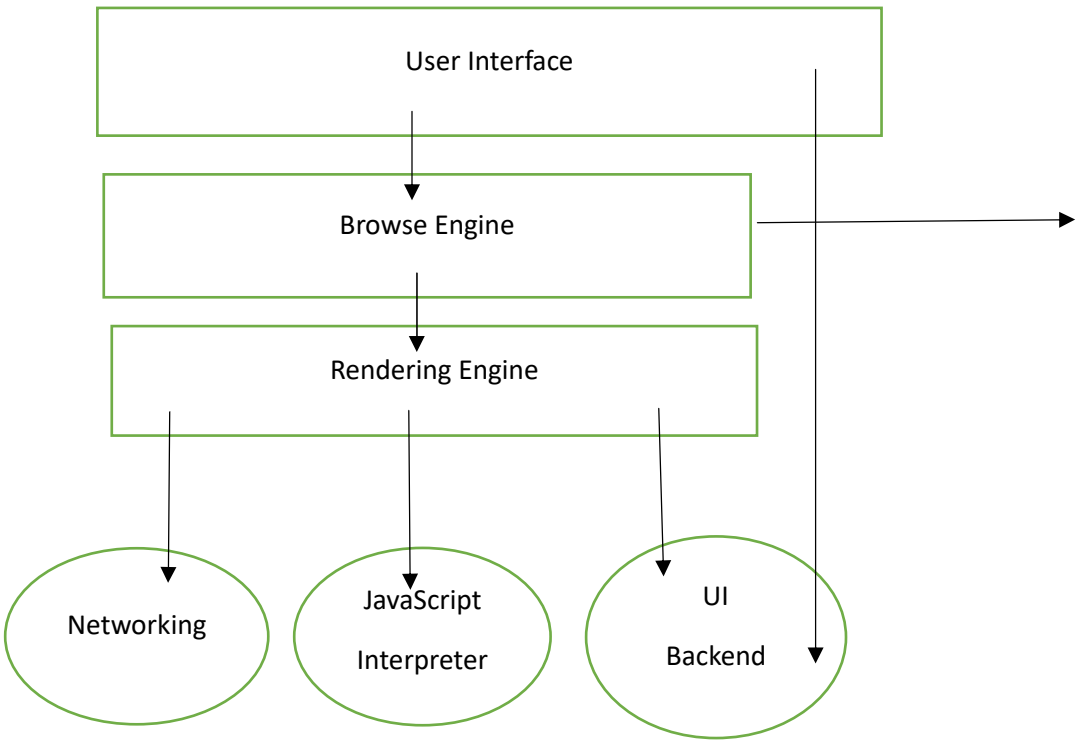
2. **Stop button:** It is used to cancel the communication of the web browser with the server and stops loading the page content. For example, if any malicious site enters the browser accidentally, it helps to save from it by clicking on the stop button.
3. **Home button:** It provides users the option to bring up the predefined home page of the website.
4. **Web address bar:** It allows the users to enter a web address in the address bar and visit the website.
5. **Tabbed browsing:** It provides users the option to open multiple websites on a single window. It helps users to read different websites at the same time. For example, when you search for anything on the browser, it provides you a list of search results for your query. You can open all the results by right-clicking on each link, staying on the same page.
6. **Bookmarks:** It allows the users to select particular website to save it for the later retrieval of information, which is predefined by the users.

URL orUniform Resource Locator

A **uniform resource locator** is the address of a resource on the internet or the World Wide Web. It is also known as a web address or uniform resource identifier (URI). For example, **https: www.javatpoint.com**, which is the URL or web address for the javatpoint website. A URL represents the address of a resource, including the protocol used to access it.

10.2.1 Functions and Working Principles of Web Browser

The primary components of a browser are shown in the below image:



User Interface: The user interface is an area where the user can use several options like address bar, back and forward button, menu, bookmarking, and many other options to interact with the browser.

1. **Browser Engine:** It connects the UI (User Interface) and the rendering engine as a bridge. It queries and manipulates the rendering engine based on inputs from several user interfaces.
2. **Rendering Engine:** It is responsible for displaying the requested content on the browser screen. It translates the HTML, XML files, and images, which are formatted by using the CSS. It generates the layout of the content and displays it on the browser screen. Although it can also display the other types of content by using different types of plugins or extensions. such as:
    - Internet Explorer uses **Trident**
    - Chrome & Opera 15+ use **Blink**
    - Chrome (iPhone) & Safari use **Webkit**
    - Firefox & other Mozilla browsers use **Gecko**

3. **Networking:** It retrieves the URLs by using internet protocols like HTTP or FTP. It is responsible for maintaining all aspects of Internet communication and security. Furthermore, it may be used to cache a retrieved document to reduce network traffic.
4. **JavaScript Interpreter:** As the name suggests, JavaScript Interpreter translates and executes the JavaScript code, which is included in a website. The translated results are sent to the rendering engine to display results on the device screen.
5. **UI Backend:** It is used to draw basic combo boxes and Windows (widgets). It specifies a generic interface, which is not platform-specific.
6. **Data Storage:** The data storage is a persistence layer that is used by the browser to store all sorts of information locally, like cookies. A browser also supports different storage mechanisms such as IndexedDB, WebSQL, localStorage, and FileSystem. It is a database stored on the local drive of your computer where the browser is installed. It handles user data like cache, bookmarks, cookies, and preferences.

When a user enters a web address or URL in the search bar like javatpoint.com, the request is passed to a **domain name server (DNS)**. All these requests are routed via several routers and switches.

The domain name servers hold a list of system names and their corresponding IP addresses. Thus, when you type something in the browser search bar, it gets converted into a number that determines the computers to which the search results are to be displayed.

The browser acts as a part of the client-server model. A browser is a client program that sends the request to the server in response to the user search queries by using Hypertext Transfer Protocol or HTTP. When the server receives the request, it collects information about the requested document and forwards the information back to the browser. Thereafter, the browser translates and displays the information on the user device.

In Brief:

- When a user enters something (like javatpoint.com) in the browser. This request goes to a domain name server.
- The browser sends the user request to the server using an IP address, which is described by the domain name server.
- The domain name server sends an IP address to the web server that hosts the website.
- The server sends the information back to the IP address, which is defined by the browser at the time of the request. The requested page may include links to other files on the same server, like images, for which the browser also requests the server.
- The browser gathers all the information requested by the user, and displays on your device screen in the form of web pages.

10.2.2 List of Internet Browsers

There are various types of internet browsers, which are as follows:

1. Microsoft Edge: Microsoft Edge is a web browser that comes pre-installed with Windows 10 operating system and Windows Server 2016. It was introduced to replace the Internet Explorer Web browser, and its code name was Spartan. It offers various types of features such as freestyle writing over Web page displays, refined search, and presentations for e-books and other reading resources.Microsoft Edge was developed under the **Spartan codename** Project. In April 2015, Microsoft changed the project Spartan name as Microsoft Edge. Although Internet Explorer and Edge are included with Windows 10, Edge act as a default browser. It combines new web technology evaluations and enhances the speed of browsing.

Although, Internet Explorer 11 was available in Microsoft Windows operating system, Microsoft Edge has become the default browser in Windows 10. It needs at least 1 gigabyte of memory. It offers several types of features, such as annotation features, a new rendering engine, and easy-to-use icons, etc. Furthermore, it also

provides better security as compared to Internet Explorer, and it can be combined with Cortana, Microsoft's virtual personal assistant.

Features of Microsoft Edge

- It provides support for Firefox and Chrome add-ons.
- It has the ability to fill the form automatically.
- It can be integrated with Cortana.
- It provides faster page rendering.
- It has more security features and also allows private browsing.
- It is modern, lightweight, and reduces resource consumption.

Latest versions of Edge browser

Platform	Versions	Release Date
Window 10	79.0.309.71	22-01-2020
Window 10 Mobile	40.15254.603	21-01-2020
Xbox One	40.15063.0	30-08-2018

2. Amazon Silk: Amazon silk is a proprietary Internet browser. It was released for Fire OS devices on 15 November 2011. It is based on the open-source Chromium project and derives most of the features from the Google Chrome browser. It divides the task of loading webpages between Amazon's servers and Fire.Silk is the default browser on most Amazon hardware devices as well as on app-based Kindle devices, TV, Fire, and compatible Echo devices. Furthermore, it is the first new mass-market, client software delivery mechanism, which should be built from the base of the cloud, not only the web.

Rendering pages on EC2 When the all contents of a page have been fetched on EC2, it renders the pages for display in the client's browser window. It depends on the amount of load and the client's network conditions.The components that can be handed off to EC2 to speed up browsing are: HTML, CSS, Networking, JavaScript, Block building, Marshalling, Native OM, etc.

3. Opera: An Opera web browser was first conceived at Telenor company in 1994, later bought by the Opera Software on 1 April 1995. It was designed for desktop and mobile interfaces, but it is more popular now for mobile phones. It is based on Chromium, and it uses the blink layout engine. An opera mini was released for smartphones on 10 August 2005 that could run standard web browsers. It can be downloaded from the google play store or Apple play store.

4. Apple Safari: Safari is an internet browser available for the Macintosh, and Windows operating systems included the iPhone, iPad, and iPod Touch. It was developed by Apple, Inc. on 30 June 2003. It is the default browser for the operating system in its products, such as OS X for the MacBook and Mac computers and iOS for the iPad and iPhone mobile devices. It is at number four in the browser market after Microsoft Internet Explorer, Mozilla Firefox, and Google Chrome. It uses the WebKit engine, which is used for rendering fonts, displays graphics, determining page layout, and running JavaScript.

5. Google Chrome: Google Chrome is an open-source internet browser. It is developed by Google on 11 December 2008 for Windows, Linux, Mac OS X, Android, and iOS operating systems.

6. Mozilla Firefox: The Mozilla Firefox web browser is developed by the Mozilla Foundation and its subordinate company, Mozilla Corporation. It was first released as beta on 23 September 2002. Although it was released as the Mozilla Browser, it was internally code-named Phoenix. The First version 1.0 of Firefox was introduced on 9 November 2004.

7. Internet Explorer: It is a web browser that is manufactured by Microsoft Corporation, and it is included with the Microsoft Windows operating system. But It was removed in Window 10 in support of Microsoft's new Edge Browser.

10.3 PLUG-INS &HELPER APPLICATIONS

Plugins are software extensions that can be loaded on a program to improve its functionality. For instance, you may require a plugin if you want to watch videos. Your browser will be unable to grasp how to play the video without plugins. Another example, a Photoshop plug-in like Eye Candy helps you to manipulate images with extra filters. Plugins allow you to customize website content, as well as computer

programmes, web browsers, and apps in general. The browser plug-in like Apple QuickTime or Macromedia Flash provides users the ability to play multimedia files within their Internet browser.

There are also plugins like VST, which help to add effects for sequencing programs and audio recording. In favor of using browser extensions, the use of plugins in web browsers has decreased; however, their use is continuing to add-ons to customize programs and apps. As plug-ins are an appropriate way to enhance the capabilities of the program; therefore, in modern times, plug-ins are used by most audio programs and graphics. Although most plug-ins are developed by third parties, and some may be shipped with the program. Also, they are sold separately.

Usually, these plug-ins can be downloaded as free from the Internet, as companies that make browser plug-ins are competing for a standard, like QuickTime and Flash. Additionally, plugins optimize the content you put out as an online creator. In terms of making more than big blocks of text, plugins also help websites and web pages. They also assist you with displaying YouTube and Vimeo videos on your website, improving the ranking of your blog entries, and customising the fonts on your website.

Some good Plugins

The use of once widely used web browser plugins are decreasing due to popular internet browsers are no longer supporting them and even replacing plugins with browser extensions. There are still few plugins that offer users the benefit in terms of providing customization of the content of websites as well as customization of computer programs, web browsers, and apps. Also, they can be used everyday computing and browsing. Few good plugins are given below:

Adobe Acrobat Reader: In modern times, people are more common to view PDF files. Also, the PDF format is ideal for the document. Therefore, this plugin enables the users to open and view those important documents.

Bukkit Plugins: Bukkit plugins are a kind of plugin that is more appropriate for people into Minecraft, as these types of plugins provide a lot of ways to customize how the sandbox video game can be played.

HP Print Service: From an Android device, these plugins allow the users to send to an HP printer. You can download this plugin from the Google Play Store like an app.

Samsung Print Service: This kind of plugin enables the users to print a document from Samsung mobile devices, including other Android devices. Also, with the help of this plugin, print jobs can be sent to different types of printers, which include Xerox, Lexmark, Canon, Brother, and Sharp. And, this plugin can be downloaded from the Google Play Store.

WordPress Plugins: These plugins allow you to customize the content and look of your website if you are a blogger on WordPress.

Plugins for Browser

There are different official websites for each web browser where you can download multiple plugins as well as install them. These plugins are also known as "add-ons" or "extensions." Furthermore, like Google, Apple, Mozilla, the web browser publisher first checks the plugins before downloading or installing that they are not harmful. There is a need to concentrate that downloading plugins directly from the browser's official website will be better for you. Below is a table that contains the name of web browsers as well as links for downloading plugins for them.

Browser	Where to get plugins
Mozilla Firefox	https://addons.mozilla.org/en-US/firefox/
Google Chrome	https://chrome.google.com/webstore/category/extensions
Microsoft Edge	https://www.microsoft.com/en-in?rtc=1
Apple Safari	https://apps.apple.com/us/story/id1377753262

Opera	https://addons.opera.com/en/
Brave	https://chrome.google.com/webstore/category/extensions
Vivaldi	https://chrome.google.com/webstore/category/extensions

Brave and Vivaldi browsers also use Chrome extensions because they are based on Chromium. Therefore, you can download extensions for these browsers from the Chrome Web Store.

Use of Plugins

Plugins are software additions that are used to add or extend functionality to your website. For instance, you have required a plugin to handle everything if you are going to take donations or sell products on our site. There are also some other plugins that are mainly useful for WordPress websites, such as a WordPress SEO plugin, a WordPress forms plugin, a WordPress security plugin, a WordPress backup plugin.

WordPress provides power to 37% of all websites on the internet as it is an outstanding Content Management System (CMS) for websites. But WordPress has no potential to do everything on its own, even if it offers more power for websites on the internet. That's why plugins come in the use that helps you out to enhance functionality to your website.

Examples of browser plugins

There are several types of Internet browser plugins, which will be detailed further below, that can be put in a browser to improve its capabilities.

- **RealPlayer:** RealPlayer provides users the ability to listen to live audio on the internet, which is originally known as RealAudio. As it allows you to listen to audio life; therefore, it saves your downloading time. Additionally, over the internet, it was considered one of the first widely available streaming audio players. It also allows you to listen to streaming audio as well as watch streaming video because

RealPlayer compresses the functionality of RealAudio due to the popularity of RealAudio.

- **QuickTime:** QuickTime is software that makes it capable of computer users to play video files. Apple developed this software and released it on 2 December 1991. This software can be used on Apple and IBM compatible computers running Microsoft Windows or any Macintosh operating system. QuickTime is also extensively used for playback. Files with the MOV extension can be played on IBM-compatible systems
- **Shockwave:** Developed by Adobe, Shockwave is similar to Adobe Flash that allows the users to view interactive, animated content in the browser window. Generally, it is a multimedia file format and application platform, which provides the developer more freedom to create kinds of applications.
- **VRML:** Virtual Reality Modelling Language (VRML) is a programming language that allows users to create a three-dimensional virtual environment on the Internet. If a user uses the proper VRML plug-in, he is able to view or enhance a virtual space or object with the help of VRML on the web.
- **X3D:** X3D is a royalty-free open standard file format, which is considered by many to be the next generation of VRML. It makes use of XML to show 3D scenes and objects.
- **Silverlight:** Microsoft Silverlight is similar to an Adobe Flash file, which is responsible for delivering. Users can access NET-based media and rich application experiences via the Internet. It's a cross-platform, cross-browser, and cross-device plug-in with files compressed into a XAP (pronounced like zap) file.
- **Adobe Acrobat:** Developed by Adobe, PDF allows users to capture the local appearance of a document. It is a file extension and file format, stands for Portable Document Format. It enables users to view and print the documents in the same format on any device. The use of PDF is very common in forms that need the user to fill in data and legal documents like bank statements, tax papers.
- **Adobe Flash:** Flash, a popular authoring software that allows users to create animated works that are saved as .FLV. It was originally released by Macromedia in 1996, which is

used to create vector graphics-based animation programs. Over the next decade, it gained many new features such as full-screen navigation interfaces, resizable file format, simple interactivity in an ant aliased, graphic illustrations.

- **Java:** Java was initially known as oak when it was created by James Gosling and colleagues at Sun Microsystems. It's an object-oriented programming language that's frequently used to create web apps and other software. It was first made publicly available in 1995, and Oracle now manages and owns it. When Java is used on the Internet, it enables the browser to perform features that are not normally available because it allows applets to be downloaded through the browser.

STOP TO CONSIDER

➢ According to W3Counter, as of November 2023, the top 5 most used browsers in the market are:

- Google Chrome: 69.6% of the market share.
- Safari: 15.1% of the market share.
- Microsoft Edge: 4.0% of the market share.
- Mozilla Firefox: 3.0% of the market share.
- Opera: 1.3% of the market share.

CHECK YOUR PROGRESS

1. The protocol is the client/server program used to retrieve the

A) IP

B) header

C) document

D) cache

2. In World Wide Web (WWW), an electronic store (e-commerce) can use a cookie for its

A) client shopper

B) server usage

C) server data

D) client data

3. The Uniform Resource Locator (URL), is a standard for specifying any kind of information on the

A) server-end

B) client-end

C) webpage

D) internet

4. The World Wide Web (WWW), was originally designed as a

A) stateless document

B) stateless program

C) stateless entity

D) stateless ip

5. The webpages are stored at the

A) server

B) client

C) domain

D) mail server

10.4 SUMMING UP

1. Web browser architecture typically consists of several key components working together to render web pages and provide a user interface. Here is a summary of the main components:

2. The User Interface includes elements like the address bar, back/forward buttons, bookmarks bar, tabs, and menus.

3.It provides the interaction points for users to navigate the browser and access its features.

4. Browser Engine is responsible for managing interactions between the UI and rendering engine.

5. It parses HTML and CSS, processes JavaScript, and renders content on the screen.

6. Rendering Engine interprets HTML, XML, CSS, and other content types to display web pages.

7. Different browsers use different rendering engines (e.g., Blink in Chrome, Gecko in Firefox, WebKit in Safari).

8. Networking handles network requests such as fetching HTML, CSS, JavaScript, images, and other resources from servers. It implements protocols like HTTP, HTTPS, FTP, etc.

9. JavaScript Engine executes JavaScript code embedded in web pages.

10. Engines like V8 (used in Chrome), SpiderMonkey (used in Firefox), and JavaScriptCore (used in Safari) are commonly employed.

11. UI Backend draws UI elements like windows, buttons, and menus using native operating system APIs. It ensures consistency with the native look and feel of the platform.

12. Data Storage manages local data storage including cookies, cache, local databases, and web storage (e.g., localStorage, sessionStorage).

13. Plugins are optional components that extend browser functionality to handle specific types of content (e.g., Adobe Flash Player, PDF viewer).

14. Modern browsers tend to phase out support for plugins in favor of native web technologies.

15. Extensions/Add-ons allow users to customize their browsing experience by adding additional features or modifying existing ones.Examples include ad blockers, password managers, and developer tools.

16. Security Components implements security measures to protect users from malicious websites, phishing attempts, and other online threats.

17. Includes features like sandboxing, SSL/TLS support, and secure password management.

10.5 ANSWERS TO CHECK YOUR PROGRESS

1. A) IP

2. D) client data

3. D) Internet

4. B) stateless program

5. A)server

10.6 POSSIBLE QUESTIONS

1.What is the role of the rendering engine in a web browser?

2. Name two popular JavaScript engines used in modern web browsers.

3. What are the main components of a browser's user interface (UI)?

4. How does the networking component contribute to web browsing?

5. Explain the purpose of browser plugins.

6. What is the difference between local storage and session storage in a browser?

7. How do browser extensions/add-ons enhance user experience?

8. Why is security important in web browser architecture?

9. Describe the function of the UI backend in a browser.

10. Name one example of a widely used rendering engine and the browser it powers.

11. What is the purpose of browser plugins?

12. Give an example of a commonly used browser plugin.

13. How do plugins enhance the functionality of a web browser?

14. What technology is commonly used to develop browser plugins?

15. Can you name a type of plugin that is becoming less common in modern browsers?

16. How do users typically install browser plugins?

17.What are some security concerns associated with browser plugins?

18. Can plugins be used to play multimedia content embedded in web pages?

19. Are browser extensions the same as plugins? Explain.

20. How can users manage and control the plugins installed in their web browser?

10.7 REFERENCES AND SUGGESTED READINGS

1.Web Application Architecture: “Principles, Protocols, and Practices" by Leon Shklar and Rich Rosen.

---×---

UNIT- 11
ARCHITECTURE OF WEB-BROWSERS

UNIT STRUCTURE:

11.1 INTRODUCTION

A web browser takes you anywhere on the internet. It retrieves information from other parts of the web and displays it on your desktop or mobile device. The information is transferred using the Hypertext Transfer Protocol, which defines how text, images and video are transmitted on the web.

11.2 OBJECTIVES

This unit is an attempt to analyse the ideas of web browser architecture. After going through this unit, you will be able to-

- Create efficient, secure, and compatible web applications that deliver a positive user experience across different platforms and devices.
- Learn the web browser architecture which is essential for web developers, designers, and anyone involved in the creation and maintenance of web content.

11.3 CONCEPTUAL ARCHITECTURE OF SOME TYPICAL WEB BROWSERS

Web Browser is a highly advanced application software that helps surf the internet. It is like a gateway to access the huge potential information the internet holds with a simple, user-friendly UI (user interface). Have you ever thought about how web pages are

painted on the web browser and how it manages unlimited searches from all over the world? Browser architecture is not just fascinating but magical as well. How anyone can have answers to almost everything in a few clicks painted beautifully on their screen is worth a read.

In this chapter we will study and understand the browser architecture and how performance, compatibility, and security are maintained in the browser. The availability of many browsers like Chrome, Firefox, Safari, and Edge gives users the option to compare performance and choose what is needed as per their requirements.

Web browser

A web browser is an application that serves as a gateway interface between the user and the server. It loads and compiles the results from the server in the form of an HTML web page and paints it on the screen of the user. It is basically an interaction tool for using the internet.

A web browser helps us find any information available on the internet in the form of texts, photos, and videos. Examples of web browsers are Chrome (one of the most popular browsers used all over the world), Firefox, Safari, Edge, Brave, and many more.

Browser Architecture

Browser architecture is designed to provide a faster, more secure, and more feature-rich platform that helps users interact easily with the internet. The browser architecture is broadly divided into seven parts.

- The user interface of a browser is designed such that it allows personalization, as every individual has different interests. This personalization is achieved by providing basic features like groups, collections, bookmarks, and themes. Each browser can have a different user interface and features.

- Browser Engine: The browser engine is responsible for coordinating web content that is fetched from the server and user interactions. It keeps a note of which button is clicked, which URL is asked to parse, and how the web content will be processed and displayed on the browser.

- Rendering Engine: The rendering engine, on the other hand, interprets and renders web content. In most browsers, both the browser engine and the rendering engine work together to provide better results to the user.
- Networking Layer: This layer handles the communication part. When the user enters or clicks on a URL, the network layer starts a HTTP request to the web server to load the requested web page. It also manages fetching resources from HTML files, images, stylesheets, and more. Have you seen those cookie notifications while searching for information on the internet? Mostly, the network layer works behind the scenes for those cookies and cache.
- JavaScript Engine: The JavaScript Engine is the core component of browser architecture, with the ability to manipulate web content and introduce dynamic behaviour in web pages.
- Data Storage: A large part of the browser goes into storing various types of data, which include not only user preferences, browsing history, passwords, and other regular data updates as well (address, name, and contact).
- UI backend: The UI backend provides dynamic and interactive behaviour on the web page and enhances the overall functionality and performance of the browser.

Fig 1 depicts a typical browser architecture.

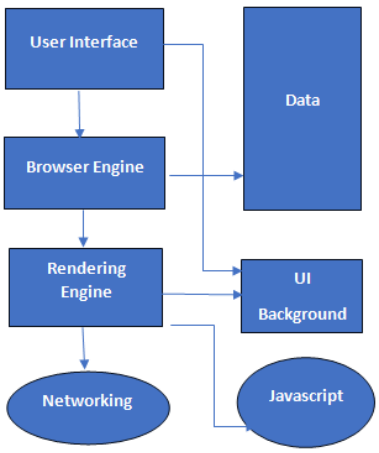


Fig 1: A typical Browser Architecture

Understanding the architecture of a web browser helps in appreciating the complexities involved in rendering web pages and providing a seamless user experience. Different browsers may have variations in their architecture, especially when it comes to the rendering engine and JavaScript engine they use.

STOP TO CONSIDER

- Studying web browser architecture is essential for web developers, designers, and anyone involved in the creation and maintenance of web content.
- **Chrome** is the official web browser from Google, built to be fast, secure, and customizable.

Conceptual architecture of some typical web browser

The conceptual architecture of a typical web browser involves various components that work together to provide users with a seamless and interactive browsing experience. Here is a high-level conceptual overview of the architecture:

User Interface (UI) Layer:

Browser Window: The visible interface that users interact with.

Address Bar: Allows users to enter URLs or search queries.

Navigation Buttons: Back, forward, refresh, and home buttons for navigation.

Bookmarks and History: Features for managing and accessing saved bookmarks and browsing history.

Tabs: Support for multiple open tabs within a single browser window.

Browser Engine:

Rendering Engine: Interprets and renders HTML, CSS, and JavaScript to display web content.

Document Object Model (DOM): Represents the structure of a web page as a tree of objects, facilitating dynamic content updates.

Networking Layer:

HTTP/HTTPS Protocol Handling: Manages communication with web servers using standard protocols.

URL Handling: Resolves and processes URLs entered by users.

JavaScript Engine:

Interpreter: Executes JavaScript code embedded in web pages.

Just-In-Time (JIT) Compilation: Translates JavaScript code into machine code for improved performance.

Data Storage:

Cookies: Small pieces of data stored on the user's device, often used for session management.

Local Storage: Persistent storage for web applications.

Browser Cache: Temporary storage for speeding up page loading.

Security and Privacy:

SSL/TLS Handling: Ensures secure communication by encrypting data between the browser and web servers.

Security Protocols: Implements security measures to protect against malicious activities.

Privacy Settings: Controls for managing cookies, tracking, and other privacy-related features.

User Input Handling:

Mouse and Keyboard Events: Captures and processes user input for interaction with web pages.

Form Handling: Manages user input in HTML forms.

UI Backend and Frontend:

UI Backend: Draws basic widgets and UI elements using the operating system's graphical capabilities.

UI Frontend: Manages the user interface components and interacts with the UI Backend.

Extensions and Add-ons:

Extension API: Provides a framework for third-party developers to create browser extensions.

Extension Management: Allows users to install, enable, or disable extensions.

Browser Kernel/Shell:

Integration Layer: Coordinates the interaction between different components.

Event Handling: Manages events such as clicks, scrolls, and keyboard input.

Error Handling: Addresses errors and exceptions that may occur during browsing.

Platform Integration:

Operating System Interaction: Communicates with the underlying operating system for tasks like file handling and integration with system settings.

This conceptual architecture represents the high-level organization of components within a web browser. Different browsers may have variations in their specific implementations and technologies used, but these core components are common across most modern web browsers.

CHECK YOUR PROGRESS

1. A_____ is an application that serves as a gateway interface between the user and the server

2. _____ is the visible interface that users interact with.

3.Document Object Model (DOM) represents the structure of a web page as a tree of _____, facilitating dynamic content updates.

4. SSL ensures secure communication by _____ data between the browser and web servers.

5._____ is a temporary storage for speeding up page loading.

11.4 SUMMING UP

1. The architecture of a web browser is a complex system that involves various components working together to provide users with the ability to browse the internet. Here's a simplified overview of the typical architecture of a web browser:
2. Browser Window is the visible part of the browser that users interact with.
3. Address Bar allows users to input URLs or search queries.
4. Back/Forward Buttons is enable navigation through previously visited pages.
5. Bookmarking allows users to save and organize favourite websites.
6. Browser Engine is responsible for processing the HTML and CSS, interpreting JavaScript, and handling Document Object Model (DOM) manipulation.
7. The two main browser engines are Blink (used by Chromium-based browsers like Google Chrome) and Gecko (used by Mozilla Firefox).
8. Rendering Engine renders the content of web pages by interpreting HTML, XML, CSS, and other web technologies.
9. Networking manages the network communication, handling the requests and responses between the browser and web servers. Implements protocols such as HTTP, HTTPS, and WebSocket.
10. JavaScript Engine interprets and executes JavaScript code embedded in web pages.Examples include V8 (used by Chrome), SpiderMonkey (used by Firefox), and JavaScriptCore (used by Safari).
11. UI Backend draws basic widgets like buttons, text fields, etc., on the browser window.

12 It uses the operating system's user interface methods to display these elements.

13. UI Frontend handles the user interaction components like buttons, forms, etc. It communicates with the UI Backend to display the UI elements.

14. Data Storage manages data storage, including cookies, local storage, and the browser cache. It enables the persistence of user data and preferences.

15. Security and Privacy Components implements security features like SSL/TLS for encrypted communication. It manages permissions, cookies, and other privacy-related settings.

16. Extensions/Add-ons allows users to enhance browser functionality by installing extensions or add-ons.These can modify or add features to the browser.

17. Browser Kernel/Shell: The core part of the browser that integrates all components and provides the basic functionality of browsing.

11.5 ANSWERS TO CHECK YOUR PROGRESS

1. Web Browser
2. Browser Window
3. Objects
4. Encrypting
5. Browser Cache

11.6 POSSIBLE QUESTIONS

1) How would you describe the difference between a web architect and a web developer?
2) What is Web browser?
3) Draw and explain browser architecture.
4) How can you explain a basic web architecture?
5) What is the technology behind web architecture?
6) What is web design architecture?

11.7 REFERENCES AND SUGGESTED READINGS

1.Web Application Architecture: “Principles, Protocols, and Practices" by Leon Shklar and Rich Rosen.

---×---

BLOCK- II

Unit 1: Introduction to Client/Server Computing

Unit 2: Web Servers-I

Unit 3: Web Servers-II

Unit 4: Server-side scripting basics

Unit 5: PHP: Hypertext Preprocessor

Unit 6: Distributed Object based models

Unit 7: Advanced web Technologies-I

Unit 8: Advanced web Technologies-II

Unit 9: Web Security

UNIT- 1
INTRODUCTION TO CLIENT-SERVER COMPUTING

UNIT STRUCTURE

1.1 INTRODUCTION

The client server computing is a process where clients always send some requests and on that basis server machine response according to the client's request. A single machine can be either a client or a server depending on the configuration of the machine. The user of a web browser is making the clients request to the web server. In this unit, we will discuss about the client-server architecture and different types of Client-Server systems. We will also discuss the differences between the fat Client and fat Server.

1.2 OBJECTIVES

After going through this unit learner will able to:

- Understand the basics of Client-Server computing.
- Learn the different types of Client-Server systems.
- Understand the concept of middleware systems.
- Understand the concept of 2-tier/3-tier/N-tier systems.
- Learn about fat client and fat server.

1.3 ARCHITECTURE OF CLIENT-SERVER COMPUTING

There are mainly two processes involved in Client-Server computing. These two processes are described as follows.

1.3.1 Client Process

The client process sends a request to a server for requesting some process that has to be performed by the Server. Generally the client process is managed in the user-interface. The client side process is a front-end application where client can see and interact with it. The key element of client process is Graphical User Interface (GUI). In GUI based application, user or client can send their request to the server.

1.3.2 Server Process

The main job of server is to perform the operation on the requests that are sending by the clients. Server programs generally receive the requests from the client and then retrieve the required information from the stored databases. The server process performs the back-end tasks. The client-server architecture is shown in the Figure1.1. Here the client sending their request through the web browser.

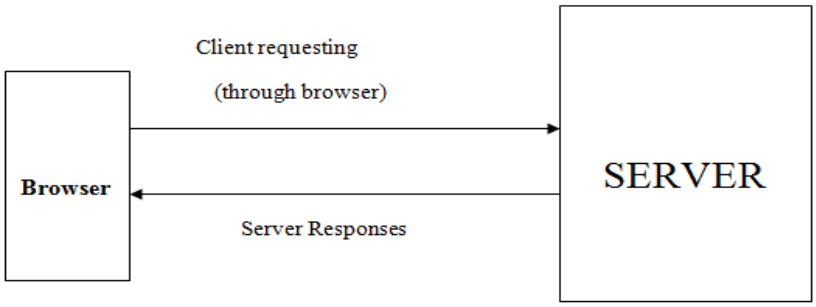


Figure 1.1: Client-Server Architecture

There are different types of server-side technologies available for responding the request receives from the clients. Some of these technologies are as follows.

(i) **Active Server Page (ASP):** ASP was developed by Microsoft to allow creating a dynamic web-site. ASP is the first server-side script engine and now it is superseded by the ASP.NET.An ASP script typically has .asp extension.

(ii) **PHP(Hypertext Preprocessor):** PHP is a server side scripting language. User can create dynamic Web pages using PHP. PHP includes some predefined functions that create, open, read, write and close files stored on the server.PHP provides a Document Object Model(DOM) to access XML elements. The PHP files have .php extension.

(iii) **JSP (Java Server Page):** The JSP technology is used to develop dynamic Web application.JSP pages havean extension .jsp which contains the java code.

(iv) **SSI(Server Side Includes):** The SSI involves the embedding of small code inside the HTML page. An SSI page typically has .shtml as its file extension. The SSI can be used to display the size of a file before downloading.

(v) **Python:** Python is a general purpose language. It is often used as a server side scripting language. The extension of a Python file is .py. The advantages of using Python are:

- Python works on different platforms like windows, Mac, Linux, etc.
- The syntax of Python is simple and it is similar to the English languages.
- Python codes are interpreted.

Using the above web technologies, it has become easier to maintain web pages and it is especially helpful to managing a large web-site. The developer need to include the server side scripting code in the HTML pages. This code is passed to appropriate interpreter which will processed the instruction and generate the final HTML. The final HTML will now displayed by the browser. The main task of the browser is to just receive the HTML code. The browser is not aware of the functioning of the server. Client-server system has distributed application architecture that distributes the tasks between service requesters that is the client and the service provider that is known as the server.

1.4 CHARACTERISTICS OF CLIENT-SERVER SYSTEMS

The client-server architecture is scalable horizontally that means more clients can be added and vertically means more servers can be added in the architecture. The characteristics of client-server architecture are as follows.

- The client or front-end portion that always interacts with the user and server or back-end portion that interacts with the shared resources. The client process contains the solution specific logic and provides an interface between the user and the other applications. The server process acts as a software engine that manages shared resources such as databases, printers, modems etc.

- In Client-server systems the client and the server have to follow a common communication protocols so that they can interact each other easily.

- A server can only processed a limited number of client's requests. So, server processes the request on system based priority.

- In client-server computing processes the requirement of resources of client and server are different.

- In Client-server system, the client and the server can be scaled horizontally and vertically.

1.5 FUNCTIONS OF CLIENT-SERVER SYSTEMS

There are different functions operated in the client and server process. Clients and servers both operate on a computer network with their own specific functions. The following are the functions of the client process.

- Client always handles the user interface.
- It translates the user's request into the desired protocol.
- It sends the request to the server.
- It waits for the responses from the server.
- Translates the responses into human readable format.

The functions for the server process are as follows.

- Server always waits for client's request. It always listensto the client's query.
- Server retrieves the required information from the stored information and processes the client's query.
- It returns the results back to the client.

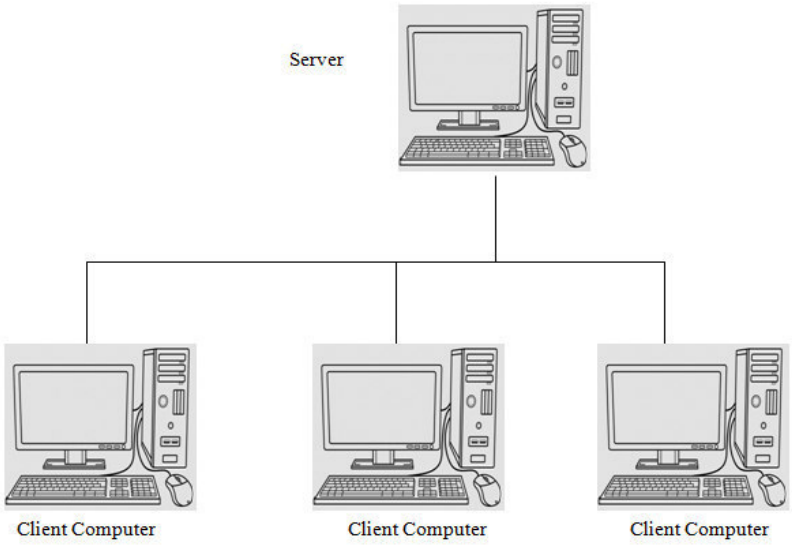


Figure 1.2: Connecting nodes in Client Server Mechanism

In Client Server architecture there are different types of Clients. These clients are categorized as Non-GUI clients and GUI clients. Non-GUI client do not use graphical user interface. The GUI clients interact with the server with the help of Graphical User Interface**.**

1.6 DIFFERENT TYPES OF SERVERS

In Client-Server system different clients have different requests and the server responses according to the requirement of the requests. Different types of servers for responding different types of requests are discussed as follows.

1.6.1 File Servers

The file server is useful for storing and sharing of files over a network. The operating system is working as file server if the system is configured with Network File Server (NFS). The NFS are used for various applications in which different processors used different hardware and software applications. The client-server computing describes the relationship between information and programs. User can share information without physically transfer files. In file server, user also benefit from remote access. The file servers can also be used as backup server as it is accessible to the

other network users. The main computing part is taken as client machine that interacts with the client and provides mechanism to communicate with the server. The server machine will decide from where data will be extracted. Remote Procedure Call (RPC) is an example of client-server computing. In RPC mechanism, client sends an RPC message to a server and server is always waiting for an RPC call. It is a powerful technique for constructing distributed, client-server based applications.There are two types of file server as discussed below.

(i) **Dedicated file server:** Dedicated file server provides the services to the other computer solely. It dedicated to only one purpose being a file server. A dedicated file server has sufficient storage space for the website and it is comparatively more secure.

(ii) **Non-dedicated file server:**Any workstation in the network can act as a non-dedicated file server. This type of file server simultaneously working as server and as a workstation according to the requirement. A non dedicated file server has less storage space and less secured.

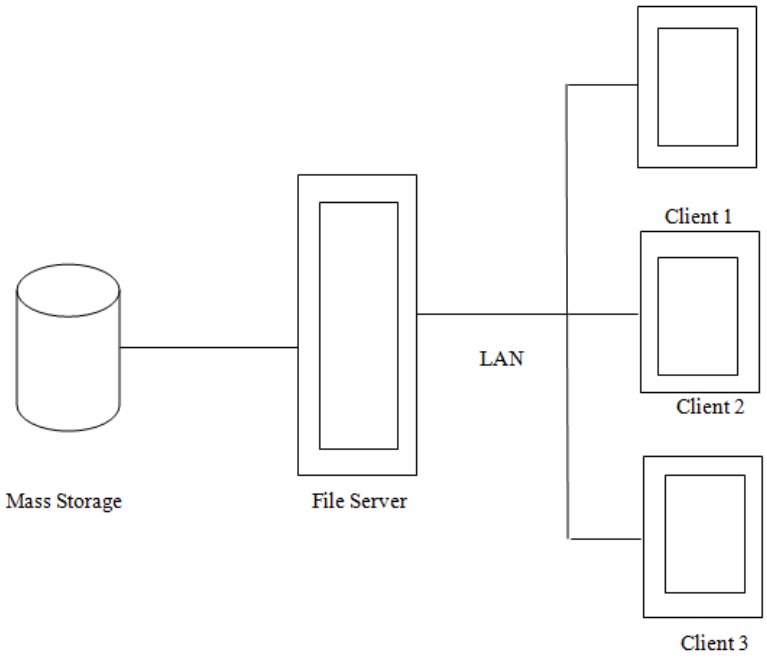


Figure 1.3 File Server Mechanisms

1.6.2 Network Server

Network server is a node or computer that provides the services, data, application or other network information to the computer which is termed as a client on the network. The main aim of the network server is to facilitates or manage the resources among the network devices. The network server categorized based on the physical characteristics, number of processors they contain, instruction set and types application they supported.

The network server provides the authoring system that provides multimedia based application. The network server provides a high-speed technical support for digital data transmission in the network. It also controls timestamp tagged with sending and receiving data packets. Different transmission technologies and network management are controlled by the network server. The network server provides a suitable framework for sending and receiving information in the client-server computing.

1.6.3 Database Server

Database server uses the database applications and provides the services to the other computer system which is termed as a client in the client-server system.Database management systems (DBMS)frequently provide database-server functionality. Database server uses their programs to retrieve the information according to the requirement of a client. The database server is also known as SQL engine. User sends their queries to the database server using front end application which is running in the user's computer. The server runs in the back end and handles the task such as data analysis and storage space management.

1.6.4 Web Server

Web-server is considered to be a new client-server model consists of a client communicating with a superfast server. Web server uses HTTP protocols to provide the responses of the request which are comes from the client side. There are two types of Web-servers one is static server and the other is the dynamic web server. In static web server, the content is static and in the dynamic web server, the content can be updated or changed. Some of the

examples of web server are Apache HTTP server, Nginx, Microsoft Internet Information Services (IIS) etc.

Features of web servers are:

(i) **Content hosting:**Web server store and serve all the web content. The web content includes images, videos, any other multimedia files.

(ii) **Security:** Web server contains different security mechanism to protect the server from any unauthorized access.

(iii) **Load balancing:** Some web serverdistributed the upcoming traffics to ensure optimal performance and the availability.

(iv) **Caching:** To reduce the server load the web server cache the frequently access files.

CHECK YOUR PROGRESS I

1. Multiple Choice Questions

(i) Client-server computing is a type of distributed computing application where information exchanged between the_____.

(a) Server and clients (b) Loader and Linker
(c)Protocol and HTTP protocol (d) ISP and Internet

(ii) Java Server page is a ____.

(a) Client technology (b) server technology
(c)HTTP request (d) HTTP responses
.

(iii) Network server controls timestamp tagged with sending and receiving__.

(a) Datagrams (b) Datasheet
(c)Dataports (d)Data packets

(iv) Apache HTTP server is an example of_____.

(a) Network server (b) Database server
(c) File server (d) Web server

(v) The database server is also known as_____.

(a) DNS server (b) Object based server
(c)SQL engine (d) Groupware server

2. Fill in the blanks

(i) The _____process is a front-end application.

(ii) ASP was developed by _____ .

(iii) Microsoft Internet Information Services (IIS) is an example of_____.

(iv) Remote Procedure Call (RPC) is an example of _____.

(v) JSP pages have an extension_____

1.7 ADVANTAGES OF CLIENT-SERVER COMPUTING

There are various advantages of client-server computing. These advantages are as follows.

(i) **Performance and reduced workload:** In client-server computing the processing is performed on distributed fashion. Unlike the traditional PC database, the speed of the DBMS packages is not tied to the speed of the workstation. The workstation has only the power of running the front-end software, which increases the usable lifetime of older PCs. This will also affect the working load on the network that connects the workstations. Instead of exchanging the entire database file in the busy network, the network traffic is reduced to queries and responses from the database server. Some database server can even store and run procedures on the server itself, which will reduce the traffic even more.

(ii) **Independence of workstation:** In client-server system user are not limited to one type of system or platform. Application independence is achieved as the workstation doesn't need to use the different type of DBMS application software. User can continue to use familiar software to access the database and the developers can design front-end application for the workstation on which the software will run.

(iii) **System interoperability:** Client-server system makes possible for different type of component system like client, network and server to work together.

(iv) **Scalability:** In client-server system, one module can be replaced without affecting the rest of the system. For example, it is possible to upgrade the server to a more powerful machine, with no visible change in the system for the end user. This ability to change component system makes client-server more powerful in terms of both hardware and software.

(v) **Data integrity:** Client-server system preserves the data integrity. DBMS system can provide number of services like encrypted file storage, real time backup, disk mirroring and other transaction processing that keeps track the changes made to a database and correct the problems in case of system crash.

(vi) **System administration:** In client-server system data and data management can be centralized. Some of the system administration functions are security, data integrity and backup and recovery.

(vii) **Integrated services:** In client-server model, all information required for a client is available at the desktop. There is no requirement of changing to another system to access the information.

(viii) **Location independence:** In client-server system there is no concern of location or technology for processing data.

(ix) **Sharing resources:** Sharing of resources is an important advantage of client-server system. The application processes provide data entry, storage and reporting by using distribution to clients and servers.

1.8 DISADVANTAGES OF CLIENT-SERVER COMPUTING

Disadvantages of client-server computing are presented as follows.

(i) **Maintenance cost:** The major disadvantage of client-server system is the increasing cost of administrative and support personal to maintain the database server.

(ii) **Training cost:** Training cost is treated as start-up cost in client-server computing. It includes the cost for training the persons as they may be unfamiliar with the system.

(iii) **Hardware and software cost:** The overall cost of hardware and software is usually higher than that of a traditional PC based multiuser system.

(iv) **Complexity:** There are many different components in client-server system. If one component is failed due to some reason then the whole system is inoperative. Due to the complexity of the system it is harder to find out the problems when the database of the system is crashed.

1.9 MIDDLEWARE

Middleware serves the purpose of a network between components of a client-server system and it can be run on both on client machine as well as on the server machine. Middleware provide the communication between different types of computer systems that enables cross-platform computing in a client-server environment and allowing different types of clients for accessing same data. Client portion has user interface and the server portion always have data. The Middleware server refers to a server on any middleware platform. Developer use middleware to simplify their development process.

The following are the more common use cases of middleware.

(i) **In Game Development:** Game developer use middleware as a game engine. In the development of game, developer required to communicate various images, videos, audios etc for different systems. The game engine provides the facilities for communicating with the different systems.

(ii) **Electronics:** Electronic engineers use middleware to accommodate different types of sensors with their controllers. The middleware allows these sensors to communicate with the different components through a standard structure.

(iii) **Software Development:** The middleware play a vital role in development of software. It provides a common or standard API to manage the required input and output data from the component. The linking of different components in middleware is hidden from the user.

(iv) **Distributed System:** The distributed system usually consists of the front-end and back-end. The front-end applications are the software used in computer and other electronic devices. On the other hand the back-end applications are the software that handles data processing, business logic and the resource management tasks.

1.10 TYPES OF CLIENT-SERVER SYSTEM

The main types of Client-server system are as follows.

(i) **1-tier system:** 1-tier is the simplest and all are familiar with this system. The 1-tier system is similar to running application on user's personal computer. In the 1-tier system, all the necessary components required to run an application reside on the same computer or machine. Such system does not form a part in the network and cannot be used to designed application in the web. The 1-tier system can be designed easily but has little scalability.

(ii) **2-tier system:** The basic architecture of client-server system is 2-tier system in which client is in the front-end and server is in the back-end. In this system front-end application contains business logic and interacts with user. Java, VB etc. are common example of front-end application. Back-end applications are used to store data. MySQL and Oracle are the examples of back-end applications.

(iii) **3-tier system:** In 3-tier system, middle tier is inserted in between client and server. The components of 3-tier system are divided into three layers namely presentation layer, functional layer and data layer. The layers are logically separated to each others. The 3-tier system overcomes some limitations of 2-tier system. The middle tier is coded in a highly portable language

like C. The clients in the 3-tier system connect with middle tier with the help of some protocols such as API, RPC and DLL. The middle tier communicates with the server via some standard database protocols. The middle tier contains most of the applications and logic for communicating with the server.

(iv) **N-tired systems:** The 3-tier system can be extended to the N-tiered system where the middle tier provides connections to various types of services to integrate and connect them with the clients. A client-server system is told to have N-tiered when the system has 3 layers of applications that are front-end, component and back-end. In the 3-tier system, the client sends HTTP requests to the server. At the receiving end the server sends the responses directly or passes it on to a specific application server. Then the application server creates a script for dynamic content and extracts the database or files from a database server according to the requirement. The N-tiered architecture is shown in Figure 1.4.

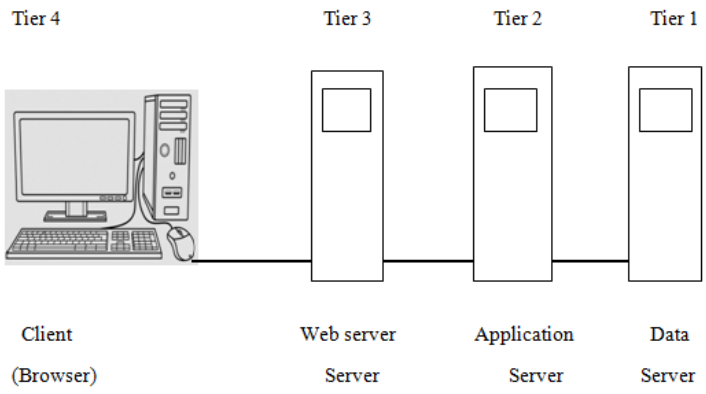


Figure 1.4: N-Tiered Architecture

The advantages of N-Tiered system are as follows.

- Different application can be developed in N-tiered system.

- Applications are rolled out independently in N-tiered system.
- In N-tiered system, server can be utilized separately for database and other application server functions.
- Server can be configured according to the requirement for each tier in the N-tiered system.
- N-Tier model makes applications more readable and reusable.

STOP TO CONSIDER

N-tier system offers innovations in the standard client-server technology that give rise to the internet itself.

1.11 FAT CLIENTS AND FAT SERVERS

The main parts of the client-server system are clients, servers and middleware. There are different types of clients, servers and middleware. Clients can be categorized as fat clients and thin client. Fat client have huge amount of logic and applications and thin client have less responsibility in the system. The client in a database system can be considered as fat client because the database system put more responsibilities on client side. The fat client is used in development of personal software and in decision support system. A fat server creates more levels of abstraction for providing services.

1.11.1. Fat Clients

A fat client requires minimum periodic connections in the network to a central server. A thin client heavily depends on to the server applications. While designing a client-server system a decision has to be taken on the assignment of tasks. These types of decisions are vital because it will directly effect on the cost and security of the client-server system. Due to decrease in price of the software and increase in growth of the technology, the fat client is more promising in comparison to the thin client.

Advantages of fat clients are as follows.

- **Less requirement of server**: Client-server system having a fat client has no requirement of high performance level as it required for a thin client.
- **High level of multimedia performance**: Fat client offer better performance in comparison to the thin client to handle multimedia such as graphics, audios, videos, etc.
- **Flexibility:** Fat client is more flexible in client-server system. Some operating system has own resources available locally. But it is not an easy task for running in case of thin client.
- **High Server performance**: Since in fat client, more works is done on the client side, so the load on the server is less in comparison to the thin client.

1.11.2 Fat Servers

Fat server performs all the application and processing required in the client-server system. A fat server has most of the program codes residing on it rather than in the client machines. Web server is a good example of fat servers. In client-server system having a fat server with thin client offer more advantages of easy updating of software. It is required only to change in the codes that resides in the fat server rather than changing the codes in the clients associated to that server. The application codes on fat servers are easy to deployment over the network as most of the codes reside on the server.

Advantages of fat server

- Fat server application are easier to manage and deployment.
- In fat server, most of the codes run on the servers.

Check Your Progress II

2. Multiple Choice Questions

(i) A client-server system is told to have N-tiered when the system has ___.

(a) Three layers (b)two layers
(c) 1 layer (d) None of these

(ii) A client that have huge amount of logic and applications is called___.

(a) Fat client (b) Thin client
(c) Middle client (d) short client.

(iii) A client is that heavily depends on to the server applications is known as____.

(b) Fat client (b) Data client
(c)Thin client (d) web client

(iv) In fat client more works done on the ___.

(b) Middle part (b) Client side
(c) Server side (d) HTTP side

(v) In fat server, most of the codes run on the___.

(a) client (b) server
(c) middle ware (d) network

3. State True or False

(i) The Middleware server refers to a server on any middleware platform.
(ii) Middleware cannot be run on client machine.
(iii) 1-tier is the simplest and all are familiar with this system.
(iv) Fat client is more flexible in client-server system.
(v) End user can not access the fat server resources directly.

1.12 SUMMING UP

- The client server computing is a process where clients always send some requests and server responses according to the requests.
- A single machine can be either a client or a server depending on the configuration of the machine.
- The client side process is a front-end application.
- The server process performs the back-end tasks.
- Clients and servers both operate on a computer network with their own specific functions.
- In Client-server system, the client and the server can be scaled horizontally and vertically.
- Dedicated file server provides the services to the other computer solely.
- In client-server system, users are not limited to one type of system or platform.
- DBMS system can provide number of services like encrypted file storage, real time backup, disk mirroring and other transaction processing that keeps track the changes made to a database and correct the problems in case of system crash.
- The 1-tier system is similar to running of application on user's personal computer.
- In 3-tier system, middle tier is inserted in between client and server.
- The components of 3-tier system are divided into three layers namely presentation layer, functional layer and data layer.
- N-Tier model makes applications more readable and reusable.
- A fat client requires minimum periodic connections in the network to a central server.
- Fat server performs all the application and processing required in the client-server system.

1.13 ANSWER TO CHECK YOUR PROGRESS

1. (i) (a)Server and clients (ii) (b)server technology
(iii) (d) Data packets

(iv) (d)Web server (v) (c) SQL engine

2. (i) Client-side (ii) Microsoft (iii) Web-server (iv) Client-server computing (v) .jsp

3. (i) (a)Three layers (ii) (a)Fat client (iii) (c) Thin client

(iv) (b) Client side (v) (b) Server

4. (i) True (ii) False (iii) True (iv) True (v) False

1.14 POSSIBLE QUESTIONS

1. What is Client-server computing?
2. Explain the architecture of client-server computing.
3. Explain the different types of server technologies.
4. Describe the characteristics of client-server computing.
5. Explain the functions of client-server system.
6. How server is classified in client-server in client-server computing?
7. Explain the advantages and disadvantages of client-server computing.
8. What are the roles of middle ware in client-server computing?
9. Differentiate between 1 tier, 2tier and 3 tier systems.
10. Explain the N-tiered system in client-server computing.
11. What is fat client? Describe its advantages.
12. What is thin client?
13. What is fat server? Describe it's advantages.

1.15 REFERENCES AND SUGGESTED READINGS

- Dewire, D. T. (1993). *Client/server computing*. McGraw-Hill, Inc..
- Smith, P. N., & Guengerich, S. L. (1994). Client/server computing.
- Simon, A. R., & Wheeler, T. (2014). *Open Client/Server Computing and Middleware*. Academic Press.

---×---

UNIT-2
WEB SERVER-I

UNIT STRUCTURE:

2.1 INTRODUCTION

XML-focused data exchange systems on the internet, known as web services, facilitate application-to-application (A2A) communication and interaction. These operations involve the exchange of messages, programs, documents, and objects. A notable feature of web services is their ability to enable applications written in diverse languages to communicate by exchanging data through a web service connecting clients and servers.

The digital age has transformed the way we communicate, collaborate, and access information. At the heart of this transformation lies the intricate web of web services and web servers, seamlessly orchestrating the exchange of data and facilitating the delivery of content across the vast expanse of the World Wide Web. The emergence of web services and their associated products is poised to revolutionize the software landscape in the future. This transformation is anticipated to provide significant opportunities for entrepreneurs globally, with a notable impact on the entrepreneurial ecosystem in countries like India. This chapter delves into the multifaceted realm of web services and explores the underlying functionalities of web servers, examining their composition and the crucial process of registration.

2.2 OBJECTIVES

After going through this unit learner will able to

- Understand the fundamental framework of web-services
- Learn about different types of web services
- Understand the functions and components of web services
- Learn the attributes and benefits of web services
- Understand the concept of web-server and its functionalities.
- Learn about the static web-server and dynamic web-server.
- Understand the application of web server and its security

- Learn the concept of web-server composition and registration.

2.3 WEB SERVICES: A FUNDAMENTAL FRAMEWORK

(i) Definition and Purpose

Web services, at their core, are software systems designed to enable communication and data exchange between disparate applications. The purpose of web services is to facilitate interoperability, allowing different systems to interact seamlessly, irrespective of the underlying technology stack.

The Internet represents the global connectivity of numerous computers across various networks. Within the World Wide Web, a web service serves as a standardized means for transmitting messages between client and server applications. Functioning as a software module designed for specific tasks, a web service in cloud computing can be discovered and invoked over the network, delivering anticipated functionalities to the invoking client.

Essentially, a web service constitutes a collection of open protocols and standards that facilitate the exchange of data between diverse applications or systems. It offers interoperability for software programs, written in different programming languages and operating on various platforms, to exchange data seamlessly over computer networks, such as the Internet. Any software, application, or cloud technology utilizing standard web protocols (HTTP or HTTPS) for connecting and exchanging data messages, typically in XML format, is categorized as a web service.

Web services use Simple Object Access Protocol (SOAP) to transmit XML data between applications, with the data being transferred over standard HTTP. The data sent from the web service to the application is commonly referred to as a SOAP message.

STOP TO CONSIDER

A web service is a software system created to facilitate computer-to-computer interaction over a network, independent of the specific hardware and software configurations involved.

One notable advantage of web services is their ability to enable interaction between programs developed in different languages, fostering connectivity by exchanging data over the internet. In this process, a client initiates a web service by submitting an XML request, to which the service responds with an XML-based response.

In this context, there are several crucial terms to consider. These are as follows

- Web services are not dependent on software or hardware: This implies that Web Services should function on any hardware—such as a CPU or architecture—as well as any software—such as an operating system or a programming environment. Web services can facilitate communication between (a) Java programs utilizing an Intel CPU running on Windows and (b) C# programs using Sun hardware running on UNIX operating systems. For example, in the picture above, a Java application may technically be the client, requesting a task from an ASP.NET page (a C# program). Because of this, the architecture is extremely adaptable and loosely connected.
- Computer-to-computer communication occurs via Web services: Web services are designed to facilitate communication between programs. Web services are not meant for communication between human or between human and computers. However, other forms of interaction, such as human-computer interactions, may have Web Services as their "end point."
- Over a network, a Web Service operates: This implies that while not mandatory in all cases, they are typically characterized by a distributed nature.

(ii) Traditional server-side web and Distributed computing Technologies

Certainly, if we consider the broad category of web services, distinctions arise when comparing them to other technologies.

(a) Traditional server-side web technologies, like Java Servlets/JSP, Microsoft's ASP.NET, open-source PHP, or Struts,

typically involve direct communication between clients and servers in a request-response manner. Unlike web services, these technologies are often platform-dependent, requiring specific runtime environments and combining both business and presentation logic.

(b) Distributed computing technologies, including CORBA, DCOM, and RMI, operate on an object-oriented communication model with tightly coupled interactions. In contrast, web services, being more loosely coupled, prioritize standardized communication through widely accepted web standards. This facilitates interoperability across diverse platforms and systems on the web.

STOP TO CONSIDER

Web Services are not tied to specific technologies and are independent of the underlying technology stack. Their primary objective does not involve generating HTML pages for user display, a task better suited for established web technologies like Java Servlets/JSP, ASP.NET, PHP, and others.

Put differently, the primary purpose behind the creation of Web Services was never to replace server-side Web Technologies but rather to enhance them in various ways. The responsibility of presenting diverse information to users would still be carried out by these technologies and not by Web Services. It's worth noting that, although it might be perplexing, a Web Service can be implemented using a Java Servlet or an ASP.NET page. However, in such instances, the objective shifts for these "Web Services enabled" Servlets or ASP.NET pages, transitioning from serving HTML content to delivering specific business services to the calling application, i.e., the client.

The initial method of communication between distant computers involved the use of Remote Procedure Call (RPC), enabling one computer to send a request for executing a task to another computer over a network. The term "procedure" denotes that these calls were made to a remote function, which typically was a C language function. This approach gained widespread popularity initially with

C/UNIX, later extending to C++/UNIX, and eventually spreading to various other technologies.

In the realm of Remote Procedure Call (RPC), a client procedure (equivalent to a function) would remotely invoke a server procedure (also a function). With the rise of Object-oriented systems, procedural programming witnessed a decline in its prominence. RPC, in turn, was eventually replaced by DCOM, CORBA, and RMI. These technologies gained considerable popularity in their specific domains, experiencing widespread success. Nevertheless, they facilitate binary communication between objects.

Conversely, Web Services enable communication between a client and a server through the exchange of a human-readable XML document. This approach enhances the interaction between the client and the server in a manner that is more user-friendly, although it might not be as conducive to machine processing.

Through the utilization of Web Services, clients and servers engage in the exchange of human-readable XML messages. These messages are encapsulated within a designated tag known as the Envelope, a convention specific to Web Services. It's important to note that this tagging is merely a customary practice in Web Services.

(iii)Types of Web service

There are two types of web services(Fig.2.1):

- SOAP - It is an XML-based protocol for accessing web services.
- RESTful - It is an architectural style, not a protocol.

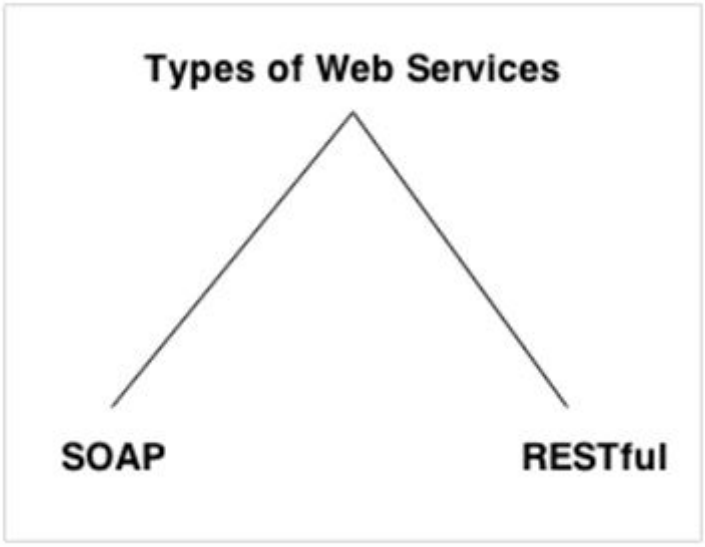


Fig. 2.1 Types of Web service

(iv) Functions of Web Services

- Either intranet or internet networks can be used to access it.
- A standardized protocol for messaging using XML.
- Irrespective of the programming language or operating system.
- By adhering to the XML standard, it possesses self-descriptive characteristics.
- It is possible to employ a straightforward location method to find it.

2.4 WEB SERVICE COMPONENTS

Numerous technologies can be used to develop and implement web services. The two most widely used implementations remain unchanged: Microsoft Web Services and Java Web Services. Implementations of Java Web Services can be found from Sun directly, from a variety of vendors, and most notably from Apache (via the use of a program known as Apache Axis). Web Services are integrated into Microsoft's.NET platform in such a way that an ASP.NET page may be quickly transformed into a Web Service.

2.4.1 SOAP (Simple Object Access Protocol)

The acronym for "Simple Object Access Protocol" is SOAP. It is a message protocol that is independent of transport. The foundation of SOAP is the transmission of XML data via SOAP messages. Every message has an attachment called an XML document. The content of the XML document is not patterned; only its structure is. The fact that all data is transmitted via HTTP, the industry-standard web protocol, is the finest thing about Web services and SOAP.

Every SOAP document must have a root element, also referred to as the element. The root element is the initial element in an XML document. There are two portions to the "envelope." The body is inserted after the header. The header contains the routing data, or the information that specifies which client the XML document should be sent to. The body will include the true message.

The following list specifies the features of SOAP:

- SOAP is a communication protocol.
- SOAP communicates between applications.
- SOAP is a format for sending messages.
- SOAP is designed to communicate via Internet.
- SOAP is platform independent.
- SOAP is language independent.
- SOAP is simple and extensible.
- SOAP allows you to get around firewalls.
- SOAP developed as a W3C standard.

2.4.2 UDDI (Universal Description, Discovery, and Integration)

UDDI, or Universal Description, Discovery, and Integration, is a standard designed for articulating, publishing, and exploring the online services of a service provider. It offers a specification that facilitates the storage of data through web services. UDDI functions as a repository where WSDL (Web Services Description Language) files can be stored, enabling client applications to locate a WSDL file and gain insights into the diverse functionalities provided by a web service. Consequently, client applications can access the

comprehensive database within UDDI, containing all relevant WSDL files.

The UDDI registry contains essential details about the online service, analogous to how a telephone directory holds the name, address, and phone number of a specific individual. This allows a client application to determine the location of the service.

2.4.3 WSDL (Web Services Description Language)

In the absence of the ability to locate a web service, it becomes inaccessible for utilization. The client initiating the web service must possess knowledge about the web service's location. Additionally, the client application needs a clear understanding of the web service's functionality to invoke the appropriate one. This is achieved through the use of WSDL, or Web Services Description Language. The WSDL file, being another XML-based document, elucidates the web service's functionalities to the client application, ensuring that it comprehends both the location and usage procedures of the web service.

Some Important elements used in Web Services Description language are as follows:

- <message>: The message element in WSDL is used to define all different data elements for each operation performed by the web service.
- <portType>: The port type element is used to determine the operation which can be performed by the web service. This operation can have two messages one is input and the second one is the output message.
- <binding>: This element contains the used protocol.

2.5 FUNCTIONING MECHANISM OF A WEB SERVICE

An extremely basic illustration in Fig. 2.2 of how a web service might operate is shown in the diagram. Requests are used by the client to send a series of web service calls to the server hosting the web service.

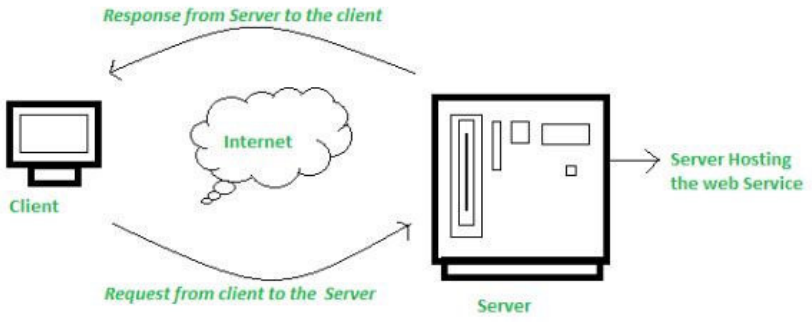


Fig.2.2 The Operational Process of a Web Service.

Requests are made using remote procedure calls, where calls to the methods hosted by the respective web service are referred to as Remote Procedure Calls (RPC). For instance, Flipkart provides a web service that presents prices for items listed on Flipkart.com. While the front end or presentation layer may be developed in .Net or Java, communication with the web service is possible using either programming language.

The critical element in the design of a web service is the exchanged data between the client and the server, which is in XML format. XML, or Extensible Markup Language, serves as a straightforward intermediary language comprehensible to diverse programming languages and operates as a counterpart to HTML. Consequently, when programs engage in communication, they utilize XML, establishing a universal platform for applications developed in various programming languages to interact with one another.

Web services utilize SOAP (Simple Object Access Protocol) to transmit XML data between applications. The transmission of data occurs through standard HTTP. A SOAP message represents the information sent from the web service to the application, containing nothing more than an XML document. The client application, making the web service call, can be developed in any programming language, as the content is composed in XML.

2.6 ATTRIBUTES OR TRAITS OF WEB SERVICES

The following characteristics are associated with web services.

- **XML Based**

XML is used by a web service's record transportation and information representation layers. When using XML, no operating system, platform, or networking is required. Applications based on online offerings are very compatible at the middle level.

- **Loosely Coupled**

An internet service provider's customer isn't always associated with that service provider. A web service provider's user interface is subject to change over time without affecting the user's ability to communicate with the provider. Decisions made by the customer and server are tightly linked in a strongly coupled system, meaning that any modifications made to one interface also need to be made to the other.

Loosely connected architecture facilitates simpler integration between various structures and makes software systems easier to maintain.

- **Capability to be Synchronous or Asynchronous**

The client's relationship to the function's execution is referred to as synchronization. In order to resume synchronous invocations, the client must wait for the service to finish operating after being blocked. A client can initiate one job and go on to other tasks thanks to asynchronous operations.

When the service is finished, synchronous clients receive their effects instantly, whereas asynchronous clients receive their outcomes later. Weakly coupled systems cannot function without asynchronous capabilities.

- **Coarse-Grained**

Java and other object-oriented systems make their services accessible via distinct methods. A character technique is simply too fine an operation to be of any value at the corporate level. Developing many fine-grained strategies from the ground up is necessary when building a Java application. These strategies are then integrated to create a rough-grained provider that is used by a buyer or a service.

Both corporations and the interfaces they expose ought to be coarse-grained. A simple method for defining coarse-grained services with sufficient access to commercial enterprise logic is web services generation.

- **Supports Remote Procedural Call**

Through web services, users can call functions, methods, and procedures on remote objects using an XML-based protocol. The input and output framework that distant systems present must be supported by a web service.

Development of components for the entire organization JavaBeans (EJBs) and.NET Components have been more common in enterprise and architectural deployments during the past few years. Both technologies are allocated and accessed via various RPC mechanisms.

A web function can translate incoming invocations into an EJB or.NET component invocation, or it can provide its own services akin to those of a traditional role in order to facilitate RPC.

- **Supports Document Exchanges**

The ease with which XML can communicate with complicated entities and data is one of its most attractive qualities. A request for a quote or a whole book can be included in these records, or they can be as straightforward as a conversation with the user's present address. Web administrations make it easier to exchange an archive, which helps with reconciliation.

2.7 BENEFITS OF WEB SERVICES

Employing web services offers the subsequent benefits, these are as follows

- **Business operations can be made accessible via the Internet**

A web service is a part of managed code that provides functionality to end users or client apps. This feature is accessible using the HTTP protocol, allowing users to access it from any location on the internet. Web services are becoming more and more valuable because all apps can now be accessed online. Web services are becoming more and more valuable because all apps can now be accessed online. In other words, the online service has the necessary capabilities and can be found anywhere on the internet.

- **Interoperability**

Web administrators facilitate communication and the sharing of services and information between various apps. Web services can also be used by other programs. For instance, a.NET program can speak with Java web administrators and vice versa. Web administrators are employed to make the innovation and application stage self-contained.

- **Cost-effective communication**

Web services can be implemented using your current inexpensive internet connection since they use the SOAP over HTTP protocol. In addition to SOAP over HTTP, other robust transport protocols, including FTP, can also be used to construct web services.

- **A universally comprehensible standard protocol**

Web services exchange information using a standard industry protocol. All four layers (Service Transport, XML Messaging, Service Description, and Service Discovery) of the web services protocol stack employ well-defined protocols.

- **Reusability**

Multiple client apps can use the same web service at the same time.

2.8 WEB SERVER FUNCTIONALITY

A web server operates as a computer system dedicated to storing, managing, and distributing files for websites to web browsers. It consists of both hardware and software elements, employing the Hypertext Transfer Protocol (HTTP) to respond to requests made by internet users via the World Wide Web. Using this protocol, web servers retrieve and deliver the requested webpage to the user's browser, such as Google Chrome. Furthermore, for handling files intended for email or storage, web servers make use of the Simple Mail Transfer Protocol (SMTP) and the File Transfer Protocol (FTP).

So, what comprises a web server? From a hardware perspective, a web server establishes a connection to the internet, enabling the sharing of data or files with other connected devices. The shared data encompasses various forms, including HTML files, images, JavaScript files, and CSS style sheets. Notably, web server hardware includes the corresponding web server software.

The web server software regulates how users on the web access the hosted files. It consists of multiple components, with one key element being the HTTP server. This component is a software entity capable of interpreting HTTP queries and understanding URLs.

(i) Roles of a Web Server

The roles of a web server can be categorized into various functions.

- Storing and dispensing web content:
  The main purpose of a web server is to store and provide web content, including web pages, images, videos, and various files, to clients upon their request. This process encompasses handling incoming requests, fetching the requested content from the server's storage devices, and

delivering the content back to the client in the format of an HTTP response.

- Managing HTTP requests:
  A web server manages requests made through the Hypertext Transfer Protocol (HTTP) from clients or users. HTTP is the established protocol for communication between clients and web servers. When a client seeks a web page or other content, the request is transmitted to the web server via the internet. Subsequently, the web server handles the request, fetches the required content from its storage devices, and transmits it back to the client in the structure of an HTTP response.
- Producing responses to HTTP requests:
  A web server produces responses to HTTP requests by executing web applications that dynamically generate content, considering user input and various factors. For instance, in the case of an e-commerce platform, a web application may dynamically generate web pages displaying product details, pricing, and availability based on user searches or selections.
- Overseeing and sustaining web applications:
  A web server administers and sustains web applications offering diverse services like e-commerce, social networking, and content management. This encompasses tasks such as installing, configuring, and updating web application software, overseeing performance, and resolving any issues that may occur.
- Providing simultaneous support to multiple clients:
  A web server caters to numerous clients or users concurrently by managing multiple HTTP requests simultaneously. This is accomplished through the implementation of multi-threading or other methodologies that enable the server to process requests concurrently.

(ii) Web Server Architecture

Web server architecture defines the structure, components, and configurations necessary for its efficient functioning and the delivery of web-based services to users. It represents a crucial aspect of managing web servers, significantly impacting their performance, reliability, and security.

- Concurrent Approach or Multi-Tier Architecture:
  In the simultaneous approach, for every client request, the web server generates an individual process or thread. Each process or thread manages a single request at a time, enabling the concurrent processing of multiple requests.
  As an illustration, consider a scenario where a web server simultaneously receives ten client requests. Under the concurrent approach, the server will initiate ten distinct processes or threads, each dedicated to handling an individual request concurrently. This methodology is commonly employed in conventional web servers such as Apache, known for its capacity to manage a substantial volume of simultaneous connections.

- Single-Process-Event-Driven Approach or Single-Tier(Single Server) Architecture:
  In the single-process-event-driven approach, a web server employs a solitary process or thread to manage all client requests. The server remains alert for events, such as new connection requests and incoming data, addressing them one by one in a non-blocking fashion.
  As an illustration, imagine a scenario where a web server receives ten client requests simultaneously. In the single-process-event-driven approach, the server will utilize a solitary process or thread to sequentially manage all ten requests in a non-blocking manner. This strategy is frequently implemented in contemporary web servers, such as Node.js, specifically crafted to efficiently handle a substantial volume of lightweight connections.

STOP TO CONSIDER

In single-tier architecture, a lone server handles both the processing of requests and the delivery of web content. This configuration is well-suited for smaller websites or

2.9 TYPES OF WEB SERVERS

The following are a few of the most popular kinds of web servers:

- Apache HTTP Server:

The open-source Apache HTTP Server is a web server program that is extensively utilized globally. It is renowned for its exceptional performance, dependability, and adaptability and is compatible with a large number of operating systems, such as Windows, Linux, and macOS.

- Microsoft Internet Information Services (IIS):
  IIS, developed by Microsoft, is web server software specifically crafted to operate on Windows operating systems. Renowned for its scalability, security attributes, and compatibility with Microsoft technologies like ASP.NET and Microsoft SQL Server.
- Nginx:
  Nginx is an open-source web server software recognized for its high performance, speed, scalability, and user-friendly interface. Frequently employed as a reverse proxy server or load balancer, it has the capability to serve both static and dynamic content.
- Lighttpd:
  Lighttpd is open-source web server software known for its lightweight nature, engineered for optimal performance with minimal resource utilization. It acknowledged for its speed, stability, and security features.

2.10 FUNCTIONING PROCESS OF A WEB SERVER?

The ability to view a page on the internet occurs when the browser requests the page from the web server, and the web server responds by providing that specific page. The following figure illustrates a basic diagrammatic representation of this process(Fig.2.4).

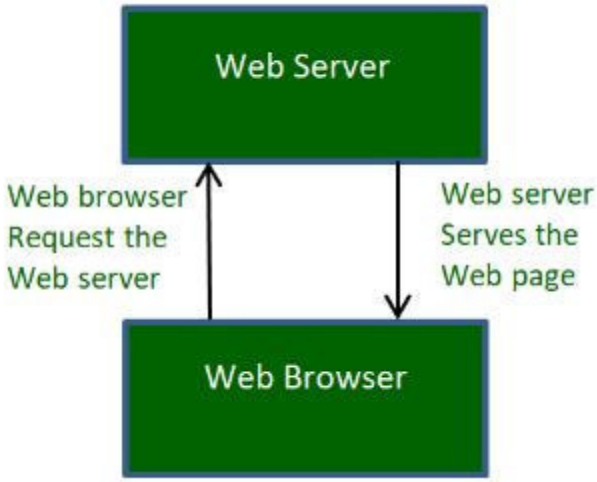


Fig.2.4 Web server and Web browser

Here is a fundamental overview of the operations of a web server shown in Fig.2.5.

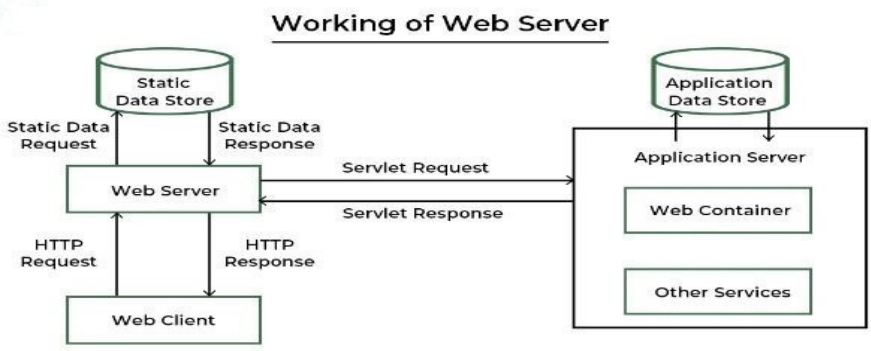


Fig.2.5 Web server basic operation

- Obtaining the IP Address from the domain name: The web browser initially acquires the IP address associated with the domain name (such as www.nameofwebsite.org). This IP address retrieval can occur through two distinct methods. The IP address is obtained through two methods: either by checking for it in the cache or by making a request to DNS Servers.
- Browser requests the full URL: Upon acquiring the IP address, the browser then requests the complete URL from the web server.

- Web Server Responds to the request: The web server replies to the browser by delivering the requested pages. If the pages are not found or an error occurs, the web server sends the relevant error message.
  For instance, you might have encountered the Error 404 message when attempting to access a webpage, indicating that the page is nonexistent. Another frequent example is Error 401, signifying denied access due to incorrect credentials such as username or password provided by the user.
- The browser displays the web page: Ultimately, the browser retrieves the webpages and presents them or displays the appropriate error message.

This procedure commonly encompasses various communication protocols layers, which include HTTP (Hypertext Transfer Protocol), TCP/IP (Transmission Control Protocol/Internet Protocol), and DNS (Domain Name System).

Web servers have the capability to execute additional tasks, including managing databases, overseeing sessions, and conducting security validations to guarantee the secure and effective delivery of requested resources.

2.11 STATIC WEB SERVER Vs DYNAMIC WEB SERVER

There are two categories of web servers, namely static and dynamic, which vary in their methods of delivering web content.

Static web servers provide pre-existing web pages that remain unchanged in real-time. Typically authored in HTML and CSS, these pages are stored in the file system of the web server. Upon a client device requesting a page, the web server retrieves the specified file and transmits it back to the client as a response.

Conversely, dynamic web servers produce web pages dynamically in response to user requests. These servers utilize programming languages such as PHP, Python, and Ruby to construct web pages from dynamic data sources like databases and APIs. Upon receiving a page request from a

client device, the server executes the required code to generate the page and delivers it back to the client as a response.

2.12 APPLICATIONS OF WEB SERVER

Web servers find extensive applications and purposes, serving as the fundamental infrastructure of the internet and the mechanism through which web resources are dispensed to users. Below are several prevalent applications of web servers:

- Hosting websites:
  Web servers are employed for hosting websites, ensuring their accessibility to users globally. They deliver HTML, CSS, JavaScript, and various other web resources that constitute the components of web pages.
- Running web applications:
  Web servers are utilized for hosting web applications that operate within the browser, including online shopping carts, social media platforms, and productivity tools.
- Streaming media:
  Web servers are employed to stream audio and video content online, facilitating users in accessing multimedia resources such as music and movies.
- Managing online databases:
  Web servers find application in overseeing online databases, enabling users to access and modify data through interfaces on the web.
- Cloud computing:
  Web servers play a role in driving cloud computing services, granting users internet-based access to computing resources and data storage.
- Internet of Things (IoT):
  Web servers are employed to enable the functioning of Internet of Things (IoT) devices, facilitating their communication and data exchange via the internet.
- File sharing:
  Web servers are utilized for distributing files online, enabling users to retrieve and download various types of files, including documents, images, and videos.

2.13 WEB SERVER SECURITY

Security for a web server pertains to the methods and procedures employed to safeguard it against unauthorized access, malicious attacks, and potential security risks. An adeptly secured web server guarantees the safeguarding of sensitive data and resources, preventing unauthorized access and upholding the confidentiality, integrity, and availability of web assets.

Below are several standard security precautions for web servers:

- Regular software updates:
  Ensuring the web server software and applications are regularly updated is crucial to addressing potential security vulnerabilities or flaws.
- Secure configuration:
  Configuring the web server with robust security settings is imperative, including secure file permissions, SSL/TLS encryption, and reliable authentication mechanisms.
- Access control:
  Configure the web server to limit access to sensitive data and resources, encompassing actions like password protection for directories and implementation of firewall rules.
- Security testing:
  Routine security testing and vulnerability scanning need to be conducted to discover and resolve potential security threats.
- Monitoring and logging:
  Observing the web server logs and activities aids in identifying and responding promptly to security incidents.
- Backup and recovery:
  Conduct periodic backups of both the web server and data to guarantee swift recovery in the event of data loss or other security incidents.

2.14 BENEFITS OF WEB SERVER

Utilizing web servers provides numerous benefits, such as:

- Scalability: Web servers are capable of managing a considerable volume of concurrent connections, rendering

them well-suited for websites experiencing high levels of traffic.

- Reliability: These servers are crafted for uninterrupted functioning and have the ability to gracefully recover from failures.
- Security: Security features are integrated into web servers to guard against typical web threats such as DDoS attacks and SQL injection.
- Customization: Configurations of web servers can be customized to meet the specific requirements of an application.

2.15 WEB SERVER COMPOSITION

Web server composition refers to the process of creating and organizing a web server by combining various components and modules to meet specific requirements and functionalities. This composition is crucial for building a robust and efficient web infrastructure capable of handling diverse tasks and serving the needs of users. Here are key points to consider regarding web server composition:

- Module Integration:

Web servers are composed of various modules, each responsible for specific functions such as handling HTTP requests, managing security, processing dynamic content, or interfacing with databases. The integration of these modules ensures a comprehensive and well-rounded server setup.

- HTTP Server Component:

A fundamental element of web server composition is the inclusion of an HTTP server component. This module manages the basic communication protocols, handling incoming requests from clients and delivering the requested web content.

- Security Modules:

Web servers often include security modules to protect against common threats such as unauthorized access, DDoS attacks, and data breaches. Security features may involve

firewalls, encryption protocols, and authentication mechanisms to safeguard the server and its hosted content.

- Load Balancing:

For web servers dealing with high traffic, load balancing modules are integrated to distribute incoming requests across multiple servers. This ensures efficient resource utilization and prevents server overload.

- Content Management Systems (CMS):

Some web servers incorporate content management systems to facilitate the creation, modification, and organization of digital content. CMS modules simplify website management and enable collaborative content development.

- Database Integration:

Web servers often interact with databases to retrieve and store dynamic content. Integration with database modules allows seamless communication between the web server and database systems, supporting dynamic and data-driven web applications.

- Scripting and Programming Support:

Web server composition includes support for various scripting and programming languages, enabling developers to create dynamic and interactive web pages. Common languages include PHP, Python, Ruby, and others.

- Caching Mechanisms:

To enhance performance and reduce latency, web servers incorporate caching mechanisms. These modules store frequently accessed content, reducing the need to generate it repeatedly and improving response times for users.

- Logging and Monitoring:

Effective web server composition includes logging and monitoring modules to track server activities, identify potential issues, and generate performance metrics. These

features are crucial for maintaining server health and troubleshooting.

- Scalability:

Web servers should be composed with scalability in mind. The architecture should allow for easy expansion to accommodate growing user demands and increased workloads. This may involve the use of scalable server configurations and cloud-based solutions.

- Configuration Management:

Web server composition involves configuring and managing various parameters to optimize performance, security, and resource utilization. This includes settings related to server protocols, security policies, and performance tuning.

2.16 WEB SERVER REGISTRATION

Web server registration is a process by which a web server is formally documented and identified within a network or on the internet. This registration is essential for establishing a structured and organized web infrastructure, enabling effective communication, and supporting various internet-related activities. The following are key aspects and considerations related to web server registration:

- Information Gathering:

The registration process begins with the collection of crucial details about the web server. This includes obtaining information such as the server's IP address, domain name, geographical location, and any other relevant identifying data.

- Registry or DNS Provider Selection:

Depending on the specific requirements and context, a decision is made regarding the appropriate registry or Domain Name System (DNS) provider. This entity will maintain and store the registration information for the web server.

- Creating an Account:

If using an external registry or DNS provider, individuals or organizations typically need to create an account or log in to an existing account. This account is used for managing and updating the web server's registration details.

- Accessing Registration Interface:

The chosen registry or DNS provider offers a registration interface where users can initiate and complete the registration process. This interface may involve filling out forms and providing the necessary server details.

- Entering Server Information:

Users input specific information about the web server into the registration form. This usually includes the server's name, IP address, domain name, and administrative contact information.

- Verification of Information:

Accuracy is crucial in web server registration. Users should carefully review and verify all entered information to ensure that it is correct. Inaccurate details can lead to communication issues and hinder the server's accessibility.

- Agreeing to Terms and Conditions:

Users typically review and agree to any terms and conditions stipulated by the registry or DNS provider. This step may involve acknowledging legal responsibilities and obligations associated with the registration.

- Submission of Registration Form:

After completing the necessary fields and agreeing to terms, users submit the registration form through the provided interface. This action initiates the official registration of the web server.

- Payment (if applicable):

Depending on the registration context, there may be associated fees. Users are required to complete any payment processes according to the guidelines established by the registry or DNS provider.

- Confirmation and Record Keeping:

Following a successful registration, users receive confirmation details. It is essential to save or record this information for future reference and to maintain an accurate record of the server's registration.

- Regular Updates:

To ensure continued accuracy, users should periodically review and update the registered information. This is especially important when there are changes to the server's details or when the registration expires.

Check Your Progress

State True or False

(i) Web services use Simple Object Access Protocol (SOAP) to transmit XML data between applications.
(ii) Web site allows user to view a webpage.
(iii) Nginx is open-source web browser software.
(iv) Static web servers provide pre-existing web pages that remain unchanged in real-time.
(v) To enhance performance and reduce latency, web servers incorporate caching mechanisms.

2.17 SUMMING UP

- This chapter provides a comprehensive exploration of web services, web server functionality, composition, and the critical process of registration.

- By understanding the fundamental components of web-services, one can grasp the intricate web that underlies our digital experiences, shaping the present and future landscape of the World Wide Web.

- The architecture of web servers is a fundamental component of the internet's structure, enabling the distribution of web content and applications. Their primary role is to ensure broad access to information and services for a worldwide audience.
- A comprehensive understanding of how web servers operate their structure and security measures is highly significant for web developers, IT professionals, and individuals interested in the intricacies of the online domain.
- A web server is a fundamental component of the internet infrastructure, responsible for hosting and serving web content.

- This section provides a comprehensive definition of web servers and elucidates their crucial role in responding to client requests, managing connections, and delivering web pages or services.

2.18 ANSWER TO CHECK YOUR PROGRESS

(1) (i) True (ii) False (iii) False (iv) True (v) True

2.19 POSSIBLE QUESTIONS

(1) What are the advantages of having XML based Web services?
(2) Differentiate between traditional server-side webs and distributed computing Technologies**.**
(3) What is the purpose of WSDL in a web service?
(4) What is DOM?
(5) How response is sent in XML-RPC?
(6) What is Remote Procedure Call (RPC)?
(7) Explain the core layer in Web Service Protocol Stack.
(8) Explain the security issues with web services.
(9) Explain the functions of web-server protocol.
(10) Describe the architecture of web-server.

(11) Explain the different types of web-servers.
(12) Explain the process of web-server registration.

2.20 REFERENCE AND SUGGESTED READINGS

1. Godbole, A.S. and Kahate A. *Web Technologies*. TATA McGRAW HILL.2011

2. https://www.geeksforgeeks.org/what-is-a-web-server-working-and-architecture/

---×---

UNIT-3
WEB SERVER-II

UNIT STRUCTURE:

3.1 INTRODUCTION

In the ever-evolving landscape of the digital world, understanding the fundamental building blocks of web communication is paramount. This chapter delves into the basics of HTTP, IP

addresses, DNS, and Ports – the essential components that form the backbone of the internet. As we embark on this exploration, we aim to demystify the core concepts that facilitate seamless data exchange and communication across the World Wide Web.

3.2 OBJECTIVES

After going through this unit learner will able to

- Understand the basics of HTTP and its request methods
- Understand the concept of IP address and its different types.
- Learn about the DNS and its working
- Understand the concept of classification of IP addresses
- Understand the concept of web server and its architecture.
- Learn about different types of ports and their uses.

3.3 BASIC OF HTTP

Hypertext transfer protocol, or HTTP, is the protocol used to send data over the Internet. The HTTP protocol is used to retrieve resources, including HTML documents. It is the basis for all data interchange on the Internet and is a client-server protocol, meaning that the Web browser, or other recipient, initiates requests.

The protocol is text-based and uses client-server communications architecture, similar to the majority of Internet protocols.

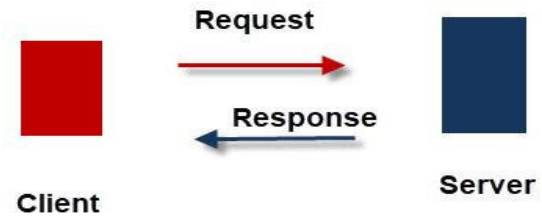


Fig.3.1 HTTP Protocol Basics

The HTTP protocol is used to retrieve resources, including HTML documents. It is the basis for all data interchange on the Internet and is a client-server protocol, meaning that the Web browser, or other recipient, initiates requests. The various sub-documents that are

retrieved—such as text, layout descriptions, photos, videos, scripts, and more—are assembled into a single document.

Instead of exchanging a stream of data, clients and servers communicate by sending and receiving individual messages; requests are issued by the client, which is often a Web browser, and responses are sent by the server in response.

Developing over time, HTTP is a flexible protocol known for its extensibility. It operates at the application layer and is transmitted over TCP or a TLS-encrypted TCP connection, with the potential to use any reliable transport protocol. Beyond fetching hypertext documents, it is utilized for retrieving images and videos, as well as posting content to servers, such as HTML form results. Additionally, HTTP can be employed to obtain specific document parts, facilitating on-demand updates to web pages.

3.3.1 System Elements Based on HTTP

The user-agent (or a proxy acting on its behalf) is the only entity that sends requests across the client-server HTTP protocol. The user-agent is often a web browser.

Every request is routed to a different server, which responds to it by providing an answer known as the response. Numerous entities commonly referred to as proxies; exist between the client and the server. They carry out various functions, such as serving as gateways or caches.

In actuality, there are more computers—routers, modems, and other devices—between a browser and the server processing the request. These are concealed within the network and transport layers of the Web because of its tiered architecture. At the application layer, HTTP is the top protocol. While crucial for identifying network issues, the underlying layers are largely unimportant for understanding HTTP.

(i) Client: the user-agent

The user-agent is any tool that acts on behalf of the user, commonly fulfilled by the Web browser; however, it can also be executed by

applications utilized by engineers and web developers for debugging their applications. The request is consistently initiated by the browser. For the presentation of a webpage, the browser initiates an initial request to retrieve the HTML document that signifies the page.

(ii) The Web server

Situated on the other end of the communication channel is the server, responsible for providing the document as per the client's request. Although a server may seem like a singular entity, it can, in reality, be a group of servers distributing the load through load balancing. Alternatively, it might involve additional software components like caches, a database server, or e-commerce servers, either fully or partially generating the document on request.

(iii) Proxies

The HTTP messages are sent by multiple computers and machines between the Web browser and the server. The majority of them function at the transport, network, or physical levels of the Web stack due to its tiered structure; they become transparent at the HTTP layer and may have a major effect on performance. Proxies are typically those who work at the application layers. These can be non-transparent, meaning they will modify the request before sending it to the server, or transparent, meaning they will forward requests without changing them in any way. Proxies can serve a variety of purposes:

- caching (the cache can be public or private, like the browser cache)
- filtering (like an antivirus scan or parental controls)
- load balancing (to allow multiple servers to serve different requests)
- authentication (to control access to different resources)
- logging (allowing the storage of historical information)

3.3.2 Fundamental Elements of HTTP

(i) Connectionless

HTTP operates in a connectionless manner. When the HTTP client launches the browser, it instigates an HTTP request. Following the request, the client disconnects from the server and awaits the

response. Once the response is prepared, the server re-establishes the connection to deliver the response to the client, after which the client terminates the connection. Therefore, both the client and server are aware of each other only during the current request and response phase.

(ii) Media Independent

HTTP is not limited to a specific type of media. It has the capability to transmit various forms of data, provided that both computers involved can interpret it.

(iii) Stateless

HTTP operates in a stateless manner, meaning that the client and server are only aware of each other for the duration of the ongoing request. If the connection is terminated, and two computers intend to reconnect, they must exchange information again, treating the connection as if it were the initial one.

STOP TO CONSIDER

- HTTPS stands for Hypertext Transfer Protocol Secure. HTTPS has a secure transfer.

3.3.3 HTTP Request Methods

(i) GET

This method uses a provided URL to retrieve data from the specified server; a GET request can be used to obtain the data. Other effects cannot be applied to the data by it.

(ii) HEAD

The HEAD method is identical to the GET method, serving the purpose of transmitting solely the status line and header section.

(iii) POST

The POST request transmits data to the server, such as file uploads, customer information, etc., through HTML forms.

(iv) PUT

The PUT method is employed to transmit a document from the server to the client.

(v) DELETE

The DELETE method is utilized to eliminate all existing representations of the specified resource identified by the URL.

(vi) CONNECT

The CONNECT method is employed to create a tunnel to the server identified by a provided URL.

3.4 IP ADDRESS

Communication between all the computers worldwide on the Internet network occurs through underground or underwater cables or wirelessly. To perform actions like downloading a file or loading a webpage on the internet, my computer needs to possess an address. This address enables other computers to locate and find mine to deliver the requested file or webpage. In technical terms, this address is referred to as an IP Address or Internet Protocol Address.

Illustrating with another example, consider the process of sending mail – one must possess your home address for the mail to reach you. Similarly, a computer requires an address so that other computers on the internet can communicate without the risk of sending information to the wrong destination. Hence, each computer globally is assigned a unique IP Address, serving as a distinct identifier for computers or nodes on the internet. This address is a numerical string presented in a specific format, commonly as a set of numbers like 192.155.12.1. Each number in the set falls within the range of 0 to 255. In essence, an IP address spans from 0.0.0.0 to 255.255.255.255, and these addresses are allocated by IANA, also known as the Internet Corporation for Assigned Names and Numbers.

3.4.1 Types of IP Address

There are two kinds of IP addresses:

(i) IPv4 Addresses (Internet Protocol version 4)

A 32-bit address known as an IPv4 address is used to uniquely identify a device's connection to the internet, such as a computer or router.

There are four numbers in all, divided by dots. Each decimal number can range from 0 to 255. However, computers convert decimal numbers—which only have two possible values—to binary numbers, which only have two possible values: 0 and 1. Thus, this (0-255) range can be expressed in binary as (00000000 – 11111111). For every integer N, a set of eight binary digits can be used to represent it. Therefore, 32 bits of binary digits can represent an entire IPv4 binary address. Since each machine is assigned a distinct set of bits under IPv4, a total of around 2,342,967,296 devices can be assigned IPv4.

STOP TO CONSIDER

The address space of IPv4 is 2^{32} or 4,294,967,296

Two commonly used notations exist for representing an IPv4 address-binary notation and dotted-decimal notation

- Binary Notation

The IPv4 address is represented in binary notation with 32 bits, and each octet is commonly called a byte. It is frequently described as a 32-bit or 4-byte address. An illustration of an IPv4 address in binary is provided below.

01110101 10010101 00011101 00000010

- Dotted-Decimal Notation

To enhance readability, IPv4 addresses are often expressed in decimal form, employing a dotted-decimal notation with a dot between the bytes. The given address can be represented in dotted-decimal format as follows.

117.149.29.2

In Figure 2, an IPv4 address is presented in both binary and dotted-decimal formats. It's essential to recognize that since each byte (octet) consists of 8 bits, every numerical value in dotted-decimal notation falls within the range of 0 to 255.

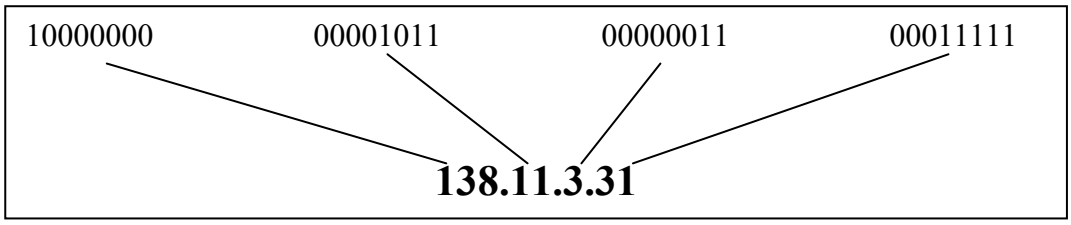


Fig.2. The IPv4 address is represented using both dotted-decimal notation and binary notation.

- Classful Addressing

IPv4 addressing initially employed the concept of classes, known as classful addressing. Classful addressing divides the address space into five classes, namely A, B, C, D, and E, with each class taking up a distinct portion of the address space. Determining the class of an address is possible by examining the binary or dotted-decimal notation. In binary notation, the initial bits directly indicate the class, while in decimal-dotted notation; the class is identified by the first byte. Both approaches are illustrated in Fig. 3 and Fig.4.

	First Byte	Second Byte	Third Byte	Forth Byte
Class A	0-127			
Class B	128-191			
Class C	192-223			
Class D	224-239			
Class E	240-255			

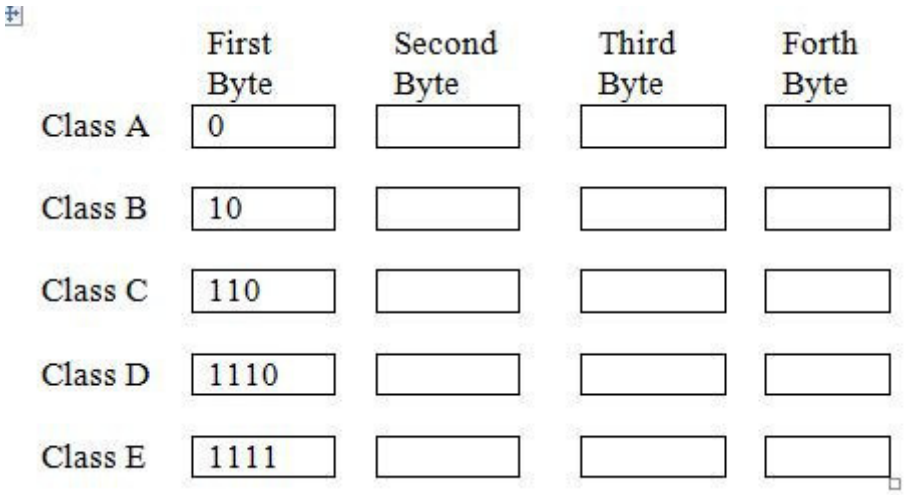


Fig.3.2 Determining the classes using binary notation

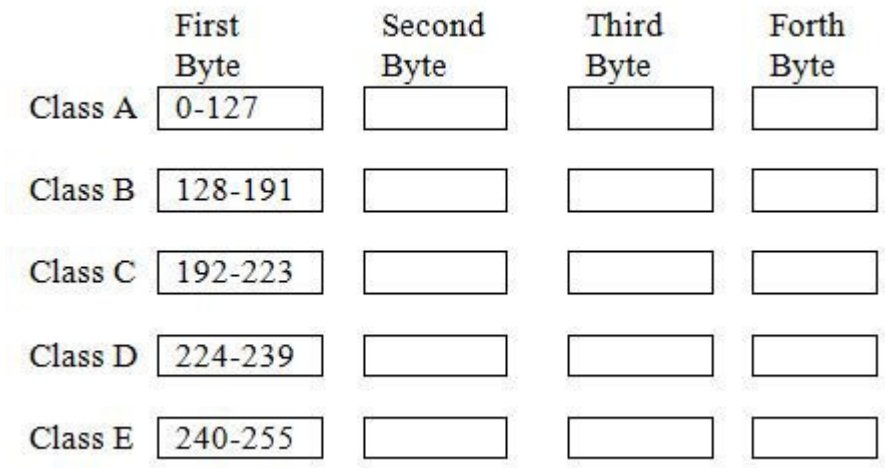


Fig.3.3 Determining the classes using dotted-decimal

- Classes and Blocks

As indicated in Table 3.1, one issue with Classful addressing is that each class is split up into a certain number of blocks, each of which has a defined size.

Let's look at the table. An organization used to be able to request a block of addresses and get one in class A, B, or C. Large enterprises with lots of connected servers or routers were the target audience for class A addresses. For midsize businesses with tens of thousands of connected hosts or routers, class B addresses were created. Small businesses with a few connected hosts or routers were the target market for class C addresses.

Table 3.1: Block size and number in Classful IPv4 addressing

Class	Number of Blocks	Block Size	Application
A	128	16,777,216	Unicast
B	16,384	65,536	Unicast
C	2,097,152	256	Unicast
D	1	268,435,456	Multicast
E	1	268,435,456	Multicast

The flaw in this design becomes apparent when considering the size of address blocks within Class A, which are excessively large for nearly all organizations, resulting in significant address wastage. Similarly, Class B address blocks are also overly spacious for many recipients, while Class C blocks are likely too small for numerous organizations.

Class D addresses were specifically created for multicasting, a concept that will be elaborated on in subsequent chapters. Each address within this class serves to designate a distinct group of hosts on the Internet. However, the Internet authorities inaccurately anticipated a requirement for 268,435,456 groups, which never materialized, leading to considerable address wastage. Additionally, class E addresses were reserved for potential future use; despite a minimal number being utilized, it resulted in further address wastage.

STOP TO CONSIDER

In Classful addressing, a significant portion of the available addresses went unused.

- Netid and Hostid

In Classful addressing, an IP address within class A, B, or C is segmented into netid and hostid components, each with different lengths determined by the class of the address. Illustrated in Fig.3 are examples of netid and hostid bytes. The netid for class A is the First byte, Class B is the First and Second byte, and Class C is the

First, Second and Third byte and the remaining byte for each class considered as hostid . It's important to note that this segmentation concept does not extend to classes D and E.

In class A, the netid is determined by one byte, while the hostid is determined by three bytes. Class B assigns two bytes for the netid and two bytes for the hostid. In class C, the netid is defined by three bytes, while the hostid is defined by one byte.

- Mask

While the lengths of netid and hostid (in bits) are fixed in classful addressing, we can employ a mask, also known as the default mask, which consists of a 32-bit sequence comprising contiguous 1s followed by contiguous 0s. Table 3.2 displays the masks for classes A, B, and C. It's worth noting that this concept is not applicable to classes D and E.

Table 3.2 The standard masks used in Classful addressing.

Class	Binary	Dotted-Decimal	CIDR
A	**11111111** 00000000 00000000 00000000	**255**.0.0.0	/8
B	**11111111 11111111** 00000000 00000000	**255.255**.0.0	/16
C	**11111111 11111111 11111111** 00000000	**255.255.255**.0	/24

The mask aids in identifying both the netid and hostid. For instance, in a class A address, where the mask comprises eight consecutive 1s, the initial 8 bits of any address within class A delineate the netid, while the subsequent 24 bits delineate the hostid.

The final column of Table 2 displays the mask represented as "/n" where 'n' can take values of 8, 16, or 24 within Classful addressing. This representation, also known as slash notation or Classless Interdomain Routing (CIDR) notation, is utilized in classless addressing, a topic to be discussed subsequently. Its introduction here is warranted as it can also be employed in Classful addressing. It will be demonstrated later that Classful addressing serves as a particular instance within classless addressing.

- Subnetting

In the period of Classful addressing, subnetting emerged as a practice. When an organization received a substantial block in class A or B, it had the option to partition the addresses into multiple consecutive groups and allocate each group to smaller networks known as subnets. In some uncommon instances, organizations could even share a portion of the addresses with neighbouring entities. Subnetting involves augmenting the number of 1s in the mask, as we will delve into further when addressing classless addressing later on.

- Supernetting

As the depletion of class A and class B addresses became prevalent, there arose a significant demand for medium-sized address blocks. The limited size of class C blocks, each accommodating a maximum of 256 addresses, proved insufficient for the majority of organizations, including midsize ones. Consequently, supernetting emerged as a solution. This technique allows organizations to merge multiple class C blocks to form a larger address range, effectively creating a supernetwork or supernet. By consolidating several networks, an organization can obtain a contiguous set of class C blocks instead of just one, enabling them to meet their address requirements. For instance, an organization requiring 1000 addresses might receive four consecutive class C blocks, which they can amalgamate to establish a single supernetwork. Supernetting reduces the number of 1s in the mask. For example, if an organization is allocated four class C addresses, the mask shifts from /24 to /22. It's worth noting that classless addressing renders supernetting unnecessary.

- Classless Addressing

To address the issue of address depletion and expand Internet access to more organizations, classless addressing was developed and put into practice. Under this system, traditional classes are eliminated, but addresses are still allocated in blocks.

In classless addressing, when an individual or organization, regardless of its scale, requires Internet connectivity, they are assigned a block of addresses. The size of this block, which dictates the number of addresses within it, depends on the nature and scale

of the entity. For instance, a household might receive only a couple of addresses, while a large corporation could be allocated thousands. Similarly, an Internet Service Provider (ISP) may be granted thousands or even hundreds of thousands of addresses, depending on the number of customers it serves.

(ii) IPv6 Addresses

The limitation of IPv4 addresses has become apparent as the number of devices seeking connectivity to the internet far exceeds the approximately 4 billion unique addresses IPv4 can provide. As a solution, IPv6 addresses, which consist of 128 bits represented in a human-readable format as eight groups of hexadecimal numbers separated by colons, are being adopted. In computer-friendly form, IPv6 addresses are expressed as sequences of 0s and 1s. This expanded addressing scheme allows for an astronomical number of unique addresses, theoretically accommodating all current and future internet-connected devices with (2^{128}) possible combinations, ensuring scalability for generations to come.

IPv6 can be written as:

2011:0bd9:75c5:0000:0000:6b3e:0170:8394

3.4.2 Classification of IP Address

An IP address is categorized into various types:

1.9.1 Public IP Address:

This address is accessible to the public and is allocated by your network provider to your router, which then subdivides it for your devices. There are two types of public IP addresses.

- Dynamic IP Address:

When you link a smartphone or computer to the internet, your Internet Service Provider assigns you an IP Address from its pool of available addresses. With this address, your device gains connectivity to the internet, enabling data transmission both to and from your device. Subsequent connections with the same device result in the provider issuing different IP Addresses each time, drawn from the same available range. This variability in IP Address upon each connection is termed as Dynamic IP Address.

- Static IP Address:

A static address remains constant, acting as a permanent identifier on the internet. These are utilized by DNS servers, which are essentially computers aiding in accessing websites on your device.

Static IP Addresses offer detailed location information such as the continent, country, city, and Internet Service Provider (ISP) associated with a particular device. Knowing the ISP enables tracking the device's internet connection. Compared to dynamic addresses, static IP Addresses offer reduced security as they are simpler to trace.

- Private IP Address:

This refers to an internal address specific to your device, which is not directed towards the internet, preventing any data exchange between a private address and the internet.

- Shared IP addresses:

Numerous websites opt for shared IP addresses, particularly when their traffic is manageable and controllable. By renting out these addresses to similar websites, they achieve cost-effectiveness. Likewise, various companies and email servers utilize the same IP address within a single mail server to reduce expenses, especially during idle server periods, saving on costs.

- Dedicated IP addresses:

A dedicated IP Address is exclusively utilized by a single company or individual, offering specific advantages such as the use of a private Secure Sockets Layer (SSL) certificate, unlike shared IP addresses. This allows for website access or logging in via File Transfer Protocol (FTP) using the IP address rather than the domain name. Dedicated IP Addresses enhance website performance during periods of high traffic and provide protection against the risk of being blacklisted due to spam on a shared IP address.

3.5 DNS (Domain Name System)

The DNS serves as a service translating hostnames to IP addresses. Implemented as a distributed database across a hierarchy of name servers, DNS functions as an application layer protocol facilitating message exchange between clients and servers. Its role is essential for the operation of the Internet. People access online information using domain names such as nytimes.com or espn.com, while web browsers communicate using Internet Protocol (IP) addresses. DNS

converts domain names into IP addresses, enabling browsers to fetch Internet resources.

Every internet-connected device possesses a distinct IP address, utilized by other machines to locate it. DNS servers remove the necessity for humans to remember IP addresses like 192.168.1.1 (in IPv4) or more intricate alphanumeric ones such as 2400:cb00:2048:1::c629:d7a2 (in IPv6).

DNS resolution entails transforming a hostname (like www.example.com) into a computer-compatible IP address (such as 192.168.1.1). Each Internet-connected device is assigned an IP address, crucial for locating the relevant device, akin to using a street address to locate a specific home. When a user intends to load a webpage, there needs to be a conversion between what the user inputs into their web browser (example.com) and the machine-readable address essential for finding the example.com webpage.

(i) Need of DNS

Each host is recognized by its IP address, yet remembering these numerical values can be challenging for individuals, especially considering that IP addresses are not fixed. Consequently, a mapping process is necessary to translate domain names into IP addresses. DNS serves this purpose by converting website domain names into their corresponding numerical IP addresses.

(ii) Types of Domain

There are several categories of domains:

- Generic domains: Generic domains include .com (commercial), .edu (educational), .mil (military), .org (nonprofit organization), and .net (similar to commercial).
- Country domain: .in (India) .us .uk
- Inverse domain: To determine a website's domain name from its IP address, we utilize DNS for the mapping. For instance, to discover the IP addresses associated with geeksforgeeks.org, one would input www. example.org". Identifying the IP address linked to a website is challenging due to the sheer number of websites. With millions of sites, efficient generation of IP addresses is crucial, necessitating a well-organized database to minimize delays in the process.
- DNS record: Domain names and IP addresses have associated validity periods, known as time to live (TTL), alongside other relevant information concerning the

domain. These records are organized in a hierarchical, tree-like structure.

- Namespace: A collection of potential names exists, which can be either flat or hierarchical. This naming system holds a set of associations between names and values, where a resolution mechanism retrieves the appropriate value when provided with a name.
- Name server: It represents the execution of the resolution process.

3.6 PORTS AND ITS TYPE

When an application on one computer communicates with an application on another computer, it utilizes both the IP Address and MAC Address. However, how does our computer discern that the data is intended for a particular application and originates from a specific application? This is where the notion of Ports comes into play.

Consider your MAC Address or IP Address as akin to the PIN code for the nearest Post Office, and your house address as a Port. When a parcel is dispatched to you, it arrives at the nearest post office and is then directed to your address for delivery. Likewise, in a computer system, data is initially received using its IP or MAC address and then directed to the relevant application based on the port number accompanying the data packets.

STOP TO CONSIDER

A port is a numerical address, represented by a 16-bit unsigned integer, assigned to each internet-utilizing application on a computer for sending or receiving data.

Each time an application transmits data, it is distinguished by the specific port from which the data originates, and the data is then routed to the recipient application based on its corresponding port. Port is frequently referred to as port number.

In the OSI Model, ports are utilized within the Transport layer. Within the headers of Transport layer protocols such as TCP and UDP, there exists a section designated for defining ports, also

known as port numbers. The network layer is not concerned with ports; its protocols solely focus on IP Addresses.

Computers, specifically their operating systems, allocate ports to various applications. Ports aid computers in distinguishing between incoming and outgoing traffic. As a 16-bit unsigned number, ports span from 0 to 65535.

(i) Types of Port

Ports are subdivided into three distinct categories:

- Well Known Port: This range spans from 0 to 1023 and is set aside for commonly used and specific services. It is utilized by widely adopted protocols and services such as HTTP (port 80), FTP (port 21), DNS (Port 53), SSH (port 22), among others.
- Registered port: This range extends from 1024 to 49151 and is allocated to applications or services that are less prevalent. However, it is utilized by those applications or services that necessitate their dedicated port. Organizations have the option to request specific port numbers within this range from IANA (Internet Assigned Number Authority).
- Dynamic Port: This range spans from 49152 to 65535 and is alternatively referred to as Ephemeral or Private Ports. These ports are designated for temporary or short-lived connections. They are not officially registered or assigned and can be utilized by any process.

(ii) The significance of Ports

Ports hold various significances, some of which include:

- Identification of service: Distinct applications or services operating on the same device can be distinguished by their respective port numbers. For instance, HTTP (port number 80) and SMTP (port number 25) on the same computer utilize different port numbers to ensure the accurate routing of their data to the appropriate service.
- Efficient Data Routing: When a network device receives data from various sources, it employs port numbers to effectively direct those data packets to the appropriate application.

- Block traffic from specific applications/services: To prevent incoming or outgoing traffic from a particular application or service, a firewall must be installed and configured with the port number of the respective application or service. Traffic to and from certain applications or services is restricted when potential threats are identified from those sources.
- Scalability of services: Numerous services can operate concurrently on a single device and can be distinguished based on their port numbers. This capability enables the device to expand and accommodate multiple services simultaneously.

(iii) Several commonly used port numbers

Commonly used port numbers associated with applications/services frequently utilized by users.

Port Number	Used By
80	HTTP(Hyper Text Transfer Protocol)
23	Telnet
25	SMTP(Simple Mail Transfer Protocol)
53	DNS(Domain Name System)
7	Echo
20/21	FTP(File Transfer Protocol)
69	TFTP(Trivial File Transfer Protocol)
443	HTTPS(Hyper Text Transfer Protocol Secure)
22	SSH(Secure Shell)
110	POP3(Post Office Protocol version 3)
67/68	DHCP(Dynamic Host Configuration Protocol)
123	NTP(Network Time Protocol)
143	IMAP(Internet Messaging Access Protocol)
1433	Microsoft SQL
3306	MySQL

3.7 WEB SERVER ARCHITECTURE

3.7.1 Apache HTTP Server

Apache is freely available and open-source web server software utilized by approximately 40% of websites globally. Officially named Apache HTTP Server, it is developed and managed by the

Apache Software Foundation. Apache enables website owners to serve content over the internet, earning it the designation of a "web server." One of the earliest and most dependable versions of the Apache web server was released in 1995. Apache manages incoming HTTP requests and processes them accordingly. It is an open-source platform developed and maintained at Apache.org.

Initially launched in 1996 by Robert McCool, the Apache web server has consistently held the title of the most popular HTTP server on the World Wide Web since its inception. It was the pioneering web server architecture adopted by Netscape Communication Corporation. Over the years, Apache has evolved to accommodate the changing landscape of the internet, supporting both static and dynamic web pages. The server is compatible with various programming languages such as PHP, Perl, Python, and MySQL. As of April 2008, the Apache Server is responsible for serving around 50% of the existing web pages. The Apache license permits the distribution of both open-source and closed-source modifications of the source code. The name "Apache" is a trademark that is registered.

- Architecture

Apache consists of two primary components, with the latter consisting of numerous smaller elements. These components are the Apache Core and the Apache Modules, which essentially expand upon the functionality of the Apache core. The server's popularity stems from its simplicity in implementation and its capacity for easy extension of capabilities through the incorporation of various modules. As evident, the Apache designers opted for a modular strategy, enabling anyone to enhance the server's basic functionality without disrupting its core implementation.

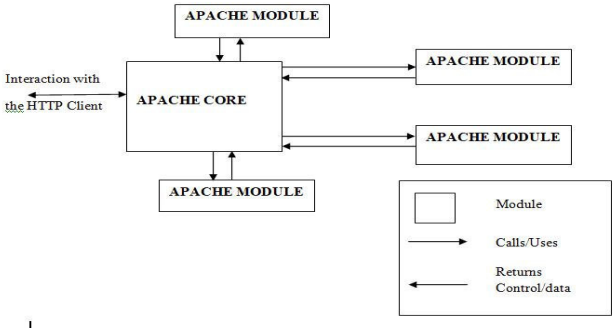


Fig.3.4 An overview of Apache

HTTP serves as the protocol utilized for transmitting data to and from the server. It's the protocol recognized by the Apache Web Server, employed for sending information back to the client machine. For a deeper technical understanding, the client machine, typically a browser, sends an HTTP Request Object to the server, which then replies with an HTTP Response Object. This exchange represents the typical interaction between the server and the browser, all of which Apache is designed to manage.

As previously mentioned, the client sends an HTTP request to the Apache Web server, which then manages and pools connections based on instructions within its core. Subsequently, the server sends a response. It's often overlooked that many individuals unknowingly use an Apache web server daily, as it currently holds the title of the most widely used web server. When you access a webpage online, it's probable that an Apache Web server is handling your request and delivering the requested webpage in return.

The Apache Web Server's general design adopts a modular approach, rather than being a singular piece of code responsible for all tasks. This approach enhances robustness and facilitates greater customization while preserving the security measures embedded within the Apache Core.

.

To realize this modular approach, the Apache designers chose to divide the server into two primary components.

1. **The Apache Core:** These components manage the fundamental operations of the server, including request allocation and the maintenance and pooling of connections. This represents the typical process within the Apache Core, providing an overview of its workflow. The Apache Core interacts with surrounding components, comprising the core architecture of Apache. The rationale behind this design was to ensure that components not reliant on each other remained separate, thus they were structured into modules.

The Apache Core consists of numerous small components that manage the fundamental aspects of what a web server should accomplish. The core components consist of a set of classes designated for specific tasks. It's important not to mix them up with modules, which are additional implementations allowing customization of Apache's functionalities.

The Apache Core delivers the essential functionality of an HTTP web server. Altering it or permitting modifications would compromise its modularity and potentially weaken security measures. Hence, modules are essential for expanding the core capabilities of Apache.

The components that constitute the Apache core are as follows:

- http_protcol.c: This component manages all operations involving direct communication with the client via the HTTP protocol. It is responsible for handling socket connections through which clients connect to the server, and facilitates all data transfers.
- http_main.c: This component oversees the initiation of the server and includes the primary server loop that awaits and accepts connections. It also handles the management of timeouts.
- http_request.c: This component manages the processing flow of requests, ensuring modules are called in the correct sequence as required. Additionally, it oversees error handling.
- http_core.c: This component implements the most fundamental functionality, barely serving documents.
- alloc.c: This component manages resource allocation pools and maintains their tracking.
- http_config.c: This component offers various utilities, such as reading configuration files and handling the information derived from them, along with support for virtual hosts. A crucial role of http_config is compiling the list of modules to be invoked during different phases of ongoing requests within the server.

2. The Apache Modules: These extensions serve as additional components of the server responsible for various types of processing, such as user authentication, among other tasks.

Due to Apache's architecture, modules are not directly aware of each other, and no single module can fully handle or process requests made to the Apache server. Typically, requests involve passing information from one module to the core, then to another module until the request is fully managed, and then it's returned to the client. Apache employs Request Phases, managed by the HTTP_REQUEST component of the core, to handle this process.

The stages or the sequence managed by the HTTP_REQUEST Module of the Apache core are outlined as follows:

The tasks controlled by the HTTP_REQUEST Module of the Apache core include the following steps:

- Converting URI to filename.
- Verifying access based on host address and other available data.
- Extracting a user id from the HTTP request and verifying its validity.
- Authorizing the user.
- Identifying the MIME type of the requested object (content type, encoding, and language).
- Performing necessary adjustments (such as replacing aliases with the actual path).
- Transmitting the data to the client.
- Recording the request in the log.

Modules serve to expand, overwrite, and implement functionalities within the Apache web server. However, they do not directly interact or have knowledge of one another. Consequently, modules are connected to the Apache core in the same manner. Since modules lack direct knowledge of each other, they must relay all information back to the core. Subsequently, the core forwards this information to the appropriate module using the HTTP_REQUEST component. This setup not only preserves the stability of the Apache Core but also adds a layer of security. Every process is required to pass information to the Core, which then verifies and manages errors through the HTTP_REQUEST component. The Apache web server features a modular design comprising a core component that establishes the fundamental operations of a web server, alongside multiple modules responsible for executing the various stages of processing an HTTP request. These modules offer handlers for different phases of the request. The core component manages HTTP connections, orchestrates the order in which module handlers are called, and services the ongoing request accordingly. Concurrency is limited to a set of persistent processes handling incoming HTTP requests on the same port. Although modules aren't implemented as separate processes, it's feasible to spawn children or collaborate with independent processes to manage request processing phases. Furthermore, Apache's functionality can be easily modified by

developing new modules that supplement or replace existing ones. The server offers high configurability across different levels, with modules having the capability to define their own configuration commands.

A notable feature of Apache contributing to its robustness and enhanced speed is its ability to dynamically initialize modules. Unlike traditional setups where all modules start up with the server, Apache selectively initializes only the necessary modules at runtime. This approach significantly accelerates request processing by ensuring that only the essential modules are activated, leading to faster overall performance.

Modules contain components known as handlers.

- Handlers: For Apache, a handler represents the operation required during a specific phase of processing a request. For instance, a file request handler would involve opening, reading, and then transmitting the file to the Apache core for delivery to the client. Modules define handlers based on the requirements of fulfilling requests, and these handlers are responsible for relaying processed information from the Apache Module back to the Apache Core's HTTP_REQUEST component.

- Modular Configuration
  When opting for a static configuration of Apache, it's important to select modules judiciously due to memory usage implications. The more modules incorporated, the greater the memory consumption. Consequently, employing a forked multi-processing module can notably impact the system's memory demands. It's essential to note that certain items are automatically included, necessitating explicit enabling or disabling of required modules. Additionally, ensure inclusion of any necessary third-party modules, such as authentication, PHP, or mod_perl, for the functioning of the web service. Utilize the "configure --help" command to access a list of available options.

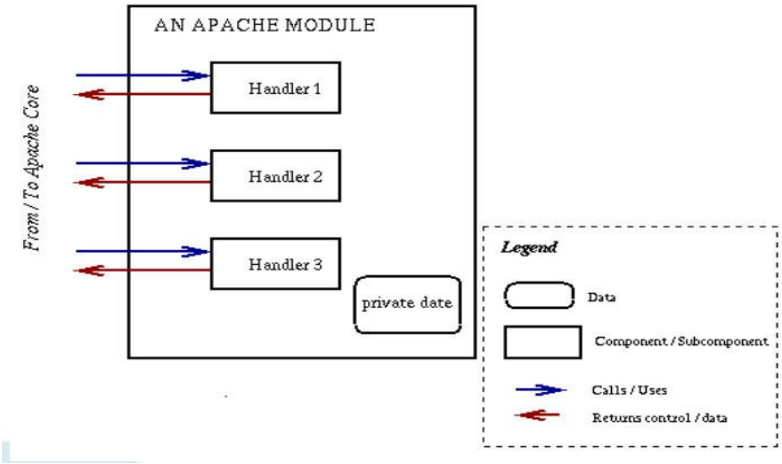


Fig.3.5 Handler system within an Apache Module

- Concurrency in Apache
  Apache offers access to two tiers of concurrency. Firstly, concurrent processes execute simultaneously, particularly when operating on distinct processors, such as in the scenario of separate processes running on a multitasking system. Additionally, if the operating system supports multithreading, a standard allowance of up to 50 threads is permitted for each process.

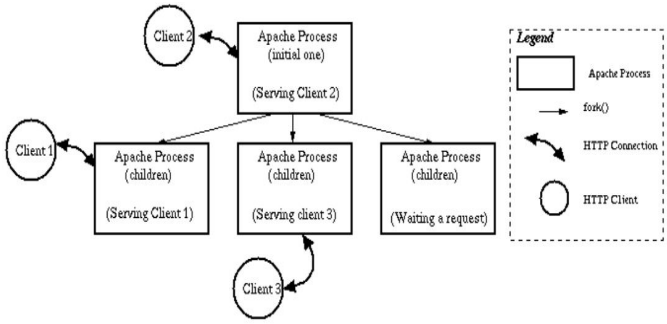


Fig.3.6 Concurrency in Apache

Each request received by the server is managed by a duplicate of the http program. Instead of generating a new instance copy upon demand and disposing of it after completing a request, Apache ensures the presence of at least 5 and at most 10 inactive children continuously. The parent process conducts regular assessments on a framework named the scoreboard,

responsible for monitoring all active server processes and their conditions.

If the number of idle servers listed on the scoreboard falls below the minimum threshold, the parent process will create additional ones. Conversely, if the scoreboard indicates an excess of idle servers beyond the maximum limit, the parent process will terminate the surplus children. Upon receiving a request, the parent process forwards it to the next available idle child listed on the scoreboard. Subsequently, the parent resumes listening for subsequent requests.

During parent and child requests, there exists a default limit of 256 for the total number of requests processed concurrently. This default setting was established by the server's creators to ensure that the scoreboard file remains sufficiently small for efficient scanning by processes, thus avoiding potential overhead issues.

The quantity of requests handled simultaneously is constrained by the maximum number of processes allowed to exist. To accommodate waiting requests, a queue is available. This queue forms when the parent delegates a request to an idle child, after which it resumes awaiting the next request. The queue can hold a maximum of around 400-600 pending requests.

The architecture of the Apache server was crafted to optimize single connections. It employs persistent connections, enabling multiple client requests to be managed through one connection instead of establishing and terminating a connection for each request. By default, the maximum number of requests permitted over one connection is set to 100. Closure of the connection occurs due to a timeout mechanism.

3.7.2 IIS (Internet Information Services)

- Introduction

IIS 7(Internet Information Services) and subsequent versions feature a request-processing structure within their architecture.

  - The Windows Process Activation Service (WAS) allows websites to utilize protocols beyond HTTP and HTTPS.
  - A customizable web server engine that allows modules to be added or removed as needed.
  - Request-processing pipelines from IIS and ASP.NET that are integrated together.

- Components in IIS

  IIS comprises multiple elements crucial for both application and web server functions in Windows Server® 2008 (IIS 7.0) and Windows Server 2008 R2 (IIS 7.5). Each element bears specific duties, such as monitoring requests directed at the server, overseeing processes, and interpreting configuration files. These elements encompass protocol listeners like HTTP.sys, as well as services such as the World Wide Web Publishing Service (WWW service) and Windows Process Activation Service (WAS).

- Protocol Listeners

  Protocol listeners are responsible for receiving requests specific to their protocols, forwarding them to IIS for handling, and then sending responses back to the requestors. For instance, when a client browser seeks a webpage from the Internet, the HTTP listener, HTTP.sys, detects the request and forwards it to IIS. Once IIS processes the request, HTTP.sys sends back a response to the client browser.

  By default, IIS employs HTTP.sys as the protocol listener for HTTP and HTTPS requests. Initially introduced in IIS 6.0, HTTP.sys serves as an HTTP-specific protocol listener. In subsequent versions like IIS 7 and beyond, HTTP.sys continues to function as the HTTP listener, now with added support for Secure Sockets Layer (SSL).

To accommodate services and applications utilizing protocols other than HTTP and HTTPS, alternative technologies like Windows Communication Foundation (WCF) can be utilized. WCF offers listener adapters, combining the roles of both a protocol listener and a listener adapter, which will be elaborated on later in this document.

- Hypertext Transfer Protocol Stack (HTTP.sys)
  The HTTP listener, also known as HTTP.sys, operates within the networking subsystem of Windows operating systems. It functions as a kernel-mode device driver responsible for receiving HTTP requests from the network, forwarding them to IIS for processing, and subsequently delivering processed responses to client browsers.

  In IIS 6.0, HTTP.sys replaced the Windows Sockets API (Winsock), which was a user-mode component utilized by earlier versions of IIS for handling HTTP requests and responses. Subsequent versions of IIS, such as IIS 7 and beyond, persist in using HTTP.sys for managing HTTP requests.
  HTTP.sys offers the following advantages:

  1. Caching in kernel mode: Cached responses are served directly from the kernel without transitioning to user mode.
  2. Request queuing in kernel mode: Requests incur lower overhead in context switching as the kernel directly routes them to the appropriate worker process. If a worker process isn't available to handle a request, it's held in the kernel-mode request queue until a worker process becomes available.
  3. Pre-processing of requests and security filtering.

- World Wide Web Publishing Service (WWW service)

  In IIS 7 and subsequent versions, tasks once managed solely by the World Wide Web Publishing Service (WWW Service) are now divided between two

services: WWW Service and a newly introduced service called Windows Process Activation Service (WAS). Both services operate under the LocalSystem account within the same Svchost.exe process and utilize identical binary files.

- The functioning of the WWW Service in IIS 6.0
  .In IIS, the role of managing worker processes is now delegated away from the WWW service. Instead, the WWW Service functions as the listener adapter for the HTTP listener, HTTP.sys. In this capacity, it's mainly tasked with configuring HTTP.sys, ensuring updates to HTTP.sys upon configuration alterations, and signaling WAS when a request enters the request queue.

  Furthermore, the WWW Service retains its responsibility for gathering performance counters for websites. These performance counters, still under the purview of the WWW Service, are specific to HTTP and don't pertain to WAS.
  In IIS 6.0, the primary responsibilities overseen by the WWW Service include:
  1. HTTP Administration and Configuration
     The WWW Service retrieves configuration details from the IIS metabase and utilizes this data to configure and maintain the HTTP listener, HTTP.sys. Furthermore, it initiates, halts, monitors, and administers worker processes responsible for handling HTTP requests.
  2. Performance Monitoring

     The WWW Service oversees performance and furnishes performance counters for both websites and the cache within IIS.

  3. Process Management
     The WWW Service is responsible for handling application pools and worker processes, including tasks such as initiating, terminating, and refreshing worker processes. Moreover, it monitors the well-being of these worker

processes and activates rapid fail detection to prevent new processes from starting if multiple worker processes fail within a customizable timeframe.

- Windows Process Activation Service (WAS)
  In IIS 7 and later versions, the Windows Process Activation Service (WAS) takes charge of application pool setup and worker processes, a role previously handled by the WWW Service. This allows for uniform configuration and processing models across both HTTP and non-HTTP sites.

  Furthermore, you have the option to operate WAS independently of the WWW Service if HTTP functionality isn't required. For instance, managing a Web service through a WCF listener adapter like NetTcpActivator doesn't necessitate running the WWW Service if there's no need to listen for HTTP requests in HTTP.sys. Further details about WCF listener adapters and hosting WCF applications in IIS 7 and beyond using WAS are available.
- Configuration Management in WAS
  During startup, WAS retrieves specific details from the ApplicationHost.config file and forwards them to listener adapters on the server. This listener adapters act as intermediaries between WAS and protocol listeners like HTTP.sys. Upon receiving configuration data, listener adapters configure their associated protocol listeners, preparing them to accept requests.

  In the case of WCF, a listener adapter encompasses the functionality of a protocol listener. Thus, a WCF listener adapter such as NetTcpActivator is set up based on information from WAS. Once configured, NetTcpActivator awaits requests utilizing the net.tcp protocol. More details about WCF listener adapters are available.

  The subsequent enumeration outlines the kind of data extracted by WAS from the configuration:

1. Overall configuration details

2. Protocol-specific configuration particulars for both HTTP and non-HTTP protocols

3. Application pool settings, including process account data

4. Site configuration specifics, such as bindings and applications

5. Application-specific configuration elements, like enabled protocols and the assigned application pools.

- Process management
  WAS oversees application pools and worker processes for both HTTP and non-HTTP inquiries. Upon receiving a client request, WAS checks if a worker process is active. If an application pool already has an operational worker process handling requests, the listener adapter forwards the request to it for processing. In the absence of an active worker process in the application pool, WAS initiates one to ensure the listener adapter can relay the request for processing.
- Request Processing in IIS
  In IIS, the merging of the IIS and ASP.NET request pipelines creates a unified approach to handling requests. The updated request-processing architecture employs a structured sequence of native and managed modules that execute specific tasks in response to requests.

  This new framework offers several advantages over previous IIS versions. Firstly, it extends features previously exclusive to managed code to all file types. For instance, ASP.NET Forms authentication and URL authorization can now be applied to static files, Active Server Pages (ASP) files, and other file types within sites and applications.

Secondly, it eliminates redundancy by integrating features from both IIS and ASP.NET. For example, when a client requests a managed file, the server utilizes the appropriate authentication module in the integrated pipeline for client authentication. In prior IIS versions, the same request would undergo authentication processes in both the IIS and ASP.NET pipelines.

Lastly, centralizing the management of all modules in one location streamlines administration tasks for sites and applications on the server, as opposed to managing features separately in IIS and ASP.NET configurations.

- IIS Architecture

IIS consists of two primary layers: Kernel Mode and User Mode. Kernel Mode encompasses HTTP.SYS, while User Mode includes WAS and the W3 service. The communications between two layers are as follows.

- When a client browser sends an HTTP request for a resource on the Web server, HTTP.sys intercepts the request.
- HTTP.sys then communicates with WAS to retrieve details from the configuration store.
- WAS queries the configuration store, known as applicationHost.config, for configuration information.
- The WWW Service receives this configuration data, including details about application pools and site configurations.
- Based on the received information, the WWW Service configures HTTP.sys accordingly.
- WAS initiates a worker process for the corresponding application pool.
- The worker process handles the request and sends back a response to HTTP.sys.
- Finally, the client receives the response.

Within a worker process, an HTTP request undergoes a series of sequential steps known as events in the Web

Server Core. Each event involves a native module handling specific aspects of the request, such as user authentication or logging information. If the request necessitates a managed module, the native ManagedEngine module establishes an AppDomain where the managed module executes tasks like user authentication via Forms authentication. Once the request completes all events in the Web Server Core, the response is sent back to HTTP.sys.

- Role of HTTP.sys in IIS
  HTTP.SYS is an integral component of the kernel mode in IIS. Each client request goes through kernel mode, where HTTP.sys organizes a queue for each application pool according to the request. Upon creating an application pool, IIS automatically registers it with HTTP.sys to distinguish it during request processing. It offers the following functionalities within IIS:
    1. Directing HTTP requests to the appropriate request queue.

    2. Storing responses in kernel mode for faster access.

    3. Conducting all textual logging for the WWW service.

    4. Enabling quality of service features like connection limits, timeouts, queue-length limits, and bandwidth regulation.

- ISAPI Filter
  ISAPI filters, represented by DLL files, serve to adjust and enhance the capabilities of IIS. Operating consistently on an IIS server, they examine each request until identifying one that requires action. These filters are registerable with IIS to influence server behavior and can undertake various functions:

1. Altering request data (such as URLs or headers) sent by the client
2. Managing the mapping of physical files to URLs
3. Governing the user credentials used in anonymous or basic authentication
4. Modifying or analyzing requests post-authentication
5. Adjusting responses sent back to the client
6. Executing operations upon request completion
7. Performing operations upon closure of the connection with the client
8. Conducting specialized logging or traffic analysis
9. Handling encryption and compression functionalities.

Check Your Progress

1. State true or false

   (i) . The HTTP protocol cannot used to retrieve resources, from the internet.

   (ii) HTTP operates in a connectionless manner.

   (iii) The IPv4 address is represented in binary notation with 64 bits.

   (iv) IPv4 addressing initially employed the concept of classes.

   (v) In the OSI Model, ports are utilized within the Transport layer.

   (vi) The Windows Process Activation Service (WAS) allows websites to utilize protocols beyond HTTP and HTTPS.

3.8 SUMMING UP

- Hypertext transfer protocol, or HTTP, is the protocol used to send data over the Internet.
- The HTTP protocol is used to retrieve resources, including HTML documents.
- The user-agent (or a proxy acting on its behalf) is the only entity that sends requests across the client-server HTTP protocol. The user-agent is often a web browser.
- Get method uses a provided URL to retrieve data from the specified server.
- A computer requires an address so that other computers on the internet can communicate without the risk of sending information to the wrong destination.
- In Classful addressing, an IP address within class A, B, or C is segmented into netid and hostid components, each with different lengths determined by the class of the address.
- When user links a smartphone or computer to the internet, then the Internet Service Provider assigns user an IP Address from its pool of available addresses.
- A static address remains constant, acting as a permanent identifier on the internet. These are utilized by DNS servers, which are essentially computers aiding in accessing websites on your device.
- When an application on one computer communicates with an application on another computer, it utilizes both the IP Address and MAC Address.
- Apache is freely available and open-source web server software utilized by approximately 40% of websites globally.

3.9 ANSWER TO CHECK YOUR PROGRESS

1(i) False (ii) True (iii) False (iv) True
(v) True (vi) True

3.10 POSSIBLE QUESTIONS

(1) Why Port Number is used?

(2) What are HTTP Request Messages?

(3) What are the different roles of HTTP?

(4) What is HTTPS?

(5) Convert the given IPv4 addresses from their binary representation to dotted-decimal notation.

(i) 10000001 00001011 00001011 11101111

(ii) 11000001 10000011 00011011 11111111

(6) Convert the given IPv4 addresses from their dotted-decimal notation to binary representation.

(i) 129.11.11.239

(ii) 193.131.27.255

(7) How can you categorize a DNS as a TCP or UDP?

(8) What is MAC Address?

(9) How do IP addresses facilitate communication across the internet?

(10) Explain some key differences between various web server architectures.

(11) What are some best practices for configuring and optimizing web server performance?

(12) How does load balancing come into play in the context of web server infrastructure?

3.11 REFERENCE AND SUGGESTED READINGS

1. Godbole, A.S. and Kahate A. *Web Technologies*. TATA McGRAW HILL.2011

3.Forouzan, B.A. *Data Communication and Networking*. Tata McGraw Hill,2017

2. https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview

---×---

UNIT- 4
SERVER-SIDE SCRIPTING BASICS

UNIT STRUCTURE

4.1 INTRODUCTION

We have already learnt about the different areas related to the web development in the earlier chapters. In this unit we are going to discuss about the basic concepts of server-side scripting. In the dynamic web development process, server-side scripting plays a very important role. Efficient Server-side scripting is required to developcustomized, interactive and secured web applications. Overview of Common Gateway Interface (CGI), Active Server Pages (ASP), and JavaServerPages (JSP) are also presented in this unit. Finally, connectivity between web application and the corresponding database using Open Database Connectivity (ODBC) is also discussed in this unit.

4.2 OBJECTIVES

After going through this chapter, we will be able to learn:

- About the basic concepts of Server-side scripting.
- About the basic concepts of CGI, ASP and JSP.
- About web database connectivity.
- The use of ODBC for wed database connectivity.

4.3 INTRODUCTION TO SERVER-SIDE SCRIPTING

In web development, Server-side scripting is a scripting technique where scripts are developed and executed on the web server. Different server-side operations are handled by using Server-side scripting. For example: customized response to the user, updating dynamic contents, database access, providing appropriate responses depending upon the user requests etc.Server-side scripting is responsible for maintaining communication between the server and its clients. Server-side scripting is used with the following steps.

- In the first step, user request for a service by performing an event in a webpage like clicking a button, submitting a form etc.
- In the second step, the user request is moved to the web server and then in the server, corresponding server-side script is executed. In this step, depending upon the requirement, server-side script may connect to the corresponding databases for database records so that the script can be able to perform its job as per the user request.
- In the third step, the server-side script generates the final result in the server and then the server returnsthat result to the user's browser.

The main characteristics and functionalities of Server-side scripting are presented in the following points.

- The server-side scripts are stored in the web server. These are also executed on the web server depending upon the requests that are sent by the users.Server-side scripting provides the mechanism for the communication between users and the server. Server-side scripting can be used to provide customized responses to the users as per their requirements.
- Server-side scripts are more secured than the client-side scripts because users do not have any direct access to the server-side scripts. Different important and confidential information used in Server-side scripting are handled in the server.
- Server-side scripting is not dependent on specific web browsers or devices used by the users. It can perform its job irrespective of the user's browser or device.

- We already know that Server-side scripting is essential in the development of dynamic web pages.
- Server-side scripts can be used to connect with databases and access database records. Server-side scripts can also insert new data to a database and update existing data in a database.
- Server-side scripting is used to handle different business related operations like processing payments, estimating costs, validation of forms etc.
- Performance of web applications can be improved by using server-side scripting as the corresponding operations are performed in the server.
- Error handling mechanism can be provided with server-side scripting.

STOP TO CONSIDER

A script is group of code thataccomplish a specific job on a web page or a web server.

Some of the popular languages for Server-side scripting are presented in the following points.

- **PHP (Hypertext Preprocessor)** is an open-source language that can be used to develop server-side scripts. It is largely used to develop dynamic web pages and web applications.
- **Python** is also an open-source language that can be used to develop server-side scripts. It becomes very popular language because of its simplicity and robust competences.
- **ASP.NET** is a Microsoft web framework that can be used to develop web applications by using .NET languages like Visual Basic .NET, C# etc.
- **JavaScript** can also be used to develop server-side scripts by using **Node.js** which is a JavaScript runtime environment.

4.4 OVERVIEW OF COMMON GATEWAY INTERFACE (CGI)

Common Gateway Interface (CGI) is the first dynamic web technology introduced in 1990. It is a specification that enables the communication between a web server and a web browser by using the HTTP protocol. It means that CGI is a standard protocol that facilitates web servers to work with scripts so that dynamic contents can be generated on the web browsers. It is also work as a medium to interact with the databases. Different programming languages can be use to write CGI scripts. For example, we can use C, Shell scripting, Perl etc. to write CGI scripts.So, the major features of CGI are presented in the following points.

- CGI was the base of dynamic web development technology.
- CGI scripts are Server-side scripts.
- CGI helps to create dynamic web pages and it facilitates the interaction with the databases.
- CGI scripts can be written by using different programming languages. For example: C, Perl, shell scripting etc.
- CGI facilitates a fast and simple approach to execute the fundamental web tasks.
- In CGI, it is easy to use existing CGI scripts as a significant repository of CGI scripts is available for different web tasks.
- Finally, CGI is extensively recognized and greatly compatible with all the web browsers.

Different steps in the execution of CGI scripts are presented in the following points.

- The first step is to read user input from different controls available in the corresponding HTML form and send the user inputs or request to the web server as an element of the HTTP request of the corresponding web browser.
- In the second step, the web server receives the HTTP request and if it finds that the requested URL connectsto a CGI script then it identifies the

corresponding CGI script. In general CGI scripts are located in a particular directory on the web server.

- In the third step, the web server runs the identified CGI script.
- In the fourth step, the CGI script produces the required output and it is returned to the web server.
- In the next step, the web server obtains the output from the CGI script and it adds the output with the proper HTTP response codes. Then the HTTP response is sent to the corresponding browser.
- In the final step, the browser obtains the HTTP response from the web server and it displays the output.

We have already learnt about different useful features of CGI but it has also different drawbacks. In general the following drawbacks are associated with CGI.

- The main drawback of CGI is the overhead associated with the loading of web pages. In CGI, a new process has to be produced by the operating system for each web page loading.As a result, each request for a web page requires server resources and processing time and it introduce overhead in the server. This overhead is increased significantly in case of web pages requested by a large number of clients at the same time. The performance of the web server may be significantly degraded due to this overhead.
- CGI scripts may be difficult to understand.
- Security threats may be associated with a CGI script if it is not properly developed by the programmer or user.
- In case of CGI scripts, data caching in memory is not possible between web page loads.

4.5 OVERVIEW OF ACTIVE SERVER PAGES (ASP)

Active Server Pages (ASP) is a dynamic web development technology created by Microsoft. In 1996, ASP was first presented. It was a part of the Internet Information Services (IIS). Compared to CGI, ASP is a faster and simpler technology. In case of ASP, scripts are allowed to run on the web server. In general, VBScript is used

in ASP as scripting language. In recent times, ASP is anon-operational technology but it was an effective mechanism for developing dynamic web pages. Instead of ASP, ASP.NET is used in the current times. The extension of an ASP file is '.asp'. The functionality of an ASP file is similar to an HTML file but it also includes server-side scripts that are executed on the server.Steps in the execution of ASP file are presented in the following points.

- In the first step, a required ASP file is requested by the Web browser.
- In the second step, IIS sent the corresponding request to the ASP engine.
- In the next step, the ASP engine executes the script available in the ASP file.
- Finally, the response is sent back to the browser as plain HTML.

Different useful features of ASP are presented in the following points.

- In case of dynamic web development, ASP provides a fast, simple and less expensive approach.
- ASP is a language independent technology.
- The process of web page designing can be separated from the process of scripting by using ASP.
- ASP can be used to update and delete the contents of web pages. It can also be used to insert new contents to the web pages.
- ASP can be used to serve user requests through HTML forms.
- ASP can be used to interact with databases and retrieve required data from the databases.
- ASP code can be hidden from the users and hence web security can be maintained.

An ASP file consists of HTML tags and server scripts as per requirement.In an ASP file, scripts are written in between <% and %> tags. An example of an ASP file is presented in Example 4.1.

Example4.1:

```
<html>
    <body>
        <%
            Response.Write("Welcome to GUCDOE.")
        %>
    </body>
</html>
```

In the above example, Response.Write () method is used to display the output in the browser. So the output of the above ASP file will be: **Welcome to GUCDOE.**

The default scripting language in ASP is VBScript. If we want to change the scripting language to any other scripting language like JavaScript then the above ASP code has to be modified as presented in Example 4.2.

Example4.2 :

```
<%@ language="javascript"%>
<html>
    <body>
        <%
            Response. Write ("Welcome to GUCDOE")
        %>
    </body>
</html>
```

4.6 OVERVIEW OF JAVASERVER PAGES (JSP)

The first official version of Java Server Pages (JSP) was introduced by Sun Microsystems in 1999 and it is a part of Java Enterprise Edition (Java EE). JSP is also a Server-side scripting technology used in the development of dynamic web pages and web applications.JSP contains both HTML tags and JSP tags.JSP tags are used to include Java program in the HTML pages.JSP is actually theextendedform of the Servlet technology. JSP technology provides

all the features that are offered by the Servlet technology. JSP is easier to maintain than Servlets. In case of JSP, lesser number of codes is required to create dynamic web pages and web applications than the Servlet.

Different important features of JSP are presented in the following points.

- JSP can be used to develop dynamic web pages and web applications that are utilized to interact with clients in real-time.
- It is easy to write code in JSP. No advanced knowledge of Java is required to use JSP. It can be used with the basic knowledge of Java programming. Java codes can be embedded within JSP pages as per requirements.
- In JSP, different pre-built tags and custom tags are offered to add different functionalities to the web pages.
- In JSP, the length of codes to develop dynamic web sites and web applications can be reduced by using different built-in objects as per requirements.
- Interaction with databases can be easily performed in JSP.
- Exception handling is possible in JSP.
- JSP is a platform-independent technology and web security can be implemented in it.

STOP TO CONSIDER

Servlet technology is a web-based technology that is used to develop dynamic web pages. A Servlet is a Server-side script that is developed by using Java programming.

The extension of a JSP file is '.jsp'. After the development of a JSP file, its execution involves the following steps.

- If the JSP page is requested for the first time by the user or it has been altered then the JSP translator transformsthe JSP page into a servlet.

- The JSP compiler compile the servlet obtained from the earlier step and generate a class file with the file extension '.class'.
- Generated class files are loaded into the memory by the servlet container using the class loader.
- The servlet container creates an object of the servlet class file.
- 'jspInit()' method is invoked to initialize the JSP page.
- 'jspService()' method is invoked by the container to generate the HTML response. The HTML response is returned to the corresponding browser.
- Finally 'jspDestroy()' method is invoked to release the resources utilized in the execution of the JSP page.

We can develop JSP scripts by using the following scripting elements.

- In a JSP file, comments can be placed by using the following syntax.

<%-- *Comments*--%>

- In JSP files, <%@ *directive* %> tag is utilized when the web page is translated so that specific commands can be offered to the web container.
- In JSP files, <%! *declarations* %> tag is used to declare variables and methods that will be used in the Java code written in the JSP file.
- In JSP files, Java code can be written within <% %> tag.
- In a JSP file, scripting language expression can be placed by using <%= *expression* %> tag. JSP engine evaluate the expression and convert it to strings.

In Example 4.3, an example of a basic JSP page is presented.

Example 4.3 :

File Name: start.jsp

```
<html>
<head>
        <title>First JSP PAGE</title>
</head>
<body>
```

```
<%-- Display the string “Welcome to Gauhati University” using Java code --%>
        <% out.println("Welcome to Gauhati University"); %>
        <%-- Display the string “Welcome to CDOE” using Expression element --%>
        </br><%= "Welcome to CDOE"  %>
</body>
</html>
```

The output of the above script will be:

Welcome to Gauhati University
Welcome to CDOE

In Example 4.4, an example of a JSP page is provided where the use of <%@ *directive* %> tag and<%! *declarations* %> tag are presented.

Example 4.4 :

File Name: Display_age.jsp

```
<%@ page language="java" %>
<html>
        <head>
                <title>Use of JSP Directive and Declaration Tag</title>
        </head>
<body>

<%-- Declare variable and method--%>

<%!
                int age = 40;

publicintgetAge()
```

```
{
returnage;
}
%>

<p>Age is :<%= getAge() %></p>
</body>
</html>
```

The output of the above script will be:

Age is : 40

Now to run the .jsp file on the web browser, we have to first install the Tomcat server. In the next step, the .jsp file is stored inside a folder and that folder is placed inside the 'webapps'directory of Tomcat. Then we have to start the Tomcat server. Now the .jsp file can be run on the browser by using the URL of that file.An example of a possible URL of a .jsp fileis presented as follows.

http://localhost:8080/myFolder/myPage.jsp

Two general drawbacks associated with JSP are presented in the following points.

- It is difficult to debug JSP pages for errors.
- When a JSP page is first requested, it has to be compiled by the JSP compiler and as a result the first time access of a JSP page can be slower.

4.7 WEB DATABASE CONNECTIVITY

The process to connect a database to a web server is referred as the Web database connectivity. A web page or a web application can be able interact with the databases through the Web database connectivity.Web database connectivity allows web pages or applications to insert, delete and update information in the databases. We can use Open Database Connectivity (ODBC) to implement Web database connectivity.

4.7.1 Open Database Connectivity (ODBC)

In 1990, Microsoft had developed Open Database Connectivity (ODBC) that is a standard Application Programming Interface (API). It is utilized to access database management systems (DBMS). It offers a common interface todifferent programming languages for the interaction with multiple database systems using a standardized set of methods. ODBC allows application programs to utilizeStructured Query Language (SQL)for accessing data from different database systems.Major components of ODBC are presented in the following points.

- **Application component:** Application component of ODBC is basically responsible for calling the ODBC API methods and submitting SQL queries to the data sources. It retrieves the desired results from the data sources and process errors. It also request commit or rollback action for the purpose of transaction control.
- **Driver Manager:** The Driver Manager in ODBC is a dynamic link library (DLL). It is utilizedto load and manage ODBC drivers. It locates and loads the suitable driver that is required to set up a connection to the corresponding database.
- **ODBC Driver:** ODBC driveroffers the required functionalities to interact with a specific database system. It is responsible for processing ODBC method calls, submitting SQL queries to the corresponding data source, and sending the desired results to the application.
- **Data Source:** A data source in ODBC is the source of the desired data and the connection information required to access it. In general, it consists of the information which includes location of the server, name of the database, authentication details like login ID and password, and different configuration parameters required by the ODBC driver to establish connection with the data source.

Important features of ODBC are presented in the following points.

- ODBC can be utilized to interact with different database systems like Oracle, SQL Server, MySQL etc. In ODBC, to work with multiple database systems, a proper diver has to be installed for each database system.
- ODBC provides a significant support for metadata. ODBC offers several functions for querying metadata which will help applications to find out the structure of the corresponding databases at the run time.
- ODBC offers a uniform set of API functions that can be used to access databases without depending on the corresponding database systems.
- An application can be able to interact with different databases at the same time by using ODBC.
- ODBC provide user authentication mechanism at the time of interaction with the databases.
- ODBC offers a uniformapproach for error handling and reporting without depending upon the corresponding database systems.
- In ODBC, existing database connections can be reused. It decreases the overhead associated with creating the new connections at each time.
- ODBC provides commit and rollback operations to support transaction control mechanism. Concurrency control is possible in ODBC and so, several users can access and modify the same data concurrently by maintaining data integrity and consistency.
- Multiple data types like integer, string, float etc. are supported by ODBC.
- ODBC can be operated in multiple platforms like Linux, Windows, macOS etc.
- Different programming languages like Java, C, Python, etc. support ODBC.

General drawbacks associated with ODBC are presented in the following points.

- ODBC has an extra layer of abstraction and as a result it may perform slower than the other approaches like Java Database Connectivity (JDBC).

- Efficiency of ODBC may be reduced in case of large databases.
- It is difficult to construct and maintain ODBC drivers.
- ODBC may not support all the recent features provided by the present day database systems.

The general steps to use ODBC in an application are presented as follows.

Step1: Proper ODBC driver for the corresponding database system is installed on the server or the client system where the application will be executed.

Step2: A Data Source Name (DSN) is configured to store the connection details.

Step 3: ODBC connectivity is implemented by using ODBC API functionsin the application program to set up a connection to the corresponding database by using the DSN.

Step 4: SQL queries are executed through ODBC API functions to access the corresponding database and perform different operations like insert, delete and update.

Step 5: The results returned by the database system are handled in the application program and errors are managed efficiently.

Step 6: In the final step, database connections should be properly closed and used resources should be released so that optimal performance and optimal utilization of memory can be maintained.

CHECK YOUR PROGRESS

1. Choose the correct option

(i) Which of the following is true for Server-side scripting?

a) Server-side scripts are created on the server and executed on the client side.

b) Server-side scripts are stored in the web server.

c) Client-side scripts are more secured than the Server-side scripts.

d) None of the above.

(ii) Which of the following is a Server-side scripting technology?
a) CGI
b) ASP
c) JSP
d) All of the above

(iii) Which of the following tag is used to include Java code inside a JSP page?
a) <%-- --%>
b)<%@ %>
c)<%! %>
d)<% %>

(iv) Whichof the following tag is used to add comments inside a JSP page?
a) <%-- --%>
b) <%@ %>
c) <%! %>
d) <% %>

(v) Which of the following Server-side scripting technology is related to Microsoft's IIS web server?
a) ASP
b) JSP
c) CGI
d) None of the above

(vi) Whichof the following ODBC component offers the required functionalities to interact with a specific database system?
a) Application component
b) Driver Manager
c)ODBC driver
d)None of the above

(vii) In ODBC, Driver Manageris responsible for_____.
a) Calling the ODBC API functions
b) Requesting commit operation
c) Loading and managing ODBC driver.
d) None of the above.

2. Fill up the blanks

(i) Common Gateway Interface (CGI) is a _____.

(ii) ASP can be used to serve user requests through _____ forms.

(iii) JSP is actually theextendedform of the _____.

(iv)_____ method is invoked to initialize the JSP page.

(v) Open Database Connectivity (ODBC) is a standard _____ Interface (API).

4.8 SUMMING UP

- Server-side scripting is a scripting mechanism where scripts are created and executed on the web server.
- Different server-side operations like customized response to the user, updating dynamic contents, database access etc. are handled by using Server-side scripting.
- Communication between the web server and its clients can be maintained by the Server-side scripting.
- PHP (Hypertext Preprocessor), Python, ASP.NET are the examples of Server-side scripting technologies.
- Common Gateway Interface (CGI) is the first dynamic web technology introduced in 1990. It is a specification that enables the communication between a web server and a web browser by using the HTTP protocol.
- CGI is a standard protocol that facilitates web servers to work with scripts so that dynamic contents can be generated on the web browsers.
- Active Server Pages (ASP) is a dynamic web development technology presented first by Microsoft in 1996.
- ASP is a faster and simpler Server-side scripting technology than CGI.
- The file extension of an ASP file is '.asp'.
- The first official version of Java Server Pages (JSP) was introduced by Sun Microsystems in 1999 and it is a part of Java Enterprise Edition (Java EE).
- JSP is also a Server-side scripting technology. JSP contains both HTML tags and JSP tags where JSP tags are used to include Java program in the HTML pages.
- JSP is the extended form of the Servlet technology.
- Scripting elements of JSP are <%-- *Comments*--%> tag, <%@ *directive* %> tag, <%! *declarations* %> tag, <% %> tag to include Java code ,and <%= *expression* %> tag.

- The file extension of a JSP page is '.jsp'.
- The process to connect a database to a web server is referred as the Web database connectivity.
- In 1990, Microsoft had developed Open Database Connectivity (ODBC) that is a standard Application Programming Interface (API).
- ODBC is utilized to access database management systems (DBMS). It offers a common interface to different programming languages for the interaction with multiple database systems using a standardized set of methods.
- ODBC offers application programs to utilize Structured query language (SQL) for accessing data from different database systems.
- Major components of ODBC are Application component, Driver Manager, ODBC Driver and Data Source.

4.9 ANSWERS TO CHECK YOUR PROGRESS

1.

(i) b) Server-side scripts are stored in the web server.
(ii) d) All of the above
(iii) d)<% %>
(iv) a) <%-- --%>
(v) a) ASP
(vi) c)ODBC driver
(vii) c) Loading and managing ODBC driver.

2.

(i) Server-side scripting technology
(ii) HTML
(iii) Servlet technology.
(iv) 'jspInit()'
(v) Application Programming.

4.10 POSSIBLE QUESTIONS

1. Write down the different characteristics of Server-side scripting.
2. Write down the difference between Server-side scripting and Client-side scripting?

3. Write a short note on Common Gateway Interface (CGI).
4. Write down the different useful features of Active Server Pages.
5. Write down about different scripting elements of Java Server Pages.
6. What is Open Database Connectivity? Explain different major components of Open Database Connectivity.
7. Write down the different features of Open Database Connectivity.

4.11 REFERENCES AND SUGGESTED READINGS

- Kamal, R. *Internet and Web technologies*. Tata McGraw-Hill Education. (2002).
- Godbole, A.S. and Kahate A. *Web Technologies: TCP/IP Architecture, and Java Programming*. TATA McGRAW HILL.(2011)
- Harvey, Harvey Deitel, and Abbey Deitel. "Internet and World Wide Web: How to Program." (2011).

---×---

UNIT- 5
PHP: HYPERTEXT PREPROCESSOR

Unit Structure

5.1 INTRODUCTION

PHP, originally known as "Personal Home Page," now stands for "PHP: Hypertext Preprocessor," is a widely-used open-source scripting language designed for web development. It can be embedded into HTML and runs on servers, processing scripts to generate HTML pages dynamically. Created by Rasmus Lerdorf in 1994 as CGI binaries in C for managing his personal homepage, PHP has evolved into a versatile tool supporting tasks like database access, form processing, and dynamic content generation.

Key features of PHP include its open-source nature, making it freely available for download and use across different platforms like Windows, Linux, and macOS. It excels in server-side scripting, enabling interactions with databases such as MySQL, PostgreSQL, and SQLite. PHP boasts an extensive library of extensions for tasks like image processing and encryption, supported by a robust community offering ample resources and support through forums and tutorials. Applications of PHP span from content management systems like WordPress to frameworks such as Laravel for building scalable web applications and even powering social networking sites like Facebook, demonstrating its flexibility and widespread utility in modern web development.

5.2 OBJECTIVES

After going through this unit, learner will be able to:

- understand what PHP is and its uses.
- download and install PHP.
- use XAMPP and WAMP
- write, save, and run a PHP file.
- understand the basic operations of PHP.
- connect PHP with MySQL to perform database operations.

5.3 INSTALLING PHP

PHP can be downloaded from the official PHP website: php.net

5.3.1 On Windows

 Download the PHP zip file from the official website.
 Extract the contents to a directory, e.g., C:\php.
 Add the PHP directory to your PATH environment variable.

 Verify the installation by opening a command prompt and typing php -v. You should see the PHP version information.

5.3.2 On Linux

 Open the terminal.
 Update the package repository: sudo apt update.
 Install PHP: sudo apt install php.
 Verify the installation: php -v.

5.3.3 On Mac

1. Install Homebrew: If you don't have Homebrew installed, open Terminal and install it with the following command:

/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"

2. Install PHP: Once Homebrew is installed, you can install PHP by running:

brew install php

This command installs PHP and any necessary dependencies.

3. **Verify Installation**: After installation completes, verify PHP by typing php -v in the terminal. This command should display the PHP version information if PHP is installed correctly.

5.4 PHP USING XAMPP SERVER

Instead of installing PHP individually, you can use XAMPP, which includes PHP along with other essential software packages. XAMPP is a free and open-source cross-platform web server solution stack package developed by Apache Friends. It primarily consists of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in PHP and Perl. XAMPP stands for:

- **X** (Cross-platform)
- **A** (Apache)
- **M** (MariaDB)
- **P** (PHP)
- **P** (Perl)

Installing XAMPP on Windows

1. Download XAMPP:

- Visit the official XAMPP website.
- Download the latest version of XAMPP for Windows.

2. Run the Installer:

- Locate the downloaded file (usually in your Downloads folder) and double-click it to run the installer.
- You might receive a warning about User Account Control (UAC). Click "Yes" to continue.

3. Select Components:

- The installer will prompt you to select components you want to install. By default, all components are selected. You can customize the installation by deselecting components you don't need.

4. Choose Installation Folder:

- Choose the folder where you want to install XAMPP (default is C:\xampp). Click "Next" to proceed.

5. Start the Installation:

- Click "Next" and then "Install" to begin the installation process. It may take a few minutes to complete.

6. Complete the Installation:

Once the installation is complete, you can launch the XAMPP Control Panel by checking the box that says "Do you want to start the Control Panel now?" and then click "Finish".

Writing and Running a PHP File in XAMPP

1. Open XAMPP Control Panel:

- Launch the XAMPP Control Panel (you can find it in the start menu or the installation folder).
- Start the Apache server by clicking the "Start" button next to Apache.

2. Create a PHP File:

- Open a text editor (like Notepad or a code editor like VS Code).
- Write your PHP code. For example, create a simple PHP file with the following content:

```
<?php

echo "Hello, World!";

?>
```

3. Save the PHP File:

- Save the file with a .php extension. For example, save it as hello.php.
- Save the file in the htdocs directory inside your XAMPP installation folder. By default, this is located at C:\xampp\htdocs.

4. Run the PHP File:

- Open your web browser.
- Type http://localhost/hello.php in the address bar and press Enter.
- You should see "Hello, World!" displayed in your browser.

5.5 PHP USING WAMP SERVER

WAMP stands for Windows, Apache, MySQL, and PHP. It is a software stack for Windows that allows you to create and manage web applications locally. With WAMP, you can develop, test, and deploy websites and web applications in a local environment before making them live.

Installing WAMP Server on Windows

1. **Download WAMP Server:**

    - Go to the WAMP Server official website.
    - Download the latest version of WAMP Server suitable for your Windows version (32-bit or 64-bit).

2. **Run the Installer:**

    - Double-click the downloaded executable file.
    - Follow the installation wizard steps.
    - Choose the default options unless you have specific preferences.
    - Select the directory where you want to install WAMP (e.g., C:\wamp64).

3. **Configure WAMP:**

    - During installation, you'll be asked to choose your default browser and text editor. Select the ones you prefer.
    - Allow WAMP Server to access your network if prompted by Windows Firewall.

4. **Finish Installation:**

    - Complete the installation process.
    - Launch WAMP Server by clicking on the desktop shortcut or from the Start menu.

5. **Verify Installation:**

    - After launching WAMP Server, look for the WAMP icon in the system tray. It should be green, indicating that the server is running correctly.
    - Open your web browser and type http://localhost/ in the address bar. You should see the WAMP Server homepage.

Writing and Running a PHP File

1. **Create a PHP File:**

    - Open your text editor (e.g., Notepad++, Sublime Text).
    - Write a simple PHP script. For example:

```
<?php

    echo "Hello, World!";

?>
```

- Save the file with a .php extension, such as hello.php.

Troubleshooting Tips

- If the WAMP icon is not green, it means the server is not running correctly. Check for errors in the Apache logs located in the C:\wamp64\logs directory.
- Ensure no other applications are using port 80, which is the default port for Apache. Applications like Skype can sometimes interfere.
- If you make changes to your PHP files or configuration, remember to restart the WAMP server to apply the changes.

5.6 BASIC PHP SYNTAX

PHP is embedded within HTML and starts with `<?php` and ends with `?>`. The PHP code is processed on the server, and the result is sent to the client's web browser as plain HTML. This allows developers to create dynamic content that interacts with databases and other data sources.

Basic PHP Structure

Every PHP script should start with the PHP opening tag `<?php` and end with the closing tag `?>`.

```
<?php

// Your PHP code goes here

?>
```

PHP and HTML Integration

PHP can be easily embedded within HTML. This allows you to create dynamic web pages that can display different content based on user interaction or other variables.

Example:

```
<!DOCTYPE html>

<html>

<head>

<title>Welcome Page</title>

</head>

<body>

<h1><?php echo "Welcome to Gauhati University Centre for Distance and Online Education!"; ?></h1>

</body>

</html>
```

In this example, the PHP code is embedded within the HTML `<h1>` tag. When the page is loaded, the PHP code is executed on the server, and the result ("Welcome to Gauhati University Centre for Distance and Online Education!") is sent to the browser as HTML.

Comments in PHP

Comments are used to explain the code and are ignored by the PHP engine. PHP supports three types of comments:

- Single-line comments: Use `//` or `#`
- Multi-line comments: Use `/* ... */`

Example:

```
<?php

// This is a single-line comment
```

```
# This is also a single-line comment

/*

This is a multi-line comment

that spans multiple lines.

*/

echo "Comments are not displayed!";

?>
```

Echo and Print Statements:

The `echo` and `print` statements are used to output data to the browser. While they are similar, `echo` can take multiple parameters, while `print` can only take one argument and always returns 1.

Example:

```
<?php

echo "Welcome to Gauhati University Centre for Distance and Online Education!";

print " This is a great place to learn.";

?>
```

Case Sensitivity:

PHP keywords (such as `if`, `else`, `while`, `echo`, etc.), functions, and class names are not case-sensitive. However, variable names are case-sensitive.

Basic PHP Script Example

Below is a complete PHP script demonstrating basic syntax, comments, and HTML integration.

```
<!DOCTYPE html>

<html>

<head>

<title>PHP Basics</title>

</head>

<body>

<?php

// Display a welcome message

echo "Welcome to Gauhati University Centre for Distance and Online Education!";

/* This is a multi-line comment

   It will not be displayed in the output

*/

// Define a variable

$university = "Gauhati University";

// Display the variable's value

echo "<p>$university Centre for Distance and Online Education</p>";

?>

</body>

</html>
```

In this script:

- PHP code is embedded within an HTML document.

- A welcome message is displayed using `echo`.
- A variable `$university` is defined and used to display a complete message within an HTML `<p>` tag.

5.7 PHP VARIABLES AND DATA TYPES

Variables in PHP are used to store data, and they start with a $ sign followed by the name of the variable.

Data Types:

- String: Sequence of characters
- Integer: Whole numbers
- Float: Decimal numbers
- Boolean: True or false
- Array: Collection of values
- Object: Instances of classes
- NULL: Variable with no value

Example:

```
<?php
$string = "Gauhati University Centre for Distance and Online Education";
$integer = 2024;
$float = 3.14;
$boolean = true;
echo $string; // Outputs: Gauhati University Centre for Distance and Online Education
echo $integer; // Outputs: 2024
echo $float; // Outputs: 3.14
echo $boolean; // Outputs: 1 (true)
?>
```

5.8 PHP OPERATORS

Operators in PHP are used to perform operations on variables and values.

Types of Operators:

- Arithmetic Operators: +, -, *, /, %
- Assignment Operators: =, +=, -=, *=, /=
- Comparison Operators: ==, !=, >, <, >=, <=
- Logical Operators:&&, ||, !

Example:

```
<?php
$a = 10;
$b = 20;
// Arithmetic
$sum = $a + $b; // 30
// Comparison
$isEqual = ($a == $b); // false
// Logical
$isBothTrue = ($a < $b && $b > 15); // true
echo $sum; // Outputs: 30
echo $isEqual; // Outputs: (nothing, as it's false)
echo $isBothTrue; // Outputs: 1 (true)
?>
```

5.9 PHP CONTROL STRUCTURES

5.9.1 Conditional Statements

Conditional statements are used to perform different actions based on different conditions.

Types:

- **if:** Executes code if a condition is true
- **else:** Executes code if the same condition is false
- **elseif:** Executes code if the first condition is false and the second is true
- **switch:** Selects one of many blocks of code to be executed

Example:

```
<?php
$university = "Gauhati University Centre for Distance and Online Education";
if ($university == "Gauhati University Centre for Distance and Online Education") {
   echo "Welcome, Gauhati University student!";
} else {
   echo "Unknown institution.";
}
?>
```

5.9.2 Looping Statements

Loops are used to execute a block of code repeatedly as long as a specified condition is true.

Types:

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

1. while loop:

Example:

```
<?php
$i = 1;
while ($i <= 5) {
   echo "Iteration $i <br>";
   $i++;
}
?>
```

Explanation:

- **Initialization:** $i = 1; - A variable $i is initialized with the value 1.
- **Condition:** while ($i <= 5) - The loop will continue to execute as long as the value of $i is less than or equal to 5.
- **Body:** echo "Iteration $i
"; - Inside the loop, the current value of $i is printed along with the string "Iteration " and a line break (
).
- **Increment:** $i++; - The value of $i is incremented by 1 after each iteration.

Output:

Iteration 1
Iteration 2
Iteration 3
Iteration 4
Iteration 5

2. for Loop

Example:

```
<?php
for ($j = 1; $j <= 5; $j++) {
   echo "Iteration $j <br>";
}
?>
```

Explanation:

- **Initialization:** for ($j = 1; - A variable $j is initialized with the value 1.
- **Condition:** $j <= 5; - The loop will continue to execute as long as the value of $j is less than or equal to 5.
- **Increment:** $j++) - The value of $j is incremented by 1 after each iteration.

- **Body:** echo "Iteration $j
"; - Inside the loop, the current value of $j is printed along with the string "Iteration " and a line break (
).

Output:

Iteration 1

Iteration 2

Iteration 3

Iteration 4

Iteration 5

3. foreach Loop

Example:

```
<?php
$courses = array("PHP", "JavaScript", "HTML");
foreach ($courses as $course) {
   echo "Course: $course <br>";
}
?>
```

Explanation:

- **Initialization:** $courses = array("PHP", "JavaScript", "HTML"); - An array $courses is created with three elements: "PHP", "JavaScript", and "HTML".
- **Loop:** foreach ($courses as $course) - The foreach loop iterates over each element in the $courses array. For each iteration, the current element is assigned to the variable $course.
- **Body:** echo "Course: $course
"; - Inside the loop, the current value of $course is printed along with the string "Course: " and a line break (
).

Output:

Course: PHP

Course: JavaScript

Course: HTML

CHECK YOUR PROGRESS 1

1. Choose the correct option:

A. Which of the following scenarios best demonstrates the advantage of using PHP variables in a dynamic web application for Gauhati University Centre for Distance and Online Education?
a) Displaying a static webpage
b) Storing user input data temporarily for processing
c) Hardcoding content directly into HTML
d) Using CSS for styling elements

B. Given the PHP code $x = 10; $y = 20; $result = $x + $y; echo $result;, what would be the output and why?

a) 30, because PHP performs arithmetic addition correctly.
b) 1020, because PHP concatenates strings by default.
c) 10, because PHP only recognizes the first variable.
d) 0, because PHP does not support arithmetic operations.

C. How can the use of conditional statements in PHP improve the functionality of a feedback form on Gauhati University's website?

a) By making the form aesthetically pleasing
b) By validating user input and displaying appropriate messages
c) By ensuring the form is always visible
d) By storing form data directly in CSS files

D. Consider the following PHP code snippet:

```
$a = "Gauhati";
$b = " University";
$c = $a . $b;
echo $c;
```

What is the output and what does it illustrate about PHP's string handling capabilities?

a) Gauhati University, illustrating string concatenation.
b) Gauhati, illustrating string assignment.
c) University, illustrating string replacement.
d) a b, illustrating variable misassignment.

E. Why is it important to understand PHP's data types and operators when developing a database-driven website for Gauhati University Centre for Distance and Online Education?

a) To apply appropriate CSS styles to the website
b) To ensure correct data manipulation and logic implementation
c) To guarantee the website loads faster
d) To prevent the use of JavaScript in the application

5.10 PHP FUNCTIONS

Functions are blocks of code that can be reused. They are used to perform a specific task.

Syntax:

```
function functionName() {
   // Code to be executed
}
```

Example:

```
<?php
function welcomeMessage() {
   return "Welcome to Gauhati University Centre for Distance and Online Education!";
}
echo welcomeMessage(); // Outputs: Welcome to Gauhati University Centre for Distance and Online Education!
?>
```

Functions with Parameters:

Example:

```
<?php
function greetStudent($name) {
   return "Hello, $name! Welcome to Gauhati University Centre for Distance and Online Education!";
}
echo greetStudent("Rajib"); // Outputs: Hello, Rajib! Welcome to Gauhati University Centre for Distance and Online Education!
?>
```

16.11 PHP and MySQL Connectivity

Connecting PHP with MySQL allows for dynamic data handling and storage in web applications. This section covers connecting to MySQL, creating databases and tables, and performing CRUD operations (Create, Read, Update, Delete) using PHP.

5.11.1 Connecting to MySQL

Connecting to the Database

To connect PHP to a MySQL database, use the mysqli extension. The connection requires the server name, username, password, and database name.

Example:

```
<?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "GUCDOE";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
   die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully to GUCDOE database";
?>
```

Explanation:

1. Database Credentials:

```
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "GUCDOE";
```

- $servername: The MySQL server address (usually localhost for local development).
- $username: The MySQL username (commonly root for local development).
- $password: The password for the MySQL user.

- $dbname: The name of the database (GUCDOE).

2. Create Connection:

$conn = new mysqli($servername, $username, $password, $dbname);

- new mysqli(...): Creates a new connection to the MySQL server using the provided credentials.

3. Check Connection:

```
if ($conn->connect_error) {
   die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully to GUCDOE database";
```

 Checks if there was a connection error.
 Terminates the script if the connection fails and outputs the error message.
 Outputs a success message if the connection is established.

5.11.2 Creation of Database and Table

Creating a Database

To create a database, use the CREATE DATABASE SQL statement.

Example:

```
<?php
$sql = "CREATE DATABASE GUCDOE";
if ($conn->query($sql) === TRUE) {
   echo "Database created successfully";
} else {
   echo "Error creating database: " . $conn->error;
}
// Select the database
$conn->select_db("GUCDOE");
?>
```

Explanation:

- CREATE DATABASE GUCDOE: SQL statement to create a database named GUCDOE.
- $conn->query($sql): Executes the SQL statement.
- Checks if the database was created successfully.

Creating a Table

To create a table, use the CREATE TABLE SQL statement.

Example:

```
<?php
$sql = "CREATE TABLE students (
   id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
   name VARCHAR(50) NOT NULL,
   course VARCHAR(100),
   reg_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
ON UPDATE CURRENT_TIMESTAMP
)";

if ($conn->query($sql) === TRUE) {
   echo "Table students created successfully in GUCDOE database";
} else {
   echo "Error creating table: " . $conn->error;
}
?>
```

Explanation:

1. Create Table:

```
$sql = "CREATE TABLE students (
   id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
   name VARCHAR(50) NOT NULL,
   course VARCHAR(100),
```

```
  reg_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
ON UPDATE CURRENT_TIMESTAMP
)";
```

- Creates a table named students with columns for id, name, course, and reg_date.
- id INT(6) UNSIGNED AUTO_INCREMENT PRIMARY KEY: An auto-incrementing primary key.
- name VARCHAR(50) NOT NULL: A variable-length string for the student's name.
- course VARCHAR(100): A variable-length string for the course.
- reg_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP: Automatically sets and updates the registration date to the current timestamp.

2. Execute Query:

```
if ($conn->query($sql) === TRUE) {
  echo "Table students created successfully in GUCDOE database";
} else {
  echo "Error creating table: " . $conn->error;
}
```

- Executes the SQL statement to create the table.
- Checks if the table was created successfully.

5.11.3 Inserting Records

To add data to the table, use the INSERT INTO SQL statement.

Example:

```
<?php
$sql = "INSERT INTO students (name, course) VALUES ('Arjun Singh', 'B.Sc in Computer Science from Gauhati University')";
```

```
if ($conn->query($sql) === TRUE) {
   echo "New record for Arjun Singh created successfully";
} else {
   echo "Error: " . $sql . "<br>" . $conn->error;
}
?>
```

Explanation:

- INSERT INTO students (name, course) VALUES ('Arjun Singh', 'B.Sc in Computer Science from Gauhati University'): SQL statement to insert a new record into the students table.
- The name field is set to "Arjun Singh" and the course field is set to "B.Sc in Computer Science from Gauhati University".
- Executes the SQL statement.

5.11.4 Updating Records

To update existing data, use the UPDATE SQL statement.

Example:

```
<?php
$sql = "UPDATE students SET course='M.Sc in IT from Gauhati University' WHERE name='Arjun Singh'";

if ($conn->query($sql) === TRUE) {
   echo "Record for Arjun Singh updated successfully";
} else {
   echo "Error updating record: " . $conn->error;
}
?>
```

Explanation:

- UPDATE students SET course='M.Sc in IT from Gauhati University' WHERE name='Arjun Singh': SQL statement to

update the course field to "M.Sc in IT from Gauhati University" for the record where the name is "Arjun Singh".

- Executes the SQL statement.

5.11.5 Deleting Records

To remove data from the table, use the DELETE SQL statement.

Example:

```
<?php
$sql = "DELETE FROM students WHERE name='Arjun Singh'";
if ($conn->query($sql) === TRUE) {
   echo "Record for Arjun Singh deleted successfully";
} else {
   echo "Error deleting record: " . $conn->error;
}
?>
```

Explanation:

- DELETE FROM students WHERE name='Arjun Singh': SQL statement to delete the record where the name is "Arjun Singh".
- Executes the SQL statement.

5.11.6 Displaying Records

To retrieve and display data, use the SELECT SQL statement.

Example:

```
<?php
$sql = "SELECT id, name, course FROM students";
$result = $conn->query($sql);
if ($result->num_rows > 0) {
   // Output data of each row
   while($row = $result->fetch_assoc()) {
      echo "id: " . $row["id"]. " - Name: " . $row["name"]. " - Course: " . $row["course"]. "<br>";
```

```
    }
} else {
    echo "0 results";
}
?>
```

Explanation:

1. Select Data:

```
$sql = "SELECT id, name, course FROM students";
$result = $conn->query($sql);
```

- SQL statement to select the id, name, and course columns from the students table.
- Executes the SQL statement and stores the result in the $result variable.

2. Check and Display Results:

```
if ($result->num_rows > 0) {
    while($row = $result->fetch_assoc()) {
        echo "id: " . $row["id"]. " - Name: " . $row["name"]. " -
Course: " . $row["course"]. "<br>";
    }
} else {
    echo "0 results";
}
```

- Checks if there are any rows returned by the query.
- Iterates over each row in the result set.

- Outputs the id, name, and course fields of each row.

CHECK YOUR PROGRESS 2:

2. Fill in the gaps:

(a) In a while loop, the condition is checked _____ the loop executes the block of code.

(b) The for loop consists of three parts: initialization, condition, and _____.

(c) In a foreach loop, you can iterate over each element in an _____.

(d) To update an existing record in a MySQL table using PHP, you use the _____ SQL statement.

(e) To retrieve and display data from a MySQL database in PHP, the _____ SQL statement is used.

5.12 SUMMING UP

- PHP (Hypertext Preprocessor) is an open-source scripting language designed for web development.
- PHP can be installed on various operating systems including Windows, Linux, and Mac.
- XAMPP and WAMP are popular server solutions that include PHP, enabling local development environments.
- Basic PHP syntax involves embedding PHP code within HTML using <?php and ?> tags.
- PHP supports various data types such as strings, integers, floats, booleans, arrays, objects, and NULL.
- PHP operators include arithmetic, assignment, comparison, and logical operators.

- Control structures in PHP include conditional statements and loops for decision making and iterative operations.
- PHP can connect to MySQL databases to perform CRUD operations, enhancing dynamic web development.

5.13 ANSWERS TO CHECK YOUR PROGRESS

1.

A. b) Storing user input data temporarily for processing
B. a) 30, because PHP performs arithmetic addition correctly.
C. b) By validating user input and displaying appropriate messages
D. a) Gauhati University, illustrating string concatenation.
E. b) To ensure correct data manipulation and logic implementation

2.

(a) before
(b) increment
(c) array
(d) UPDATE
(e) SELECT

5.14 POSSIBLE QUESTIONS

Short answer type questions:

1. What is PHP and what are its main uses?
2. How do you install PHP on a Windows system?
3. Explain the purpose of the echo statement in PHP.
4. What are PHP variables and how are they declared?
5. Describe the role of arrays in PHP.
6. How do you connect PHP to a MySQL database?
7. What is the significance of the if statement in PHP?
8. How do you create a simple function in PHP?

9. What is the use of the foreach loop in PHP?
10. Explain the concept of PHP operators with an example.

Long answer type questions:

1. Describe the process of installing XAMPP and writing a basic PHP file to display "Hello World!".
2. Explain PHP control structures with suitable examples.
3. How do you perform CRUD operations (Create, Read, Update, Delete) using PHP and MySQL? Provide code examples.
4. Discuss the integration of PHP and HTML to create dynamic web pages, using examples relevant to Gauhati University Centre for Distance and Online Education.
5. Illustrate with examples the different data types in PHP and how they are used in web development.

5.15 FURTHER READINGS

1. "Learning PHP, MySQL & JavaScript" by Robin Nixon
2. "PHP and MySQL Web Development" by Luke Welling and Laura Thomson
3. "Head First PHP & MySQL" by Lynn Beighley and Michael Morrison

---×---

UNIT- 6

DISTRIBUTED OBJECT BASED MODELS

Unit Structure:

6.1 INTRODUCTION

We have learnt from the previous chapters that a Distributed system must have the capability of distributing data, functions and modules freely and transparently defined on an application structure. In distributed object based systems, the concept of object plays a very crucial role in implementing distribution transparency. In these systems everything is treated as an object. Client-Server systems are one of the many ways of implementing distributed systems or applications where clients are offered services and resources in the form of objects that they can invoke. Distributed objects form an important paradigm because it is relatively easy to hide distribution aspects behind an object's interface.

6.2 OBJECTIVES

In this unit you will be able to:

- Understand a Distributed object based system Model
- Learn different Distributed object based models like: CORBA, DCOM,EJB
- Understand the architectures of CORBA, DCOM and EJB.
- Explain the features of CORBA, DCOM and EJB
- Understand the basic features of EJB.

6.3 DISTRIBUTED OBJECT BASED MODELS

The distributed object based models that we are going to study in this unit are: CORBA, DCOM and EJB. Let's start with the very first model which is CORBA. CORBA stands for Common Object Request Service Broker Architecture. It is an industry defined standard for distributed systems. Though CORBA is replaced by other advanced models, it is still used today in many industries, such as telecommunications, finance, and government, where the cost and risk of replacing them with newer technologies may be high.

6.4 CORBA

The architecture of CORBA stick to it's reference model OMG (Object Management Group) which was laid down in 1997. OMG consists of four architectural elements connected to Object Request Broker (ORB). ORB is responsible for carrying out all the

communications that take place between the objects and their clients while concealing the issues related distribution and heterogeneity. The global architecture of CORBA is depicted in Fig 6.1:

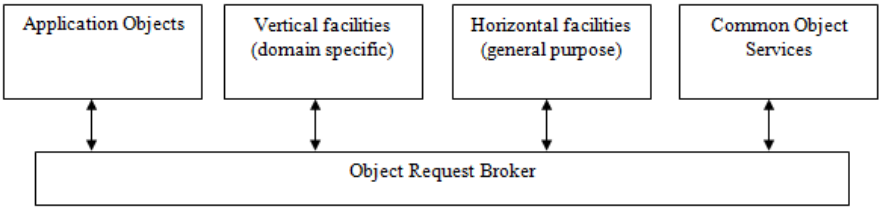


Fig 6.1: The global architecture of CORBA

Horizontal services consist of general purpose high level services that are independent of application domains such as those for user interfaces, information management system and task management. On the other Vertical facilities consist of general purpose high level services that are independent of application domain such as e-commerce, banking, manufacturing etc.

Object Model

As CORBA is organized as a collection of clients and object servers, the general organization of CORBA system is shown in Fig 6.2:

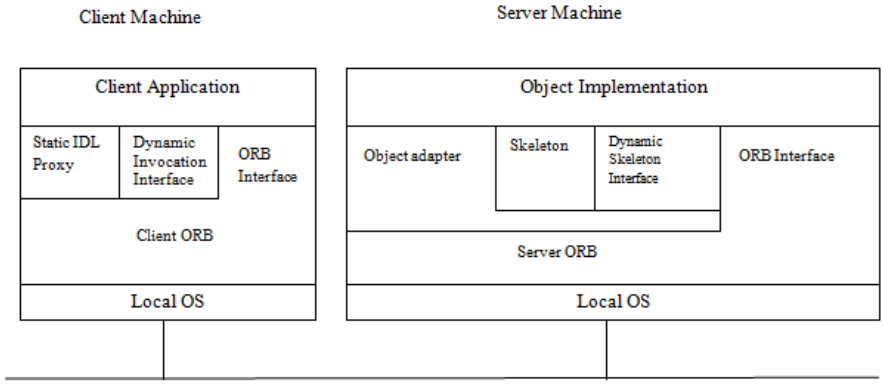


Fig 6.2: The general organization of a CORBA system

Objects and services are specified in the CORBA Interface Definition Language (IDL). CORBA IDL is similar to other IDL's providing precise syntax for expressing methods and their parameters. Interface specifications can only be given by means of IDL. Beneath any process in CORBA lies the ORB. ORB is the runtime system that is responsible for the communication between the client and an object. It ensures that an invocation is sent to the object's server and that the reply is passed back to the client. An ORB also offer operations to marshall and unmarshal object

references so that they can be exchanged between processes as well as operations for comparing references. Another service offered by ORB is that it provides a means to obtain an initial reference to an object implementing a specific CORBA service.

A Dynamic Invocation Interface (DII) is offered by CORBA to clients for allowing them to construct an invocation request at runtime. CORBA provides an object adapter to take care of forwarding incoming request to the proper object. As we can see in Fig 2, the skeleton in CORBA are the stubs that are responsible for unmarshaling at the server side. When using a dynamic skeleton, an object will have to provide the proper implementation of the invoke function as offered to the client.

CORBA Services

The list of services offered by CORBA is listed in Fig 6.3:

Service	Description
Collection	Provides the means to group objects into lists, queues, stacks, sets etc
Query	Provides the means to query a collection of objects using a declarative query language
Concurrency	Provides locking mechanisms by which clients can access shared objects
Transaction	Allows a client to define a series of method invocations across multiple objects in a single transaction
Event	Provides the means by which clients and objects can be interrupted on occurrence of a specified event
Notification	Advanced facilities for asynchronous communication
Externalization	Deals with marshalling objects in such a way that they can be stored on disk or sent across a network
Life Cycle	Provides means for creating, destroying, copying and moving objects
Licensing	Allows developers of objects to attach a license to their object and enforce a specific licensing policy
Naming	A means by which objects can be given human relatable name that maps to the object's identifier
Property	Allows clients to associate (attribute, value)-pairs with objects

Trading	Allows objects to advertise and find the services that it has to offer
Persistence	Offers facilities for storing information on disk in the form of storage objects
Relationship	Offers services to explicitly relate two or more objects
Security	Mechanisms for secure channels, authorization, auditing etc.
Time	Returns the current time within specified error ranges

Fig 6.3: Services offered by CORBA

6.4.1 Communication

The mode of communication in CORBA was a very simple one in which the client invokes a method on an object and waits for an answer. This model was thought to be too simple and so later on some communication facilities were added to enhance the model. First, let us go through the object invocation models

Object Invocation Models

The three invocation models supported by CORBA are:

- Synchronous: This mode is useful in cases when the client expects an acknowledgement or response. When proper response is received CORBA guarantees that the method has been invoked exactly once. If no response is received, the client simply invoke the method and continue with it's own execution as soon as possible.
- One-way: In this asynchronous invocation, no guarantee is given to the caller that the invocation request is delivered to the object's server as it doesn't gives any response.
- Deferred synchronous: In this mode a request is a combination of one way request and letting the server asynchronously send the result back to the client. As soon as the client sends its request to the server, it continues without waiting for any response from the server. Thus a client can never learn for sure whether it's request is actually delivered to the server.

Event and Notification Services

Event service deals with signalling the occurrence of an event so that the clients amenable to that event could take appropriate action. An event is produced by a supplier and received by a consumer. An

event channel delivers the events which is placed in between the suppliers and the consumers as shown in Fig 6.4. This type of event model is referred to as push model. In this model when an event occurs, supplier produces the event and pushes it to the consumers through the event channel

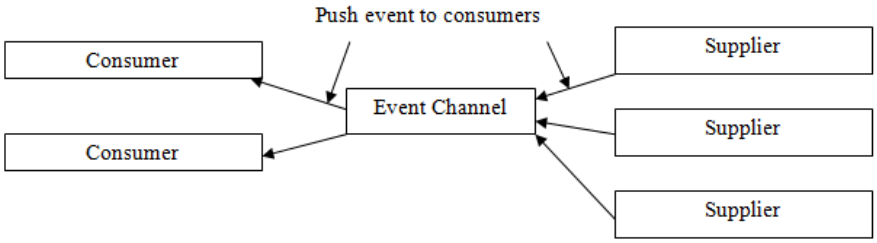


Fig 6.4: Logical organization of suppliers and consumers of events following the push style model

There is also an alternative model, the pull model shown in Fig 6.5 where the consumers poll the event channel to check whether an event has occurred and the event channel in turn polls the various suppliers.

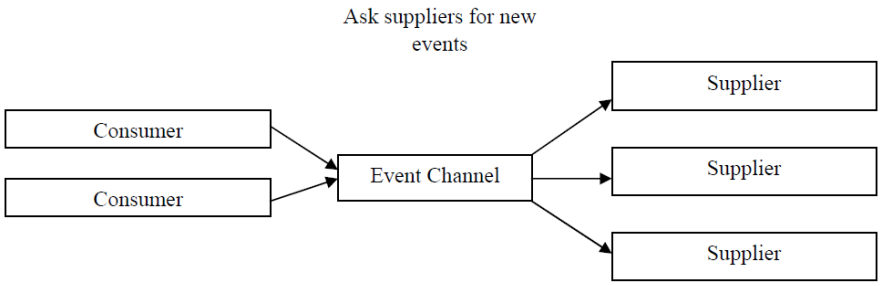


Fig 6.5: Pull Model for event delivery in CORBA

Communication Models

For persistent communication, CORBA supports the message queuing model through an additional messaging service.

In callback model, a client provides an object with an interface containing callback methods which can be called by the underlying communication system to pass the result of an asynchronous invocation.

In polling model, the client is offered a collection of operations to poll its ORB for incoming result

Interoperability
CORBA uses General Inter-ORB Protocol (GIOP) as it's standard to overcome the lack of interoperability which existed in early versions of CORBA. It helps in communication between the server and the client that adheres to this standard protocol.

6.4.2 Processes
Client and server are the two types of processes that are distinguished by CORBA. The client- side software of CORBA is kept minimum by compiling the IDL specifications of an object into a proxy that marshals invocation requests for example IIOP request messages and unmarshals the corresponding reply messages into results that are handed back to the invoking client. The task of the proxies in CORBA is to connect a client application to the underlying ORB. A client can also dynamically invoke objects through the DII other than only generating an object-specific proxy.
On the other hand, servers were initially left open to a variety of applications and were given minimal support in the form of a basic object adapter.

6.4.3 Naming
CORBA also has a naming service that allows client for using a character-based name. A name in CORBA consists of a sequence of name components which is of the form (id, kind) pair. Here id and kind are both strings. For example the object name "dayne" could have kind "dir" telling that it is a directory object. The name components in CORBA do not have a separator instead names are to be passed explicitly as a sequence of name components. The representation of the sequence is dependent on the language but remains opaque to a client that calls the naming service.
The structure of a naming graph does not have any restrictions and each node in the naming graph is treated as an object. A naming context is an object that stores a table mapping name components to object references and it does not have any root context.

6.4.4 Synchronization
Synchronization in CORBA is facilitated by the concurrency control and transaction service.
Concurrency control provides the locking service by using a central lock manager. It provides locks such as read and write locks and also supports many other locks that are applicable at different granularities often needed in a database. A transaction in CORBA is initiated by a client and consists of a series of invocations on objects. An object when invoked for the first time automatically

becomes part of the transaction. As the object is participating in a transaction at present, a notification is sent to the object's server. This information is implicitly passed to the server when invoking the object.

6.4.5 Caching and Replication

CORBA doesn't offer any generic caching and replication services. The lack of support of these services implies that the application developers will have to resort to an ad-hoc approach when replication is needed. This requirement is met by CASCADE. The goal of CASCADE system is to provide a generic, scalable mechanism that allows any kind of CORBA object to be cached. The caching service offered by CASCADE is implemented as a potentially large collection of object servers each referred to as Domain Caching Server(DCS). Each DCS is an object server running on a CORBA ORB. The object servers may spread across a wide area network such as the Internet.

6.4.6 Fault Tolerance

Fault tolerance is achieved in CORBA through replication by replicating objects into object groups. These groups consists of one or more identical copies of the same object. An object group can be referred as if it is a single object. Replication is transparent to the clients by offering the same interface as the replicas it contains. The object groups are referenced by using a special kind of IOR called an Interoperable Object Group Reference (IOGR). The IOGR contains multiple references to different objects.

STOP TO CONSIDER

CORBA can be used to build a language-independent, O/S-independent distributed system and also useful when building a plain old distributed object system

CHECK YOUR PROGRESS

1. A _____ is offered by CORBA to clients for allowing them to construct an invocation request at runtime.

2. An event channel delivers the _____ which is placed in between the suppliers and the consumers

3. In_____ model, the client is offered a collection of operations to poll its ORB for incoming result

4. The name components in CORBA do not have a _____ instead names are to be passed explicitly as a sequence of name components

6.5 Distributed COM

As the name suggests Distributed COM(DCOM) is an extension of Component Object Model. DCOM is popular in the Windows Operating System in a networked environment. Compared to DCOM, CORBA or any other distributed system has still a long way to go.

Overview of DCOM

The basis of DCOM is formed by Microsoft's component object technology COM. The goal of COM is to support the development of components that can be dynamically activated and that can interact with each other. A component in COM is executable code either contained in a library or in the form of an executable program. COM services are offered in the form of a library which is linked to a process. It was originally developed to support the compound documents. Microsoft supports a myriad of compound documents, so it needs a general way to distinguish between the various parts and be able to glue them together. This led to the formation of OLE (Object Linking and Embedding). This version was soon modified to a morenewer version which was built on top of COM which is a more flexible layer. The general organization is shown in Fig 6.6

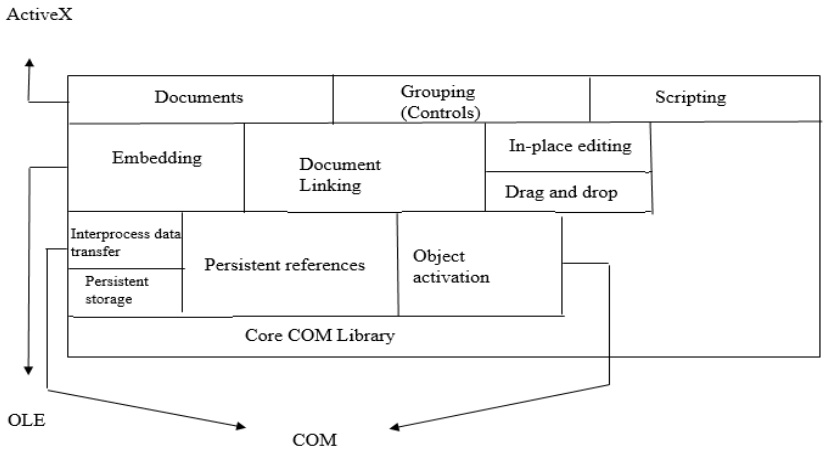


Fig 6.6: The general organization of ActiveX, OLE and COM

In the figure we can see ActiveX, which is the new term given to cover everything with all the new features like flexible capabilities for starting components in different processes, support for scripting and a more or less standard grouping of objects which is called as ActiveX controls. DCOM helps a process to communicate with components that are placed on another machine. The mechanism for

exchange of information in DCOM is similar to COM and the differences that COM and DCOM exhibit are often hidden by various interfaces.

Object Model

The remote-object model is adopted by DCOM. The DCOM object model is simply an implementation of an interface where a single object can implement several interfaces at the same time. DCOM have only binary interfaces. The interfaces in DCOM has a unique 128 bit identifier called it's Interface Identifier (IID). No two interface can have the same IID as IID is globally unique. A DCOM object is created as an instance of a class. A class object in DCOM represents a collection of objects of the same type that implements the same set of interfaces. A class identifier (CLSID) is used to refer each class object in DCOM. The difference between CORBA and DCOM is that all objects in DCOM are transient which means as soon as the object has no more clients, the object is destroyed. Dynamic invocation of objects is also supported by DCOM.

DCOM Architecture

The overall architecture of DCOM is shown in Figure 6.7:

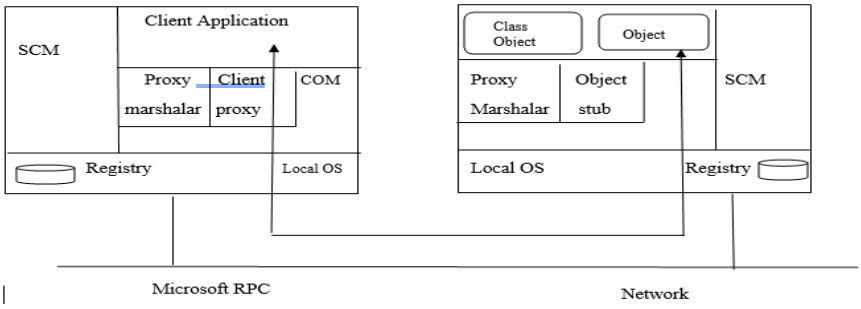


Fig 6.7: Architecture of DCOM

The client side of DCOM consist of a process that is given access to the SCM and the registry to help look for and set up a binding to a remote object. A proxy is being offered by the client implementing the object's interface. The object server consists of a stub for marshalling and unmarshalling invocations, which are passed to the actual object. By means of RPC communication between the client and the server takes place. DCOM offers other communication primitives as well.

DCOM Services

DCOM is nothing but extension of COM. Again COM has evolved itself into a more newer version COM+. The services that are

specifically offered by DCOM can be understood with the help of the figure given below:

CORBA Service	DCOM/COM+ Service	Windows 2000 Service
Collection	ActiveX Data Objects	-
Query	None	-
Concurrency	Thread concurrency	-
Transaction	COM+ Automatic Transactions	Distributed Transaction Coordinator
Event	COM+ Events	-
Notification	COM+ Events	-
Externalization	Marshalling utilities	-
Life Cycle	Class factories, JIT activation	-
Licensing	Special case factories	-
Naming	Monikers	Active Directory
Property	None	Active Directory
Trading	None	Active Directory
Persistence	Structured storage	Database access
Relationship	None	Database access
Security	Authorization	SSL, Kerberos
Time	None	None

Fig 6.8: Overview of DCOM services in comparison to CORBA services

6.5.1 Communication

Initially synchronous communication was in DCOM. Though different forms of invocation are now supported by DCOM, synchronous communication is still the default one.

Object Invocation Models

Pure synchronous invocations is offered by means of callback interfaces. Callback interfaces are supported by connectable objects which are special DCOM objects that can be handed a pointer to a callback interface as implemented by a client. The callback interface is specified by a connectable object, it expects that client will implement. When a connectable object is binded by the client, it must provide a pointer to the callback interface and when the

connectable object is on a remote host , the interface pointer needs to be marshalled and shipped to that machine. Asynchronous calls in DCOM require that the client and object are up and running. Communication in DCOM is transient and persistent communication is supported through message queuing.

Events

The basic idea of events in DCOM are very much same as the push-style event model in CORBA. An event is modelled by means of a method call having only input parameters. An event class is a group of events that are grouped together which is represented by a regular DCOM class object with it's own CLSID. An event class in DCOM is not required to be implemented separately as it provides a default implementation. After instantiating an event class into an object, a supplier can simply generate events by calling the appropriate methods of that object.

Let us assume that an event is represented by a method m_event. A subscriber will have to provide an implementation for that method. To subscribe, it passes a pointer to the interface that implements that method to the event system. Whenever the supplier calls m_event, the event system will take care that the version of m_event at each subscriber is also invoked. The process of event processing in DCOM is depicted in Figure 6.9

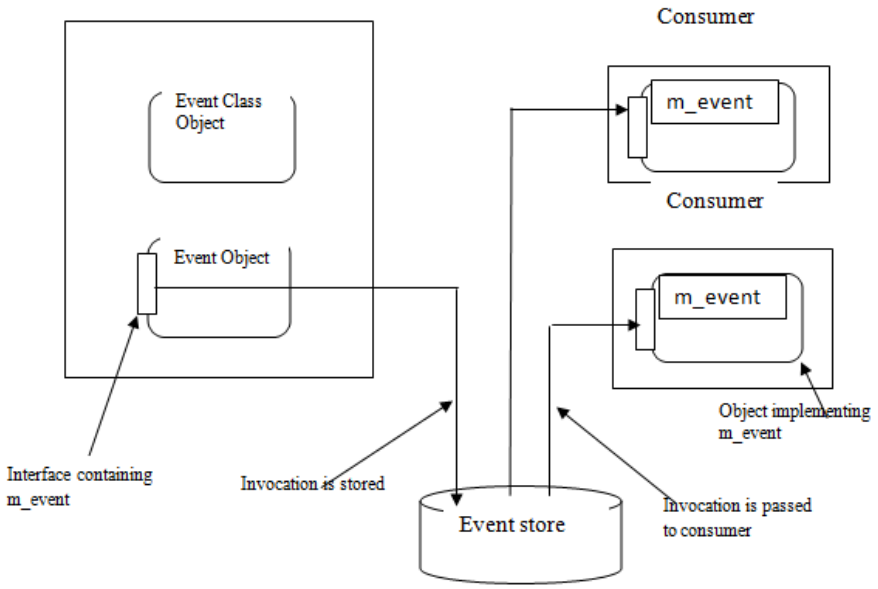


Fig 6.9: Event Processing in DCOM

The events in DCOM can be stored and so if a subscriber is not active when an event happens, it can later pick up the event if it wishes. If this situation arises, the event system instantiates the supplier's event class and replays the queued events by invoking the associated methods the associated methods, one after the other. In case of non-persistent events, the effect is that the associated event method is also called the subscriber.

Messaging

Persistent asynchronous communication is accomplished with the help of **Queued Components (QC).** QC is actually an interface to **Microsoft Message-Queuing (MSMQ).** An important characteristic of QC is that it hides MSMQ completely from clients and objects. In case of QC asynchronous calls are restricted to interfaces containing methods that have only input parameters excluding those that have return values or output parameters. QC offers only one-way asynchronous method invocations. A pointer to an interface containing methods that can be called asynchronously is returned to client when it binds to an object that supports QC. Whenever a method like this is invoked by a client, the invocation is automatically marshalled and stored locally by the QC subsystem at the client. When an interface is released by calling Release, the marshalled and the stored methods are sent across MSMQ thus indicating that the client has finished invoking methods.

Processes

In this section we will go through the organization of client and servers in DCOM respectively

Clients

One of the important issue in DCOM is handling object references. It's not enough for a process A to simply pass it's interface pointer, to pass an object reference to another process B. To pass an interface pointer it is necessary for process A to pass also the IID of that interface along with the location of the object as well as also the transport protocol that will be used. Process B is assumed to have the implementation of a proxy for that interface that is locally available, so it simply loads the implementation and starts invoking the methods. Sometimes this standard way of marshalling may not be the best way to handle invocation requests. DCOM also supports custom marshalling by which a developer can completely decide on how a proxy and it's associated object communicate. Passing an object reference with custom marshalling takes place by marshalling

the sender's proxy, transferring the marshalled proxy to the receiver and unmarshaling it again. Custom marshalling is depicted in Fig 6.10

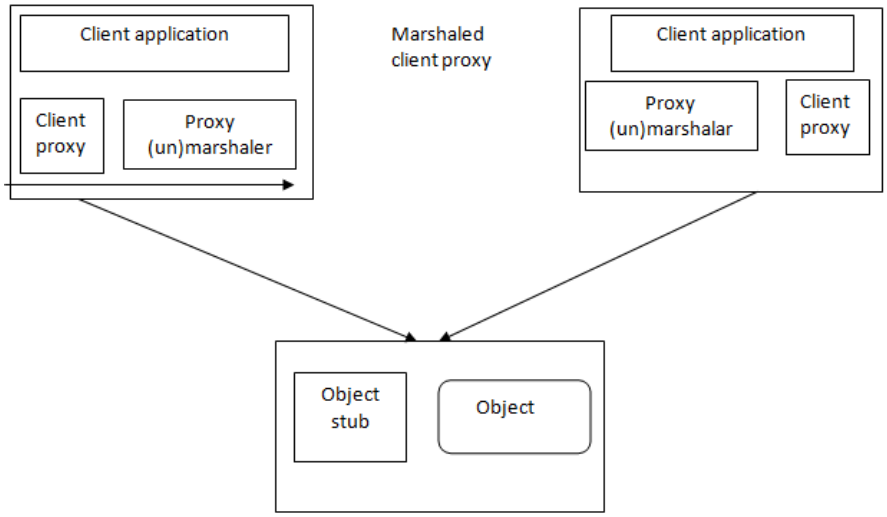


Fig 6.10: Custom Marshalling

Servers

A special Service Control Manager (SCM) will be executed by a host supporting DCOM objects. The SCM is responsible for activating objects. It works as follows. Let us assume that a client holds a CLSID of a class object. A new object is instantiated when it passes that CLSID to it's local registry where a lookup is done for the host on which the object server should be placed. The CLSID is passed to the SCM associated with that host. The CLSID is then looked up by the SCM in it's local registry to find the file to load the class object from which it can instantiate new objects. This approach lets the server to a developer handle all the issues related to managing objects. DCOM takes care of object management with the help of the facility JIT(just-in-time) activation which allows a server to take control of activating and destroying objects. With JIT activation, an object is activated by a request from a client to instantiate a new object from a given class and also a server may decide to destroy an object to make room for the new objects whenever it likes such as when it is running out of memory.

6.5.3 Naming

DCOM provides persistent object references called monikers that can be shared between multiple processes. DCOM has no high-level naming service but it makes use of the Windows Active Directory. Let's discuss the concept of monikers along with the Windows Active Directory service.

Monikers

A moniker contains all the necessary information to reconstruct the referenced object and bring that object into the state it's last client had left in. DCOM provides different kind of monikers which are discussed below:

File moniker: It refers to an object constructed from a file. The moniker will contain not only the path name of the file that is used to construct the object but also the CLSID of the class object that is needed to actually create the object from that class.

URL moniker: It refers to an object constructed from a URL.

Class moniker: It refers to class object.

Composite moniker: It refers to a composition of monikers.

Item moniker: It refers to a moniker in a composition

Pointer moniker: It refers to an object in a remote pass.

Active Directory

DCOM makes use of a worldwide directory service known as Active Directory. Active Directory is a system that is assumed to be partitioned into domains, where each domain consists of a number of users and resources. There is one or more domain controllers in every domain to keep track of the users and resources in a domain. A domain has an associated DNS name.Different domains are allowed by Active Directory to be grouped into trees and forests. The task of a domain tree is to form a subtree in the DNS name space whereas a domain forest simply groups a number of independent domains. The main benefit of having a domain group is that the Active Directory provides a single , global index for that group called a global catalog.

6.5.4 Synchronization

The synchronization mechanism in DCOM is implemented in the form of a transaction mechanism. This will be discussed in details in the Fault Tolerance section.

6.5.5 Replication

DCOM doesn't have any particular support for incorporating replication. The special services in Active Directory has the replication mechanism. In case of user applications, a developer will have to construct his own solutions.

6.5.6 Fault Tolerance

DCOM supports fault tolerance by means of automatic transactions, which forms an integral part of COM+. Automatic transactions allow a developer to specify that a series of method invocations, possibly on different objects, should be grouped into a transaction. Depending on the configuration of the respective class objects, transactions on objects are created automatically. A transaction attribute is present in each class object to determine the transactional behaviour of it's objects. The possible attribute values are:

Attribute_Value	Description
REQUIRES_NEW	A new transaction is always started at each invocation
REQUIRED	A new transaction is started if not already done so
SUPPORTED	Join a transaction only if caller is already part of one
NOT_SUPPORTED	Never join a transaction
DISABLED	Never join a transaction, even if told to do so

Fig 6.11: Transaction attribute values for DCOM objects

6.5.7 Security

DCOM handles security in two ways: Declarative security and Programmatic Security. Declarative security defines what an object requires with respect to activation control, access control and authentication for specifying in the registry. It is based on the use of roles and what credentials are required with respect to the role identification are also specified in the registry. Activation specifies which user or user group is allowed to create an instance of that class and access control specifies the access privileges for each user or user group. The several authentication levels used in DCOM are:

Authentication level	Description
NONE	No authentication is required
CONNECT	Authenticate client when first connected to server
CALL	Authenticate client at each invocation
PACKET	Authenticate all data packets
PACKET_INTEGRITY	Authenticate data packets and do integrity check
PACKET_PRIVACY	Authenticate, integrity check, and encrypt data packets

Programmatic security defines how the applications can set security levels as well as also choose between different security services. One of the important feature of programmatic security is that a process can also refer to the security service it expects to use to handle security. The five authentication services of programmatic security are depicted in Fig:

Service	Description
NONE	No authentication
DCE_PRIVATE	DCE authentication based on shared keys
DCE_PUBLIC	DCE authentication based on public keys

WINNT	Windows NT security
GSS_KERBEROS	Kerberos authentication

STOP TO CONSIDER

DCOM can transparently use a variety of network protocols, including User Datagram Protocol, NetBIOS, TCP/IP and Internetwork Packet Exchange/Sequenced Packet Exchange.

CHECK YOUR PROGRESS

5. COM services are offered in the form of a _____which is linked to a process.

6. DCOM helps a process to _____with components that are placed on another machine.

7. In DCOM a _____ is being offered by the client implementing the object's interface.

8. Communication in DCOM is _____and persistent communication is supported through message queuing.

9. DCOM provides persistent object references called _____that can be shared between multiple processes.

10. _____is a system that is assumed to be partitioned into domains, where each domain consists of a number of users and resources.

6.6 EJB (Enterprise Java Beans)

EJB are the software component module which was developed with the sole purpose to build/support enterprise specific problems and it is a reusable server-side software component. Enterprise JavaBeans facilitates the development of distributed Java applications, providing an object-oriented transactional environment for building distributed, multi-tier enterprise components. Being a remote object, it needs the services of an EJB container in order to execute. WORA

(Write Once Run Anywhere) is the primary goal of EJB. Enterprise JavaBeans takes a high-level approach to building distributed systems by making the application developer free to concentrate on programming only the business logic without the need to write all the “plumbing” code that's required in any enterprise application. This task is accomplished by the server vendor.

6.6.1 EJB Architecture

A typical EJB consists of the following components:

EJB server The EJB server contains the EJB container, which provides the services required by the EJB component. EA Server is an EJB server.

EJB client An EJB client usually provides the user-interface logic on a client machine. The EJB client makes calls to remote EJB components on a server and needs to know how to find the EJB server and how to interact with the EJB components. An EJB component can act as an EJB client by calling methods in another EJB component.

An EJB client does not communicate directly with an EJB component. The container provides proxy objects that implement the components home and remote interfaces. The component’s remote interface defines the business methods that can be called by the client. The client calls the home interface methods to create and destroy proxies for the remote interface.

EJB container The EJB specification defines a container as the environment in which one or more EJB components execute. The container provides the infrastructure required to run distributed components, allowing client and component developers to focus on programming business logic, and not system-level code. In EA Server, the container encapsulates:

- The client runtime and generated stub classes, which allow clients to execute components on a remote server as if they were local objects.
- The naming service, which allows clients to instantiate components by name, and components to obtain resources such as database connections by name.
- The EA Server component dispatcher, which executes the component’s implementation class and provides services such as transaction management, database connection pooling, and instance lifecycle management.

EJB component implementation The Java class that runs in the server implements the bean's business logic. The class must implement the remote interface methods and additional methods for lifecycle management.

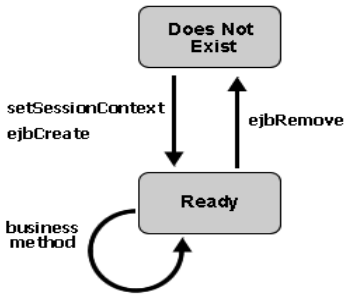
6.6.2 EJB Types

There are mainly three types of EJB:

1) Session Bean: For each session, a session bean is being generated from the client and expires as soon as the client exits. The duration of a session bean cannot exceed the range from the beginning of the session till the end of the usage of the system by the user. Session beans can be classified into two types:

- Stateless Session Bean: As the name indicates, here the state of the session is not managed and can handle multiple request from multiple client. After each method call, the container may destroy the stateless bean or recreate it or clears all the information pertaining to the last invocation method. Some of the examples of stateless session beans are:
  ■ A user verification service
  ■ An encoding engine
  ■ Any service that given some input always produces the same result

Life Cycle of Stateless Session Bean



Imgsrc:
https://docs.oracle.com/cd/E13226_01/workshop/docs81/doc/en/wls/guide/images/SLSBLifeCycle.gif

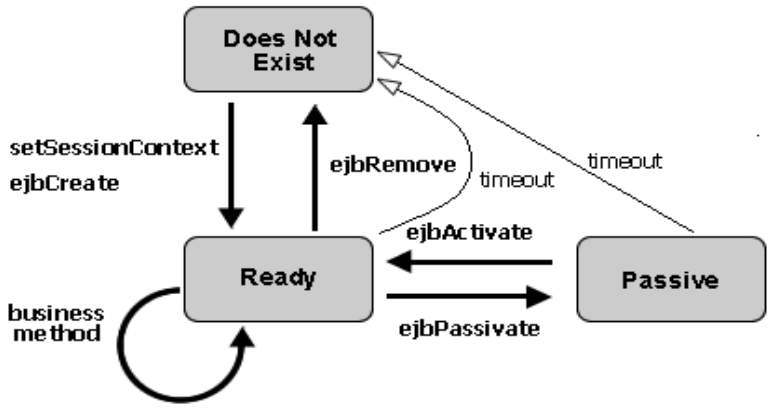
A stateless session bean has two states:

- **Does Not Exist**. In this state, the bean instance simply does not exist.

- **Ready**. When WebLogic server is first started, several bean instances are created and placed in the Ready pool. More instances might be created by the container as needed by the EJB container.

    - Stateful Session Bean: Act on behalf of a single client and maintain client-specific session information (called conversational state) across multiple method calls and transactions. They exist for the duration of a single client/server session. Some of the examples of stateful session bean are:
      ■ E-commerce web store with a shopping cart
      ■ Online bank
      ■ Tax declaration

Life Cycle of Stateful Session Bean



Imgsrc: https://docs.oracle.com/cd/E13226_01/workshop/docs81/doc/en/wls/guide/images/SFSBLifeCycle.gif

A stateful session bean has three states:

- **Does Not Exist**. In this state, the bean instance simply does not exist.

- **Ready**. A bean instance in the ready state is tied to particular client and engaged in a conversation.

- **Passive**. A bean instance in the passive state is passivated to conserve resource.

Methods in Session Bean

setSessionContext (SessionContext): The EJB container calls this method to set the associated session context.

ejbCreate(…): The container calls this method so that you can initialize your session bean instance. A client creates a stateful instance using the create bean class methods defined in the session bean's home interface. The container calls the corresponding ejbCreate method.

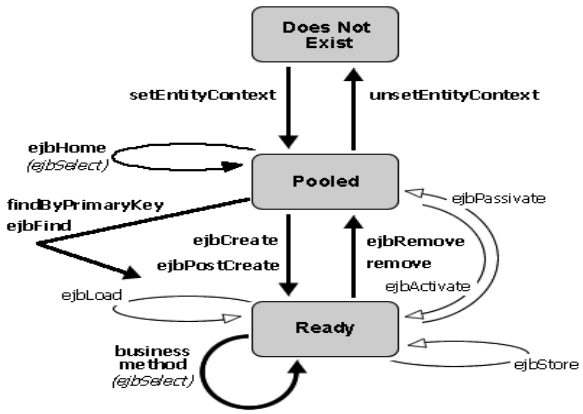
ejbActivate(…) : This method is called when the instance is activated from its passive state.

ejbPassivate(…): This method is called when the container intends to passivate the bean instance.

ejbRemove(…): A container invokes this method before it ends the life of the session object. This happens as a result of a client's invoking a remove operation, or when a container decides to terminate the session object after a timeout.

2) Entity Bean: Entity beans are persistent objects that can be stored in permanent storage. It lives on the entity or database layer of the 3-tier architecture.The entity bean data is the physical set of data stored in the database. An entity bean consists of the same files as a session bean. There are mainly two types of entity beans: Bean-managed persistent or container-managed persistent. Entity beans that manage their own persistence are called bean-managed persistence (BMP) entity beans. Entity beans that delegate their persistence to their EJB container are called container-managed persistence (CMP) entity beans.

Life Cycle of a Entity Bean



Imgsrc: https://docs.oracle.com/cd/E11035_01/workshop102/ejb/entity/images/EntityBeansLifeCycle.gif

An entity bean has the following three states:

- **Does not exist**: In this state, the bean instance simply does not exist.
- **Pooled state:** When WebLogic server is first started, several bean instances are created and placed in the pool. A bean instance in the pooled state is not tied to particular data, that is, it does not correspond to a record in a database table. Additional bean instances can be added to the pool as needed, and a maximum number of instances can be set.
- **Ready state**. A bean instance in the ready state is tied to particular data, that is, it represents an instance of an actual business object.

Methods for Entity Bean

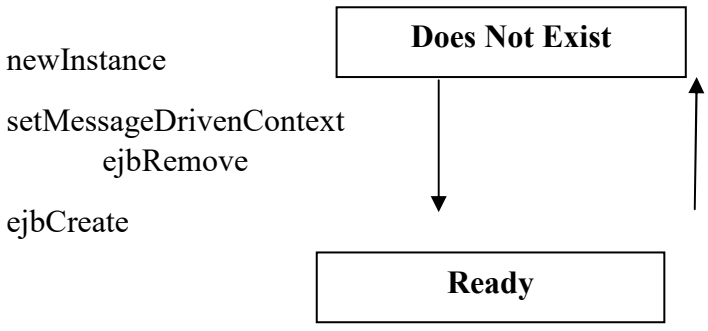
ejbCreate — An ejbCreate method is to be implemented corresponding to one create method declared in the home interface. When the client invokes

	the create method, the container first invokes the constructor to instantiate the object, then it invokes the correspondingejbCreate method. The ejb Create method performs the following: • creates any persistent storage for its data, such as database rows • intializes a unique primary key and returns it
ejbPostCreate	The container invokes this method after the environment is set. For each ejbCreate method, an ejbPostCreate method must exist with the same arguments. This method can be used to initialize parameters within or from the entity context.
ejbRemove	The container invokes this method before it ends the life of the session object. This method may perform any required clean-up, for example closing external resources such as file handles.
ejbStore	The container invokes this method right before a transaction commits. It saves the persistent data to an outside resource, such as a database.
ejbLoad	The container invokes this method within a transaction when the data should be reinitialized from the database. This normally occurs after the transaction begins.
setEntityContext	Associates the bean instance with context information. The container calls this method after the bean creation. The enterprise bean can store

	the reference to the context object in an instance variable, for use in transaction management. Beans that manage their own transactions can use the session context to get the transaction context. Resources can be allocated that will exist for the lifetime of the bean within this method. Resources are released in unset EntityContext.
Unset Entity Context	Unset the associated entity context and release any resources allocated in set EntityContext.
ejbActivate	Implement this as a null method, because it is never called in this release of the server.
ejbPassivate	Implement this as a null method, because it is never called in this release of the server.

3) Message Bean: Message Beans don't have a home, local home, remote or local interface but they have a single, weakly typed business method. They don't have any return values but however it is possible to send a response to the client. MDBs cannot send exceptions back to clients. They are stateless and can be durable or nondurable subscribers. Durable means that the subscriber receives all messages, even if it is inactive.

Life Cycle of Message Bean



A message driven bean has the following two states:

• Does not exist: In this state, the bean instance simply does not exist. Initially, the bean exists in the; does not exist state.

• Pooled state: After invoking the ejbCreate() method, the MDB instance is in the ready pool, waiting to consume incoming messages. Since, MDBs are stateless, all instances of MDBs in the pool are identical; they're allocated to process a message and then return to the pool.

Methods for Message Beans

onMessage(Message): This method is invoked for each message that is consumed by the bean. The container is responsible for serialising messages to a single message driven bean.

ejbCreate(): When this method is invoked, the MDB is first created and then, added to the ‘to pool’.

ejbCreate(): When this method is invoked, the MDB is removed from the ‘to pool’.

setMessageDrivenContext(MessageDrivenContext): This method is called for, as a part of the event transition that message driven bean goes through, when it is being added to the pool. This is called for, just before the ejbCreate().

Here we have discussed only the basics of EJB.

Stop to Consider

In a web-centric framework, EJBs handle the business logic behind web components such as servlets and JSPs.Client applications like Swing applications can also use EJBs, similar to web-centric applications.EJBs can serve as a bridge between network technologies commonly used in B2B e-commerce and enterprise systems.

6.7 SUMMING UP

1.CORBA stands for Common Object Request Service Broker Architecture.

2. It is an industry defined standard for distributed systems.

3. Objects and services are specified in the CORBA Interface Definition Language (IDL).

4. CORBA IDL is similar to other IDL's providing precise syntax for expressing methods and their parameters.

5.A Dynamic Invocation Interface (DII) is offered by CORBA to clients for allowing them to construct an invocation request at runtime.

6. Various services are offered by CORBA such as notification, licensing, security, trading etc.

7. The mode of communication in CORBA was a very simple one in which the client invokes a method on an object and waits for an answer.

8. The three invocation models supported by CORBA are: synchronous, one way and deferred synchronous.

9.Event service deals with signalling the occurrence of an event so that the clients amenable to that event could take appropriate action.

10. For persistent communication, CORBA supports the message queuing model through an additional messaging service.

11. Client and server are the two types of processes that are distinguished by CORBA.

12. A name in CORBA consists of a sequence of name components which is of the form (id, kind) pair. Here id and kind are both strings.

13. Synchronization in CORBA is facilitated by the concurrency control and transaction service.

14. A transaction in CORBA is initiated by a client and consists of a series of invocations on object.

15. CORBA doesn't offer any generic caching and replication services.

16. The goal of CASCADE system is to provide a generic, scalable mechanism that allows any kind of CORBA object to be cached.

17.Fault tolerance is achieved in CORBA through replication by replicating objects into object groups.

18.The basis of DCOM is formed by Microsoft's component object technology COM. The goal of COM is to support the development of components that can be dynamically activated and that can interact with each other.

19. The remote-object model is adopted by DCOM.

20. By means of RPC communication between the client and the server takes place in CORBA.

21.An event class in DCOM is not required to be implemented separately as it provides a default implementation.

22. Persistent asynchronous communication is accomplished with the help of **Queued Components (QC).** QC is actually an interface to **Microsoft Message-Queuing (MSMQ).**

23. DCOM provides persistent object references called monikers that can be shared between multiple processes

24. DCOM makes use of a worldwide directory service known as Active Directory.

25.DCOM supports fault tolerance by means of automatic transactions, which forms an integral part of COM+.

26. Automatic transactions allow a developer to specify that a series of method invocations, possibly on different objects, should be grouped into a transaction.

27. DCOM handles security in two ways: Declarative security and Programmatic Security.

28. EJB are the software component module which was developed with the sole purpose to build/support enterprise specific problems and it is a reusable server-side software component.

29. A typical EJB consists of the following components: server, client and container.

30. There are mainly three types of EJB: session bean, message bean and entity bean.

6.8 Answers to Check Your Progress

1. Dynamic Invocation Interface (DII)

2. Events

3. Polling model
4. Separator

5. Library

6. Communicate

7. Proxy

8. Transient

9. Monikers

10. Active Directory

6.9 Possible Questions

1. What is a distributed object-based model?

2. Describe the global architecture of CORBA with the help of a diagram.

3. What is Dynamic Invocation Interface?

4. Mention the different services offered by CORBA.

5. What are the three invocation models supported by CORBA?

6. How is the naming service accomplished in CORBA?

7. How is fault tolerance achieved in CORBA?

8. Describe about the underlying principle of DCOM.

9. Describe with the help of a diagram the overall architecture of DCOM.

10.Explain about the event processing in DCOM.

11. What is custom marshalling? Explain.

12. What is a moniker? Explain the different types of moniker.

13. How is the security mechanism handled in DCOM?

14. What is EJB?

15. Explain the architecture of EJB.

16.What are the different types of EJB? Explain.

6.10 References and Suggested Readings

1. Van Steen, M., & Tanenbaum, A. S. (2017). *Distributed systems* (p. 20). Leiden, The Netherlands: Maarten van Steen.

2.https://infocenter.sybase.com/help/index.jsp?topic=/com.sybase.help.eas_5.2.easfg/html/easfg/BABEFEGE.htm

3.https://docs.oracle.com/cd/E13226_01/workshop/docs81/doc/en/wls/guide/session/LifeCycle.html?skipReload=true

4.https://testbook.com/full-form/ejb-full-form

5. https://egyankosh.ac.in/bitstream/123456789/10464/1/Unit-1.pdf

6.https://docs.oracle.com/cd/E24329_01/web.1211/e24446/ejbs.htm#INTRO255

7.http://st.inf.tu-dresden.de/files/teaching/ss11/cbse/slides/12-ejb.pdf

---×---

UNIT- 7
ADVANCE WEB TECHNOLOGIES-I

Unit Structure:

7.1 INTRODUCTION

A web service (WS) is a service offered by an electronic device to another electronic device, communicating with each other via the Internet or a server running on a computer device, listening for requests at a particular port over a network, serving web documents (HTML, JSON, XML, images).

In practice, a web service commonly provides an object-oriented web-based interface to a database server, utilized for example by another web server, or by a mobile app, that provides a user interface to the end-user. Many organizations that provide data in formatted HTML pages will also provide that data on their server as XML or JSON, often through a Web service to allow syndication. Another application offered to the end-user may be a mashup, where

a Web server consumes several Web services at different machines and compiles the content into one user interface.

7.2 OBJECTIVES

After going through this unit learner will able to

- Learn about different web services
- Understand the concept of ISAPI
- Understand the concept of SOAP
- Understand the concept of UDDI
- Learn about the WSDL

7.3 WEB SERVICES AND RELATED TECHNOLOGIES

The term "Web service" describes a standardized way of integrating Web-based applications using the XML, SOAP, WSDL and UDDI open standards over an Internet Protocol backbone. XML is the data format used to contain the data and provide metadata around it, SOAP is used to transfer the data, WSDL is used for describing the services available and UDDI lists what services are available.

A Web service is a method of communication between two electronic devices over a network. It is a software function provided at a network address over the Web with the service always-on as in the concept of utility computing.

Several software systems are used by many businesses for management. Data sharing between two software systems is a common requirement, and a Web service is a communication mechanism that makes this possible over the Internet. A service provider is a software system that processes requests for data, while a service requester is a software system that makes data requests.

- It is a communication application, or client-server application.
- The protocol used to communicate over a network between two devices.
- It is a software system designed to facilitate interoperable communication between machines.
- It is a group of guidelines or protocols that allow data to be transferred between two applications or devices.

Let us understand it by the figure 7.1 given below:

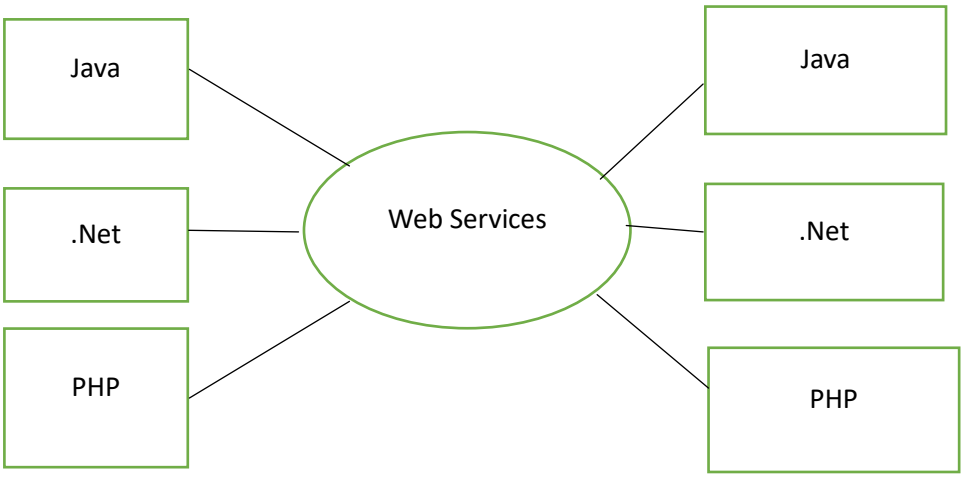


Figure 7.1 Web services and related technologies

As you can see in the figure 7.1, Java, .net, and PHP applications can communicate with other applications through web service over the network. For example, the Java application can interact with Java, .Net, and PHP applications. So, web service is a language independent way of communication.

STOP TO CONSIDER

Web service is a software function provided at a network address over the Web, always-on, like utility computing.

7.4 ISAPI

An Internet server API (ISAPI) is a collection of pre-written Windows program calls that programmers and developers can use to extend or customize HTTP servers or web servers that comply with ISAPI standards. These improvements are known as Internet Server Extension Applications (ISA) and ISAPI filters. They include features like dynamic web page generation for display on a Web client (browser) and database invocation.

Internet Server API: An Overview via Techopedia

Writing database apps, such as an order form entry system or a bespoke catalogue, is made simpler by the Internet Server Application Programming Interface. Using an HTML form and

ISAPI, a programmer can obtain user data and use that data to generate a customized page for the user.

A programmer can create two different kinds of server extensions using ISAPI:

Internet server applications (ISA) are similar to CGI applications in functionality, but they are thought to be faster because they are dynamic link libraries (DLLs), which load into memory and do not require location and re-reading like executable CGI applications do.

ISAPI filters are employed by

- Compression
- Encryption
- Custom authentication
- Logging schemes

7.5 SOAP

SOAP stands for Simple Object Access Protocol. It is an XML-based protocol for accessing web services.

SOAP is a W3C recommendation for communication between two applications.

SOAP is XML based protocol. It is platform independent and language independent. By using SOAP, you will be able to interact with other programming language applications.

Advantages of Soap Web Services

WS Security: SOAP defines its own security known as WS Security.

Language and Platform independent: SOAP web services can be written in any programming language and executed in any platform.

Disadvantages of Soap Web Services

Slow: SOAP uses XML format that must be parsed to be read. It defines many standards that must be followed while developing the SOAP applications. So, it is slow and consumes more bandwidth and resource.

WSDL dependent: SOAP uses WSDL and does not have any other mechanism to discover the service.

7.6 UDDI

UDDI, or Universal Description, Discovery and Integration, is an Extensible Language Markup (XML)-based standard to describe, publish and find information about web services. It is also a registry for businesses providing web-based services to list themselves and find partners on the Internet.

The UDDI standard includes a set of specifications that define a distributed registry service for web services. It is an XML-based and platform-independent framework that uses Web Service Definition Language (WSDL) to describe interfaces to web services. It is also an open, industry-wide framework since it lets all kinds of businesses list themselves on the Internet and discover, interact with, and carry out transactions with each other.

UDDI is an XML-based standard to describe, publish and find information about web services.

UDDI is often compared to a traditional telephone book's white, yellow, and green page, which correspond to UDDI's categories of Organizations, Categories and Bindings respectively. The project lets businesses list themselves by name, product, location, or web services offered. The ultimate goal of UDDI is to streamline online transactions by helping companies find one another on the Web and make their systems interoperable for e-commerce.

The UDDI specification utilizes World Wide Web Consortium and Internet Engineering Task Force standards such as XML, Hypertext Transfer Protocol, and Domain Name System protocols. It has also adopted early versions of the proposed Simple Object Access Protocol (SOAP) messaging guidelines for cross-platform programming. UDDI can also communicate via

Common Object Request Broker Architecture and Java Remote Method Invocation Protocol.

The two main functions of UDDI are to provide a SOAP-based protocol and a set of registries. The registries are globally replicated, and the protocol defines how clients communicate with these registries. A UDDI registry contains the metadata of web services and includes a pointer to the WSDL description of a service. UDDI also includes a set of WSDL port type definitions to manipulate and search the registry.

UDDI's XML schema for SOAP messages defines documents that describe a business and provide information about its services. The schema also includes a common set of APIs (Application Program Interfaces) for querying and publishing information to UDDI directories, plus an API to replicate directory entries between peer UDDI nodes.

UDDI's XML schema for SOAP messages defines documents that describe a business and provide information about its services.

UDDI relies on a distributed registry of businesses plus their service descriptions, implemented in a common XML format, to manage and enable the discovery of web services. Web service providers advertise their services on the UDDI registry that manages all the information about the provider, service implementations and service metadata. Organizing and cataloguing web services on a registry allows for sharing and reuse within an enterprise or with an external partner. Today, these registries are based on the UDDI 3.0 specification.

A UDDI registry is a central repository for three primary nodes labelled Providers, Services, and Bindings. Two types of registries are available: private and public. A private registry lets businesses publish and test their internal web applications in a secure, private environment. A public registry consists of peer directories containing information about businesses and their web services. Both private and public registries comply with the same specifications.

UDDI registry directory structure

UDDI has a very specific directory structure and three types of directory listings:

- Organizations (also known as Providers)
- Services
- Bindings (also known as end points)

The Provider’s folder is where all the service providers are listed in the UDDI server. It includes a Relationships tab to specify organizational structures (parent-child) and partnerships (peer), which describe a company's operational structure and convey business relationships.

Steps for publishing services to the UDDI registry, which users can think of as white pages where businesses that provide web-based services can list themselves and find partners on the Internet.

The Services, which are logical containers, are below the service providers, and the Bindings for the services are below the services. Services include a collection of Bindings and Categories. The Bindings listing refers to endpoints or physical locations where the services exist. They are the services' own children. A single service, in turn, is a child of an organization (provider).

Categories are listed at every level in the hierarchy (meaning each level has a Categories tab), and that helps to make UDDI a powerful and expressive standard. A category classifies the different levels through metadata. When a user adds a category to a level of the UDDI structure (providers, services, bindings), they can either use a standard category or create a custom category.

Bindings are the lowest level in the UDDI hierarchy. They represent implementation locations or service details, like WSDLs. In a binding for a service, the access point could be a URL to where the service exists. A WSDL deployment is another possible access point. Regardless of the access point, it's usually better to expose the service and/or its WSDL for simplicity instead of putting all the

details of the service, such as its parameters and other technical information, in the UDDI registry.

Thirty-six companies announced UDDI 1.0 in September 2000. Microsoft, IBM, Ariba, Dell, Merrill Lynch, Sun Microsystems, and SAP were some of the companies that spearheaded the project. In the years since the original announcement, the UDDI project has grown into an industry-wide initiative sponsored by the Organization for the Advancement of Structured Information Standards (OASIS). It now includes hundreds of companies, including some of the biggest names in the corporate world.

In November 2000, UDDI entered its public beta-testing phase. Each of its three founders -- Microsoft, IBM, and Ariba -- set up registry servers that were interoperable with servers from other members. As information goes into a registry server, it is shared by servers in other businesses. A UDDI server is a directory of available service providers and their services.

The Microsoft UDDI server was originally part of the Windows Server platform. It later became part of BizTalk Server. IBM maintains two public registries. Its test registry lets businesses test and validate their web services and experience the UDDI registration process without placing the service in an official registry.

7.7 WSDL

The Web Services Description Language (WSDL /ˈwɪz dəl/) is an XML-based interface description language that is used for describing the functionality offered by a web service.The acronym is also used for any specific WSDL description of a web service (also referred to as a *WSDL file*), which provides a machine-readable description of how the service can be called, what parameters it expects, and what data structures it returns. Therefore, its purpose is roughly similar to that of a type signature in a programming language.

The latest version of WSDL, which became a W3C recommendation in 2007, is WSDL 2.0. The meaning of the

acronym has changed from version 1.1 where the "D" stood for "Definition".

The WSDL describes services as collections of network endpoints, or ports. The WSDL specification provides an XML format for documents for this purpose. The abstract definitions of ports and messages are separated from their concrete use or instance, allowing the reuse of these definitions. A port is defined by associating a network address with a reusable binding, and a collection of ports defines a service. Messages are abstract descriptions of the data being exchanged, and port types are abstract collections of supported operations. The concrete protocol and data format specifications for a particular port type constitutes a reusable binding, where the operations and messages are then bound to a concrete network protocol and message format. In this way, WSDL describes the public interface to the Web service.

WSDL is often used in combination with SOAP and an XML Schema to provide Web services over the Internet. A client program connecting to a Web service can read the WSDL file to determine what operations are available on the server. Any special datatypes used are embedded in the WSDL file in the form of XML Schema. The client can then use SOAP to actuallycall one of the operations listed in the WSDL file, using for example XML over HTTP.

The current version of the specification is 2.0; version 1.1 has not been endorsed by the W3C but version 2.0 is a W3C recommendation. WSDL 1.2 was renamed WSDL 2.0 because of its substantial differences from WSDL 1.1 By accepting binding to all the HTTP request methods (not only GET and POST as in version 1.1), the WSDL 2.0 specification offers better support for RESTful web services, and is much simpler to implement. However support for this specification is still poor in software development kits for Web Services which often offer tools only for WSDL 1.1For example, the version 2.0 of the Business Process Execution Language (BPEL) only supports WSDL 1.1.

WSDL 1.1 Term	WSDL 2.0 Term	Description
Service	Service	Contains a set of system functions that have been exposed to the Web-based protocols.
Port	Endpoint	Defines the address or connection point to a Web service. It is typically represented by a simple HTTP URL string.
Binding	Binding	Specifies the interface and defines the SOAP binding style (RPC/Document) and transport (SOAP Protocol). The binding section also defines the operations.
PortType	Interface	Defines a Web service, the operations that can be performed, and the messages that are used to perform the operation.
Operation	Operation	Defines the SOAP actions and the way the message is encoded, for example, "literal." An operation is like a method or function call in a traditional programming language.
Message	—	Typically, a message corresponds to an operation. The message contains the information needed to perform the operation. Each message is made up of one or more logical parts. Each part is associated with a message-typing attribute. The message name attribute provides a unique name among all messages. The part name attribute provides a unique name

		among all the parts of the enclosing message. Parts are a description of the logical content of a message. In RPC binding, a binding may reference the name of a part in order to specify binding-specific information about the part. A part may represent a parameter in the message; the bindings define the actual meaning of the part. Messages were removed in WSDL 2.0, in which XML schema types for defining bodies of inputs, outputs and faults are referred to simply and directly.
Types	Types	Describes the data. The XML Schema language (also known as XSD) is used (inline or referenced) for this purpose.

- History

WSDL 1.0 (Sept. 2000) was developed by IBM, Microsoft, and Ariba to describe Web Services for their SOAP toolkit. It was built by combining two service description languages: NASSL (Network Application Service Specification Language) from IBM and SDL (Service Description Language) from Microsoft.

WSDL 1.1, published in March 2001, is the formalization of WSDL 1.0. No major changes were introduced between 1.0 and 1.1.

WSDL 1.2 (June 2003) was a working draft at W3C, but has become WSDL 2.0. According to W3C: WSDL 1.2 is easier and more flexible for developers than the previous version. WSDL 1.2 attempts to remove non-interoperable features and defines the HTTP 1.1 binding better. WSDL 1.2 was not supported by most SOAP servers/vendors.

WSDL 2.0 became a W3C recommendation in June 2007. WSDL 1.2 was renamed to WSDL 2.0 because it has substantial differences from WSDL 1.1. The changes are the following:

- Added further semantics to the description language
- Removed message constructs
- Operator overloading not supported
- PortTypes renamed to interfaces
- Ports renamed to endpoints

- **Subset WSDL**

Subset WSDL (SWSDL) is a WSDL with the subset operations of an original WSDL. A developer can use SWSDL to access Subset Service, thus handle subset of web service code. A Subset WSDL can be used to perform web service testing and top-down development. Slicing of a web service can be done using a Subset WSDL to access Subset Service. Subset Service can be categorized into layers using SWSDL. SWSDLs are used for Web service analysis, testing and top-down development. AWSCM is a tool that can identify subset operations in a WSDL file to construct a subset WSDL.

Security considerations

Since WSDL files are an XML-based specification for describing a web service, WSDL files are susceptible to attack.To mitigate vulnerability of these files, limiting access to generated WSDL files, setting proper access restrictions on WSDL definitions, and avoiding unnecessary definitions in web services is encouraged.

Check Your Progress

1.What Does Internet Server API Mean?

2. What are the functions of UDDI?

3.What are UDDI registries?

4. What is the history of UDDI?

5. What is WSDL?

7.8 SUMMING UP

- Web services and related technologies are the foundation of modern internet-based communication and data exchange.
- Web service and technologies include HTTP, REST, SOAP, XML, JSON, APIs, GraphQL, Microservices Architecture, and Service-Oriented Architecture.

- ISAPI, a technology used in Microsoft's Internet Information Services (IIS), allows developers to create dynamic, interactive web applications by extending its functionality.
- ISAPI extensions offer better performance compared to CGI programs and can handle tasks like processing requests, generating dynamic content, performing authentication, and managing sessions.
- GraphQL is a query language and runtime for APIs, allowing clients to request only the data they need in a single query.
- SOAP, or Simple Object Access Protocol, facilitates communication between systems over the internet using XML-based messages. It supports advanced features like security, reliability, and transnationality, making it suitable for enterprise environments.
- SOAP services are described using WSDL, an XML-based language that defines the service interface, operations, and message formats.
- UDDI, or Universal Description, Discovery, and Integration, is a specification for creating and managing distributed registries of web services. It enables businesses to publish, discover, and integrate web services over the internet.
- UDDI registries are hierarchically organized, with top-level nodes representing large-scale registries and lower-level nodes representing individual businesses or service providers. It is based on open standards and protocols, making it interoperable across different platforms and programming languages.
- WSDL documents are written in XML and typically consist of multiple elements representing the service interface, operations, messages, and bindings.
- WSDL promotes interoperability by providing a standardized way to describe web services, allowing clients to understand how to interact with the service regardless of the platform or programming language used.
- Various tools and frameworks exist for generating WSDL documents from existing code (e.g., Java, .NET) and for generating client code from WSDL documents, simplifying the development and consumption of web services.

- WSDL supports versioning to accommodate changes in the service interface over time, allowing clients to adapt to new versions of the service without breaking compatibility.

7.9 ANSWER TO CHECK YOUR PROGRESS

1. Programmers and developers can utilize an Internet server API (ISAPI), which is a set of pre-written Windows program calls, to expand or modify HTTP servers or web servers that adhere to ISAPI standards. The terms Internet Server Extension Applications (ISA) and ISAPI filters refer to these enhancements. They have functions like database invocation and dynamic web page production for display on a Web client (browser).

2. UDDI offers two primary services: a set of registries and a SOAP-based protocol. The protocol specifies how clients interact with these registries, which are globally replicated. Web service metadata is stored in a UDDI registry, which also has a link to the service's WSDL description. A collection of WSDL port type definitions for registry manipulation and search is also included with UDDI.

   Documents that give information about a business and its services are defined by UDDI's XML schema for SOAP messages. The schema also offers an API to replicate directory entries between peer UDDI nodes, as well as a common set of APIs (Application Program Interfaces) for publishing and querying data to UDDI directories. Documents are defined by UDDI's XML for SOAP messages.

3. UDDI relies on a distributed registry of businesses plus their service descriptions, implemented in a common XML format, to manage and enable the discovery of web services. Web service providers advertise their services on the UDDI registry that manages all the information about the provider, service implementations and service metadata. Organizing and cataloguing web services on a registry allows for sharing and reuse within an enterprise or with an external partner.

Today, these registries are based on the UDDI 3.0 specification.

4. Thirty-six companies announced UDDI 1.0 in September 2000. Microsoft, IBM, Ariba, Dell, Merrill Lynch, Sun Microsystems, and SAP were some of the companies that spearheaded the project. In the years since the original announcement, the UDDI project has grown into an industry-wide initiative sponsored by the Organization for the Advancement of Structured Information Standards (OASIS). It now includes hundreds of companies, including some of the biggest names in the corporate world.

   In November 2000, UDDI entered its public beta-testing phase. Each of its three founders -- Microsoft, IBM, and Ariba -- set up registry servers that were interoperable with servers from other members. As information goes into a registry server, it is shared by servers in other businesses. A UDDI server is a directory of available service providers and their services.

5. The Web Services Description Language (WSDL /ˈwɪz dəl/) is an XML-based interface description language that is used for describing the functionality offered by a web service. The acronym is also used for any specific WSDL description of a web service (also referred to as a *WSDL file*), which provides a machine-readable description of how the service can be called, what parameters it expects, and what data structures it returns. Therefore, its purpose is roughly like that of a type signature in a programming language.

   The latest version of WSDL, which became a W3C recommendation in 2007, is WSDL 2.0. The meaning of the acronym has changed from version 1.1 where the "D" stood for "Definition".

7.10 POSSIBLE QUESTIONS

1. What is web service and how does it differ from a traditional web application?

2. What is Internet Server API?

3. What are the uses of SOAP?

4. Explain the advantages and disadvantages of SOAP.

5. Explain the working of UDDI.

6. Explain the directory structure UDDI registry.

7. How is WSDL used in the web services over the internet?

8. What is Subset WSDL?

9. Explain the term binding in WSDL.

10. Explain the history of WSDL.

7.11 REFERENCES AND SUGGESTED READINGS

1. www.javatpoint.com
2. www.wikipedia.com

---×---

UNIT-8
ADVANCED WEB TECHNOLOGIES-II

Unit Structure:

8.1 INTRODUCTION

The evolution of the Internet has brought about a paradigm shift in the way we interact with information, services and each other. As the Internet continues to evolve, so do the technologies that underpin its functionality. Here in this unit we will discuss about the concept of AJAX and its uses. In this unit we also discuss about AJAX XML file and .NET architecture with different languages.

8.2 OBJECTIVES

After going through this unit learner will able to

- Understand the concept of AJAX
- Learn the concept of HTTP request and response
- Learn how AJAX is related to XML file.
- Learn the working of AJAX with PHP and ASP
- Understand the concept of .NET frame work.

8.3 INTRODUCTION TO AJAX

8.3.1 The Evolution of Web Technologies -

Web technologies have come a long way from simple static web pages to dynamic and interactive applications. In the beginning, web servers served static HTML pages as the main content on the Internet. But the Internet began to change with the advent of server-side programming languages such as PHP and Python, as well as technologies such as JavaScript. This change makes it easier to create dynamic web applications that can respond instantly to user input.

The advent of Web 2.0, which emphasized user-generated content, teamwork, and interactive user experiences, further changed the Internet. AJAX (asynchronous JavaScript and XML) technologies have improved the responsiveness and interactivity of online applications by facilitating the smooth transfer of data between client and server.

8.3.2 What is AJAX?

AJAX is a set of web development tools that allow the browser and server to communicate asynchronously. Because AJAX enables seamless communication in the

background, parts of a web page can be dynamically updated without having to completely reload the page. This contrasts with traditional web applications, which rely on synchronous communication, where every user action results in a complete page refresh. AJAX is basically XMLHTTPRequest (XHR) object, a JavaScript API that makes it easy to connect to a server, is the basis of AJAX. Without interfering with the user experience, developers can send HTTP requests to the server, receive data, and asynchronously update the web page using an XHR object.

8.3.3 Working of AJAX

As the client interacts with an HTML form, a specific event triggers JavaScript code within the client's browser. This code initiates an AJAX request, often implemented using an object known as XML HTTPRequest, to the web server. While the client remains engaged with the form, the web server receives the AJAX request and triggers the appropriate server-side code to process it. This could involve database queries, calculations, or any other server-side operations necessary to fulfil the request. Following the processing on the server side, an AJAX response is generated and passed back to the web server. This response typically contains data or instructions requested by the client. Meanwhile, the client continues its activities uninterrupted, unaware of the communication occurring behind the scenes. Upon receiving the AJAX response, the browser automatically updates the relevant parts of the webpage, reflecting the results of the server-side processing.

8.3.4 Benefits of AJAX

First, by enabling smooth, asynchronous changes to web pages, AJAX improves the user experience by reducing latency and providing a more responsive experience. Relentlessly waiting for the page to completely reload is no longer a problem for users, resulting in a significantly improved, interactive and smooth browsing experience. Second, by allowing developers to update specific sections of a website without having to reload all of the content, AJAX helps improve performance. For this optimization, web applications run faster and become more scalable. It also uses less bandwidth and puts less strain on the server. Last but not least, AJAX enables greater interactivity, allowing programmers to develop dynamic web applications that quickly respond to user input such as mouse clicks, button presses, and form submissions.

STOP TO CONSIDER

AJAX revolutionized web development by enabling asynchronous communication between the client and server.

8.4 REQUEST AND RESPONSE IN HTTP

Communication on the World Wide Web is based on the Hypertext Transfer Protocol (HTTP). It defines the format and method of transmitting messages between clients and servers. Since HTTP uses a TCP/IP connection, ensuring reliable data delivery.

HTTP requests are initiated by clients using various methods, each serving a distinct purpose:

- GET: Retrieves data from the server.

- POST: Submits data to be processed by the server.
- PUT: Updates existing data on the server.
- DELETE: Removes data from the server.

HTTP responses are accompanied by status codes, providing information about the outcome of the request. These codes are categorized into different ranges, including:

- 1xx: Informational responses.
- 2xx: Successful responses.
- 3xx: Redirection messages.
- 4xx: Client errors (e.g., 404 Not Found).
- 5xx: Server errors (e.g., 500 Internal Server Error).

- **Client-Server Interaction**

Client-Side Request Handling: Sending Data to the Server –

When a client initiates a request, it creates an HTTP message containing the relevant data and sends it to the server. This process involves specifying the HTTP method, URL, headers, and optionally message body. Commonly used methods include GET and POST, each of which is suitable for different types of interactions.

The process begins with an event that triggers an AJAX request in client-side JavaScript code. This event could be a button click, a form submission, or even a page load. When an event occurs, JavaScript initializes an XML HTTPRequest (XHR) object, which serves as a bridge for asynchronous communication between the client and server. The XHR object is then configured with important

request details such as the HTTP method (e.g. GET, POST), server-side resource URL, optional request headers, and any data to be sent in the request body. These configurations determine the nature and extent of communication between the client and server. Once the XHR object is configured correctly, it sends an asynchronous request to the server. Unlike traditional synchronous requests, where the browser waits for the server to respond before continuing, AJAX requests allow the browser to continue performing other tasks while waiting for the server to respond.

Server-Side Processing: Handling Requests and Generating Responses –

When an HTTP request from a client reaches the server, the server first parses the incoming data to extract relevant information. This includes checking parameters, headers, cookies, and request method (GET, POST, PUT, DELETE, etc.). After parsing the request, the server determines the appropriate handler or controller to handle the request based on the requested URL (Uniform Resource Locator) and any predefined routing rules defined in the application configuration.

Before performing the requested action, the server may perform authentication and authorization checks to verify the client's identity and permissions. Authentication involves validating the credentials provided by the client, while authorization determines whether the client has the necessary permissions to perform the requested operation. Once authenticated and authorized, the server executes the appropriate business logic associated with the requested operation. This may include querying databases, calling

external services or APIs, performing calculations, or any other calculation required to complete the request.

After the business logic is executed, the server creates an HTTP response to send back to the client. This response includes several components, including an HTTP status code indicating the result of the request (e.g. success, resource not found, server error), headers providing additional metadata about the response, and a message body containing the actual data or content to be found delivered to the customer. The server sends the crafted HTTP response back to the client over the network. This involves packaging the response into TCP/IP packets and routing them over the Internet to the client device.

Returning Responses to the Client: Data Formats and Content Types –

The client sends a request to the server, the server processes the request and generates a response to return to the client. The format of this response data is critical for effective communication. Two common formats are JSON (JavaScript Object Notation) and XML (Extensible Markup Language). JSON is lightweight and understandable by both humans and machines, making it ideal for transmitting structured data, especially in AJAX requests. On the other hand, XML provides a hierarchical structure suitable for complex data and is widely used to exchange data and enable interoperability between different systems. Content negotiation techniques allow clients and servers to agree on the most appropriate format based on factors such as the client's preferences and the server's capabilities.

8.5 AJAX and XML File

AJAX (Asynchronous JavaScript and XML) is a technique used in web development to create dynamic and interactive web applications. It allows us to make requests to a web server asynchronously, without needing to reload the entire page. Although the term includes "XML," it's common to use other data formats like JSON instead of XML due to JSON's simplicity and flexibility.

(i) How AJAX related to XML

- AJAX allows us to send and receive data asynchronously, that means the user can continue interacting with the web page while data is being fetched from or sent to the server in the background.
- AJAX is implemented using JavaScript, the primary scripting language of web browsers. JavaScript code is responsible for initiating and handling AJAX requests.
- In AJAX, the XMLHTTP Request object is used to make HTTP requests from the client-side JavaScript code to the server. It provides methods and properties for sending and receiving data over HTTP.
- The term "XML" is part of AJAX, it's more of a historical artefact. Initially, XML was commonly used as the data format for exchanging data between the client and server. However, due to XML's verbosity and complexity, JSON (JavaScript Object Notation) has become the preferred format for AJAX requests and responses due to its simplicity and ease of use.
- AJAX is commonly used for various purposes in web development, such as loading content dynamically, submitting forms asynchronously, fetching data from a

server for updates without refreshing the entire page, and interacting with web services and APIs.

- While XMLHTTPRequest is still used in some cases, modern web development often utilizes newer APIs like Fetch API or libraries/frameworks such as Axios or jQuery.AJAX(), which provide simpler and more powerful ways to perform AJAX requests.

8.6 AJAX WORKING WITH PHP

AJAX with PHP is a powerful combination for creating dynamic and interactive web applications. It allows us to fetch and send data to the server asynchronously, without needing to reload the entire page.

- AJAX is a set of web development techniques that enables web pages to be updated asynchronously by exchanging data with a web server in the background. While the name includes "XML," modern applications often use other data formats like JSON due to its simplicity.
- PHP is a server-side scripting language used for web development. It's often used to generate dynamic content, interact with databases, and handle server-side logic.
- Working of AJAX with PHP
    a) We can use JavaScript to make asynchronous requests to the server. This is typically done using the XMLHTTPRequest object or the newer Fetch API.
    b) PHP handles the requests received from the client, performs any necessary operations

(such as fetching data from a database), and generates a response.

c) Once the server processes the request, it sends a response back to the client. This response can be HTML, XML, JSON, or any other format. JavaScript on the client-side then processes this response and updates the web page accordingly.

- **Use cases:**

a) AJAX can be used to fetch data from the server without refreshing the entire page. This is commonly used in applications like social media feeds or news websites.

b) Instead of submitting a form traditionally, AJAX can be used to send form data to the server and handle the response dynamically.

c) AJAX allows us to update parts of a web page in real-time, such as chat applications or stock market tickers.

- **Security**

a) Ensure that our application is protected against CSRF attacks by using techniques like CSRF tokens.

b) Handle CORS headers properly to prevent unauthorized access to our server from other domains.

- **Practices**

    a) Always sanitize and validate user input to prevent security vulnerabilities such as SQL injection or XSS attacks.

    b) Implement proper error handling both on the client and server-side to provide a better user experience and to debug issues efficiently.

8.7 AJAX WORKING WITH ASP

AJAX with ASP (Active Server Pages) involves using JavaScript to asynchronously communicate with an ASP server to perform actions like fetching data, submitting form data, or updating parts of a web page without reloading the entire page.

- ASP is a server-side scripting language used for web development, similar to PHP. It's often used to generate dynamic content, interact with databases, and handle server-side logic. ASP.NET is a more recent iteration of ASP, providing a framework for building dynamic web applications.
- We use JavaScript to make asynchronous requests to the server. This is typically done using the XMLHTTPRequest object or the newer Fetch API.
- ASP handles the requests received from the client, performs any necessary operations (such as fetching data from a database), and generates a response.
- Once the server processes the request, it sends a response back to the client. This response can be HTML, XML, JSON, or any other format. JavaScript on the client-side then

processes this response and updates the web page accordingly.

Use Cases:

- AJAX can be used to fetch data from the server without refreshing the entire page. This is commonly used in applications like social media feeds or news websites.
- Instead of submitting a form traditionally, AJAX can be used to send form data to the server and handle the response dynamically.
- AJAX allows us to update parts of a web page in real-time, such as chat applications or stock market tickers.

Security:

- Ensure that our application is protected against CSRF attacks by using techniques like CSRF tokens.
- Handle CORS headers properly to prevent unauthorized access to our server from other domains.

Practices

- Always sanitize and validate user input to prevent security vulnerabilities such as SQL injection or XSS attacks.
- Implement proper error handling both on the client and server-side to provide a better user experience and to debug issues efficiently.

8.8 AJAX DATABASES

AJAX can be used to interact with a database dynamically, allowing us to fetch, insert, update, or delete data without reloading the entire web page.

Use of AJAX with a database:

- Using JavaScript, initiate an AJAX request from the client-side to the server-side script.
- This can be done using the XMLHTTPRequest object or modern alternatives like Fetch API or libraries like Axios or jQuery.AJAX().
- On the server-side, use a server-side scripting language like PHP, ASP.NET, Node.js, or Python (with frameworks like Django or Flask) to handle the request.
- Connect to the database using appropriate database connectors or libraries
- Execute SQL queries or use an ORM (Object-Relational Mapping) library to interact with the database.
- Depending on the AJAX request (e.g., fetch data, insert data, update data, delete data), execute the corresponding database operation.
- Once the database operation is completed, generate a response on the server-side.
- This response can be in various formats like HTML, JSON, XML, or plain text, depending on the application requirements.
- On the client-side, receive the response from the server-side script.
- Depending on the response format, update the web page content dynamically.

- Implement proper error handling both on the client and server-side to provide a better user experience and to debug issues efficiently.
- Ensure that the server-side code is secure and protected against SQL injection, Cross-Site Scripting (XSS), and other security vulnerabilities.
- Optimize database queries and server-side code for performance to ensure efficient data retrieval and processing.
- Consider implementing caching mechanisms or pagination for large datasets to improve performance.
- Use techniques like long-polling, WebSockets, or Server-Sent Events (SSE) for real-time updates if the application requires instant data synchronization between clients and the server.

8.9 AJAX APPLICATIONS AND EXAMPLES

Some of the common applications and examples of AJAX are:

- **Dynamic Content Loading**: AJAX allows us to load content dynamically without refreshing the entire web page. This is useful for components like comments sections, search results, or news feeds, where new content can be fetched and displayed without disrupting the user's experience.
- **Form Submission**: Instead of traditional form submission, AJAX can be used to submit form data asynchronously. This provides a smoother user experience, as the page does not need to reload after form submission. Form validation and feedback can also be implemented dynamically.
- **Auto-Suggest and Auto-Complete**: AJAX can power auto-suggest or auto-complete functionality in search bars or

input fields. As the user types, AJAX requests can be made to the server to fetch relevant suggestions or completions in real-time.

- **Infinite Scrolling**: With AJAX, we can implement infinite scrolling, where additional content is loaded as the user scrolls down the page. This is commonly used in social media feeds or product listings to provide a seamless browsing experience.
- **Real-Time Updates**: AJAX enables real-time updates of content on the web page without requiring manual refreshes. This is useful for applications like chat rooms, live sports scores, or stock market tickers, where data needs to be updated continuously.
- **Interactive Maps**: AJAX can be used to fetch and display data dynamically on interactive maps. This allows users to interact with the map and view additional information without reloading the entire page.
- **Drag-and-Drop Interfaces**: AJAX can power drag-and-drop interfaces, where elements can be rearranged or repositioned dynamically without page refreshes. This is commonly seen in to-do list applications or project management tools.
- **Single-Page Applications (SPAs)**: AJAX is a fundamental technology used in SPAs, where the entire application runs within a single web page. SPAs provide a desktop-like user experience with smooth transitions between views and no page reloads.
- **User Authentication and Authorization**: AJAX can be used for user authentication and authorization workflows,

such as logging in, logging out, or managing user sessions, without disrupting the user's browsing session.

- **Data Visualization**: AJAX can fetch data from the server for visualizations such as charts, graphs, or dashboards, allowing users to interactively explore and analyse data without reloading the page.

8.10 INTRODUCTION TO .NET FRAMEWORK

Microsoft aimed to make the World Wide online more lively by allowing individual devices, computers, and online services to collaborate intelligently to give richer solutions to consumers. Users may construct a wide range of value-based applications by intelligently integrating web sites on the Internet, including unified banking services, electronic bill payment, stock trading, insurance services, and complete supply chain management. Microsoft refers to this as 'Web services', and the software approach for building and delivering these services is known as '.NET'.

.NET is a software platform that covers everything needed to create applications for online services. It combines presentation technologies, component technologies, and data technologies on a single platform, allowing users to create Internet applications as simply as they would on desktop systems. The .NET Framework is one of the tools accessible through the.NET infrastructure and tools component of the platform.

(i) Architecture of .NET Framework

The.NET framework offers a new environment for developing and deploying scalable, distributed applications via the Internet. The.NET Framework provides a platform for designing, creating, and operating online services and applications. It consists of three different technologies:

- Common Language Runtime
- Framework Base Classes
- User and program interfaces (ASP .NET and WinForms)

- COMMON LANGUAGE RUNTIME (CLR)

CLR is the.NET Framework's main component, responsible for loading and running C# and other.NET languages. The top layer includes classes for constructing web services and dealing with the user interface. The CLR, is the heart and soul of the.NET Framework. CLR is a runtime environment that executes programs in C# and other languages. It also promotes cross-language compatibility.

The Common Language Runtime offers several kinds of services, which include:

- Loading and executing programs
- Isolating memory for applications
- Verifying type safety
- Compiling intermediate language (IL) into native executable code
- Providing metadata
- Managing memory (automatic garbage collection)
- Enforcing security
- Interoperability with other systems
- Managing exceptions and errors
- Supporting debugging and profiling

- Common Type System (CTS)

The.NET Framework supports several languages via the Common Type System, which is incorporated into the CLR. The CTS supports a wide range of types and actions present in most programming languages, thus calling one language from another does not need type discussions. Although C# is specifically built for the.NET platform, we may create.NET applications in a variety of other languages, including C++ and Visual Basic.

- *Common Language Specification*

The Common Language Specification offers a set of guidelines for interoperability on the.NET platform. These guidelines provide guidance to third-party compiler designers and library builders. The CLS is a subset of CTS, so the languages that support it can utilize each other's class libraries as if they were their own. CLS-compliant Application Program Interfaces (APSs) may be simply utilized by all.NET languages.

- *Microsoft Intermediate Language (MSIL)*

MSIL, or IL, is the instruction set into which all.NET programs are built. It is similar to assembly language, with instructions for loading, storing, initializing, and invoking functions. When we build a C# program or any program written in a CLS-compliant language, the source code is converted to MSIL.

- *Managed Code*

As we all know, the CLR is in charge of managing the execution of code written for the.NET platform. Managed code refers to code that satisfies the CLR at runtime to execute. Compilers that work with the.NET platform produce managed code. For example, the C# compiler produces managed code. C# (and other compilers that can

output managed code) produces IL code. The JIT compilers subsequently translate the IL code into native machine code.

- FRAMEWORK BASE CLASSES

.NET provides a set of fundamental classes we may utilize to develop applications easily. We can utilize them by simply instantiating them and using their methods, or by inheriting them through derived classes and enhancing their capabilities. Much of the functionality in the base framework classes lies in the massive namespace for many different functions, including:

- Input/output operation
- String handling
- Managing arrays, lists, maps, etc.
- Accessing the registry
- Accessing files and files systems
- Security
- Windowing
- Windows messages
- Database Management
- Evaluation of mathematical functions
- Connecting to the Internet

- USER AND PROGRAM INTERFACES

The.NET Framework includes the following tools for managing user and application interfaces:

- Windows forms
- Web forms
- Console Applications
- Web services

These tools allow users to create user-friendly desktop and web-based applications on the.NET platform utilizing a broad range of languages.

- VISUALS STUDIO .NET

Visual Studio.NET (VS.NET) provides an Integrated Development Environment (IDE) with a wide range of capabilities and productivity tools. These capabilities and technologies enable developers to create web apps more quickly and easily. Using web services and XML, regardless of the language used for development, there is now a single environment to learn, set up, and use. We do not need to switch between environments to build, debug, and deploy our code. VS.NET includes features that enhance assistance across the development lifecycle.

- .NET LANGUAGES

The.NET Framework is language-neutral. Currently, we can develop.NET applications using a variety of programming languages. They include:

- Native to .NET
  - C# (Specially created for .NET)
  - C++
  - Visual Basic
  - JavaScript
- Third-party Languages
  - COBOL
  - Eiffel
  - Perl
  - Python
  - SmallTalk
  - Mercury
  - Scheme

All.NET languages are not the same. Some languages may use components from other languages, some can build objects from classes created in other languages, and some languages can extend the classes of other languages using inheritance capabilities.

- BENEFITS OF THE .NET APPROACH

Microsoft has promoted the.NET approach to benefit both developers and users. Some of the primary benefits expected are:

- Simple and faster systems development
- Rich object model
- Enhanced built-in functionality
- Many different ways to communicate with the outside world
- Integration of different languages into one platform
- Easy deployment and execution
- Wide range of scalability
- Interoperability with existing applications
- Simple and easy-to-build sophisticated development tools
- Fewer bugs
- Potentially better performance

Check Your Progress

1. How does AJAX differ from traditional web development approaches?
2. Give examples of benefits of using AJAX in web development
3. What are the most commonly used HTTP methods and what are their purposes?
4. What are the advantages of using JSON and XML as data formats?
5. What do HTTP status codes indicate?

8.11 SUMMING UP:

- Web technologies have come a long way from simple static web pages to dynamic and interactive applications.
- In the beginning, web servers served static HTML pages as the main content on the Internet. But the Internet began to change with the advent of server-side programming languages such as PHP and Python, as well as technologies such as JavaScript
- AJAX is a set of web development tools that allow the browser and server to communicate asynchronously.
- AJAX is basically XMLHTTPRequest (XHR) object, a JavaScript API that makes it easy to connect to a server, is the basis of AJAX.
- AJAX with PHP is a powerful combination for creating dynamic and interactive web applications. It allows us to fetch and send data to the server asynchronously, without needing to reload the entire page.
- .NET is a software platform that covers everything needed to create applications for online services.
- CLR is the.NET Framework's main component, responsible for loading and running C# and other .NET languages.
- The Common Language Specification offers a set of guidelines for interoperability on the.NET platform
- Managed code refers to code that satisfies the CLR at runtime to execute.
- Visual Studio.NET (VS.NET) provides an Integrated Development Environment (IDE) with a wide range of capabilities and productivity tools.

8.12 ANSWER TO CHECK YOUR PROGRESS

1. In traditional web models, an HTML request results in a full page refresh. AJAX enables a web application user to interact with a web page without the interruption of constant web page reloading.
2. Some of the benefits of using AJAX are:
    - Enhances performance.
    - Improve response time.
    - Supported by multiple browsers.
3. The primary or most commonly-used HTTP methods are POST, GET, PUT, PATCH, and DELETE. These methods correspond to create, read, update, and delete (or CRUD) operations, respectively.
4. What are the advantages of using JSON and XML as data formats?

   JSON supports numbers, objects, strings, and Boolean arrays. XML supports all JSON data types and additional types like Boolean, dates, images, and namespaces.
5. HTTP response status codes indicate whether a specific HTTP request has been successfully completed.

8.13. POSSIBLE QUESTIONS

1. What is AJAX?
2. Explain the working of AJAX.
3. What are the benefits of AJAX?
4. Explain the request-response process in HTTP.
5. What is client-server architecture?
6. How are AJAX related to XML files?
7. Explain the AJAX working with PHP?

8. Explain the uses of AJAX with databases.

9. What is ASP?

10. Explain AJAX working with ASP.

10. Explain the AJAX and its applications.

11. Explain the architecture of .NET frame work.

12. What is VISUAL STUDIO.NET?

13. Explain the benefits of .NET approaches.

8.14 REFERENCES AND SUGGESTED READINGS

- Programming in C# by E Balagurusamy
- www.javatpoint.com
- www.stuedu.com

---×---

UNIT-9
WEB-SECURITY

UNIT STRUCTURE

9.1 INTRODUCTION

Web security is the mechanism that protectsthe data or information for unauthorized access or some malicious use in the web. In this unit, we will discuss the different problems and issues related to web security. Here we also discuss the basic concept of firewall and its configuration. In this unit, we will discuss the network layer firewall and application layer firewall. Here we also discuss about proxy server and its application in the web.

9.2 OBJECTIVES

After going through this unit learner will able to:

- Understand the concept of security in the web.
- Learn about some common web threats.
- Understand the concept of secure naming and DNS spoofing.
- Understand the concept of Secure Socket Layer and how it builds a secure connection between two sockets.
- Learn the concept of firewall and its types.

- Understand the concept of Application layer firewall and Network layer firewall.
- Learn the concept of Proxy server and how it protects the data or information from unauthorized access.

9.3 PROBLEMS AND ISSUES IN WEB SECURITY

Web security is concerned about different techniques and technologies which are designed to protect website, web applications and web services from various threats. Web security is divided into three parts-

(i) Firstly, secured naming mechanism for objects or resources.
(ii) Secondly, establishment of secured authenticated connections.
(iii) Lastly,required actions when a website sends a client a piece of executable code.

The following are some of problems and issues related to the web security

9.3.1 Threats

Web threats are some destructive or malicious activities and attacks that targets the web-sites, web applications and web services. These activities create serious problems such as data breaches, financial losses,damage the reputation of the web-sites and many more. The following are the common web threats

(i) **Malware:** Malware means designing malicious software to damage a computer system. This includes designing viruses, worms, Trojans and spyware. Malware is distributed through infected web-sites or web applications.

(ii) **Phishing:** phishing is a malicious work which attempt to deceive users into disclosing sensitive information such as user-name and password and credit card or debit card details by posing a trusted entity. It involves fake emails and messages. It uses some fraudulent emails, text messages, phone calls or web-site to initiate the people in sharing some confidential records.

(iii) **SQL injection:** It is also known as SQLI which is a common attack technique that involves inserting malicious SQL statement into web application's entry. In SQL injection an attacker inserts malicious SQL statements into an entry field. This can allow attackers to manipulate or retrieve data from the database.

(iv) **Cross-site Scripting (XSS):** In XSS attack, an attacker uses web-pages or web-application to send malicious code and compromise user to interact with this malicious code.XSS attacks can steal session cookies, hijack user's session and redirect user to malicious sites.

(v) **Distributed Denial of Service (DDoS):** DDoS is a cybercrime where overwhelming a web-server or network with a flood of internet traffic from multiple sources to make it inaccessible or degrade its performance. The DDoS attacks are a type of denial-of-service(DOS) attack, but they are more effective because they use more attacking traffic. Financial losses may occur due to DDoS.

(vi) **Man-in-the_Midde(MitM):** MitM is a cyber-attack, where attacker secretly steal data such as log-in credentials of financial information. Attacker can get this sensitive information like login name and password by exploring some insecure communication channels.

(vii) **Session Hijacking:** It refers to the malicious act that taking control of a user's web session. In session hijacking taking control over an active session by stealing session cookies or session IDs.

(viii) **Credential Stuffing:** It is a cyber-attack in which usercredential (username and password) are stolen to gain unauthorized access to user accounts on various web-sites. This type of attack is based on the

assumption that many people use the same username and password across multiple accounts.

(ix) **Social Engineering:** Social engineering is a technique of manipulating, influencing, a victim in order to gain control over the computer system or to steal personal and financial information. It uses psychological manipulation to trick user into making security mistakes or giving away sensitive information.

(x) **Zero-day-Exploits:** Zero-day exploits can be used to attack the targeted website or organizations. Cybercriminals can exploit these vulnerabilities to compromise systems, steal data or creates some other types of attacks before the vendor aware this issues and releases a patch.

9.3.2 Secure Naming

Secure naming refers to a process to provide names for some variables, files, databases etc. in such a way that it increases the security and reduces the risk. The following are some practices for secure naming.

(i) **Avoiding sensitive information:** Do not use some sensitive information like passwords, cryptographic keys in names. Use some authentic storage mechanism for sensitive information.

(ii) **Avoid ambiguity and confusion:** Use some distinct name and avoid similar names that cause some confusion and lead to some accidental misuse.

(iii) **Avoid SQL injection:** In case of naming databases, fields, tables and queries ensure that these names do not contain character that could be exploited in SQL injection attack. For this purpose, use some parameterized queries to reduce the SQL injection risks.

(iv) **Consistency:** Keep the consistency in naming conventions across your project or organization. It will help for reducing confusion and adds some extra security. Practice some security reviews which

maintain the security standard throughout your project.

(v) **Documentation:** Document the naming conventions and guidelines so that developers understand and support secure naming practices throughout the development life cycle.

9.3.3 DNS Spoofing

DNS spoofing means tricking a DNS server into installing a false IP address. A cache that holds an intentionally false IP address is called poisoned cache. In DNS spoofing,attacker finds out some weakness in the DNS system and takes overall control and then it will redirect to a malicious website. DNS spoofing technique is shown in the Figure 9.1

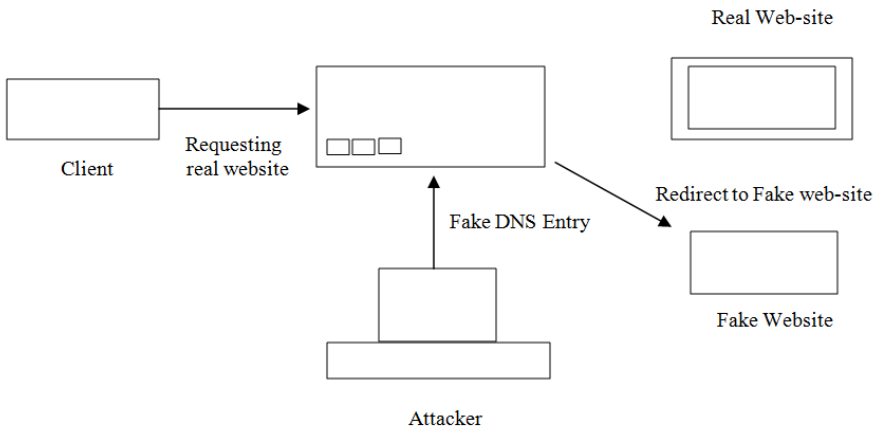


Figure 9.1: DNS Spoofing

In the Figure 9.1, client initially request for a real website he or she wishes to visit. It goes to the DNS server to resolve the IP address of that website. In the fake DNS entry part hacker already take control over the DNS server by detecting some flaws in the real website. After gaining the control the attacker add some false entries to the DNS server. Since there is a fake entry in the DNS server, it will redirect the user to the Wrong website.

To prevent DNS spoofing, following techniques can be utilized.

(i) DNS Security Extension (DNSSEC) is used to add an additional layer of security in the DNS resolution

process.DNSSEC provide digitally signing data which protect DNS spoofing.

(ii) To prevent DNS spoofing, source authentication can be used to verify the source of the DNS request. This can be achieved using some techniques like IPsec or TLS. It authenticates the requestor and ensures that the request has been tampered in transit.

(iii) Response Rate limiting (RRL) is a technique that limits the rate at which a DNS server responds to queries. This will help to prevent DNS spoofing attack.

9.3.4 Secure DNS

In 1994, IETF setup a working group to make DNS fundamentally secure. This is actually a project and is known as DNSsec (DNS security). The output of DNSsec is presented in RFC 2535. DNSsec is based on public-key cryptography. Every DNS zone has a public or private key pair.

DNSsec has three fundamental services

(i) Proof of where data actually sent.
(ii) Public key distribution
(iii) Transaction and request authentication.

The first serviceverifies that data being returned has been approved by the Zone's owner. The second one useful for storing and retrieving public keys securely. The final or the third one is required to guard against the spoofing attack.

DNS records are grouped into sets called RRsets(Resource Record sets). RRsets can be cached anywhere even at an untrustworthy servers without endangering the security.

DNSsec introduces several new record types. The first of these is the key-record, which holds the public key of a zone user, host and cryptographic algorithm used for signing purpose. It also holds the protocol used for transmission and some other bits.

The second new record type is the SIG record. It holds the signed hash according to the algorithm specified in the key record. The signature applies to all the records in the RRset.

The DNSsec designed in such a way that zone's private key can be kept offline. Once or twice in a day the client of zone's database can be manually transported to an offline machine on which the private key is located. All the RRsets can be signed there and the SIG record thus produced.

This method of pre signing RRsets tremendously speed up the process of answering queries, since no cryptographic has to be done in the fly.

9.3.5 SSL- The Secure Socket Layer

When web is on public view, it was initially for just distributing static pages. In 1995, Netscape Communication Corporation introduced a security package called SSL. This software and its protocols are now widely used.

SSL builds a secure connection between two sockets, which includes the followings.

(i) Parameter negotiation between client and server
(ii) Manual authentication of client and server
(iii) Secret communication
(iv) Data integrity protection

The positioning of SSL in the protocol stack is shown in Figure 9. 2. The SSL layer is placed between the application layer and the transport layer. The SSL layer is accepting the requests from the browser and sending them down to TCP for transmission to the server.

Application (HTTP)
Security (SSL)
Transport(TCP)
Network(IP)
Data link(PPP)
Physical(modem, ADSL, cable TV)

Figure 9. 2:Layers for a home user browsing with SSL

Once the secure connection established, SSL's main job is to handling compression and encryption. When HTTP is used over SSL, it is called HTTPS (secure HTTP). The SSL protocol has gone through several versions and it supports variety of different algorithms and options.

SSL consists of two sub protocols, one for establishing a secure connection and other for using it. SSL supports multiple cryptographic algorithms from which the strongest algorithm uses triple DES with three separate keys for encryption and SHA-1 for message integrity. Due to the combination of these, it is relatively slow. It is mostly used in banking and other application where highest security is required.

For an ordinary e-commerce application, RC4 is used with 128-bit for encryption and MD5 is used for message authentication.

For actual transport, a second sub-protocol is used as shown in Figure 9. 3. Messages from the browser are broken into units of upto 16 KB. If compression is enabled, each unit is then separately compressed. After that, a secret key is derived and parameter key is concatenated with the compressed text and the result hashed with the hashed algorithm MD5. This hash is appended to each fragment as the MAC. Now the compressed fragment plus MAC is then encrypted with the agreed-on-symmetric encryption algorithm. Finally a fragment header is attached and the fragment is transmitted over TCP connection.

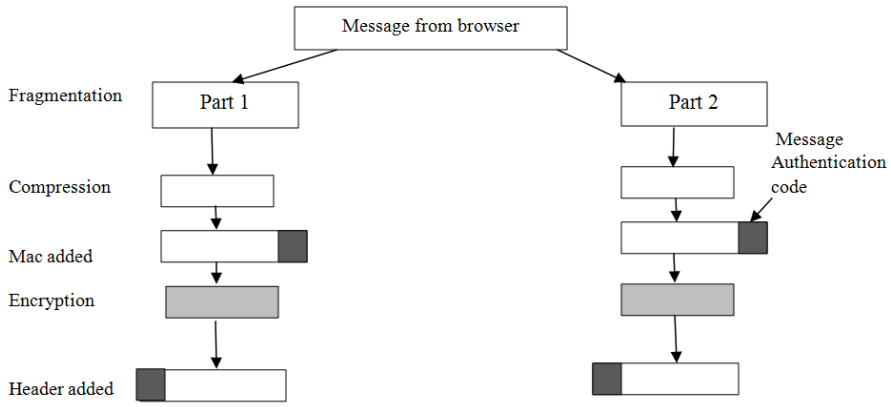


Figure 9. 3: Data transmission using SSL

Advantages of SSL:

(i) **Data protection:**SSL provide protection of our data and sensitive information which is communicated and exchanged in online. It protects all the sensitive

information like IDs, password, credit and debit card details etc. SSL helps the encryption of data which is very difficult for hackers or some attackers to decrypt them.

(ii) **Secure website:**Using SSL,the website is more secure in comparison to HTTP version website. Modern browser denotes user whether a website using SSL certificate or not. Using this security, user feels more comfortable in surfing and accessing web pages in that website.

(iii) **Reduce the risk of phishing attacks:**Using SSL on your website indicates that you are a valid person in a company or group. Therefore it is highly advised that without prior investigation never ever disclose your personal information in an untrusted website.

(iv) **Secure customer payment:**SSL uses encryption on both the end of browser as well as on the server. In terms of payment information that has been encrypted by SSL is secured and cannot be decrypted or read by third parties.

(v) **Easy to install:**To install an SSL certificate, no need for professional guidance. Anyone without prior technical knowledge can do the task quite easily. The hosting provides helps us in installing certificates and activating them in our own website.

(vi) **SEO Ranking:**The search engine Google has announced that they have updated their algorithm which states that the website which are using SSL definitely get an upper hand in comparison to the HTTP website.

(vii) **Protects from Google warning:**Many latest and updated browsers alert user by giving some warning messages if an SSL certificate is not installed. Therefore SSL certificates must be installed to avoid such warning messages.

(viii) **PCI/DSS Requirements:**The website which accepts online payments or performed online transaction, the SSL certificate is mandatory for such website. The website must follow the rules and regulations mentioned by the authority.

Disadvantages of SSL:

(i) **Cost:** The SSL certificate is expensive and the cost is depends upon the nature of security that user need on their website. The multi-domain certificate which covers domain and sub-domain obviously cost more in comparison to the single domain certificate.

(ii) **Slow processing:** The SSL makes website slow. Encryption has been performed at the both ends when server and browser go for handshaking process for establishing a good and secure connection. Once the connection is established SSL encrypts and decrypts the data and information before they are made available for further processing.

(iii) **Redirects:** When you redirect your website from HTTP to HTTPS, it is advised to redirect all the traffic from HTTP to HTTPS otherwise Google webmaster prompts errors like mixed content found which hampers SEO ranking.

(iv) **Renew SSL:** The SSL has to be renewed and updated periodically, especially every year. If SSL certificate is not renewed in the proper time, SSL stop validation and protection of your website. This maylead to serious drop in the revenue for the website which is especially deals in e-commerce and online transaction.

(v) **Caching issues:**To deal with encryption, additional servers are placed just after the encryption reaches the caching server. For this reason the data is properly encrypted and presented to the destination.

STOP TO CONSIDER

SSL provides maximum level of security. Organization such as Google recommends that websites use HTTPS to protect user privacy and security.

CHECK YOUR PROGRESS – I

1. **State True or False**
   (i) Malware means designing malicious software to damage a computer system.
   (ii) Phishing is not a malicious work.
   (iii) Secure naming reduces the security of the web.
   (iv) DNS Security Extension is used to prevent DNS spoofing.
   (v) The SSL layer is placed between the application layer and the Network layer.

9.4 FIREWALL

Firewall is analogous to that old medieval security standby: digging a deep moat around a castle. This design forced everyone entering or leaving the castle to pass over a single drawbridge where the police can inspect them. In network same policy also possible: a company can have many LAN connected in arbitrary ways, but all the network traffic to or from the company is forced through an electronic drawbridge called firewall.

Firewall configuration has two components: one is router that does packet filtering and another is the application gateway. Firewall typically defined as a system or group of systems that enforces an access control policy between two networks. Firewall can also be defined as a mechanism used to protect your network from unauthorized access. Firewall acts as gatekeeper between a company's internal network and the outside network.

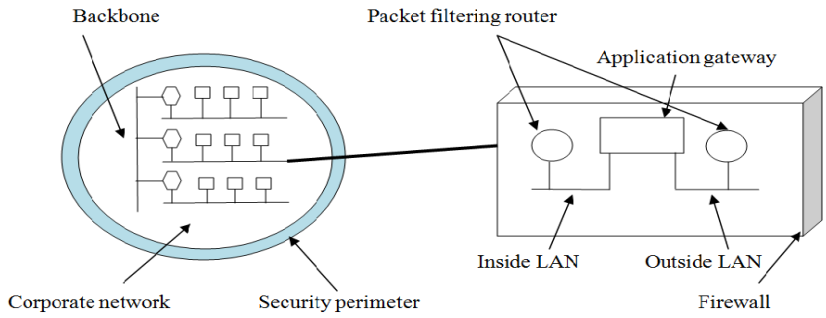


Figure 9. 4: A firewall consisting of two packet filters and an application gateway

A firewall consists of two packet filters and an application gateway as shown in the Figure 9.4. Firewall basically performs two important functions.

(i) **Gate keeping by firewall**: As firewall is acting as gate keeper, it always examines the sources and the location from where data is entered in the network. After detection of the sources and its location, based on the instruction according to policy of the firewall, it is decided whether or not to allow that information to enter in the network.

(ii) **Monitoring by firewall**: Data and information monitoring is one of the most important aspect of firewall design. It includes logging off all the system activities and generation of report for system administration. Monitoring of firewall can be categorized as active monitoring and passive monitoring. In active monitoring firewall notifies a manager when some incident is occurs in the system. Firewall alert administration through e-mail or a pager about any incident occurs in the network. In passive monitoring, a firewall logs a record of each incident in a file or a disk. The manager analyzes the log periodically to determine whether attempts to access the organization have increased or decreased.

9.4.1 Network Layer Firewall

Network layer firewall operates on network layer of OSI model. Network layer firewall is also known as the packet-filtering firewall. Main responsibility of the network layer firewall is to monitor and control network traffic based on IP-address, protocols and port numbers. The network layer firewall mainly protect against unauthorized access and certain types of malicious traffic.

Advantages of Network layer firewall:

(i) **Enterprise Network:** It is often deployed at the perimeter to protect corporate networks from external threats.

(ii) **Improving privacy:** It creates a privacy barrier between local network and the external internet.

(iii) **Access control:** Network layer firewalls can enforce access privileges and limit access to sensitive resources. It also controls internal network segmentation to regulate access and bandwidth.

(iv) **Early filtering:** It reduces the load by performing early high throughput filtering.

(v) **Real time monitoring:** Network layer firewall is fast and provides some tools for monitoring network activities in real time.

Disadvantages of Network layer firewall:

(i) **Limited inspection Capabilities:**Network layer firewall may not effectively detect or prevent sophisticated threats like advanced malware, zero-day exploits or some targeted attacks.

(ii) **No deep packet filtering:** Network layer firewall may not inspect the contents of packets deeply. Due this reason some malicious payloads may pass through network layer firewall.

9.4.2 Application Layer Firewall

The application layer firewall operates at the application layer that is in the layer 7of the OSI reference model. Application layer firewalls check the content of data packets to provide more depth control and security. Application layer firewall can be active and passive. Active application layer firewall inspect the entire incoming request actively against some known malicious attacks such as SQL injections, parameter tampering and cross site scripting. It passes only the request which is termed as "clean".Passive application layer firewall acting as intrusion detection system(IDS) where they also inspect all incoming request against some known vulnerabilities, but they don't actively reject or deny the requests if a potential attack is occurs.

The features of Application layer firewall are as follows.

(i) **Deep packet inspection:**It analyzes the payload of packets that means it allows inspecting the actual data being transmitted,for example HTTP request and

responses. It also identifies some malicious content such as malware.

(ii) **Application specific rules:**It enforces the rules based on some specific application or services such as browsers, email and file transfer protocols.

(iii) **User authentication:**It includes the mechanism for user authentication. It allows only authorized user to access specific applications or servers. It also enforces the security mechanism based on user roles and identities.

(iv) **Protection against common threats:**Application layer firewall specially designed to protect against application layer attacks, such as SQL injection, cross-site scripting (XSS) and Denial-of-service attacks.

(v) **Logging and Reporting:**It provides detailed logging of application traffic, helping in monitoring and auditing the applications.

9.5 PROXY SERVER

Proxy server refers a server which is working between the client and the web as shown in the Figure 9.5. There are different types of proxy server available according to the requirement for server and client communication. The basic purpose of proxy server is to protect the direct connection of internet clients and internet resources. The proxy server prevents the identification of IP addresses of client when it makes any request to the server.

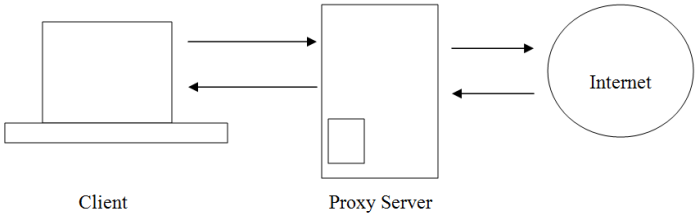


Figure 9.5: Proxy server between client and web

The components of proxy server are as follows.

- **Client:** The client layer includes any devices like computer, smartphones which are used to place a request for resources in the web. Here client using the devices send a request to the proxy server instead of directly to the destination server.
- **Proxy server layer:** The proxy server always receives the request from the client for immediate processing.
- **Request handler:** It always receives the incoming request, checking them and gives responses according to the rules(for example: access control, filtering).
- **Caching Mechanism:**It stores the resources locally to reduce the loading times and usage bandwidth.
- **Response handler:** The main job of the response handler is to receive the responses from the destination server and send them back to the client according to the requirement of the client.
- **Forwarding Engine:**It forwards the request to the destination server if the requested resources are not cached.
- **Logging and Monitoring:** It will keep tracks of different requests and responses for analyzing and security purposes.

Requirement of proxy server are discussed as follows.

- **Prevent the hacker activity:**To protect personal or organizational confidential data from unauthorized access or malicious use, some organization implement some different methods and techniques. But still there is a possibility that vital information whether personal or institutional can be hacked. To protect such type of data or resources, proxy server is used.
- **Content filtering:** Caching the content of website, proxy server helps fast access of data or information which is frequently accessed.
- **Controlling the uses of internet:** proxy server monitoring the activities of kids as well as employee in a particular organization. It uses some mechanism to restrict them by giving some suitable message for any harmful access.
- **Saving bandwidth:** Proxy server helps organizations or companies to get better performance by saving bandwidth.

CHECK YOUR PROGRESS-II

2. State True or False

(i) Application gateway is a component of firewall.

(ii)Network layer firewall operates on Application layer of OSI model.

(iii) Network layer firewall is fast and provides some tools for monitoring network activities in real time.

(iv) Proxy server refers a server which is working between the client and the web.

(v) Logging and Monitoring stores the resources locally to reduce the load time.

9.6 SUMMING UP

- Web security is the concern about different techniques and technologies which are designed to protect website.
- Web threats are some destructive or malicious activities and attacks that targets the web-sites, web applications and web services.
- Secure naming refers to a process to naming some variables, files, databases etc. in such a way that it increases the security and reduces the risk.
- DNS spoofing means tricking a DNS server into installing a false IP address.
- The SSL layer is placed between the application layer and the transport layer.
- Using SSL, the website is more secured in comparison to HTTP version website.
- Firewall typically defined as a system or group of systems that enforces an access control policy between two networks.
- The responsibility of the network layer firewall is to monitor and control network traffic based on IP-address, protocols and port numbers.
- Active application layer firewall inspect the entire incoming request actively against some known malicious attacks such as SQL injections, parameter tampering and cross site scripting.
- Proxy server refers to a server which is working between the client and the web.

9.7 ANSWER TO CHECK YOUR PROGRESS

1. (i) True (ii)False (iii) False (iv)True (v) False
2. (i) True (ii) False (iii)True (iv) True (v) False

9.8 POSSIBLE QUESTIONS

(1) What is web security?
(2) What is web-threats and how it targeted web services and websites?
(3) What is secure naming? Explain different practices for secure naming.
(4) What is DNSspoofing?
(5) Explain the fundamental services of DNS security.
(6) What is SSL?Explain how SSL builds a secure connection between two sockets.
(7) Explain the advantages and disadvantages of SSL.
(8) What is firewall? Explain the different components of firewall configuration.
(9) What is Network layer firewall?
(10) What is Application layer firewall?
(11) Explain the working of proxy-server.
(12) What are the requirements of proxy-server?

9.9 REFERENCES

Tanenbaum, A. S. (2003). *Computer networks*. Pearson Education India.

9.10 SUGGESTED READINGS

- Jackson, J. C. (2006). *Web Technologies*. Pearson India.
- Edition, W. S. F. (2023). Cryptography And Network Security.

---×---